Facilitating the Process of Enabling Applications Within Grid Portals

Maciej Bogdanski, Michal Kosiedowski, Cezary Mazurek, and Maciej Stroinski

Poznan Supercomputing and Networking Center, ul. Noskowskiego 10,
61-704 Poznan, Poland
{bogdan,kat,mazurek,stroins}@man.poznan.pl

Abstract. Grid technologies have had a major impact on the scientific research recently, creating great opportunities for utilization of powerful computing resources. Researchers all around the world have started to use the grid to solve important scientific problems in relatively short periods of time. Many scientific applications have already been enabled on the grid and many of them are waiting to become grid-aware. It is important to make this process of enabling applications on the grid easy. The tools used to create grid portals and grid access environments can help to achieve this task and facilitate the process of enabling applications on the grid and within grid portals. Many outstanding works have been conducted in that field. In this paper we present our contribution to these works that we believe can help to create an environment in which enabling applications within a grid portal is easy and requires little effort.

1 Introduction

Poznan Supercomputing and Networking Center has been involved in the research concerning grid technologies within a few major projects. These projects include the European Union funded GridLab [1] and Crossgrid [2] as well as some national grants. One of such national grants is the PROGRESS project, which has been co-funded by the State Committee for Scientific Research and Sun Microsystems Poland. This project, undertaken within the scope of the PIONIER program [3], has aimed to deliver a flexible access environment to computing resources and services [4]. The regular PROGRESS project works were conducted between December 2001 and the end of 2003. They resulted in the deployment of the PROGRESS grid-portal environment which is enabled under the PROGRESS HPC Portal [5]. This environment and this grid portal are, however, a subject to continuous research and development to provide the end users with better, more reliable and friendlier grid tools.

One of the paths that we have followed to facilitate the use of the PROGRESS HPC Portal, and its possible reproductions, has been fulfilling the users' need of more specialized and more user-friendly grid access environment. Having been introduced with a functional grid user interface providing multiple opportunities of utilization of grid resources the users seem not to be completely satisfied.

H. Jin, Y. Pan, N. Xiao, and J. Sun (Eds.): GCC 2004, LNCS 3251, pp. 175–182, 2004.
© Springer-Verlag Berlin Heidelberg 2004

They require an access environment which would be much easier to use for non-advanced users. Furthermore, the portal and website operators require an easy procedure to provide the functionality of the grid services to the users of their respective portals and websites. This is why we introduce the PROGRESS Portlet Framework which allows to build reusable web components and co-operates with PROGRESS grid services to deliver a flexible environment that facilitates the process of enabling applications within grid portals and other websites. We present these solutions in this paper. First, we shortly review some of the known grid portals in section 2 and the PROGRESS grid-portal environment in section 3. The PROGRESS Portlet Framework is overviewed in section 4. Further on, in section 5, we discuss how this framework and other PROGRESS grid tools can be employed to create reusable components and facilitate the building of a specialized and user-friendly access environment to grid-enabled applications. Eventually, we draw some conclusions in section 6.

2 Grid Portals

There are numerous outstanding grid portal solutions deployed around the world. However, the architectures of many of them, unlike the architecture of the PROGRESS HPC Portal and a few others, do not facilitate the building of new portals, enabling new applications within the portals and do not make the overall portal workplace advanced and comfortable to use. While we take a closer look at the PROGRESS HPC Portal itself in sections 3, 4 and 5, let us investigate some of the other grid portals now.

To actually investigate the portals we must look for the tools utilized therewith. One of the best known grid portals is HotPage [6]. HotPage is an implementation of the Grid Portal Toolkit (GridPort) [7] infrastructure. GridPort version 3 comes as a collection of grid and web services and enables easy utilization of grid resources by multiple remote clients, for example portals, applications or other grid services. Such architecture of a grid-portal environment facilitates the building of new portals and is very close to the architecture of the PROGRESS grid-portal environment which is described in section 3.

