User-Friendly Frameworks for Accessing Computational Resources

Bartek Palak, Paweł Wolniewicz, Marcin Płóciennik,
Michał Owsiak, and Tomasz Żok

Poznań Supercomputing and Networking Center,
ul. Noskowskiego 10, 61-704 Poznań
{bartek,pawel.wolniewicz,marcinp,michalo,tzok}@man.poznan.pl

Abstract. The grid technology, even though it gives great computational power and greatly improves the manner of exploitation of resources, also has its disadvantages. Grid systems, due to their distribution and heterogeneity, are very complex and hard to access and oversee. Thus, one of the research fields designs tools and technologies that give easy, secure and consistent access to grid applications and resources as well as seamless interoperation between various computing environments. The lack of user-friendly tools is especially annoying in cases when users need to access computing resources of different grid infrastructures. Similar problems are experienced by grid application developers who should focus on applications themselves and not on their interoperation with different grid middleware. In this article we describe the concept of an abstract grid model and its implementation in two user-friendly frameworks – Migrating Desktop and g-Eclipse. They both are intuitive graphical environments that provide: easy access to heterogeneous resources and seamless interoperation of underlying middleware solutions. Although the two products provide similar functionalities they are complementary to each other and target different user groups. The method of integration of scientific applications with both frameworks was also presented.

Keywords: grid, graphical user interface, interoperation, gLite, UNICORE.

1 Introduction

To use modern computing applications scientists often need access to computing resources, that are easily available and easily accessible. Owing to several grid initiatives, which provided efficient distributed computing infrastructures over the last decade, a significant improvement of computing power availability could be noticed. In recent years, grids have emerged as wide-scale distributed infrastructures that support the sharing of geographically distributed, heterogeneous computing and storage resources [1].

Many grid projects (such as EGEE, BalticGrid, DEISA, etc.) demonstrated the benefit of a general infrastructure for scientific and commercial applications.

M. Bubak, T. Szepieniec, and K. Wiatr (Eds.): PL-Grid 2011, LNCS 7136, pp. 191–204, 2012.
© Springer-Verlag Berlin Heidelberg 2012

However, the complexity of grid infrastructures is often discouraging to application developers and impedes the use of grid technologies in scientific application domains. Unfortunately, in many cases accessibility of the resources, understood as the ease of use experienced by users, is still not at a satisfactory level. Understanding the behaviour of grid resources is difficult and the learning curve for newcomers is too steep. It often discourages users who are non-experts in the technology and systems being used. Additionally the diversity of environments, based on various concepts and architectures along with their complexity, implies that even deeper knowledge is required to access the resources. Unfortunately, even though there is a trend towards interoperable, service-oriented implementations of grid-services, currently different grid middleware systems are in use. The most popular and widely distributed middleware systems are gLite [2], Globus Toolkit [7] and UNICORE [3,15]. While all these middleware systems offer basic services to interact with the underlying grid infrastructure, each follows a slightly different approach. Incompatibility of middleware implementations is especially annoying in cases when the nature of the problem being solved forces the user to use computing resources of various kinds.

The issues described above clearly show the need for high-level, user-friendly environments that provide: intuitive access to heterogeneous resources and seamless interoperation of underlying middleware solutions. More user-friendly and intuitive tools and user interfaces are needed, in order to make the look-and-feel of grid infrastructures similar to that of existing computer desktop systems which people are already familiar with. In this article we describe two user-friendly client platforms for users, which were developed and deployed within the PL-Grid project – Migrating Desktop [4] and g-Eclipse [5]. They both allow for interoperation with various grid or cloud infrastructures and provide user friendly, transparent access to different middlewares, although their development was driven by different requirements. The two products provide similar functionalities and are complementary with each other because they target different user groups.

2 Related Work

Among the commercial or educational products, which can be used for accessing the grid and interactions between underlying environments, one can distinguish three most popular groups: command line interfaces, portals, and advanced graphical tools. Command line interfaces (CLI) provide only the simplest text interface. They are available in all grid systems (e.g. ARC, UNICORE, gLite) but are used mainly by very experienced users (e.g. system administrators, application developers, etc). This kind of interaction with computing environment requires very deep knowledge of the system being used, so their usage could be very difficult for beginners. Portals and advanced graphical tools are much more popular among the users as a way of accessing grid remote resources. They offer intuitive and easy manner of interaction with user by providing services such as user profile management, information services, remote job submission, job

tracking, file transfer, authentication and authorisation, composition of workflow between tasks etc. Advanced graphical tools have usually more complex, flexible and extensible characteristics compared to portals that are usually limited to support only one type of middleware. Rich clients targeted to Windows, Linux/Unix and MacOS/X are in most cases Java-based, to ensure platform independency.

