International Workshop on Science Gateways

Consorzio COMETA

IWSG 2010

International Workshop on Science Gateways

Catania, Italy
20-21 September 2010

PROCEEDINGS

Edited by
Consorzio COMETA

Proceedings of the

International Workshop on Science Gateways (IWSG10)

Catania, Italy, 20-21 September 2010

Editors:

Prof. Roberto Barbera

Dr. Giuseppe Andronico

Dr. Giuseppe La Rocca

Consorzio COMETA

Via Santa Sofia, 64
95100 Catania - Italy
www.consorzio-cometa.it

Istituto Nazionale di Fisica Nucleare
Sez. di Catania
Via Santa Sofia, 64
95123 Catania – Italy
www.ct.infn.it

Property of :

Consorzio COMETA
Via Santa Sofia, 64
I-95123 Catania-Italy
Year 2010

http://www.consorzio-cometa.it

http://agenda.ct.infn.it/conferenceDisplay.py?confId=347

Cover art production by Rita Ricceri

ISBN: 978-88-95892-03-0

Programme Committee

(In Last Name Alphabetic Order)

Giuseppe Andronico
National Institute of Nuclear Physics, Italy

Roberto Barbera
Department of Physics and Astronomy of the University of Catania and National Institute of Nuclear Physics, Italy

Sandra Gesing
University of Tübingen, Germany

Luciano Milanesi
Institute of Biomedical Technologies, Milan, Italy

Steffen Moeller
University of Lubeck, Germany

Lars Packschies
University of Cologne, Germany

Miklos Kozlovszky
MTA SZTAKI, Hungary

Ivan Merelli
Institute for Biomedical Technologies, National Research Council, Italy

Organizing Committee

(In Last Name Alphabetic Order)

Giuseppe Andronico
National Institute of Nuclear Physics, Italy
Roberto Barbera
Department of Physics and Astronomy of the University of Catania and National Institute of Nuclear Physics, Italy - Chair
Giuseppe La Rocca
National Institute of Nuclear Physics, Italy

Programme

Monday 20 September 2010

Conveners: Dr. Mattiew Woitsazkec and Prof. Antonio Laganà

10:00 Registration and Welcome at IWSG2010
10:30 Woitaszek M., *The TeraGrid Science Gateways Program*
11:30 Coffee break
12:00 Kozlovszky M., *Converting P-GRADE Grid Portal into E-Science*
12:30 Bonaccorso F., *The Virtual Control Room: an advanced tool to build Scientific Gateways and Support Virtual Research Communities*

13:00 Lunch break and free time
16:00 Laganà A., *Towards a Molecular and Materials e-Science Environment*
17:00 Venuti N., *A Service-Oriented Interface to the iRODS Data Grid*
17:30 Yudin Y., *An Efficient Workflow System in Real HPC organization*
18:00 Gesing S., *Workflow Interoperability in a Grid Portal for Molecular Simulations*

Tuesday 21 September 2010

Conveners: Dr. Denis Caromel and Dr. David Manset

09:00 Caromel D., *Scientific Grid and Cloud Portal with ProActive Parallel Suite*
10:00 Szejnfeld D., *Nano-Science Gateway development with Vine Toolkit and Adobe Flex*
10:30 Wewior M., *The MoSGrid Gaussian Portlet - Technologies for the Implementation of Portlets for Molecular Simulations*
11:00 Orro A., *A web portal for management of biological data and applications*
11:30 Coffee break
12:00 Rotondo R., *A Grid Portal as an e-Collaboration environment powered by Liferay and EnginFrame*
12:30 La Rocca G., *A "lightweight" Crypto Library for supporting a new Advanced Grid Authentication Process with Smart Card*

13:00 Lunch break and free time (Sala Ciclopi)
16:00 Manset D., *neuGRID, A Grid-based Neuroscience Gateway*
17:00 Dooley R., *Recipes for Success in New Science Gateway Development*
17:30 Krüger J., *Workflows and Analysis Approaches for Molecular Dynamics Simulations*
17:45 Pierattini S., *FARO - The Web portal to access ENEA-GRID Computational Infrastructure*
18:00 IWSG 2010: Summary and Conclusions

TABLE OF CONTENTS

Converting P-GRADE Grid Portal into E-Science Gateways

A. Balasko[1], M. Kozlovszky[1], A. Schnautigel[1], K. Karóckai[1], I. Márton[1],T. Strodl[2], P. Kacsuk[1]
[1]*MTA SZTAKI,Budapest,Hungary,{m.kozlovszky,balasko,karoczka,imarton,kacsuk}@sztaki.hu*
[2]*Vienna University of Technology, Vienna, Austria, e9625482@student.tuwien.ac.at*

***Abstract*—Nowadays there is an increasing need to facilitate the knowledge and research tool sharing and realize effective high-level collaboration among members of the same Virtual Organization. E-Science Gateways are the primary solutions dedicated to support such needs. With E-Science Gateways researchers can use grid infrastructure to run shared, well-tested applications customized to their own research field. In this paper the development lifecycle of an e-Science Gateway is described, role definitions of a generic gridification process is provided and the general steps of the application porting process are also identified. In the second part of the paper a developed external module of P-Grade Grid Portal called Application Specific Module is introduced, which extends the portal's functionality set. This Application Specific Module allows developers to convert a generic P-GRADE grid portal into a domain specific e-Science Gateway. At the end of the paper two case studies are detailed, to show the development and usage possibilities of the Application Specific Module.**

***Index Terms*—Grid computing, Grid portal, P-GRADE grid Portal, E-Science Gateway.**

I. Introduction

IN service grids users and resources are grouped into community sets. Community management (membership and resource management) and the cooperation among group members are key peculiarities of such research communities. In service grids the group that shares the same computing resources is called a Virtual Organization (VO). Virtual Organizations provide a wide range of hardware and software resources for their research group members. VOs generally enable access to their resources through well defined gateways, which can be web based (like portals) or user interface machines (UIs), both solutions requires in-depth grid knowledge from the end-users.

Nowadays there is an increasing need to facilitate the knowledge and research tool sharing and high level collaboration among research group members. However several generic grid-portals are available world-wide, but they provide functionalities for advanced grid-users, who are interested in grid-systems, and have knowledge in it, but not for general researchers, who are experts in their research domain but usually they would like to use grids, not develop or port applications to distributed system, and they have very limited knowledge in grid technologies. E-Science Gateways are the primary solutions dedicated to bridge such knowledge gaps. With E-Science Gateways non-grid-aware users can use grid infrastructure to run shared, well-tested applications customized for their own research field [1]. Generally these solutions host a set of research specific applications developed by (and for) the community, and offer services through a unified user interface usually through a web-portal. Such an E-Science Gateway portal has to fulfill the following requirements: it has to comply with the specific demands, it needs to support data sharing and multi-user data management, it needs to hide (completely) the complexity of the grid infrastructure. As the end-users probably don't have knowledge about grids, and they focus on their own research area, the creation of new domain specific applications, or the usage of existing ones must be supported within their research area domain.

II. Role definition of e-Science Gateways

We have identified work phases and various development roles related to converting a generic grid portal into an E-Science Gateway. We have defined the following three well-separated entities playing key roles in the development and usage of an E-Science Gateway:

- Portal Administrator, who installs, manages and maintains the web based VO interface: the generic grid portal. A Grid Portal Administrator has knowledge about the grid portal server and about grid technology (User Interface installation, grid resource setup, etc.).
- Grid Application Developer, who ports (gridifies) the application(s) onto grid infrastructure. A Grid application Developer tests the application(s) and optimizes it to be processed on the grid with the highest performance. The Grid Application Developer can use command line or web based code compilation solutions to create the application binaries. He needs to know the foundations of distributed computing paradigms, and howto use grid technologies (submit jobs, queries the information system, manage data on storage elements) and he needs to have research domain specific knowledge too. He develops workflow structure on the grid portal and can share the workflow with the other members of the grid developer/user

communities.

- E-Science Gateway Developer, who receives the gridified application(s) from Grid Application Developers and creates an E-Science Gateway from the gridified application, where the possible users can parameterize and run the successfully gridified application on the grid-infrastructure. He creates an easy-to-use community specific web based user interface, and hides all the grid aspects of the ported application. He is in possession of the knowledge developing in Java and Java Server Pages (JSP) and using the services and JSP tag libraries provided by Gridsphere framework.

Both type of developers should have a valid grid-certificate and access to the Virtual Organization's infrastructure. The developers need to work in close collaboration during development to share the important part of the research domain specific knowledge.

III. GRID PORTALS

1. P-GRADE Grid Portal [2-4]: The P-GRADE Grid Portal (Parallel Grid Run-time and Application Development Environment) is an open source, service-rich, workflow-oriented graphical grid front-end. It supports workflows composed of sequential jobs, parallel jobs and application services. The P-GRADE grid Portal hides the complexity of the grids through its high-level graphical web interface, and it can be used to develop, execute and monitor workflow applications on grid systems built with Globus [6], EGEE [7] (LCG or gLite [8]) and ARC [9] middleware technologies. P-GRADE grid Portal installations typically provide the user access to several grids, using a single login. Workflows and workflow based parameter studies defined in the P-GRADE grid Portal are portable between grid platforms without learning new systems or re-engineering program code. More than 14 large-scale P-GRADE grid Portals are operating and serving the user communities of international multi-institutional grids and grid based virtual organizations in Europe and in the U.S.
2. UCLA Grid Portal [10]: as an educational grid-portal provides a single web interface to those computational clusters that have joined the UCLA Grid (including clusters on the TeraGrid).
3. EnginFrame [11]: Grid and Cloud Portal delivers access to Applications, Data, and the HPC compute farm (grid, cluster, or cloud) through a standard web browser, developed by NICE.

IV. E-SCIENCE GATEWAY EXAMPLES AROUND THE WORLD

A. P-GRADE grid Portal based E-Science Gateways

Many scientific and non-scientific gateways were developed in several domains. Each gateway exploits the possibilities of P-GRADE grid Portal via a simplified and specialized web-interface to meet the end-users' requirements. In all cases the end-users don't need to have any kind of knowledge about the underlying grid infrastructure.

- P-GRADE grid Portal based E-Science Gateways
  - E-Marketplace Model Integrated with Logistics (EMMIL) is a three-sided e-commerce model that integrates buyers, sellers and logistics service providers, who all participate in the same negotiation process [12].
  - Seismology E-Science Gateway
  - Simulation E-Science Gateway

Seismology and Simulation E-Science Gateways are described in more detail in Section 7, where case studies show how the Application Specific Module was used.

B. LEAD portal

Linked Environments for Atmospheric Discovery (LEAD)[14] makes meteorological data, forecast models analysis and visualization tools available to end-users. The LEAD Portal's "workflow" tool links data management, assimilation, forecasting, and verification applications into a single experiment.

C. RENCI Science Portal

The RENCI Science Portal [15] is a Teragrid Gateway and is available for all non-commercial academic uses and teaching efforts. The system enables large scale computational science by running compute jobs on TeraGrid and RENCI resources, accessible via standard web service. The RENCI Science Desktop is a Java Swing application deployed using Java Web Start (JWS) and uses RENCI's hosted standard web services (Axis2).

D. LUNARC Application Portal

The Lunarc Application Portal [16] provides easy access grid resources for commonly available applications, such as MATLAB, Python, etc. The portal provides easy to use forms for each supported application where single and multiple jobs can created and submitted to the grid and functions for controlling and monitoring jobs are also available.

V. E-SCIENCE GATEWAY DEVELOPMENT LIFECYCLE

In case of P-GRADE Portal a theoretical E-Science Gateway development lifecycle was defined with the following main steps.

1. Portal Administrator installs the P-GRADE grid Portal 2.7+ core server equipped with the Application Specific Module on a User Interface machine, which has been already pre-configured and connected to a specific Virtual Organization.
2. Grid Application Developer creates the gridified version of the application, compiles the source code and sets up the running environment of the application. He manages and transfers the input and intermediate data sets onto the storage elements, if required. He develops the workflow of the application and tests/evaluates the running of the

application, he also optimizes the fine tune of the application to achieve the best performance possible on the distributed infrastructure. Then he provides the finished workflow and the received community requirements (in/output/visualization parameters) to the E-Science Gateway Developer.

3. E-Science Gateway Developer identifies the necessary visual elements (data tables, buttons, menu sets), descriptor of services and portlets specifically for the application. Then he develops and integrates the information into web-portlets according to the requirements of the community.

4. Grid Application Developer publishes the application using a pre-developed portlet of the Application Specific Module.

5. From that point, the finished application specific web interface of the E-Science gateway is available for the end-users of the research community, and the end-users are able to parameterize and run the research domain specific application without any grid knowledge with their web browsers.

VI. Application Specific Module

In the P-GRADE grid Portal a special portal extension enables the development of new E-Science gateways. This special extension called Application Specific Module, which provides an easy-to-use solution to convert the generic P-GRADE grid Portal instance into research domain specific E-Science gateway. In this part of the paper we give a detailed description about the Application Specific Module, and describe how the module can be used by developers.

A. Structure of the Application Specific Module

Application Specific Module consists of two main components; script-layer is used for installing different parts of the module (data tables, services, portlets), and the Java-layer is used as API during the development of the web-interface (see Figure 1).