GridPort version 3 replaces the collection of Perl modules which formed version 2 of the toolkit [8] that has been utilized by many production computing portals as it delivers a good solution to enable grid resources on the web. Another portal implemented in the Perl technology is the Legion Grid Portal [9]. The logic of LGP is a Perl CGI script, which is used to process most of the user requests. Its role is to issue Legion commands on behalf of the user. LGP may be deployed for interaction with any underlying grid infrastructure, for example Globus. It, however, cannot deliver distributed grid services.

There are a few other grid portals, like numerous deployments of the Grid Portal Development Kit [10], the portal delivered within the EnginFrame package [11], the portal from University of Lecce [12] or Sun's Grid Engine Portal [13]. These, however, are also directed towards delivering basic grid functionality, thus the portal workplaces created by these portals do not offer rich functionality and good flexibility.

We have presented a range of known computing portals herewith. Some of them provide a wide range of possibilities for the end user and make their portal workplace an excellent opportunity to process the research and scientific experiments on the grid. In the next three sections we show how the PROGRESS HPC Portal compares to these solutions and how it facilitates the work of grid users and grid portals operators.

3 PROGRESS Grid-Portal Environment

The architecture of the PROGRESS grid-portal environment, which is utilized by researchers from the bio-informatics and chemistry area and such applications as, for example, Assembling DNA Sequences [14] or Gaussian [15], is presented in Fig. 1. The PROGRESS grid computing resources are managed by a Globus Toolkit installation, and are delivered by the PROGRESS Grid Resource Broker (GRB) [16], which decides where and how to run the application associated with the job submitted by a user via the grid services grouped within the Grid Service Provider (GSP) [17, 18], based on the job requirements and the current status of the grid resources. The whole infrastructure is assisted by the Data Management System [18, 19], which is used as the source of input data and the destination for results of computing experiments performed in the PROGRESS grid. The GSP and DMS services are available for use within the PROGRESS HPC Portal.

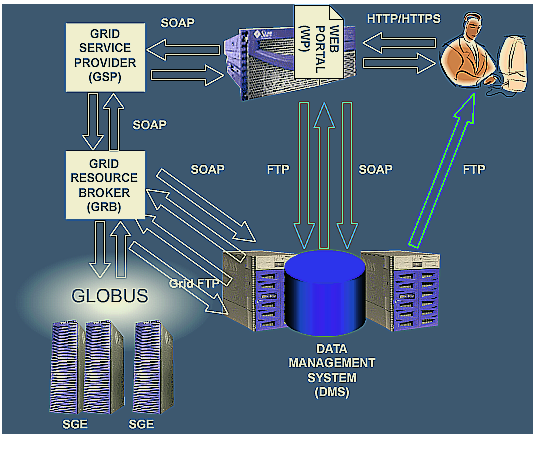

Fig. 1. PROGRESS grid-portal environment

In the PROGRESS grid-portal environment we assumed the SOAP protocol for communication between its distributed modules, thus all the modules provide Web Services interfaces. The richest Web Services interface is delivered by the GSP, which includes four independent services built using the J2EE technology. The two most important GSP services are the Job Submission and the

Application Management services. The first of these services delivers functions for building a computing job, submitting it to the grid for execution and for viewing its results; it is capable of maintaining both single-task and workflow grid jobs. The other one manages the PROGRESS application repository in a form of application descriptors, which include references to the application executables and lists of executable arguments, required environment variables and files. The main access point to the Data Management System is the Data Broker Service, which aims to deliver DMS resources and distributed modules under one external interface.

The Job Submission, Application Management and Data Broker services, which were discussed in detail in [17–19], and a few others of less importance to the grid nature of the portal, are utilized within the PROGRESS HPC Portal. The users of the portal can utilize the above-mentioned serviced and thus can manage their grid jobs, manage applications available within the application repository, and manage the data stored within the DMS.