NGS Job Submission Portal and UCLA Grid Portal [11] are good examples of grid dedicated portals. The NGS Job Submission Portal [10] can be used to access and interact with the HPC and data resources available on the NGS. Functionality available to users include: support for submission and monitoring of job computing as well as data access and transfer around the Compute and Data Grid. The UCLA Grid Portal provides a single web interface to computational clusters of UCLA Grid, and other-including clusters on the TeraGrid. The features offered cover: resource discovery, job handling, file management and results visualisation.

Vine Toolkit [14] has been successfully used as a core web platform for various Science Gateways. Vine Toolkit is a modular, extensible and easy-to-use tool as well as high-level Application Programming Interface for various applications, visualisation components and building blocks. It allows for interoperability between many different HPC and grid technologies on the service layer. Using Vine Toolkit it is possible to build a portal upon the different HPC technologies working together to deliver a complete solution to the users.

P-GRADE [12] offers workflow construction and execution mechanisms for grid including execution and performance visualisation, advanced security features, access to information systems and resource brokers, and multi-grid support. It is de facto "a hybrid" of two different user interfaces: the Workflow Manager runs as a portlet on the P-GRADE Portal server and the Workflow Editor runs as a separate application on the desktop downloaded from the portal. The P-GRADE Portal hides the low level details of grid systems with high-level, user-friendly interfaces that can be easily integrated with various middleware. It offers following services: definition of grid environments, creation and modification of workflow applications, management of grid certificates, controlling and execution of workflow applications on grid resources and monitoring and visualisation of workflows and their component jobs.

Another popular example of advanced graphical tools is UNICORE Rich Client (URC) that provides users with a graphical representation and management of UNICORE controlled resources, services or files. This, Eclipse based, graphical client for UNICORE 6 enables user to build, submit and monitor jobs (which can be scheduled also as workflows that combine several applications), as well as allows them for integrated data and storage management. For better integration of grid applications URC uses the concept of GridBeans – small software packages that provide tailored graphical user interfaces for scientific applications available on the grid.

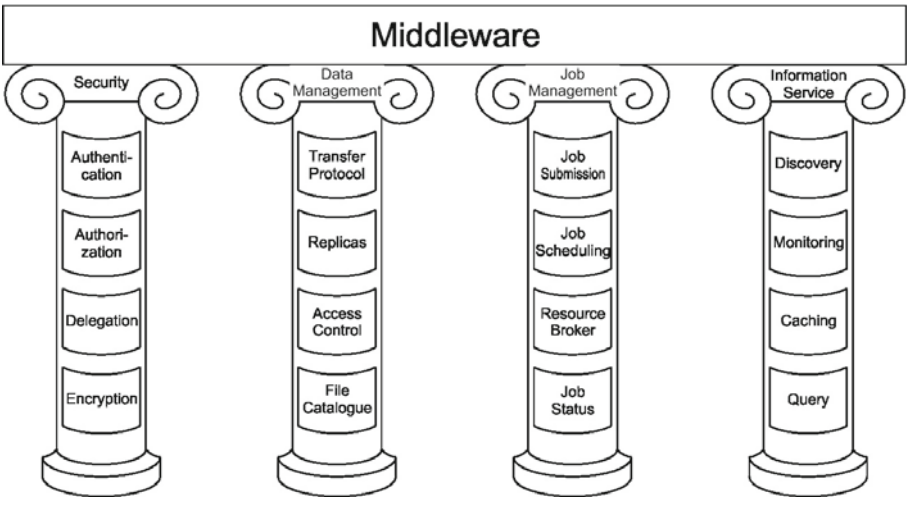

Fig. 1. The four pillars of modern grid middleware

3 Description of the Solution

The computing and storage resources in grid are connected by means of middleware in order to form a virtual distributed computing centre. Apart from the ongoing standardisation efforts, this middleware still follows different concepts and therefore builds grid infrastructures with diverging architectures. Nevertheless one can find a common subset of components which each middleware has to provide (see Fig. 1). The following list gives an overview of these components:

Authentication/Authorization Framework. As grid infrastructures consist of resources that are not parts of local administrative domains, grid middleware has to provide security frameworks to protect against intruders and faked resources. Therefore communication channels may be encrypted, users have to be authenticated and access rights have to be managed.

Data Management. One of the main use cases of a grid infrastructure is distributed data management. Users have to be able to store their data on the infrastructure without worrying about the exact location, backups/replicas and accessibility or reliability of the system. Therefore middleware provides at least some kind of secure transport protocol but also higher level services, for instance file catalogues, which are becoming more and more popular.

Job Management. The most important use case covered by grid middleware is management of the user's jobs computing. In heterogeneous and distributed computing environments this is of course not as easy as running an executable on a local machine. Incoming jobs have to be distributed on the available resources, most often according to specific requirements such as necessary libraries, OS versions or available memory. Furthermore, different jobs may have different priorities. Therefore the job management parts of

any middleware in connection with data management are normally the most challenging ones.