Basically the scripts are used for installing the Application Specific Module to an already pre-installed P-GRADE grid Portal (2.7+). During the installation a special portlet (called Publish portlet) is installed into the portlet repository of the P-GRADE grid portal instance. This portlet is a handy tool of the grid application developer, who uses it to open up and publish the newly developed application for the whole portal user community. Further more the data table management of the application specific future portlets, and the registration of services and portlets are also maintained by these scripts. The most important part of Application Specific Module is the Java-layer, namely the Grid Portal Developer can use this as an API to develop the specific web-interface for the application. As each object of two specified classes (APP_SPECInstance, APP_SPECUserPreferences) will be stored in an internal consistent Hibernate database. E-Science Gateway Developer must inherit a new class from these, specified for the application, and use that ones during the development of the required functionalities. With this method Grid Application Developer can also develop new functionalities without any modification of the original code.

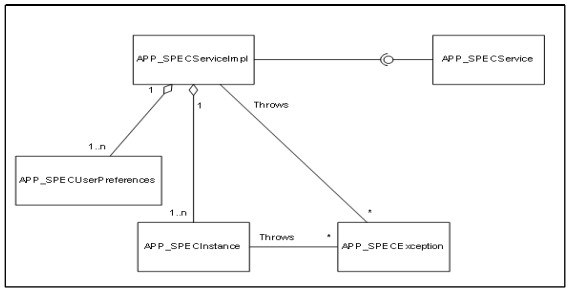


Figure 1. Application Specific Module internal structure

B. Provided interface

The Application Specific Module (java part) provides a simplified API for all the P-GRADE grid portal services, therefore the E-Science Gateway Developer does not have to know the inner structure of the whole P-GRADE grid Portal (e.g. java classes or objects). The API covers the whole life-cycle from the publishing till the downloading phase of the results. Using this API, e-Science gateway developers can reach the list of the published applications of all the Grid Application Developers. A new application can be easily created from the published ones independently from the user-space. With that solution the users can parameterize the generated application and manage the application starting, status checking and application result handling (downloading) easily. In certain cases the basic functionalities should be extended with specific ones (for example managing remote-files through a self-developed portlet such cases are also supported by the API.

C. Usage

Application Specific Module supports two scenarios. In the first scenario (see Figure 2), the user creates a new application (called instance) from an application that is published by the Grid Application Developer. Namely published applications for specified users (Grid Application Developers) must be shown to end-users who must be able to create a new one from a selected application. In another point of view it means that an object for this new application will be created and stored in the Hibernate database. Then he/she can create this application in the file-system by copying and modifying the published application. Later on the users will parameterize and manage this newly generated application instead of the published one.

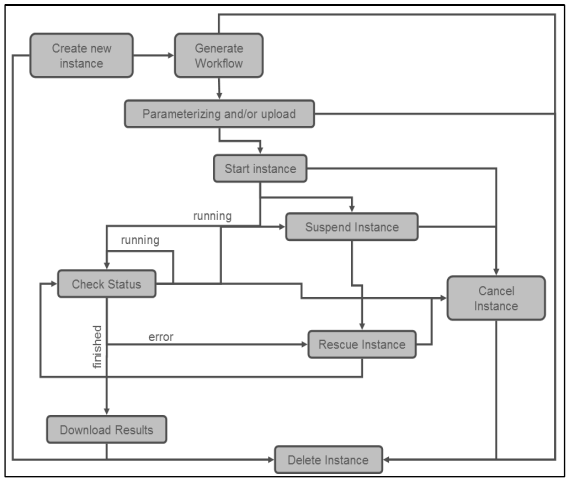


Figure 2. Usage scenario 1

In the second scenario (see Figure 3), users would like to parameterize or manage a previously generated application. Namely users must be able to load an existing application, and generate a workflow for it, parameterize or manage it depending on which step has been done previously.

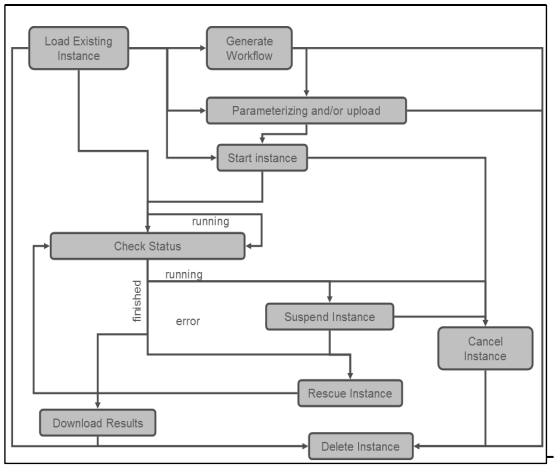


Figure 3. Usage scenario 2

VII. CASE STUDIES

In this section 3 different e-Science Gateways are introduced, that are developed using the Application Specific Module. to confirm the power of the functionalities of the API. In all Gateways' case end-users interactions (e.g. creating new instance, parameterization, execution) can be covered by the usage scenarios detailed in the previous section.

A. Seismology E-Science Gateway

The Seismology E-Science Gateway operates within the Seismology VO of the SEE-GRID-SCI infrastructure, provides unified GUI of different seismology applications (such as NMMC3D, SRA and FPS) and provides end-user access indirectly to the South-East European seismic databases.

In this paper only the Numerical Modeling of Mantle Convection (NMMC3D) [17] application will be described more detailed, which is calculating mantle convection models in 3D Cartesian domain. The main goal of this application is to study the structure and the surface manifestation (topographic and geoid anomalies) of the mantle plumes. The application is focusing on modeling and numerical analysis of mantle convection and solves the equations of thermal convection with a partly finite difference, partly spectral scheme. With the help of the automatic plume detection algorithm, it is possible to monitor the characteristics of individual plumes even in chaotic convection system, in large provinces, at high resolution, but this needs large computational capacity. Parallel execution of the application reduces the running time significantly. The application was ported in cooperation by the Geodetic and Geophysical Research Institute of the Hungarian Academy of Sciences (GGRI) and MTA SZTAKI, and the GUI modules were developed and integrated into the P-GRADE Portal based Seismology E-Science Gateway.

During the development of the NMMC3D workflow the -Parameter Study feature of the P-GRADE Grid Portal was exploited with usage of a special (so called Autogenerator) job. The Autogenerator job generates a set of input files from some pre-adjusted parameters by a Descartes multiplication. The Seismology E-Science Gateway was created using the Application Specific Module of P-GRADE grid Portal 2.7. Due to the strict grid security constraints, end users should posses valid certificate to utilize the E-Science Gateway. After login, the users should create their own workflow based application instances, which are derived from pre-developed and well-tested workflows (the NNMC3D interface is shown in Figure 4).

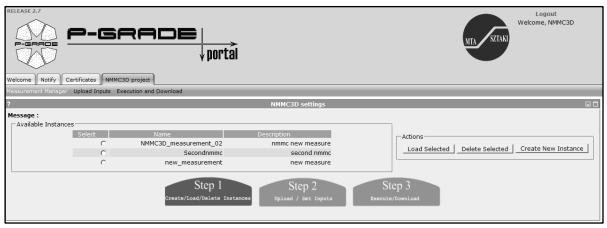


Figure 4. Measurement creation interface

Input parameters of the NMMC3D application such as the "Rayleigh numbers" (Rayleigh number), the "Diss" (viscous dissipation), the Heat (internal heating,) and the number of iteration can be set on the user interface.

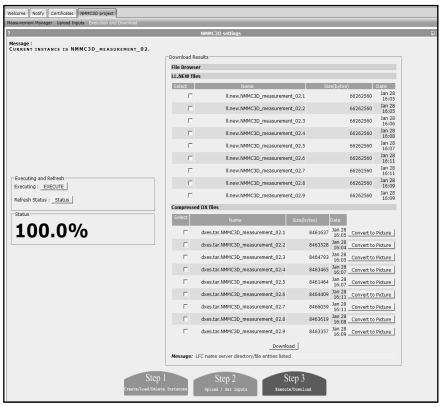


Figure 5. Execution and Download interface

After the setup of all input parameters, users can start the execution of the NMMC3D application, without any knowledge about the totally hidden grid-infrastructure and middleware (shown in Figure 5).

After application processed successfully on the grid infrastructure users can download the result files in a compressed format, and the web interface can convert the generated files to picture and show them directly in the browser (shown in Figure 6).

Behind the scenes new services are developed for managing the file-transfers between the portal machine and the Storage Elements automatically. The basic services of Application Specific module were extended to provide facility for them to visualize the output files stored on the Storage Elements. Therefore new application (called dx) was installed on the portal server to convert the downloaded output files to pictures. When users instruct the web interface to convert the results, the portal downloads the selected file from the Storage Element (about 8 MB), extracts it (it will grow up to 100 MB), converts it to picture using OpenDX [18] and ImageMagick [19], and sends the file to the browser to visualize it.

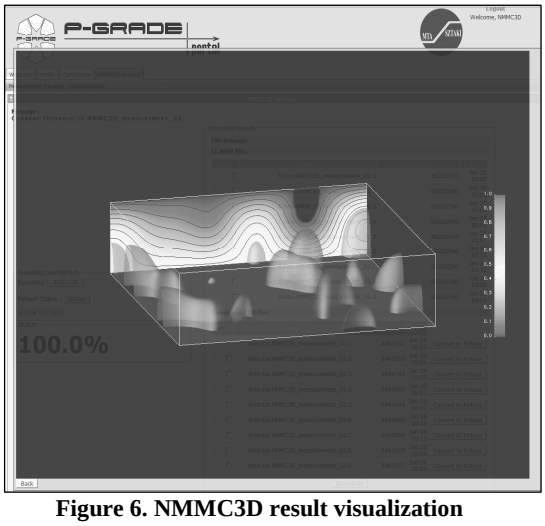


Figure 6. NMMC3D result visualization

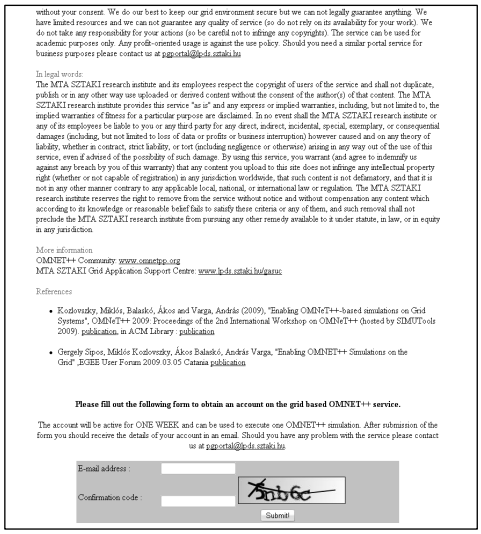
B. Simulation E-Science Gateway

Simulation E-Science Gateway is working with the workflow based version of the OMNeT++ framework. OMNeT++ is an extensible, modular, component-based C++ simulation library and framework that was gridified using P-GRADE grid Portal's Parameter study mechanism.

To enable the execution of the model as a parameter study application, the "Automatic Parameter Input Generator" element of the P-GRADE grid Portal was used. On the web interface, end users, who are not grid professionals, are able to upload their configuration and topology files, setup and submit their model into the grid infrastructure. (In this version, the end user cannot access and change the compiled binary.) The results produced by the submitted simulations are downloadable by normal web browser manually. The developed e-Science Gateway contains four main views (portlets):

On the registration portlet users should register their selves by typing their valid e-mail address and a kaptcha (shown in Figure 7). Then the portal sends automatically an e-mail that includes a one-week-valid personalized gateway access to the user. Following the link, users can automatically login and use the Simulation E-Science Gateway.

On the "Upload inputs" portlet the end user can upload the queuing topology and configuration parameter files overwriting the default files, can select the indexes or interval of indexes to run, and can add time-limitation for the execution. Because the end user can not access and change the compiled application within the workflow, he does not need to have a certificate to execute the workflow on grid infrastructure. An automatic third-party proxy certificate can be used for submission. The Simulation E-Science Gateway is using automatic proxy certificate renewing mechanism.


without your content. We do our best to keep our grid environment secure but we can not legally guarantee anything. We have limited resources and we can not guarantee any quality of service (so do not rely on its availability for your work). We do not take any responsibility for your actions (so be careful not to infringe any copyrights). The service can be used for academic purposes only. Any profit-oriented usage is against the use policy. Should you need a similar portal service for business purposes please contact us at pgp-certal@lpds.sztaki.hu

In legal words:
The MTA SZTAKI research institute and its employees respect the copyright of users of the service and shall not duplicate, publish or in any other way use uploaded or derived content without the consent of the author(s) of that content. The MTA SZTAKI research institute provides this service "as is" and any express or implied warranties, including, but not limited to, the implied warranties of fitness for a particular purpose are disclaimed. In no event shall the MTA SZTAKI research institute or any of its employees be liable to you or any third party for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to loss of data or profits or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this service, even if advised of the possibility of such damage. By using this service, you warrant (and agree to indemnify us against any breach by you of this warranty) that any content you upload to this site does not infringe any intellectual property right (whether or not capable of registration) in any jurisdiction worldwide, that such content is not defamatory, and that it is not in any other manner contrary to any applicable local, national, or international law or regulation. The MTA SZTAKI research institute reserves the right to remove from the service without notice and without compensation any content which according to its knowledge or reasonable belief fails to satisfy these criteria or any of them, and such removal shall not preclude the MTA SZTAKI research institute from pursuing any other remedy available to it under statute, in law, or in equity in any jurisdiction.