In the testbed version of the PROGRESS HPC Portal this functionality has been provided to the portal users by a set of independent Java portlets. These were designed as separate modules, each implementing the whole stack enabling them to access the Web Services interfaces of the respective services. This has been changed for the most recent version of the PROGRESS HPC Portal and the above-mentioned portlets. The new portlets have been implemented with the use of the PROGRESS Portlet Framework and deployed for use within the PROGRESS HPC Portal; we describe the features of this framework in the next section. The use of this framework has not only enabled us to create highly reusable portlet codes, but has also provided us with an opportunity to easily reuse these parts of the already implemented portlets in new portlets that are responsible for communication with the underlying services and exposition of the obtained results as easily usable objects. This allows to easily create specialized portlets for specific applications which have been enabled on the grid; we show how this is done in section 5 of this paper.

4 PROGRESS Portlet Framework

The PROGRESS Portlet Framework features four layers in its architecture and forms a clear structure of an application accessing Web Services. The bottom layer consists of *Web Service Proxy* classes, which handle the SOAP communication with WS services. The next layer features *Request Handler* classes, which are capable of translating user's requests into a proper stream of invocations of WS methods. The third layer provides *Content Generator* classes, which prepare the content to be returned to the user. Finally, the highest layer includes *Provider* classes, which can take a form of portlets. The layered architecture of the PROGRESS Portlet Framework is presented in Fig. 2.

The Web Service Proxies extend a base proxy class, which utilizes the functionality of the Web Service Invocation Framework and uses the binding classes generated with the Axis toolkit based on the WSDL descriptions of services. The

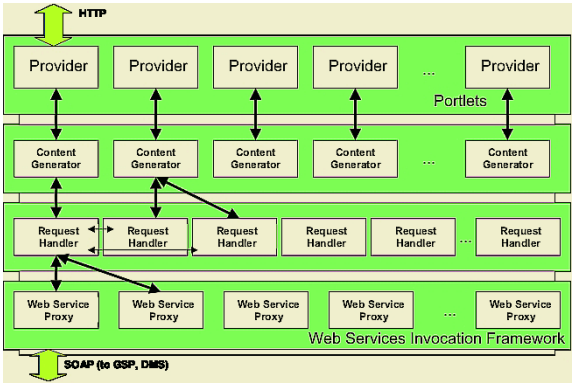

Fig. 2. Architecture of the PROGRESS Portlet Framework

Proxy classes are capable of communicating with the services using the SOAP protocol. They expose the *executeMethod()* method to enable the invocation of WS methods.

The Proxies' *executeMethod()* methods are invoked by the Request Handlers, which invoke proper WS methods based on user requests. One Request Handler method can invoke multiple WS methods. An invocation stream of Request Handler methods, which can be methods of one or of multiple Request Handlers, constitutes the action to be performed as stated in the user request. Each Http Request for portlet content contains the *action* and *page* parameters, which are analyzed by a Content Generator. Actions and pages are described in XML configuration files, in *actions.xml* and *pages.xml*, respectively. The configurations are dynamically loaded by classes referenced by Content Generators. This mechanism of dynamic load of action and page configurations allows to quickly add new pages to a portlet. When all methods necessary to deliver the content of a new page are available within the already existing Request Handlers, it is enough to add proper lines to the respective configuration files and the required actions are performed on the services, and the new page is generated on the basis of the responses returned by Request Handlers. There is no need to write any lines of Java code for such a page to be generated.

Content Generators are responsible for the preparation of the content returned by a portlet filling in a portal channel. They transform the data gathered by Request Handlers into HTML pages; in the PROGRESS HPC Portal we use the XSLT technology to build portlet contents. Each Content Generator is then utilized by exactly one Provider. Providers, which are usually portlets, can take a form of any class used to generate the content of a channel within a web portal for example the JSR 168 *Portlet* or the Sun's *ProfileProviderAdapter*, or even a generic *HttpServlet*. Providers are the top layer classes, which allow plugging the built web applications into a portal framework and the portal itself.

The layered architecture of the PROGRESS Portlet Framework enables flexible and easy development of user interfaces to programmatic Web Services. Its

structure is clear to the developer, thus facilitating his/her work. The parts of the already existing code are easily reusable when a newly designed portlet is required to utilize the same functionality of the WS services that were utilized in previously developed portlets. This last feature provides a great opportunity to create highly specialized application portlets. We show this in practice in the next section.