Information Service. Usually an ordinary user needs not be aware of the inner structure of a grid. Nevertheless, at least for operators or administrators, it is essential to have access to the current state of the infrastructure, i.e. which services are currently available and what their status is, or which jobs are scheduled or running. Such information can most often be retrieved by querying an information service or cache. Usually the information services are bound to a Virtual Organisation.

We propose that user-friendly environments act as a bridge between the middleware stacks to ensure seamless access to various underlying middleware in terms of job handling and file management. They are structured according to the above list of components. All of the component modules have the same two-layer structure: the abstraction layer provides a generalised model of grid services and grid resources, the implementation layer implements the functionality of the specific grid middleware by a number of middleware plug-ins. Wherever possible, the implementation layer makes use of common standards, such as of those defined by the OGF [6].

4 Implementation

There are two graphical frameworks available for accessing PL-Grid resources. Both implement the structure described in previous chapter and implement two layer architecture. Although they provide similar features, they are dedicated to different groups of users and can be used in different conditions.

4.1 Migrating Desktop

The Migrating Desktop Platform is based on the client-server paradigm and consists of a Java Web Start application – rich graphical client and server intermediating with grid infrastructures. MD simultaneously hides the complexity of the computing environment behind the advanced graphical user interface, and provides the users with a unified view of the infrastructures used at the interoperation level. It is a powerful and flexible user interface providing transparent user work environment and easy access to the resources. It allows the user to run applications and tools, manage data files, and store personal settings independently of the location or the terminal type.

Open architecture of Migrating Desktop and mechanism of well defined plug-ins and interfaces makes interoperations of various computing architectures available in terms of data handling, job submission and monitoring. The platform offers a framework that can be easily extended on the basis of a set of well-defined plug-ins used to access data, define job parameters, pre-process job parameters, and visualise the job results. Thus the key feature of the Migrating Desktop is

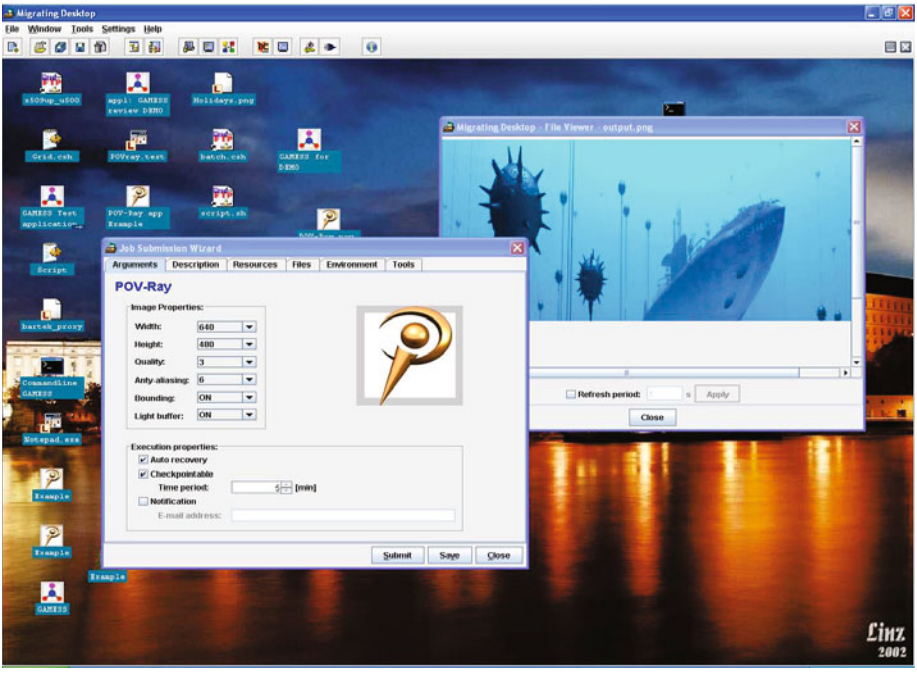

Fig. 2. Screenshot of Migrating Desktop – job defining and visualisation

a possibility of adding various tools and applications, and supporting different visualisation formats easily. Other Migrating Desktop features include:

Single sign-on mechanism. Authentication and authorisation based of X509 certificates schema allows once authenticated users to access resources independently of the underlying infrastructure.

Capability of files exchange among different kinds of storage. Migrating Desktop currently supports several file systems: local, FTP, GridFTP, SRMv2.2, LFN. The framework offers an abstract layer with well-defined interfaces that intermediate while data is exchanged between systems, so the user could have no knowledge of the characteristics of storage parts. Storage, seen as an abstract file systems, is managed by Grid Commander – a graphical tool based on the concept of well-known applications (like e.g. Total Commander) dedicated to file management.