More information
OMNeT++ Community: www.omnetpp.org
MTA SZTAKI Grid Application Support Centre: www.lpds.sztaki.hu/gasuc

References

- Kozlovszky, Miklós, Balaskó, Ákos and Varga, András (2009), "Enabling OMNeT++-based simulations on Grid Systems", OMNeT++ 2009: Proceedings of the 2nd International Workshop on OMNeT++ (hosted by SIMUTools 2009). publication, in ACM Library : publication
- Gergely Sipos, Miklós Kozlovszky, Ákos Balaskó, András Varga, "Enabling OMNET++ Simulations on the Grid" ,EGEE User Forum 2009.03.05 Catania publication

Please fill out the following form to obtain an account on the grid based OMNET++ service.

The account will be active for ONE WEEK and can be used to execute one OMNET++ simulation. After submission of the form you should receive the details of your account in an email. Should you have any problem with the service please contact us at pgp-certal@lpds.sztaki.hu

E-mail address :
Confirmation code :
Submit

Figure 7. Simulation E-Science Gateway registration page

On the „Execution and download" portlet the end user can

execute the previously parameterized model by a single button with its parameter field. As the output files are generated in parallel during the execution, users can download them (shown in Figure 8). On the user interface some external tools were also used: to avoid malicious usage of the E-Science Gateway, a google-developed kaptcha was included in the registration form and Google stats were used to check the service traffic and to be able to get some statistic about the interest for the e-Science Gateway.

Figure 8. Execution and Download portlet

VIII. CONCLUSIONS

In this paper, we have described the development lifecycle of the e-Science Gateways, and we have provided role definitions of a generic gridification process. The most important role in our point of view is the Grid-Application Developer, who is doing the application gridification, and the E-Science Gateway Developer, who creates customized web-interface with specialized services around the gridified applications for the research community. We have enumerated some of the available generic grid portal solutions and E-Science Gateways. We have introduced and provided detailed description about an external module called Application Specific Module of the P-GRADE grid Portal, which can convert a generic P-GRADE grid Portal instance into research domain specific E-Science Gateway. In the last part of the paper from the available instances we have spotted case studies, to show the capabilities of this Application Specific Module.

ACKNOWLEDGMENTS

The authors would like to thank the SHIWA project for its financial support. The SHIWA (SHaring Interoperable Workflows for large-scale scientific simulations on Available DCIs), is an Integrated Infrastructure Initiative (I3) project co-funded by the European Commission (under contract number 261585) through the Seventh Framework Programme. The SHIWA project aims to leverage existing workflow solutions and enable cross-workflow and inter-workflow federative exploitation of DCI Resources by applying both a coarse- and fine-grained strategy. Full information is available at http://www.shiwa-workflow.eu. This work makes use of results also produced by the SEE-GRID e-Infrastructure for regional e-Science, a project co-funded by the European Commission (under contract number 211338).

REFERENCES

[1] Dennis Gannon: Programming E-Science Gateways, in M. Danelutto, P. Fragopoulou and V. Getov (editors): Making Grids Work, Springer, 2008. PDF
[2] P-Grade Grid Portal: http://portal.p-grade.hu [acc. 19.08.2010]
[3] Cs. Nemeth, G. Dozsa, R. Lovas, P. Kacsuk. The P-GRADE Grid Portal. ICCSA 2004: International Conference Assisi, Italy, LNCS 3044, pp. 10-19 Globus
[4] Sourceforge.net: P-Grade Grid Portal. http://sourceforge.net/projects/pgportal [acc. 19.08.2010]
[5] P. Kacsuk, Z. Farkas, G. Sipos, G. Hermann, T. Kiss: Supporting Workflow-level PS Applications by the P-GRADE Grid portal, Towards Next Generation Grids Proceedings of the CoreGRID Symposium 2007
[6] The Globus Toolkit. http://www.globus.org/toolkit
[7] EGEE website http://www.eu-egee.org/ [acc. 19.08.2009]
[8] EGEE gLite: Lightweight Middleware for Grid Computing. http://glite.web.cern.ch [acc. 19.08.2009]
[9] ARC middleware: http://www.nordugrid.org/middleware/
[10] UCLA Grid Portal http://grid.ucla.edu/ [acc.25.08.2010]
[11] EngineFrame http://www.nice-italy.com/web/nice/products
[12] P. Kacsuk, G. Herman, A. Balaskó; L.B. Kacsukné; Simulation of the EMMIL e-marketplace model in see-grid using the P-GRADE portal, ESM'2007. The 2007 European simulation and modelling conference. St. Julian's, Malta, 2007. pp. 569-573.
[13] OMNeT++ website : http://www.omnetpp.org/ [acc.25.08.2010]
[14] LEAD Portal https://portal.leadproject.org/gridsphere/gridsphere
[15] Renci Science portal : https://portal.renci.org/portal
[16] LUNARC application portal : http://www.lunarc.lu.se/Software/lap [acc. 19.08.2009]
[17] A. Balasko, M. Kozlovszky, B. Süle : Enabling Numerical Modeling of Mantle Convection on the Grid, MIPRO Conference 2009
[18] OpenDX website : www.opendx.org [acc. 19.08.2009]
[19] ImageMagick website http://www.imagemagick.org

A. BALASKO is working as a Research Fellow at the Laboratory of Parallel and Distributed Systems at the Computer and Automation Research Institute of the Hungarian Academy of Sciences from 2006, where he is a developer team member of the P-GRADE and WS-PGRADE Portals. He received MSc degree at Eötvös Lóránt University of Sciences in 2007 as a Software Engineer Mathematician. Main part of his profile is grid user and research community support as a member of the Grid Application Support Center (GASUC). He has been involved in several European grid projects such as SEE-GRID II, SEE-GRID-SCI, EGEE II and EGEE III.

M. KOZLOVSZKY is working as Senior Research Fellow at the Laboratory of the Parallel and Distributed Systems in the Computer and Automation Research Institute of the Hungarian Academy of Sciences, he is also an Associate Professor at Obuda University (Hungary), leading there the Biotech Group at John von Neumann Faculty of Informatics. He received his PhD from the Budapest University of Technology and Economics (Hungary) in 2009, and his MSc in Computer Science from the University of Szeged (Hungary) in 2001.

The Virtual Control Room: an advanced tool to build Scientific Gateways and Support Virtual Research Communities

Fabio Bonaccorso, Alessio Curri, Daniele Favretto, George Kourousias, Milan Prica, Roberto Pugliese

ELETTRA Sincrotrone Trieste SCpA, Trieste, Italy

{fabio.bonaccorso; alessio.curri; daniele.favretto; george.kourousias; milan.prica; roberto.pugliese}@elettra.trieste.it

***Abstract*—The Virtual Control Room was born as a Grid portal to access and control remote instrumentation. In its current form is an advanced tool to build scientific Gateways and support Virtual Research Communities. Based on powerful open-source web2.0 technologies, the portal provides a complete collaborative environment. This environment enables users access the e-Infrastructure in a secure, remote, and interactive manner. A powerful application development environment and the integration of workflow engines hides the complex details of the e-Infrastructure making it transparent to the users; thus allowing them to concentrate only on their core job: doing science. The portal can be used *"as is"* or fully customized to support the specialized needs of a specific scientific community. An embedded tunneling technology and the application launcher allow a simplified integration of additional resources. After an introduction of the main features and components of the Virtual Control Room, we describe the application manager and explain how the tool can be used to support scientific communities with examples spanning from environmental monitoring, to earthquake research to experimental science in a Synchrotron Radiation Facility.**

***Index Terms*—Scientific Gateway, Grids, Clouds, Scientific Portal, Grid Portal, Remote Instrumentation, Virtual Research Communities**

I. INTRODUCTION

THE Virtual Control Room (VCR) portal has been originally introduced in the GRIDCC project (www.gridcc.org). Following the completion of the GRIDCC project, the development of the VCR has been continued by the Scientific Computing Group at Sincrotrone Trieste S.C.p.A. The VCR has been further developed inside the DORII project (www.dorii.eu) by the same team. The DORII – Deployment of Remote Instrumentation Infrastructure – project aims to deploy an e-Infrastructure where instruments, sensors and scientific equipment integrate with computing and storage resources.

The VCR is the main user interface adopted by the DORII project. DORII focuses on application from three different fields of science (Experimental, Environmental, Earthwuake). The VCR portal has already been successfully applied for applications like the on-line and batch data analysis in experimental science, oceanographic and coastal observation and modeling (using imaging or through Mediterranean Ocean Observing Network), Network Centric Seismic Simulations and Earthquake Early Warning System.

At the time of GRIDCC the focus was on remote operations and control and the name VCR was appropriate. Now the system has evolved considerably and the explicit reference to the term Control even if appropriate seams misleading.
The VCR is an open source Grid portal that allows users to search, discover, browse, control and manage Grid resources (e.g. job and workflow submission, credential management, file transfer) including remote instrumentation. The VCR provides a collaborative environment. It incorporates a set of groupware tools in support of scientific teamwork such as people browser, skype integration, e-logbook and wiki-like help system. VCR assures simple user credential management, Support for MyProxy and VOMS.

As shown in the following picture the VCR integrates with the Grid via a java common library developed within the DORII project.

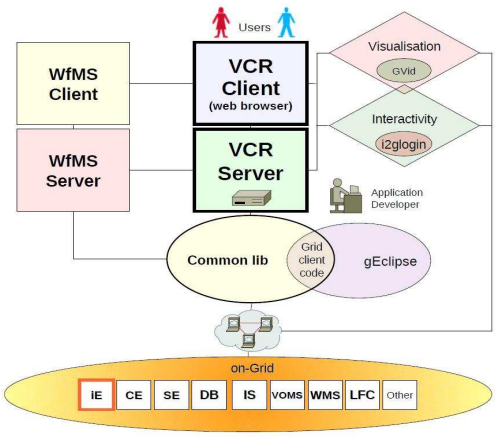


Figure 1. The VCR in the DORII Architecture

The Instrument Element (IE) component provides extensibility by allowing the inclusion of such devices as scientific instruments and sensors in the data elaboration process. In other words, the Instrument Element represents a virtualization of data sources in a Grid environment.
The main goal of the IE middleware is to provide users with a simple way of attaching their scientific instrumentation to gLite-based Grids. The IE offers a common interface for accessing instrumentation. Instruments and sensors are interfaced via Instrument Managers that allow connections to physical devices or more precisely, their control systems.
A wide variety of scientific instruments and sensors, including the SRF beamline detectors, cameras, robots of various types and environment monitoring devices have been successfully attached to the gLite Grid and remotely accessed via the Instrument Element middleware.

II. The Virtual Control Room

The Virtual Control Room (VCR) is an open source web portal that allows simplified access to the gLite Grid resources. The current VCR implementation is based on the latest version of Gridsphere 3.1, Google Web Toolkit (GWT) and the DORII Java Common Library for accessing Grid resources. The GridSphere portal framework provides an open-source portlet based Web portal (www.gridsphere.org). Its portlet API implementation is fully JSR 168 compliant. Google Web Toolkit (GWT) is a development toolkit for building and optimizing complex browser-based applications. DORII Common Library (DORII CL) is a Java client library for accessing gLite Grid resources.
VCR integrates DORII Workflow Management System and an internal Application Manager. It customizes user environment introducing tags for user-application mapping. Registered portal users may access Grid resources from the VCR using their personal certificates, or the portal's robot certificate, the latter an approach proved to be most useful for occasional users of the infrastructure.
Users are linked to the various projects through the VCR tags so each user is presented with the correct set of resources that he/she is entitled to use, and his/her proxy certificate has the correct VOMS attributes set automatically. Integration of the scientific instrumentation is provided through a graphical Instrument Element client. The VCR's tunnelling allows for remote access to the legacy control system and supports interactive application through visualization of the GVid client in a user's browser.
VCR is structured in modules (Core Module, Instruments, Remote desktop, Logbook) so that only the desired tools may be installed. In this section we will describe in more detail some of the features and components of the VCR.

A. *People browser*

Selecting the People tab, a table containing the list of the VCR's registered users may be seen. The table contains also personal information like the user's full name, e-mail address, Skype contact and Skype status for each user that has provided the necessary information during the registration procedure. If you have Skype client installed on your machine, you may initiate a call by clicking on the user's Skype-name in the table.

B. *Resources Browser*

The resources browser contains a list of all available resources, grouped by type. The Resource Browser retrieves the information from the Information System based on the Berkeley Database Information Index (BDII). The Resources Browser supports more than one BDII site but only for redundancy. In addition, it may contain also some static resource entries. Through this component a user can browse and select the resources available in the Infrastructure, namely Instruments Elements, Computing Elements, Storage Elements, Resource Brokers and File Catalogs. Moreover, in the resource browser users will find Portal File System resources. By selecting a resource listed in the Resource browser, the content of the resource is displayed on a specific component.

C. *Storage Portlet*

The Storage portlet allows you to perform traditional file system operations (e.g. browsing, creating and deleting files and directories, uploading files and downloading files and directories) on remote directories that are located on VO's Storage Elements.

The LFC (LCG File Catalog) is a data catalog containing logical to physical file mappings. LFC catalogs are managed by the Storage Element portlet. User space and Shared space on the VCR are also managed by the Storage portlet.

D. *Instrument Control*

Selecting an IE from the Resources Browser, the VCR opens a session and delegates user credentials to the IE. If the operation is successful, the Instrument Manager will appear in the Instrument Control portlet. The lifetime of the user proxy on the IE is less then the original proxy contained in the VCR.

According to his/her privileges based on the VOMS roles, an user may issue commands to the instrument. A special

type of commands that trigger state change upon their execution are called transitions. Transitions are shown separately from other commands. The list of available commands shows only those that may be executed in the current state. The state machine is representing Instrument Manager settings, not the actual instrument state machine, although the two may coincide.