5 Specialized Application Portlets

As it was already mentioned, the PROGRESS HPC Portal features three core portlets: "My computing jobs" for creating, configuring and executing grid jobs, "Applications" for managing the grid-enabled application, and "My data" for managing files stored within the DMS. These core portlets utilize the functionality of the GSP Job Submission and Application Management services and the DMS Data Broker.

The PROGRESS Portlet Framework has been used to rebuild these portlets to enable their reusability within other portals and websites. The other advantage is the opportunity to reuse some parts of the core portlet codes. Thus, the framework is also the environment used to create new portlets specialized to deliver an intuitive user interface for the management of grid jobs utilizing specific grid-enabled applications. Such specialized portlets carry a high value for the users of the grid portals. While the standard "My Computing Jobs" portlet allows to create and submit a grid job built on top of any available application, it is also quite complex for a non-advanced user. Most users require an easy to use and intuitive interface; otherwise, a new tool, even if it delivers a wide range of functionality, is not utilized and respected.

While developing the core portlets we have implemented all necessary classes to work within the four layers of the PROGRESS Portlet Framework. Therefore, when designing new portlets to provide user interfaces to specific grid-enabled applications, we could reuse the already existing code, because these new portlets utilize the same WS servicesas the three core portlets. The specialized application portlets are aimed to utilize the same grid infrastructure as the core portlets and to deliver a highly specialized interface to create, configure and execute grid jobs that use specific grid-enabled applications as opposed to the standard user interface available within the "My computing jobs" portlet, which allows to manage grid jobs based on any available grid application, yet it is not very intuitive for application users.

Thanks to the layered architecture of the PROGRESS Portlet Framework, a developer of a specialized application portlet, for example the developer of the "Gaussian" portlet, which enables the management of grid jobs utilizing the Gaussian chemistry application, could use the already existing Request Handlers to implement most of the actions performed by the new portlet. For example, the portlet can issue a request to list the content of a DMS directory or a request to retrieve the current status of a grid job. Both these requests can be handled by the Request Handlers, and the underlying Web Service Proxies implemented

during the development of the core portlets. This speeds up the job of specialized application portlets developers and enables quick development of new portlets. Furthermore, these portlets can be easily plugged into any portal thanks to both the flexible architecture of the PROGRESS Portlet Framework and the usage of a standardized Web Services interface by the underlying services.

It is important to notice that the descriptions of all jobs created and submitted for execution with the use of the specialized application portlets are stored within the GSP Job Submission Service database. Thus, these jobs can also be viewed and accessed via the core "My Computing Jobs" portlet. However, the specialized application portlets, like the example "Gaussian" portlet, deliver a much more intuitive and non-complex user interface. This is possible thanks to them being familiar with the structure of the corresponding grid-enabled application. This knowledge enables to deliver an easy to use, wizard-like interface which is user-friendly and does not require any grid knowledge from the user.

6 Conclusions

Grid portals play an important role as the deliverers of the grid services and resources. There are many grid portal solutions available; many of them being outstanding work facilitating the process of building new grid portals and enabling grid services and applications to the end users. We have shortly reviewed some of these portals in this paper and have presented our own contribution to the ongoing research and development in the area of grid access environments. With the PROGRESS HPC Portal we fulfill our goal to deliver an extensible and flexible grid access environment, which is also user-friendly and built of highly reusable components. These components are the services features by the respective modules of the PROGRESS grid-portal environment and the portlets developed with the use of the PROGRESS Portlet Framework that utilize these services.