Job submission to various infrastructures using one tool. Migrating Desktop Platform handles the whole job's life-cycle from job defining and submission, up to obtaining results and visualisation of the job outcome. Provided mechanism (based on the idea of plug-ins) allows easy integration of applications within Migrating Desktop and submission of jobs to computing resources of various grid infrastructures.

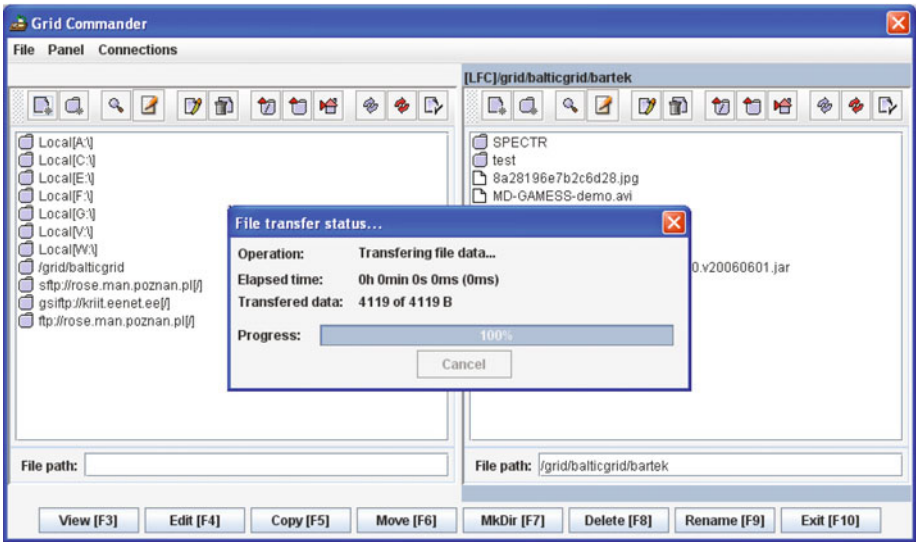

Fig. 3. Migrating Desktop – file transfer between storages of various kind

Simultaneous monitoring of jobs running on different infrastructures. Migrating Desktop features a possibility of obtaining data from heterogeneous mechanisms of job tracking of different middleware, so the status of jobs submitted by the user can be monitored using one dialog.

Migrating Desktop strictly cooperates with the Roaming Access Server (RAS), which intermediates between various grid middleware and applications. The RAS offers a well-defined set of web-services that can be used as an interface for accessing HPC systems and services (based on various technologies) in a common, standardized way. The Roaming Access Server consists of several independent modules responsible for job submission, job monitoring, user profile management, data management, authorisation, and application information management. Using features listed above MD provides users with a tool that makes grid usage much easier and more intuitive, which is essential for encouraging potential beneficiaries to reap profits from using grid infrastructure for compute- and data-intensive applications.

Migrating Desktop was created in the EU CrossGrid Project [8] and developed in the EU Interactive European Grid Project [9], where Migrating Desktop functionality for handling interactive grid applications (i.e. grid applications characterised by the interaction with a person in a processing loop) was implemented. It also proved its usefulness in everyday work of BalticGrid project users communities chosen as a common point for accessing and managing the project applications, tools, resources and services. Currently it is developed within the PL-Grid project.