Reading instrument output (variables or attributes) is always permitted. The VCR refreshes the IM variables by polling the IE at fixed time interval. In alternative, JMS monitoring may be activated on single variables by subscribing to them. Graphical clients may monitor both the variables refreshed by polling and those that await JMS events.

E. Computing Element and Resource Brokers

The Computing Element portlet is active when a computing resource (Computing Element or Resource Broker) has been selected from the "Resource Browser". It allows the submission of jobs to the specified resource and monitoring of their status. In the portlet users may see a list of Jobs that have been submitted to the computing resource. Job submission is facilitated by a wizard that appears when you select the "Create new Job" link. Starting from the top of the portlet and going down, job submission form contains respectively the name of the computing resource selected, a status window, the wizard and the list of Jobs (in case the Job is parametric the whole job tree is displayed).

The wizard prompts the users for all the input needed to create a JDL file: job description, type, executable file, output and error files, inbox and outbox files. Moreover, the Computing Element portlet takes advantage of the Jython engine of the VCR and allows the users to optionally select the scripts to be executed before and/or after the job lifetime with the required parameters. Once the the "NEW JOB" form has been filled, in order to launch the job a user should press the "Submit Job" button. JDL file will be generated and then submitted to the Grid.

Form the Job list users may see the status of the job and control the job execution: Users can cancel the job and get the job output once the job is finished.

F. Workflow Management Systems

The VCR integrates the DORII Workflow Management System. The Workflow Manager System sub-tab allows users to select a Workflow Manager and to start the Workflow Manager Editor and Monitor. All the interaction will occur through the Editor and Monitor that in turn communicate with a Workflow Engine.

G. E-Logbook

The logbook module offers a simple forum-like utility for posting logs and holding discussions. It allows users to browse posts, start new discussion threads, reply to the existing ones or edit wiki-type help posts. Visualization is dived into two portlets: Logbook and Help, the latter showing only the Help topics. A third portlet of the module is used for editing keywords, categories visible only to users with administration privileges.

H. Tunnels

The VCR provides a tunneling service, which allows to view legacy instruments that do not present a web interface and are often hidden behind firewalls. Moreover, http servers otherwise unreachable from the outside world may be accessed using the type of tunneling provided by the VCR as well as the video streaming over the http. Tunnel configuration and execution is controlled from the Remote Desktop module. The front end for the tunneling service consists of two portlets, a Wizard and a Viewer. The Tunnel Wizard is available only to portal administrators and it is used to create, modify and destroy tunnels. HTTP, VNC and Generic wizard are available. The Tunnel Viewer shows the list of available tunnels and allows to select and start a tunnel connection to the target service.

III. The Application Manager and Application Development

The Application Manager allows developers to prepare applications by means of XML files, a set of specific components and the Jython scripts.

These XML files are defined by a particular XSD created by the VCR developers which associate to every tag a particular UI graphical element (eg text boxes, free HTML etc) or services (eg Tunnels) All the functionalities available in the VCR can be used programmatically via the internal Jython engine. Thus the Application Manager allows customization without touching the VCR internals making thus the Grid and the e-Infrastructure completely transparent to the end user.

A. Application Development

From the development point of view, an application consists of a folder containing an XML file that describes the structure of the application and a folder called 'scripts' that includes the scripts, namely the application components or steps. We shall use the terms step and script almost interchangeably in the following description.

During the execution the application starts with an initial script and depending on the outcome flows from one script to the other till it exits. Depending on how the application is developed, the user can decide to move back and forth between the different application steps, repeat a step (an auto-repeat feature is available too), cancel the application, skip specific steps or move to a specific step in the application flow.

Normally, at each step a script is executed by the VCR application engine. The application engine is based on a Jython interpreter that can call an available Java library, in particular the DORII Common Library and hence access and use the e-Infrastructure. An XML file that defines an

input form and the associated action describes each application step. The user input collected in the form variables is passed to the Jython scripts or job submission and monitoring component. Steps may be set reloadable to allow a simplified form of monitoring. The user can specify the Jython script to execute before reloading, canceling a step or exiting the application.

B. Application Scripts

Application scripts are structured as a sequence containing a description, an include section (to allow integration of generic html), an arguments section containing input or special fields (generally used to collect user input) and a service section used to activate special services.

Scripts' form arguments named with a specified `<name>` (e.g. `SE_FILE_PATH`) are accessible in the python scripts as variables with the same name. Some variables are automatically set by the VCR. These include `USERNAME`, `PROXYPATH`, `SCRIPTPATH` containing respectively the user name, the path of the user proxy, and the execution path of the script. Each application when executed is run as a separate instance in a separate file space. Through Jython scripts users may program complex applications accessing all the features of the DORII Common Library and by adding 3rd party libraries, the capabilities may be further extended.

C. Using Applications

Users have to select the “Applications” tab and the select the Applications sub-tab. The “Available applications” portlet lists the applications that a user may access based on his/her associated tags. User may start an application by selecting the corresponding “Open” button. The “Active applications portlet” lists the applications that the user has currently opened. A specific application may be viewed by selecting the corresponding “View” button.

The “Application monitor” portlet shows the steps of the selected application. Users move from one step to the other by pressing the navigation buttons (Next or Back). They may exit by pressing the “Clear application” button. The application also exits after the final step has been completed. During the execution of a step, the user can see the script output and errors by pressing respectively the “View script output” and “View script error” button in the 'Application monitor' portlet.

IV. USING THE VIRTUAL CONTROL ROOM TO SUPPORT SCIENTIFIC COMMUNITIES

The VCR has been used as the scientific gateway to all the DORII communities and applications.

The seismic community is quite broad and includes many disciplines: it is interesting to understand how a building behaves during an earthquake as well as to study the nature of the event itself, evaluating the impact on a whole area. Moreover all this knowledge should be organized in a coherent way, in order to share and to improve cooperation between specialists in such different fields. The outcomes of these activities result in smarter design criteria or faster damage assessment procedures. Having this in mind, EUCENTRE proposed two applications, Network-Centric Seismic Simulations (NCSS) and Earthquake Early Warning System (EEWS) that are target of this evaluation phase and focus on the two aspects stated above.

The Network-Centric Seismic Simulations (NCSS) application gives remote access to some basic instrumentation (an actuator, a load cell and some displacement transducers) giving the possibility to perform stress tests on a wall. Everything is integrated in the DORII framework to run a model solving the reverse force-displacement problem (drift control) and estimating the material parameters of the wall under tests. Computations are performed on the CEs and results are shared. The Earthquake Early Warning System (EEWS) application monitors in real time a seismic sensors network, band pass filtering the signals and detecting a sudden increase in the ground velocity. In case more nodes meet the same conditions in a short time interval, then a potential P seismic wave have been caught. The user to analyze in more detail the interesting portion of the seismogram and characterize the P-wave, having a visual feedback and the details of the event, can launch a semi-automatic procedure.

Understanding the environment requires a quite complex setup in terms of sensors and modeling, with physical, chemical, biological and other complex phenomena involved. The common needs of the environmental community are: monitoring and collection of observations at geographically distributed places, consolidation and transmission of data in near to real time, remote processing of data and comparison with current models to improve their predictive value.

The Oceanographic and Coastal Observation and Modelling Mediterranean Ocean Observing Network (OCOM-MOON) application involves a network of oceanographic detectors passively following the sea currents in the Mediterranean basin, providing data that can be used operatively by research institutions to monitor the sea state and to drive numerical models to obtain forecasts. In the Horus-Bench application the oceanographic and coastal observation and modeling is done using imaging. These applications require interactivity, collaboration, access to instruments and to the existing e-Infrastructure. Similar integrated sea monitoring systems, deployed with ad hoc instrumentation, can be adopted in coastal environment to control the behavioral evolution of the main physical processes that characterize this environment, in order to safeguard the coastal settlements. The Ecohydros application has the goal of monitoring of inland waters and reservoirs.

Experimental stations in facilities like Synchrotrons and

Free Electron Lasers produce huge quantities of data. These data need to be analyzed on-line, which requires considerable computing power and often teamwork. The problem is even more difficult considering the increased efficiency of the light sources and detectors. Complex calculations are required to take diffraction images and convert them in a 3D protein structure. Similarly, complex calculations are required to calculate tomography and then perform an analysis of the result. The results of these analyses often need to be visualized by a distributed team and used to modify interactively the data collection strategy. Data from instruments and sensors are saved in distributed repositories, computational models are executed, and finally an interactive data mining process is used to extract useful knowledge. Applications for the SAXS, the SYRMEP beamline, the Tomolab and the XRD beamline of the Synchrotron Radiation Facility ELETTRA have been developed. The following picture describes the SAXS application.

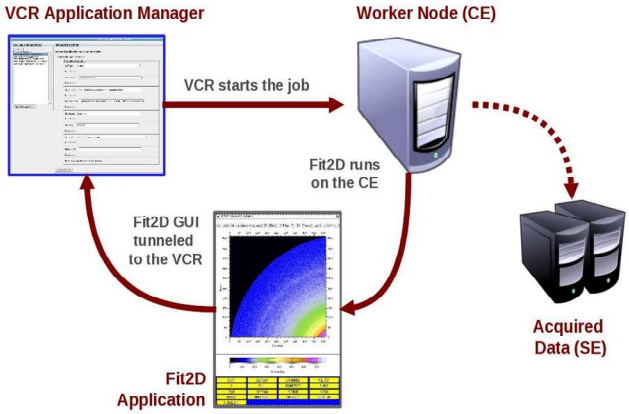


Figure 2. The workflow in the SAXS application

For the SYRMEP beamline and the Tomolab we developed a set of online and offline processing applications. The computation takes place in a specially designed Instrument Element running on a dedicated system based on virtualisation. The users access the application via the same application manager but the application is quite more responsive and interactive. There are various "logistics" in the process that the middleware takes care of. It has been observed that there are quite high latencies during the job submission phase. These latencies for certain applications are not a problem but for other they are. The IE controls a Computing entity, the PSGen application, without accessing a traditional CE. This irregular use of the IE demonstrates a case where it may even compute or more precisely: steer a computation. Applications, can be accessed from the users that have access to portal, taking into account most of the fine Grid-related security control, like Certificates, VOs, and the VCR-side TAGS. The PSGen user can start the application at the beginning of the experiment and receive feedback at any stage; even after login-off and re-login in. The input and output datasets are stored in Storage Elements (SE) that can be browsed from the web portal. In addition to the above the user may start other Applications that make full use of the Grid in standard manner. These may include online and offline, single, parametric, MPI, and interactive and any combination permitted from the gLite, the Common Library and the rest of the technologies developed during the DORII project. Finally the scientists may use features like the logbook for note-taking purposes while using PSGen. The following picture describes the user experience with the VCR with the PSGen online processing application.

Figure 3: The user experience with the new PSGen application

In the *XRD* application the user selects dataset and a set of relevant parameters and each new acquired image is transferred from the detector to the storage and processed. The application implements a simple workflow of two computing steps are executed in sequence: the first is actually a multi-step computing activity, namely the xds application implemented using openmp that takes a dataset of diffraction images (typically 180 if the k-goniometer step is one degree to even thousands if the k-goniometer step is a fraction of a degree) and produces intermediate files and a second step that takes the product of the first one and produce a HKL format structure factor file that can be passed to more advanced steps performed offline. The final goal is to solve the structure, that means, find the 3D structure of the sample where most of the functional properties reside. But, discovering the 3D structure is quite complex. It may involve different techniques that combine different datasets (each one acquired at a specific photon energy) or comparing the experimental data with online databases and similar cases or trial an error iterative processes where researchers use their knowledge and experience to guess the 3D structure an then compare their proposals with the experimental evidence. All these steps are normally performed offline. To support the offline processing users of the beamline the workflow system *taverna*, an open source and domain independent Workflow Management System has been used. Using the VCR a distributed team of researchers can remotely access all the acquired datasets, control the beamline via tunnelling and

execute workflows using collectively Grid resources and the other database resource available online like the protein data bank.

V. CONCLUSIONS

We have described the Virtual Control Room an open-source web portal that can be used to implement customized gateways to support the specific requirements of scientific communities. The tool has been used to implement scientific gateways for many different scientific communities, a large number of applications and a huge user base. Through the VCR the user can interactively access and operate on all the resources available in the e-Infrastructure and execute complex applications and scientific workflows that hide completely the details of the e-Infrastructure thus allowing the users to be distant from the complexities and technicalities of the ICT infrastructure and concentrate on the core of their job: doing science of high standard.

ACKNOWLEDGMENTS

The work has been partially supported by the DORII EU FP7 project under grant agreement RI-213110. The authors would like to thank all the researchers and scientists that allowed the improvement of the product with their continuous feedback and cooperation.