The PROGRESS grid access environment brings many advantages to flexible delivery of grid-enabled applications to the portal users. It features the Application Management Service which enables to easily manage applications available on the grid. It also features the Job Submission and Data Broker services which allow to build and execute grid jobs on the grid. Finally, thanks to the PROGRESS Portlet Framework it provides an opportunity to easily build specialized portlets aiming to deliver the functionality of corresponding grid-enabled applications. These portlets can also be reused in multiple portals, communicating the same grid services. With the GSP and DMS packages, and the portlets developed within the PROGRESS Portlet Framework, the PROGRESS project provides a solution that facilitates the work of the portal operators and the end users, and facilitates the process of enabling applications within grid portals. We think that delivery of highly specialized user interfaces to grid-enabled applications is an important factor of the grid research and development works, and believe our contribution can help grid researchers and practitioners in that field.

References

1. The GridLab project. http://www.gridlab.org/
2. The Crossgrid project. http://www.crossgrid.org/
3. Rychlewski, J., Weglarz, J., Starzak, S., Stroinski, M., Nakonieczny, M.: PIONIER: Polish Optical Internet. Proceedings of ISThmus 2000 Research and Development for the Information Society conference. Poznan Poland (2000) 19-28
4. Kosiedowski, M., Mazurek, C., Stroinski, M.: PROGRESS - Access Environment to Computational Services Performed by Cluster of Sun Systems. In: Bubak, M., Noga, M., Turala, M.: Proceedings of the 2nd Cracow Grid Workshop. Cracow Poland (2002) 45-56
5. PROGRESS HPC Portal. http://progress.psnc.pl/portal/
6. NPACI HotPage Grid Computing Portal. https://hotpage.npaci.edu/
7. Thomas, M., Boisseau, J., Dahan, M., Roberts, E., Seth, A., Urban, T., Walling, D.: Enabling the Rapid Development of Web Service-Based Grid Portals and Applications. Submitted to the Proceedings of the 13th IEEE International Symposium on High Performance Distributed Computing, accessed from http://gridport.net/
8. Thomas, M., Mock, S., Boisseau, J., Dahan, M., Mueller, K., Sutton, D.: The GridPort Toolkit Architecture for Building Grid Portals. Proceedings of the Tenth IEEE International Symposium On High Performance Distributed Computing (2001)
9. Natrajan, A., Nguyen-Tuong, A., Humphrey, M. A., Grimshaw, S.: The Legion Grid Portal. http://legion.virginia.edu/papers.html
10. Novotny, J.: The Grid Portal Development Kit. Concurrency - Practice and Experience (2000) 00:1-7
11. Franzini, B.: EnginFrame, the GridPortal in Engineering and Industry. Portals and Portlets 2003, Edinburgh UK (2003)
12. Cafaro, M.: Web Access to the Grid using the Grid Resource Broker Portal. Portals and Portlets 2003, Edinburgh UK (2003)
13. Grid Engine Portal. http://gridengine.sunsource.net/project/gridengine/
14. Blazewicz, J., Figlerowicz, M., Formanowicz, P., Kasprzak, M., Nowierski, B., Styszynski, R., Szajkowski, L., Widera, P., Wiktorczyk, M.: Assembling SARS-CoV genome - new method based on graph theoretic approach. Submitted to Acta Biochimica Polonica
15. Introducing Gaussian 03. http://www.gaussian.com/g03.htm
16. Pukacki, J.: Resource brokering in the PROGRESS Project. New Network Technologies, Grids and Portals Multi-Conference. Poznan Poland (2003), accessed from http://progress.psnc.pl/
17. Bogdanski, M., Kosiedowski, M., Mazurek, C., Wolniewicz, M.: GRID SERVICE PROVIDER: How to improve flexibility of grid user interfaces. Lecture Notes in Computer Science 2657: Proceedings of the International Conference on Computing Science ICCS 2003. Springer-Verlag. Berlin Heidelberg New York (2003) 255-263
18. Grzybowski, P., Kosiedowski, M., Mazurek, C.: Web Services Communication within the PROGRESS Grid-Portal Environment. Proceedings of the International Conference on Web Services ICWS 2003. Las Vegas USA (2003) 340-345
19. Grzybowski, P., Mazurek, C., Spychala, P., Wolski, M.: Data Management System for grid and portal services. Accessed from http://progress.psnc.pl/