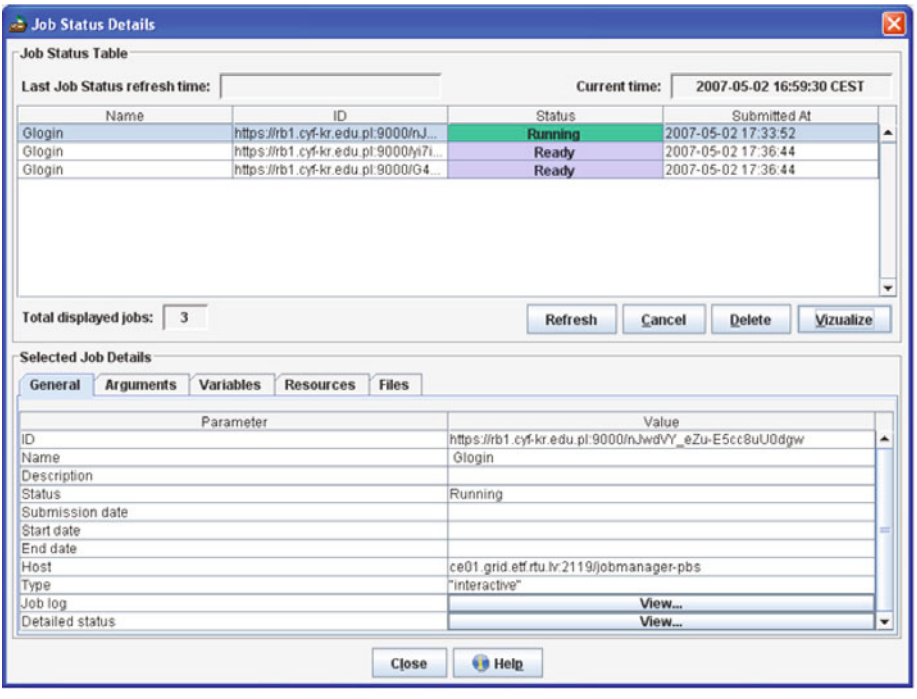

Fig. 4. Migrating Desktop – simultaneous monitoring of jobs running on various grid infrastructures

4.2 g-Eclipse

g-Eclipse is a standalone application – an integrated workbench framework based on well-known Eclipse platform. The g-Eclipse platform is developed as a general framework that can be used by grid users, grid developers and grid operators. The software developed in the g-Eclipse project consists of "core grid plug-ins" for the Eclipse platform. These enable and standardise the access of grid infrastructures from within Eclipse, independent of the grid middleware used. The intuitive interface hides the complexity of using grid services from the end user or application developer, which results in a low entry barrier especially for users and developers new to grid technology. The g-Eclipse platform can be used as a rich client application with user friendly interface to access grid resources, but can also be used as a base for writing customised grid applications using the g-Eclipse model and the g-Eclipse common grid library.

The graphical user interface follows the Eclipse UI guidelines. The intuitive interface hides the complexity of using grid services from the end user or application developer, which results in a low entry barrier especially for users and developers new to grid technology. The graphical front-end of Eclipse is organised with the concepts of views, editors, wizards, preference pages, etc. These components provide the basic functionality to integrate new GUI elements into

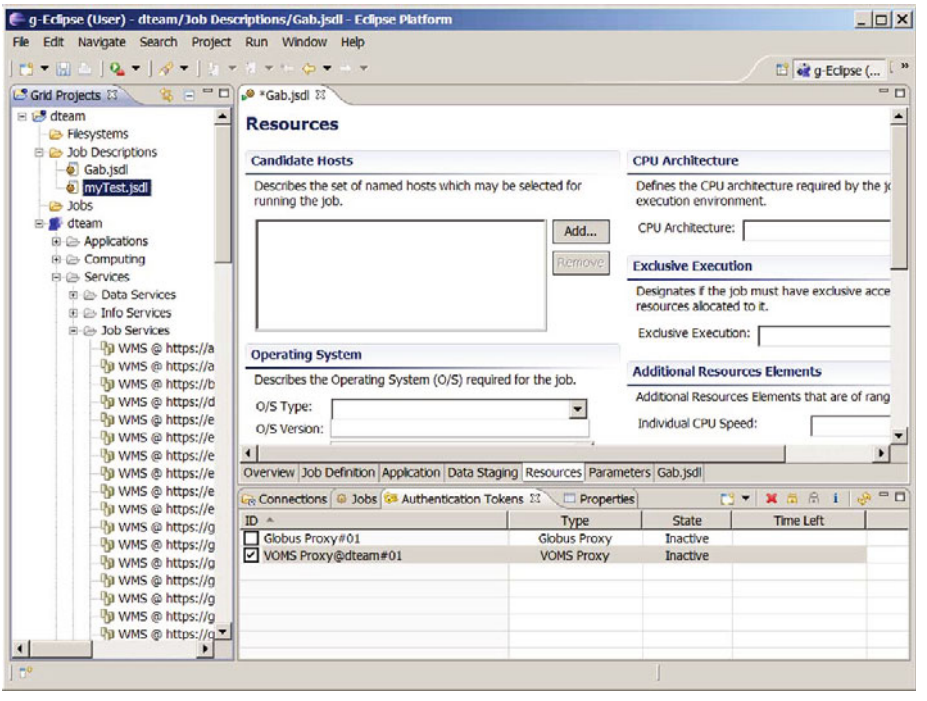

Fig. 5. Example layout of g-Eclipse components

the framework. These basic elements are grouped in so-called perspectives. Perspectives determine the visible actions and views within the workbench, but go well beyond this by providing mechanisms for task oriented interaction with resources in the Eclipse Platform. Users can rearrange their workbench and therefore customise it to their needs and habits with the help of these components. An example screenshot from g-Eclipse is shown in Fig. 5.

The g-Eclipse Framework was designed as an open platform for a wide range of tools and the initial contribution was an integrated development environment for Java. The central point of the Eclipse architecture and framework is its plug-in architecture, a component-based software architecture that leads to a clear and modular design. This is achieved by the underlying OSGi framework that defines the dependencies between different plug-ins, and how and when additional plug-ins are loaded. In addition, the Eclipse framework relies on the mechanisms of extension points and extensions. An extension point is a definition of how to enhance an existing functionality. This way of building software components leads to an extensible architecture with well-defined interfaces.