REFERENCES

[1] F. Brun, G. Kourousias, D. Dreossi, L. Mancini, "An improved method for ring artifacts removing in reconstructed tomographic images", IFMBE World Congress on Medical Physics
[2] and Biomedical Engineering, vol. 24, Springer, 2009, pp. 926–929.
[3] I.Foster, C.Kesselman, "The grid: blueprint for a new computing infrastructure", Morgan Kaufmann, 2004.
[4] Laure, E. and Fisher, S. and Frohner, A. and Grandi, C. and Kunszt, P. and Krenek, A. and Mulmo, O. and Pacini, F. and Prelz, F. and White, J. and others, "Programming the grid with glite", Computational Methods in Science and Technology 12 (2006), no. 1, 33–45.
[5] M. Prica, R. Pugliese, A. Del Linz , G. Kourousias, A. Curri, D. Favretto, F. Bonaccorso, "An advanced web portal for accessing grid resources with virtual col laboration features", 5th EGEE User Forum (Uppsala, Sweden), EGEE User Forum, April 2010.
[6] Plociennik et al., "DORII–Deployment of Remote Instrumentation Infrastructure", Relation 10 (2009), no. 1.109, 800.
[7] Prica, M. and Pugliese, R. and Del Linz, A. and Curri, A., "Adapting the instrument element to support a remote instrumentation infrastructure", Remote Instrumentation and Virtual Laboratories: Service Architecture and Networking (2010), 11.
[8] Prica, M. and Pugliese, R. and Scafuri, C. and Cano, L.D. and Asnicar, F. and Curri, A., "Remote operations of an accelerator using the grid", Grid Enabled Remote Instrumentation (2009), 527–536.
[9] R. Pugliese, M. Prica, G. Kourousias, A. Del Linz, A. Curri, "The grid as a software application provider in a synchrotron radiation facility", Remote Instrumentation Services on the eInfrastructure, vol. X, Springer, 2009. "An infrastructure for the integration of geoscience instruments and sensors on the grid", Geophysical Research Abstracts, vol. 11, EGU, 2009.
[10] "Integrating Instruments in the Grid for On-line and Off-line Processing in a Synchrotron Radiation Facility", COMPUTATIONAL METHODS IN SCIENCE AND TECHNOLOGY 15 (2009), no. 1, 21–30.

FABIO BONACCORSO is a computer scientist at Sincrotrone Trieste. His interests are in in Web Technologies, Grid Infrastructures, Remote Instrumentation, user interfaces and Web 2.0 He has participated in the EuroTeV and DORII projects.

ALESSIO CURRI is a technologist at Sincrotrone Trieste. His professional skills include system analysis and design of Grids and distributing computing environments. He has participated in EGEE I&II, GRIDCC, BIOXHIT projects and is currently involved in DORII and the Italian Grid Infrastructure.

DANIELE FAVRETTO is a software engineer at Sincrotrone Trieste. His interests are in Web technologies, Grid Infrastructures, and Remote Instrumentation. He has participated in the DORII project.

GEORGE KOUROUSIAS is a computational mathematician at Sincrotrone Trieste. His background is in Numerical Methods and Artificial Intelligence. His current interests include signal processing, medical imaging, HPC, Grid & Cloud computing.

MILAN PRICA is a senior software engineer working at Sincrotrone Trieste. His background is in Computer Science, Information Retrieval and Search Engine technologies. His interests include Grid Systems, Data Management, Reliable Systems and Web 2.0. Major projects that he has participated in include ADAPT, GRIDCC and DORII.

PROF. ROBERTO PUGLIESE is a research coordinator at Sincrotrone Trieste S.C.p.A. where he is leading the Scientific Computing Team. He is teaching E-Commerce at the University of Udine. His research interests include Web Based Virtual Collaborations and Grid technologies. He was the technical coordinator of GRIDCC and is currently coordinating the applications of DORII.

GridF: Empowering Scientific Calculations on the Grid

C. Manuali[1], A. Laganà[2]
[1]*Department of Mathematics and Informatics, Perugia, Italy, carlo@unipg.it*
[2]*Department of Chemistry, Perugia, Italy, lag@unipg.it*

***Abstract*—Grid empowered calculations are becoming an important tool for scientific advances. The possibility of simplifying and harmonizing the work carried out by computational scientists using a Web Service approach is considered here. To this end, a new Collaborative Grid Framework named GriF has been developed and validated to run on the Grid by considering as a study case quantum reactive scattering codes. Its use as a tool of Science Gateways to facilitate massive calculations also to the end of improving scientific collaboration is discussed. Accordingly, a preliminary study on how to profile the users of Virtual Organizations in order to pave the way to a systematic evaluation of the work carried out in Grid and to fostering its sustainability, is presented.**

***Index Terms*—Grid Frameworks, Quality of Users, Quality Evaluation, Reactive Scattering, Science Gateways, Service Oriented Architectures, Virtual Organizations, Virtual Research Communities, We Services.**

I. INTRODUCTION

THE growing popularity of distributed computing has fostered the formation of Virtual Research Communities (VRC)s. VRCs not only share hardware and software on the Grid computing infrastructure but also develop new approaches to collaborative research relying on the contributions of the users. This is what has occurred within the Enabling Grids for E-sciencE (EGEE) European project [1] and has stimulated the birth of specific scientific organizations committed to the design and implementation of collaborative computational endeavours on distributed platforms [2]. An example of this is the Virtual Organization (VO) COMPCHEM [3] which gathers together several scientists operating in the field of molecular sciences and that we have taken as the reference community for our investigation. Such trend has been consolidated by the recent launch of the European Grid Initiative (EGI) [4]. An important task of COMPCHEM within EGI is, in fact, to support its members and users in building collaborative computational initiatives leveraging on complementarity so as to tackle the study of highly complex systems. This makes Grid empowered calculations a tool of primary importance to the end of fostering scientific advances. For this reason, we have designed a new collaborative Grid Framework based on the Service Oriented Architecture (SOA) and on a Web Service approach [5] that simplifies and harmonizes the work carried out on the Grid platform by geographically dispersed computational scientists.

The new SOA Grid Framework (called GriF) provides the users with an efficient and easy to use workflow that enables the running on the Grid of distributed versions of the scientific codes. GriF has been first developed to run some components of GEMS (a Grid Empowered Molecular Simulator) [6], [7] and in particular the quantum mechanical atom diatom reactive scattering codes like ABC [8]. However, GriF can be easily generalized and incorporated in Science Gateways [9].

Moreover, GriF can make use of the outcomes of the existing Grid sensors to perform quality analyses based on the profiling of the users and can be adapted as well to elaborate the outcomes of new Grid sensors (see for example ref. [10]). This is because GriF puts the users at the center of the action as a point of generation, integration, control and evaluation of the computational activities.

Accordingly, the paper illustrates not only the structure of GriF but also its quality-evaluation functionalities by discussing the profiling of the users of ABC[1]. Therefore, the paper is articulated as follows: in section II the main features of the ABC code relevant to its implementation as a Java Web Service are singled out; in section III the implementation of the collaborative SOA Framework GriF is described; in section IV a first implementation workflow-enabled of the Web Service interface for the ABC program is illustrated; in section V a user profiling classification is discussed. Some conclusions and ideas for future work are outlined in section VII.

II. THE ABC CODE

The ABC quantum reactive scattering program adopts a Time Independent (TI) approach to the calculation of detailed reactive properties. The TI method is based on the integration of the stationary version of the Schrödinger equation for the nuclei at a fixed value of total energy (E). To integrate the stationary Schrödinger equation it is

[1] Fundamental to this work has been the interaction [10] with the research group working on monitoring the Grid and in developing related sensors at the Centro de Supercomputation de Galitia (CESGA, Santiago De Compostela, Spain) [11].

necessary to define a proper continuity coordinate (reaction coordinate) smoothly connecting reactants to products. Then, on a fairly dense grid of points (the interval included between two adjacent point is called sector) taken along the reaction coordinate the eigenvalues and the eigenfunctions (basis functions) of the Hamiltonian of the bound state problem in the remaining coordinates are computed. To this end, the system wavefunction is expanded at each point of the reaction coordinate grid in the basis functions. By averaging over the remaining coordinates the product of the wavefunction times every basis function one obtains a set of coupled equations in the reaction coordinate. The integration of these coupled differential equations in the reaction coordinate allows the determination of the scattering partial (fixed J) S^J matrix from the value of the partial atom diatom wavefunction $\psi^J(R,r,\Theta)$ at the asymptotes (for fixed E and all the admissible values of v, j, v' and j').

In ABC, the hyperspherical Delves coordinates are used [12] whose hyperradius ρ is the reaction coordinate. The propagation of the solution from small values of ρ (closely packed system) to the large ones (the asymptotes) is performed by linking recursively the matrix of the coefficients (g) of the expansion of ψ in the basis functions of the various regions of the reactive process through O, the overlap matrix, and U, the coupling matrix (for further details seeref. [8]), using the following matrix equation:

$$\frac{d^2g}{d\rho^2} = O^{-1}Ug. \qquad (1)$$

For the chemical system considered in our study the reaction coordinate ρ was divided into 150 sectors, and in each sector the basis functions and related eigenvalues were calculated. Accordingly, the pseudocode of ABC (see Fig. 1) is articulated in a first uncoupled loop over the different arrangements of the various ρ sectors to build the O and U matrices.

```
Do j_sectors varying from 1 to N_sectors
   Calculate sector eigenvalues and eigenfunctions
   Calculate coupling and border overlap matrices
EndDo j_sectors

Do j_events varying from 1 to N_events
   Generate initial conditions: E_tr, J ...
   Perform preliminary calculations
   Do j_sectors varying from 1 to N_sectors
      Perform sector propagation
      Map the solution into the next sector
   EndDo j_sectors
   Perform final calculations
   Write output data: E, S^J
EndDo j_events
```

Figure 1. Pseudo-code of the ABC application.

The first loop is followed by a second one generating the energy specific partial S^J matrix elements. Inside this second loop an inner recursive loop repeatedly propagates the solution from small ρ values to the asymptotes.

III. THE GRIF FRAMEWORK

The basic goal of GriF is to provide the users with a user friendly tool allowing them to exploit the innovative features of Grid computing with no need for mastering related technicalities. In other words, GriF makes Grid applications black-box like and, at the same time, leads to better memory usage, reduced cpu and wall times consumption as well as to an optimized distribution of tasks over the Grid platform. This leads also to a more efficient exploitation of the innovative features of the Grid when building applications of higher level of complexity. Thanks to its robust SOA framework nature GriF can, in fact, support collaboration among researchers bearing complementary expertise and contributing to collaborative work when they want to articulate their computational applications as a set of sequential, concurrent or alternative tasks. The SOA organization of GriF consists of two Java servers and one Java client, as sketched in Fig. 2.

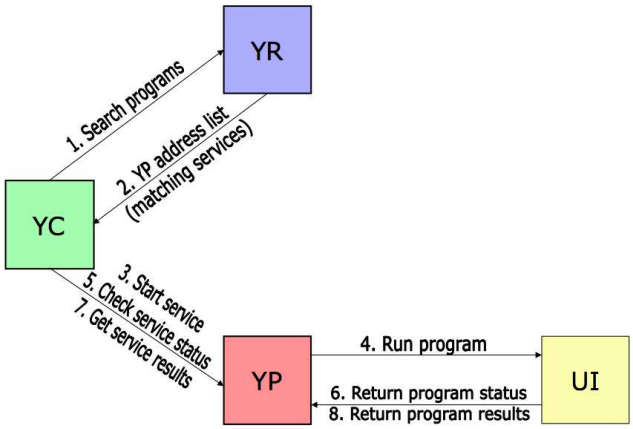


Figure 2. SOA Organization of GriF in terms of Java servers and client.

The first Java server is YR (Yet a Registry) that is based on the standard UDDI [13] (Universal Description, Definition, and Integration) protocol and acts as a directory listing to carry out service registry operations. YR is inspected to find and invoke the appropriate Web Services.

The second Java server is YP (Yet a Provider) that makes use of the Simple Object Access Protocol (SOAP) [14] (an XML-based messaging format established as transmission framework for inter-service communication via HTTP or HTTPS). YP holds the Web Services of the VRC and handles quality parameters. Both YR and YP make use of WSDL [15] (Web Services Description Language) to describe the services provided and to reference self-

describing interfaces that are published in platform-independent XML documents. The Java client is YC (Yet a Consumer) that is the entry point for operating with GriF within the Grid. YP takes care of running the jobs on the associated User Interface (UI) while YC implements all the already mentioned extensions and protocols to correctly interface the Web Services offered by GriF.
Accordingly, when running an application on the Grid, YC can be used to perform the various actions devoted to the management of the basic Grid operations. YC is also used to search applications on YR, to introduce some changes and to compile as well new executables (a different version of the same application) on the selected YP. The results of previous actions can be used to start the execution of the Grid job (after passing a user-specific input file), to monitor the jobs status and eventually retrieve the results (some Web Services wrapping the Grid middleware commands and managing their execution on the Grid have been implemented to this end).
Moreover, Web Services built specifically for the management of the Grid Proxies (and/or of the Robot Certificates [16]), the handling of GriF, the compilation of different executables, the monitoring of the status of the jobs and the retrieving of the results can be handled using YC. In addition, YC was transformed into the YA (Yet an Applet) Java applet in order to allow the use of GriF also on a client machine with no YC.
Typical fragment pseudo-codes used respectively in YPs (to wrap Grid applications into Java Web Services) and in YC (to invoke the Java Web Services exposed by YPs), are listed in Fig. 3.