The g-Eclipse framework is a general grid workbench tools that can be extended for many different grid middleware (such as gLite, UNICORE, Globus Toolkit), and comes with exemplary support for the QCG [16] and GRIA [13] middleware. Furthermore an adapter to Amazon's Web services (i.e. S3 and EC2) is available. As a first outcome of this development, Amazon's Simple Storage

Service can be easily accessed from within g-Eclipse and instances in Amazon's Elastic Compute Cloud can be controlled. With this step, g-Eclipse introduces itself to clouds and is no longer only middleware independent, but also grid independent. The important thing for application developers is that they do not need to care about the specific grid middleware support. The same application can work with different grids or infrastructures without any changes.

5 Exploitation

Migrating Desktop and g-Eclipse can be used as a general purpose tool for accessing grid but their power is also in the ability to adapt to a specific application needs. In the following sections the methods of application integration are explained and exemplary integrations of Gamess and Gaussian applications are presented. GAMESS and Gaussian applications were chosen to prove correctness of integration mechanisms and procedures as both applications are very popular within a quantum chemistry community which is one of the main grid users group.

The Migrating Desktop Platform and g-Eclipse were successfully deployed on the PL-Grid infrastructure. The deployment procedure included evaluation of products features, thorough examination of security vulnerabilities as well as general tests. Now, both tools are used by PL-Grid users as a common point for accessing and managing the project applications, tools, resources and services.

5.1 Integrating Application with Migrating Desktop

Applications whose requirements are accomplished by Migrating Desktop functionality can be integrated without much effort. In other cases application developers have to add an extra functionality to fulfill application's specific needs using concept of plug-ins as integration points between the Migrating Desktop and applications. Depending on the functionality that the developer would like to add to the Migrating Desktop framework, he/she has to choose what kind of plug-in has to be implemented. The developer can choose from the types described in this document:

- Job input plug-in – for defining specific job input parameters;
- Job process plug-in – if any additional activities should be performed before submitting a job (e.g. job parameters have to be optimised before submission);
- File viewer plug-in – if applications produce a non-standard format files that are not handled by the Migrating Desktop and that have to be presented in a user-friendly graphical format;
- Application viewer plug-in – for visualisation of job results that are produced as a set of files or that have to be pre-processed (e.g. any animations, visualisation of meteorological data such as weather forecast map, etc.);
- Tool plug-in – for adding any Java tools to the Migrating Desktop framework.

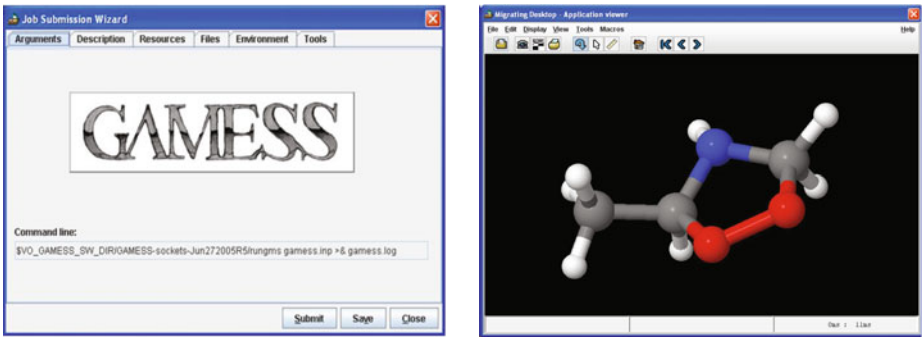

Fig. 6. GAMESS application – job submission plug-in (left) and job visualization plug-in (right)

Integration of GAMESS application within Migrating Desktop Framework includes:

- implementation of specific plug-ins for job defining to make job submission easier and more intuitive for the user (see Fig. 6),
- adaptation of the open-source JMol application as a Migrating Desktop plug-in for visualization of application results (see Fig. 6),
- preparation and publishing a user guide, describing in details the procedure of defining, monitoring and results visualising of GAMESS application.

5.2 Integrating Application with g-Eclipse

g-Eclipse can also be used as a base for users' applications which should access grid resources. The easiest way is to provide an application description which defines application parameters. Such description is then automatically converted to a form of graphical dialog containing fields for entering application parameters. With the help of the g-Eclipse framework an application developer or integrator can prepare applications, both simple or workflow, and share them with the application users group.

A more advanced way of integrating applications can be achieved by providing more advanced plug-ins such as input file editor or result visualisation. g-Eclipse provides visualisation functionality that can be used by application plug-ins. For some applications also submission support plug-in is necessary to prepare a proper job description and input scripts. Plug-ins can also register additional job description format and supporting editors (textual or graphical ones) which can submit and handle specific application input file format. By embedding applications into g-Eclipse developers need not provide client interface or grid resources management. All elements, menus and actions to submit a job, manage a job, update status or transfer files are provided by the g-Eclipse framework. An example integration with pharmaceutical integration is shown in Fig. 7.

The advanced method of integration is to prepare custom applications. By extensively using the Eclipse extension mechanism in combination with

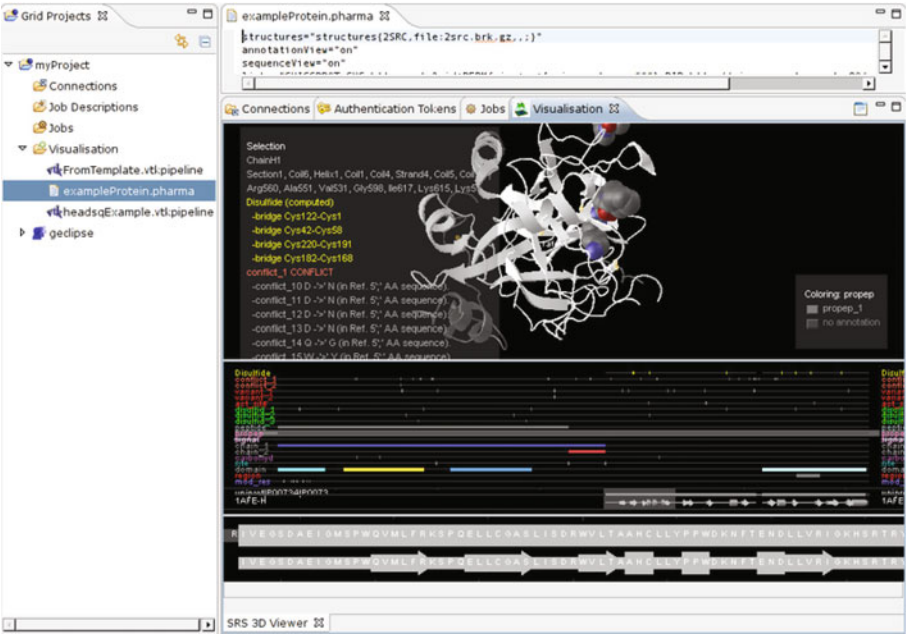

Fig. 7. Pharmaceutical application integrated into g-Eclipse

object-oriented design patterns, the framework can be easily extended by application-specific implementation. The application is not plugged into g-Eclipse, but rather built on top of it. It follows the approach of the Rich Client Platform (RCP) of Eclipse. Dedicated applications can be built by using parts of g-Eclipse as a common library to handle resources and grid access. With such an approach existing applications can be gridified. As the g-Eclipse core model is grid middleware independent, the same application can access resources from various grid middleware. Example integration with JMolEditor is shown in Fig. 8. The quantum chemistry task prepared in JMolEditor can be easily submitted to PL-Grid resources for Gaussian computation.

6 Summary

The Migrating Desktop Platform and g-Eclipse Platform interoperation mechanisms described above provide the user with unified and intuitive access to different kinds of computing infrastructures. This feature allows users to forget about the complexity of underlying environments and focus on their research without the need of understanding grid technologies. User-friendly and intuitive frameworks such as g-Eclipse and Migrating Desktop make the look-and-feel of grid infrastructures similar to that of existing computer desktop systems