YP Web Service code:

```
public class RunGrid {
   public String run(String input, String user, String date) {
      String cmd = ''rungrid.sh'' + input + user + date + ''>'' + outputfile;
      String[ ] command = {''sh'', ''-c'', cmd};
      Process p = null;
      Runtime r = Runtime.getRuntime();
      p = r.exec(command);
      p.waitFor();
      FileReader file = new FileReader(outputfile);
      BufferedReader reader = new BufferedReader(file);
      while ((line = reader.readLine()) != null) output += line;
      return output;
   }
}
```

YC Client code:

```
public class YC {
   public static void main(String [ ] args) {
      String endpoint = ''http://'' + YP_NAME + ''/RunGrid.jws'';
      Service service = new Service();
      Call call = (Call) service.createCall();
      call.setTargetEndpointAddress(new java.net.URL(endpoint));
      call.setOperationName(''run'');
      call.addParameter(''input'', XMLType.XSD_STRING, ParameterMode.IN);
      call.addParameter(''user'', XMLType.XSD_STRING, ParameterMode.IN);
      call.addParameter(''date'', XMLType.XSD_STRING, ParameterMode.IN);
      call.setReturnType(XMLType.XSD_STRING);
      String result = (String) call.invoke(new Object [ ] {input, user, date});
   }
}
```

Figure 3. Pseudo-code of wrapping and invoking web Services in GriF.

In the YC section of Fig. 3, the variable *YP_NAME* takes the value of the YP name selected by the user during the Service Discovery phase performed using YR (see Fig. 2).
Accordingly, Software Providers can expose their services in an open, standard and secure way suited to serve all kinds of users including those having little familiarity with the wrapped applications and the Grid platform. In this way, the applications gain a high level of friendliness and portability while the Grid system reaches a high level of expertise. GriF users can, in fact, carry out most of their operations using mainly a natural-like language (for example, when searching for an application of interest, the VO name, the program name, as well as some keywords matching the desired application description and functions, are sufficient). Moreover, additional procedures have been implemented to carry out various Framework-side operations like those related to the Grid resources matching for the specific applications offered.

IV. THE WEB SERVICE INTERFACE WRAPPING ABC

The GriF application considered here is the running of the ABC program to carry out a systematic quantum study of the reactive efficiency of atom diatom systems and its implementation as a Web Service. The code presents a high level of complexity that needs to be properly dealt by adopting appropriate interfaces in order both to efficiently support users having a low level of expertise (training purpose) and users wishing to customize it for advanced studies (research purpose). The ABC wrapped service has been tested on the EGEE Grid middleware by developing a minimal workflow in which the user is driven from the beginning (the generation of the input file, as illustrated in Fig. 4) to the end (the retrieval of the results from the Grid) of the process.

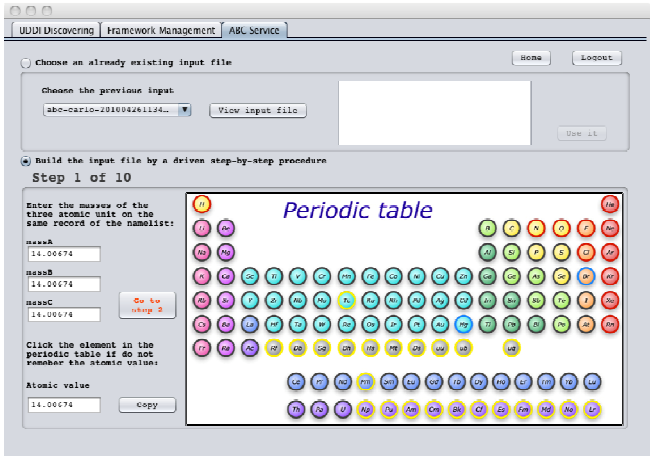


Figure 4. The first step of the Wizard process of building the input file.

In the developed procedure the user preliminarily chooses between a Classical Mechanics (QCT) and a Quantum Mechanics (QM) treatment and then selects the chemical system to be investigated. The ways these two options are implemented are different. In fact, while the choice between

a QCT and a QM treatment implies the adoption of a different application and of the related set of executables, the choice of a different system can be confined to the modification of the values of some parameters of the input file (like masses and energies) and the adoption of a different Potential Energy Surface (PES). The application workflow is, therefore, designed to search on the Web for the availability of an appropriate PES and related parameters before starting the integration of the scattering equations. If a proper PES is not available, GriF is conceived to search on the Web for the availability of a sufficiently large set of ab initio potential energy values for the system considered. If such values are available they are fitted to a proper functional form using an appropriate fitting procedure (see for example ref. [17]). If the ab initio values are not available, one has to produce them by running ab initio electronic structure calculations using again an ad hoc procedure (see for example ref. [18]). Here we consider a version of GriF implemented for the case in which various suitable PESs are available and therefore they need only to be ticked up for the Web Service execution. To better illustrate the case considered, a screenshot of the execution (Wizard Mode) of the ABC application on the Grid is shown in Fig. 5.

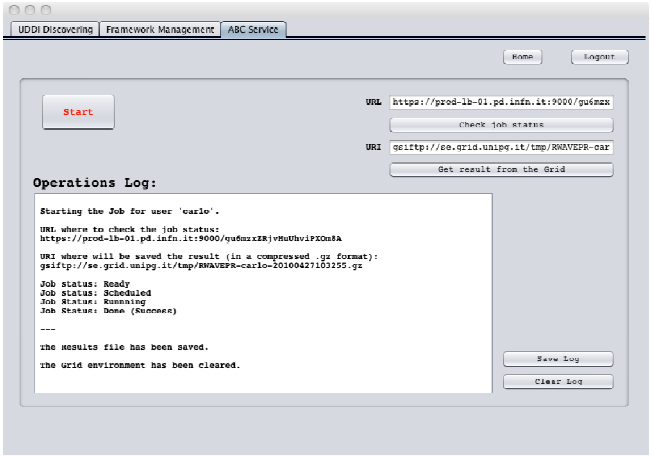


Figure 5. The execution of the ABC application on the Grid.

In the Wizard mode the users access a simple graphical interface on which they can start the execution of the ABC application just by pressing the button ”Start”. The job is submitted for the distributed execution on the Grid using an adaptive algorithm making use of the Parametric Job option [19]. After the job has started, useful information like the URL (where to check the jobs status) and the URI (where to fetch the output when the job ends) are returned. Both information are automatically fed into the appropriate fields. After receiving the typical sequence of Grid messages associated with a successful completion of the job (”*Ready*”, ”*Scheduled*”, ”*Running*” and ”*Done (Success)*”), users can access the final results. This can be done by clicking the button ”*Get Results from the Grid*” that triggers the set of Web Services responsible for the acquisition of the results via the SOAP protocol and then stores them in the YC directory selected by the user.

V. The Active And Passing User Profiling

On this application, a preliminary classification of the COMPCHEM users based on their compiling and running habits has been attempted. GriF offers, in fact, the possibility of collecting at all stages (from the selection of the PES to the collection of the results) a certain amount of data that is stored in a MySQL [20] database. This is particularly useful to optimize the management of a VRC by defining some Quality of Services (QoS) parameters. In particular, GriF allows, for example, to collect information on the type of PES preferentially utilized by the users, on how complex is the job assembled for running (e.g. the parameter values and eventually the syntactic and semantic errors made) and on which are the critical parameters for job execution (e.g. date, time, application name and if it is a new compiled version). Moreover, by combining this feature of GriF with the possibility of retrieving users data from the Grid middleware saving them into its database, it is possible to profile the users of a VRC and evaluate their quality (Quality of Users, or QoU).

In the investigation reported here, a sample of COMPCHEM users has been parameterized in terms of their ability and/or interest to run and/or modifying the code used. The results of the parametrization are illustrated in Fig. 6 where the histogram of the frequency of the value of the daily ratio between the number of job compilations (Nc) and the number of related executions (Nx) measured during the month considered is shown.

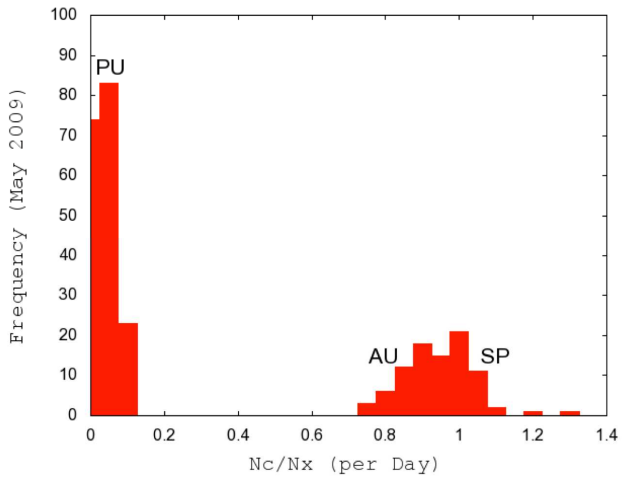


Figure 6. The first three COMPCHEM user profiles: the Passive User (PU), the Active User (AU) and the Service Provider (SP).

As apparent from the figure, the measurements are characterized by a clear bimodal structure made by a sharper peak placed on the extreme left hand side region of the plot and a smoother one placed on the opposite side. The

first peak is centred on the interval $0.05-0.10$ (with a bin width of 0.05). It clearly indicates that the largest number of jobs run using an already compiled executable and that the changes are confined to the data of the input file. On the contrary, the second peak tells us that for the second largest number of jobs (though smaller than the first one) the compilation is performed almost every time the job is executed.

The appearance of such a bimodal structure suggests the existence of two main types of users. The first type (associable with the lhs peak) can be called Passive User (PU). PUs typically run on the Grid platform a stable version of an application and, therefore, do not need to compile the application every time it is sent for execution. This type of user is more a "production scientists" who builds his/her scientific investigations on a massive production of numerical experiments carried out using well-established numerical procedures. In our study, this is the case, indeed, of the scientists carrying out extensive ABC computational campaigns in which the efficiency of the relevant reaction is calculated for a large variety of initial conditions.

The second type of user (that can be associated with the rhs maximum) is the Active User (AU). AUs typically develop on the Grid platform new applications or new versions of an application and, therefore, need to compile about each time the application is sent for execution. This type of user is a "chemistry expert" that, in the case considered by us, develops new applications of the quantum mechanics codes dealing with chemical processes. In the application considered by us (the ABC code) the main developments have been concerned with the implementation of a set of new PESs for different chemical systems (this work widens the offer of ABC services on the Grid). Most likely this type of AU turns out at a certain moment into a PU.

VI. DECLUSTERING AUS

Strict sense AUs, however, have to be distinguished from other types of users whose compilation activities exceed executions. This is already apparent from the broad shape of the $Nc/Nx \simeq 1$ maximum that indicates that there is an appreciable extent of variability as far as compilation before execution is concerned. This is due to the fact that the application is sometimes not recompiled before being executed whereas other times it is compiled more than once.

The second case is better identified as that of Service Providers (SP)s. SPs which carry out the work of wrapping computer applications into GriF Web Services (and testing them as well). For this reason, they bear characteristics similar (though not identical) to those of AUs. As a result, in the rhs structure of Fig. 6 the broad structured maximum can be split into an AU and an SP slightly displaced distributions centred one just above and the other just below the $Nc/Nx=1$ value. This is clearly the case of the sample of users investigated by us. In particular, while the AUs are mainly "expert chemists" using GriF more or less transparently to build their own or shared applications, SPs are mainly "computer scientists lent to Chemistry" for the purpose of developing Grid Services on behalf of the generic user (PUs in most cases). These SPs use GriF more wisely also adapting it for the interaction among existing applications.

Our study, however, has singled out a more extreme case of SPs that we call Software Developers (SD)s. This is clearly indicated by the structure shown in Fig. 7 in which, as in Fig. 6, the frequency of the Nc/Nx value is plotted as a histogram (though using larger bins of width 1) whose abscissa extends to 20.

Fig. 7 tells us that, in our case, SD users bear statistical importance. SDs have little interest in running existing computational scientific applications (as in the case of GriF and/or Web Services developers). Yet they are interested in building new applications and/or new features of GriF as is the case of the exploitation of new theoretical approaches to a given problem.

For this reason, they compile much more frequently than running. Consequently, for them the value of the daily Nc/Nx ratio is abundantly larger than 1 (in our case we found a peak at $Nc/Nx=11$).

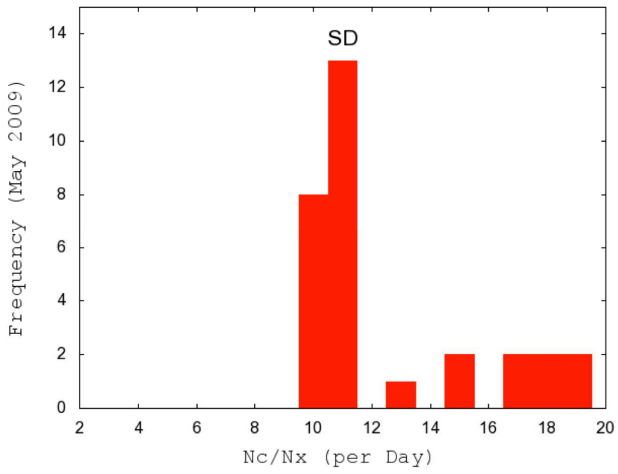


Figure 7. The fourth COMPCHEM user profile: the Software Developer (SD).

A caveat, however, has to be issued at this point. In fact, while the PU characteristics (daily Nc/Nx values close to 0) are extremely localized, Nc/Nx values ranging from about 1 to larger values are much more scattered.

Moreover, it would not come as a surprise a further smoothing of the results for larger statistical samples that would make the separation of the various kinds of users more difficult to obtain like in the case in which the same user plays different roles at the same time. This prompts a more articulated definition of the users and the development of new Grid sensors.