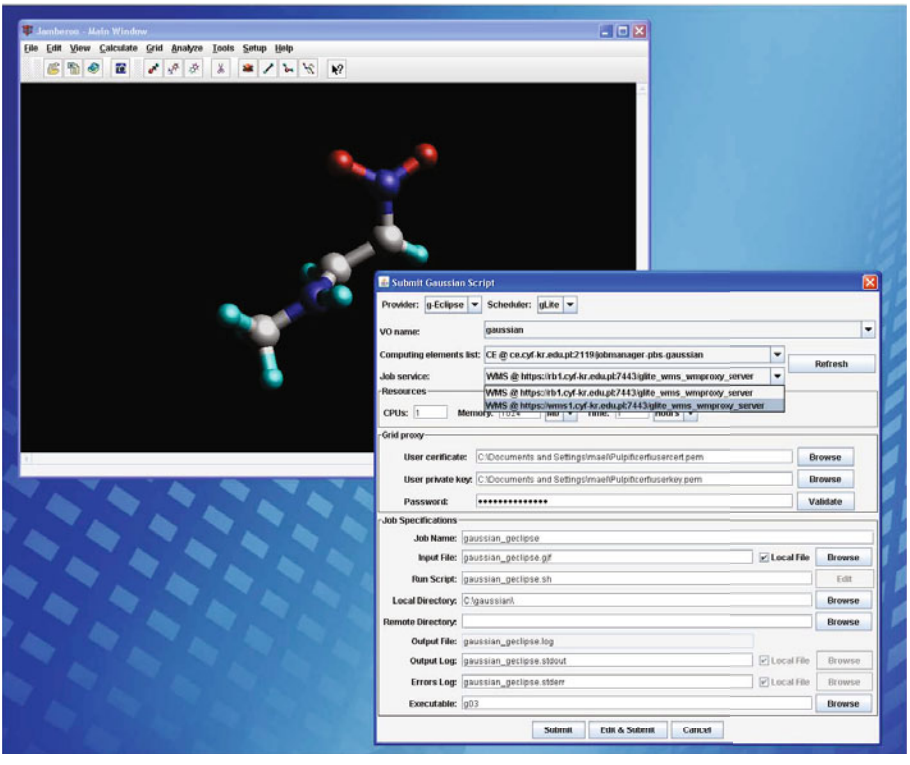

Fig. 8. JMolEditor enhanced with grid capabilities

which people are already familiar with. The ease of use of the resources independently on their type in an obvious way could attract new users to benefit from the newest computing technologies.

Diversity and complexity of computing environments very often discourages users who could benefit from usage of computing resources. Intuitive interface, open architecture and possibility of interoperation between various middlewares makes the described frameworks valuable tools, that can be used both by inexperienced users or skilled application developers, supporting their work with various grid infrastructures. The Migrating Desktop and g-Eclipse were successfully deployed in the PL-Grid project. The frameworks are not dedicated to any specific user community, and can be used for different user needs, different infrastructures, Virtual Organisations and applications. Example integration was shown for fusion, chemical and pharmaceutical applications. g-Eclipse and Migrating Desktop are open source projects with user communities gathered around it, which allows for sustainable development.

References

1. Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: Jin, H., Reed, D., Jiang, W. (eds.) NPC 2005. LNCS, vol. 3779, pp. 2–13. Springer, Heidelberg (2005)
2. Laure, E., et al.: Programming the Grid with gLite. Computational Methods in Science and Technology 12(1), 33–45 (2006)
3. Romberg, M.: The UNICORE Architecture: Seamless Access to Distributed Resources. In: Proceedings of the 8th IEEE International Symposium on High Performace Distributed Computing (HPDC), pp. 287–293 (1999)
4. Kupczyk, M., Lichwała, R., Meyer, N., Palak, B., Płóciennik, M., Stroiński, M., Wolniewicz, P.: The Migrating Desktop as a GUI Framework for the "Applications on Demand" Concept. In: Bubak, M., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2004, Part I. LNCS, vol. 3036, pp. 91–98. Springer, Heidelberg (2004)
5. Kornmayer, H., Stümpert, M., Gjermundrød, H., Wolniewicz, P.: g-Eclipse – A Contextualised Framework for Grid Users, Grid Resource Providers and Grid Application Developers. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2008, Part III. LNCS, vol. 5103, pp. 399–408. Springer, Heidelberg (2008)
6. Open Grid Forum, `http://www.gridforum.org/`
7. Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: Jin, H., Reed, D., Jiang, W. (eds.) NPC 2005. LNCS, vol. 3779, pp. 2–13. Springer, Heidelberg (2005)
8. Crossgrid project, `http://www.eu-crossgrid.org/`
9. IntEuGrid project, `http://www.interactive-grid.eu/`
10. UK National Grid Service Job Portal, `https://portal.ngs.ac.uk/`
11. UCLA Grid Portal, `https://grid.ucla.edu`
12. Kacsuk, P., Sipos, G.: Multi-Grid, Multi-User Workflows in the P-GRADE Portal. Journal of Grid Computing 3(3-4), 221–238 (2005)
13. Surridge, M., Taylor, S., De Roure, D., Zaluska, E.: Experiences with GRIA – Industrial Applications on a Web Services Grid. In: Proceedings of the First International Conference on e-Science and Grid Computing, pp. 98–105 (2005)
14. Dziubecki, P., Grabowski, P., Krysiński, M., Kuczyński, T., Kurowski, K., Piontek, T., Szejnfeld, D.: Online Web-based Science Gateway for Nanotechnology Research. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 205–216. Springer, Heidelberg (2012)
15. Benedyczak, K., Stolarek, M., Rowicki, R., Kluszczyński, R., Borcz, M., Marczak, G., Filocha, M., Bała, P.: Seamless Access to the PL-Grid e-Infrastructure Using UNICORE Middleware. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 56–72. Springer, Heidelberg (2012)
16. Bosak, B., Komasa, J., Kopta, P., Kurowski, K., Mamoński, M., Piontek, T.: New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 40–55. Springer, Heidelberg (2012)