VII. Conclusions and Future Work

In the present work the development of GriF, a Java SOA Grid Framework structuring computational applications as Web Services that can be run transparently on the Grid, is illustrated. GriF is open to be driven by a user-side control allowing the management of a domain-specific operation logic in which the user of a VO (or of a VRC) can select various options (like modifying part of an application or assembling a different input) before entering the running phase. As a result, not only it has been possible to carry out on the Grid massive computational campaigns by spending a minimum effort and achieving maximum throughput (e.g. using the Web Service implemented in this work which wraps the ABC application) but it has been also possible to profile some types of users. This feature is of particular interest because the profiling of the users is the basis on which an evaluation of the work carried out in a VO can be performed and will be the basis on which its sustainability will be grounded. Yet, the preliminary indications on the users classification discussed in this work aimed at perform an evaluation of the Quality of the Users (QoU) need to be validated by more extended statistical analysis and can also be improved by introducing new evaluation parameters that we have planned to develop in collaboration with CESGA that is the official monitor of the computational activities carried out on the EGI Grid.

Acknowledgments

The authors acknowledge financial support from the project EGEE III and from the COST CMST Action D37 ”CHEMGRID” in particular for building the collaborative network. Thanks are also due for funding to the project ”Fundamental Issues on the Aerothermodynamics of Planetary Atmosphere Re-entry” (AO/1-5593/08/NL/HE) and the ESA-ESTEC contract 21790/08/NL/HE, MIUR and ARPA. Computer time allocation has been obtained through the COMPCHEM VO of EGEE.

References

[1] The Enabling Grids for E-sciencE (EGEE); http://euegee.org/.
[2] Foster, I., Kesselman, C.: Scaling System-Level Science: Scientific Exploration and IT implications, Computer, 39(11), 31-39 (2006).
[3] Laganà, A., Riganelli, A., Gervasi, O.: On the Structuring of the Computational Chemistry Virtual Organization COMPCHEM, Lecture Notes in Computer Science, 3980, 665-674 (2006) http://compchem.unipg.it
[4] The European Grid Initiative (EGI); http://web.euegi. eu/documents/other/egi-blueprint/.
[5] The Web Services Architecture, W3C Working Group; http://www.w3.org/TR/ws-arch/ (2004).
[6] Gervasi, O., Dittamo, C., Laganà, A.: A Grid Molecular Simulator for E-Science, Lecture Notes in Computer Science, 3470, 16-22 (2005).
[7] Laganà, A., Gervasi, O.: A Priori Molecular Virtual Reality on EGEE Grid, International Journal of Quantum Chemistry, 110, 446-453 (2009).
[8] Skouteris, D., Castillo, J. F., Manolopoulos, D. E.: ABC: A Quantum Reactive Scattering Program, Comp. Phys. Comm., 133, 128-135 (2000).
[9] Specialised Support Centres; http:/knowledge.euegi. org/knowledge/index.php/Specialised Support Centres#Science Gateways.
[10] Freire, E., Simon, A., Lopez, J., Fernandez, C., Diez, R., Diaz, S., Manuali, C., Laganà, A.: Application Domain Accounting for EGI, 5th EGEE User Forum, Uppsala (SW), April 12-15 (2010); http://egee-uf5.euegee.org/.
[11] Centro de Supercomputaci´on de Galicia (CESGA); http://www.cesga.es/ (2010).
[12] Schatz, G.C.: Quantum reactive scattering using hyperspherical coordinates: results for H + H2 and Cl +HCl. Chem. Phys. Lett., 150, 92-98 (1998).
[13] The Universal Description, Discovery and Integration (UDDI) protocol 3.0.2; http://www.oasisopen.org/specs/ (2005).
[14] The Simple Object Access Protocol (SOAP) 1.2; http://www.w3.org/TR/soap (2007).
[15] The Web Services Description Language (WSDL) 1.1; http://www.w3.org/TR/wsdl (2001).
[16] INFN Certification Authority; http://security.fi.infn.it/CA/docs/ (2010).
[17] Arteconi, L., Laganà, A., Pacifici, L.: A Web-based application to fit potential energy functionals to Ab Initio values, Lecture Notes in Computer Science, 3980, (2006) 694-700.
[18] Storchi, L., Tarantelli, F., Laganà, A.: Computing Molecular Energy Surfaces on a Grid, Lecture Notes in Computer Science, 3980, (2006) 675-683.
[19] Parametric Jobs - EGEE SEE ROC Wiki; http://wiki.egee-see.org/index.php/Parametric Jobs.
[20] The MySQL database homepage; http://www.mysql.com/.

Carlo Manuali is the head of the Scientific Computing Services Office and Security Manager of the University of Perugia (Italy) and has a Master in Computer Science. He has worked in the past in the Computer Centre of the University as Network and System Manager. He is also a Senior Professional Licensed Engineer in Information Science. He has a permanent collaboration with specialized Informatics magazines on the field of Operating Systems.

Antonio Laganà is full professor in Chemistry at the University of Perugia (Italy) since 1994 and President of the Computational Chemistry Division of EUCHEMS. He was Director of the Computer Centre (1996-2001) and is Director of the Department of Chemistry (from 2002). His scientific activity focuses on the development of computational procedures for pure and applied physical sciences by exploiting innovative computing technologies. Italian representative in COST CMST he is the past Chairman, past action coordinator and present working group coordinator. Past president of ECTN he presently coordinates its computational activities. Author of more than 300 papers published in international journals, he has edited several books, organized various international conferences and delivered several invited talks.

A Service-Oriented Interface to the iRODS Data Grid

Nicola Venuti[1], Livia Torterolo[1], Alberto Falzone[1], Michael Conway[2] and Leesa Brieger[3]
[1]*NICE S.r.l. {nicola.venuti,livia.torterolo,alberto.falzone}@nice-software.com*
[2]*DICE Center – UNC*
[3]*RENCI - UNC*

***Abstract*— iRODS micro-services and rules can be used to build a data grid that implements a community's own data policy. However, often the data administrators are not the developers who customize the services or deploy the data grid. A tool that gives the data administrator intuitive access to the rules and special-purpose services of his data grid is important in separating the IT tasks from the data administration tasks. The EnginFrame (EF) cloud interface framework from Nice S.r.l. was used to build a service-oriented iRODS interface. This interface demonstrates how data grid access can be customized for community use; one view of the data grid, determined by data usage scenarios, is provided for the community user, and another view, determined by data management criteria, is provided for the administrative user.**

***Index Terms*— iRODS data grid, data grid interface, data grid access, web interface, EnginFrame, EF, Grid Portal, cloud interface, administrative interface.**

I. INTRODUCTION

DEVELOPMENT of special-purpose micro-services and rules will equip an iRODS data grid to implement specialized data access and preservation policy as required by a target community. The developers who would customize a data grid in this way may not, however, be the data administrators who determine and/or enforce data policy for that community.

Therefore, along with a customized data grid, it is imperative to offer a user-friendly interface that provides not only user access to community data, but also administrative access to the services that support and implement data policy. The data grid, with special-purpose services and with an administrative interface, then provides the data administrator with the necessary tools to curate and preserve his community's electronic data - without being an iRODS programmer to do it.

The user-friendly interface provides a separation between the data administrator and the systems administrator. It can offer intuitive access to the specialized data services, freeing up the data admin to concentrate on applying, enforcing, and verifying data policy for his community.

The authors used the EnginFrame (EF) cloud interface framework to develop a prototype of such an interface; this was used for a live demonstration of iRODS services at an NSF/NARA/NITRD iRODS presentation in August 2009. The interface was used to showcase important iRODS archival services in a real-time demo. It serves to illustrate how an interface can be customized to offer specialized views of the services implemented in a given data grid. Further, the interface presents one view of data and services for community users and another view, which includes more administrative functionalities, for the data administrator.

Several basic iRODS services were selected for the demonstration; we briefly mention implementation considerations for some of these special services, followed by a description of the EnginFrame interface and then the blending of the two technologies.

II. SPECIALIZED IRODS SERVICES

While iRODS can be viewed as a framework for implementing data policy for the curation of electronic assets, it is also a tool kit that comes with many pre-defined rules, microservices, and capabilities. Some of these enable functionalities such as audit tracking and quota checking, in support of verification of policy; others enable capabilities such as searching on user-defined metadata.

These were the sorts of functionalities, based on out-of-the-box iRODS services, that were showcased at the NSF demo; thus these were the services exposed in the EF interface to the data grid.

2.1. Audit Tracking

Audit tracking is enabled in iRODS by changing the setting of the parameter auditEnabled from "0" to "2" in iRODS_root/server/icat/src/icatMidLevelRoutines.c, then recompiling, and restarting the iRODS server. Once audit tracking is enabled, any operation that calls upon the iCAT metadata catalogue is logged - in the iCAT. Any requests, such as downloading a data object, changing permissions on a collection, deleting or creating an object, etc., are all logged in the iCAT's audit table, along with record of the change that was made if authorization for the operation was granted. Audit information can then be tracked by querying this table and presenting the results in a user-friendly format. The queries can be implemented with the *iquest*

icommand or with microservices by using *msiMakeGenQuery* and *msiExecGenQuery*.

There is need to be careful, however, with these queries. The microservice queries use an iRODS-specific syntax to approximate SQL but does not replicate it perfectly. Iquest allows a reduced form of SQL querying. Neither approach yet gives full SQL functionality. For audit table querying, there is a further complication that can result in spurious results. Consider that the audit table in the iCAT database contains the following fields:

```
AUDIT_OBJ_ID
AUDIT_USER_ID
AUDIT_ACTION_ID
AUDIT_COMMENT
AUDIT_CREATE_TIME
AUDIT_MODIFY_TIME
```

The audit table, in AUDIT_OBJ_ID, contains information about the entity (data object, collection, resource, user, etc.,) that is the object of an action that was performed and logged. It contains the ID of the target entity; however, there is no built-in mechanism to determine which it is - object, collection, user, resource, etc. Thus, at any one time, the AUDIT_OBJ_ID field of the audit table can refer to any of a number of tables containing detailed information on either a data object, a collection, a user, or a resource. The joins of the standard iRODS query services then have the effect of joining all the tables referred to by the ID, with the result that much spurious information is retrieved with the query.

By breaking down the joins into a series of simpler *iquest* queries, it is possible to separately query on each type of entity in the audit table, thereby avoiding the joins that cause spurious results to be generated. The following example for an audit procedure for an administrative user illustrates this; the *iquest* commands are run in a script so that output can be saved from one step to the next.

1.To see an audit trail for a given user, save an iRODS user name into a script variable and run the iquest command to query the audit table:

```
iquest "SELECT AUDIT_OBJ_ID,
        AUDIT_ACTION_ID, AUDIT_COMMENT,
        AUDIT_CREATE_TIME,
        AUDIT_MODIFY_TIME WHERE
        USER_NAME = '${_irods_username}'
```

2. Save the AUDIT_OBJ_ID into a script variable and use it to get query and get separate results from each entity table:

```
iquest "SELECT COLL_NAME, DATA_NAME
        WHERE DATA_ID = '${_objId}'"

iquest "SELECT COLL_NAME WHERE
        COLL_ID = '${_objId}'"

iquest "SELECT USER_NAME WHERE
        USER_ID = '${_objId}'"
```

For the NSF/NARA demo, the results of these queries were arranged into xml files to allow for formatted presentation. Additional Java filters provide an easy way to further manipulate the results and were applied in order to sort and refine the search results.

2.2. Other Services

iRODS allows users to add their own AVU triplets (attribute, value, units) to the iCAT metadata catalogue. Metadata searching of user-defined metadata was implemented for the demo using the iquest icommand to query the iCAT.

The implementation of quotas is awaited in iRODS and should be coming out in version 2.3. In the meantime, it is possible to use *iquest* to return and display usage information for each user, handling it similarly to the way quota information will be handled. This was implemented in the demo prototype.

The *irule* icommand allows users to run any iRODS rules on a command line. The interface also provided a means of pointing and clicking to edit and run selected rules.

III. ENGINFRAME

EnginFrame is proprietary software developed by Nice S.r.l. It is typically used as a computational grid portal or a cloud interface and serves as a framework for logically collecting applications, services and resources and presenting them in a web 2.0 interface that provides user-friendly access to the distributed resources. It is not a portlet container but instead delivers services that are JSR168-compliant; EnginFrame allows organizations to provide application-oriented computing and data services to both users (via Web browsers) and in-house or ISV applications (via SOAP/WSDL based Web services) so EF services could be used as portlets in another portal.

The main goal of EF is to hide the details and the complexity of the underlying infrastructure in order to improve usability and utilization. Usability goes up when end-user requirements for accessing the infrastructure go down, and utilization is improved by making the evolution of the underlying systems transparent to the end-user and enforcing the utilization policies even as infrastructure evolves.

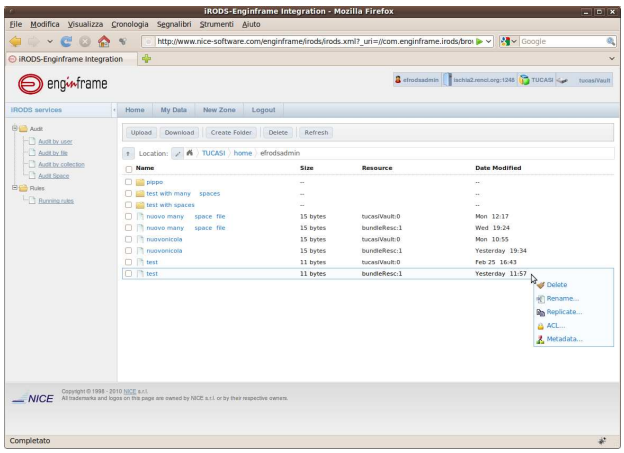


Figure 1. Metadata and ACL settings can be viewed and monitored through the browser.

EF provides a flexible authentication framework with built-in support for a wide set of well-known authentication mechanisms like OS/NIS/PAM, LDAP, and Microsoft Active Directory. It has been integrated with the iRODS challenge-response authentication mechanism. The EF authorization framework allows the definition of groups of users and access control lists, thus providing a means for tailoring the Web interface to the specific users' roles or access rights. This was used in the demo interface to distinguish between community users and administrative users of the data grid. Community users were presented, in the interface, a reduced set of services compared to administrative users.

IV. THE IRODS EF INTERFACE

The merging of the EnginFrame and iRODS technologies required development of an iRODS plug-in for EF and the wrapping of the iRODS services as EF services. The EF file manager for data browsing was also outfitted with iRODS functionalities so that some of the basic iRODS characteristics are present in the data browser.
User-defined metadata can be added, modified, queried, and deleted as part of basic iRODS functionalities. Setting and modifying ACL permissions are also included among the basic iRODS capabilities. Both these functionalities are available with the browser through the EF interface. See Figure 1.

Disk usage is queried using *iquest* and displayed.

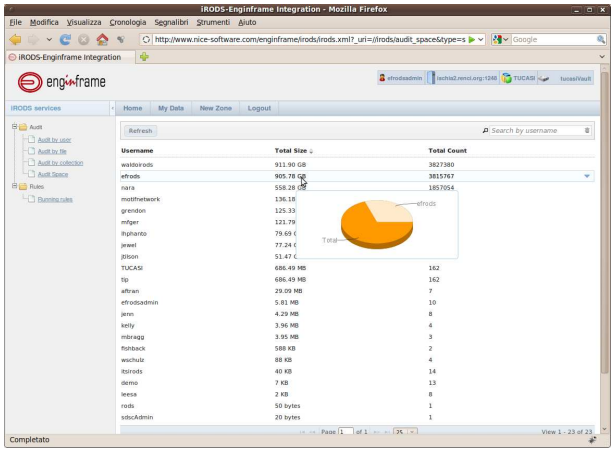


Figure 2. Usage data.

The same sort of display is planned for quotas when that functionality becomes operational. See Figure 2.
Figure 3 shows the unfiltered results of an audit table query on all entries, and Figure 4 is a snapshot of the rule editor.

V. DEPLOYING DATA GRIDS

The customization of a data grid for a user community is an important step in deploying this technology for a given user group. Beyond simply installing the data grid, data management policy must be unambiguously defined and then translated into the micro-services and rules of this technology.
Another very important step in the deployment is the development of a user-friendly interface for accessing the data grid. A custom interface can provide intuitive access to the custom services of the data grid and a user-friendly way of invoking the rules that implement and enforce data policy.
Further, the interface can be customized to various user groups that access the data and data services. As mentioned above, the EF interface was developed to show different views of the services to community and administrative users, thereby distinguishing between the different classes of services offered to the two groups. It would also be possible to adjust the view of the data grid to other user groups, so that the presentation of data and services fits with a group's own use cases.

VI. THE FUTURE

A new domain of expertise will likely grow up around this technology, embodied by those who deploy the iRODS data grids. They will likely become increasingly separate from the DICE group who develops iRODS as well as from the user communities who are the consumers of the iRODS technology.

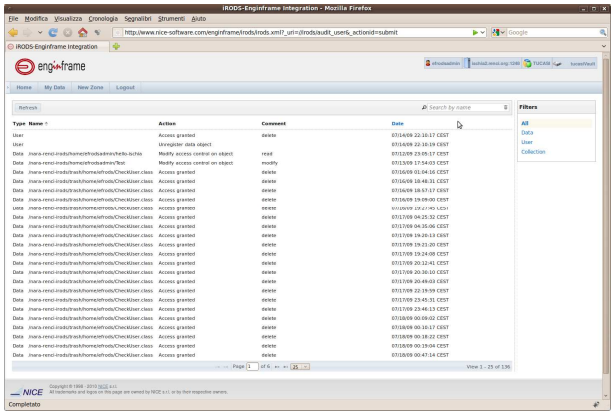


Figure 3. Data dump of the audit table.

There is in fact a need for a third group that bridges the gap between the other two. The developers know all that this technology can offer but are often not aware of the intricate details of the needs of the user groups. Users know some rudimentary aspects of the data grids but often define their needs in terms of the constraints they have learned to live with rather than exploiting the full potential of iRODS. There is increasingly a need for a group that straddles those two perspectives and brings rich iRODS capabilities to user groups with complex data needs.

These deployment groups must work closely with data specialists from the user communities in order to understand the required policy to implement in the data grids and how the administrative interfaces should operate. They will also have to understand how the users must view the data and services presented in order to meet their use cases. Policy should become easy to apply using the custom interface, and the full functionality of rich iRODS services should be delivered.

Deployment groups will promote the adoption of iRODS data grids, supporting communities who want to explore the technology and allowing its adoption even by groups who may not be well-supported with in-house IT specialists. The deployment groups will do the programming of the services and the development of the interfaces so that users and data administrators will be freed from these tasks. The upshot is that many more communities will have access to this technology.

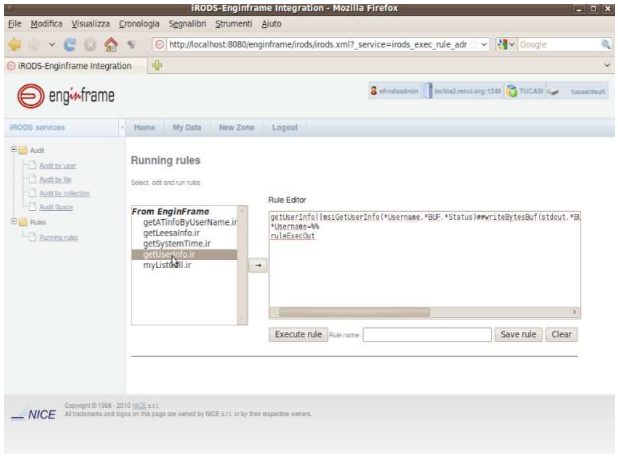


Figure 4. The rule editor.

REFERENCES

[1] Reagan W. Moore, Richard Marciano, Arcot Rajasekar, Antoine de Torcy, Chien-Yi Hou, Leesa Brieger, Jon Crabtree, Jewel Ward, Mason Chua, UNC Chapel Hill; Wayne Schroeder, Michael Wan, Sheau-Yen Chen, UCSD, "NITRD iRODS Demonstration", sponsored by NARA at NSF, 2009. Can be linked from https://www.irods.org/index.php/Publications.
"Technical Demonstration of Integrated Preservation Infrastructure Prototype", National Coordination Office for Information Technology Research and Development (NITRD) / NSF / NARA, National Science Foundation, Washington, D.C., August 4, 2009 Powerpoint Version. Combined Video and Powerpoint Slides of NITRD Demo. Can be linked from https://www.irods.org/index.php/Publications.
[2] iRODS and Data Preservation 2nd Workshop on Data Preservation and Long Term Analysis in HEP, Wayne Schroeder, SLAC National Accelerator Laboratory, Menlo Park, CA, May 26, 2009. Can be linked from https://www.irods.org/index.php/Publications.
[3] Policy-Based Distributed Data Management Systems, Open Repositories 09, Reagan Moore, Arcot Rajasekar, Mike Wan, May, 2009. Can be linked from https://www.irods.org/index.php/Publications.
[4] http://www.nice-software.com
[5] http://www.enginframe.com
[6] http://code.google.com/p/ef-irods-plugin/
[7] JSR168 http://jcp.org/en/jsr/detail?id=168
[8] NIS http://en.wikipedia.org/wiki/Network_Information_Service
[9] PAM http://kernel.org/pub/linux/libs/pam/
[10] LDAP http://en.wikipedia.org/wiki/LDAP

An Efficient Workflow System in Real HPC organization

Yudin, Y., Krasikova, T., Dorozhko, Y., Currle-Linde, N. and Resch, M.
High Performance Computing Center Stuttgart (HLRS),
University of Stuttgart, Germany,
{yudin, krasikova, dorozhko, linde, resch}@hlrs.de

***Abstract*—Currently some scientific and business organizations own sets of High Performance Computing (HPC) resources. These resources are heterogeneous. They have different purposes, architectures, performance and used software. There is a problem with providing a convenient way to run complex computational experiments in real HPC organizations. Such complex computational experiments use different resources simultaneously to start a large number of computational jobs.**

In this paper we describe experience in creating a workflow system which makes it possible to run complex computational experiments under conditions of a real HPC organization. Significant requirements to this system are high efficiency and interoperability with the existing HPC infrastructure of the organization without any changes.

***Index Terms*—HPC, HPC organization, workflow, SEGL, GriCoL.**

I. INTRODUCTION

EACH HPC (High Performance Computing) organization that provides computational power has its own scheme of resource management, security policies, concepts of access rights, limitations for the use of computational resources, disk space, memory and network. Organization may offer different cost models for different purposes of resource usage, such as commercial usage or scientific research usage. Resources can be provided free of charge or as a paid service. Accordingly, user jobs will have different priorities and limitations.

However, there are a lot of common or similar principles in the existing frameworks to organize work with a HPC resource in a real organization. Commonly, these principles are based on widespread and well-known open-source software. Usually resources run Linux/Unix-like operating systems (Linux/Unix OS are installed on 95.4% of the HPC computers in the TOP500 [1]). Most computational resources, clusters or supercomputers, have one or more access-point machines (so called front ends). A job submission is organized with the help of batch systems [1] in accordance with queue policies, priorities and limitations existing in the organization.

Users can access front end machines within the local network of the organization via SSH [3]. Access from an external network can be denied or can be restricted and provided only for fixed IP addresses. Quite often resources are accessed from an external network through VPN (Virtual Private Network). Users have their own accounts on all accessible resources of the organization. Data exchange between users and resources is organized using SSH [3] and such programs as SCP, SFTP, RSYNC.

The approach described above is widespread in organizations which offer HPC resources, because of simple installation and support, high reliability and security.

The standard scenario for working with HPC resources is the following: the user transfers input data to the selected resource. Then the user submits a job to the queue. After that the user has to check state of the job periodically. Often such a job is part of a complex computational task. A large number of computational jobs can be started within the scope of the task in parallel or sequentially. Jobs can be dependent on other jobs, if some resulting data of one job is used as an input data for another job. A user, who starts such tasks (or experiments), requires a suitable tool to create the computational experiment, to run it and to monitor it. Workflow systems [4] for HPC environments fit these requirements.

In this paper we describe our experience in development and usage of the high performance workflow system SEGL (Science Experimental Grid Laboratory). This system is designed to work in a real HPC organization. Such an organization has a set of the HPC resources and a defined standard schema for working with these resources.

The main requirements to the SEGL system were the following:

- The system must work under conditions of the real HPC organization without any changes of the infrastructure of the organization.
- Reliability of the system.
- Simplicity of using HPC resources by applied specialists (scientists, engineers) with the help of this system.

- Ability to reuse existing code. This means reusing of the predefined computational jobs as well as the complex computational experiments.
- Scalability of the system to complex simulation experiments that consist of large number of computational tasks (up to tens of thousands and hundreds of thousands).

All these requirements have the same priority.

It should be noted that we are talking about real HPC organization, but not about a virtual organization (VO) [5]. There are a number of differences between a VO and a real HPC organization [6] concerning HPC resources from the viewpoint of the workflow developer. There are such questions as administration of the resources, network communications and security. Often computational resources of the real organization are used as resources of the VO, but resources of the VO can have different settings and limitations. Also user support in a VO is a separate and complicated problem [7].

However in the case of using the HPC resources for business and industrial tasks foreground questions are security, commercial advantage, quality of user support and responsibility of the organization that provides resources. Using resources as a part of the VO supposes some changes of the infrastructure of the real organization. We solve the problem by developing a workflow system within the scope of the real HPC organization, without any infrastructure changes.

II. Workflow Description Language

The Grid Concurrent Language (GriCoL) [8] is used in the system to describe a computational experiment as a workflow. GriCol has a two-layer model for the description of the experiment. There are the control flow level and the data flow level.

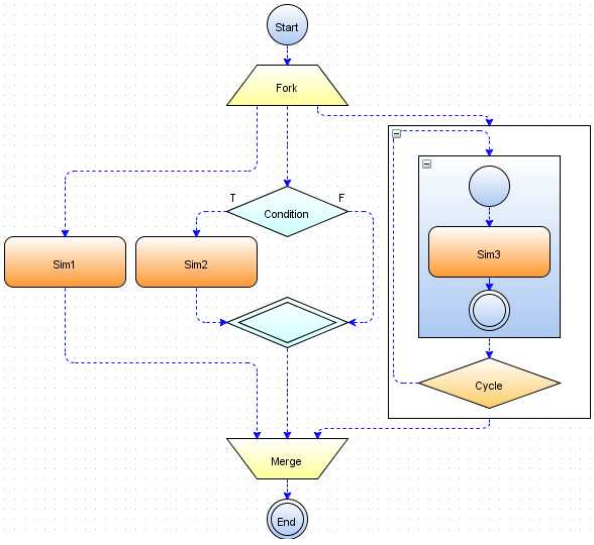


Figure 1. Control flow level.

The control flow level (Figure 1) offers possibilities to describe set of parallel and sequential tasks within the scope of the experiment. This description is the workflow of the experiment. To describe the logic of the experiment, the control flow level offers different types of blocks: solver, transition, fork/merge (parallel branching), cycle, conditional transition, nested experiment.

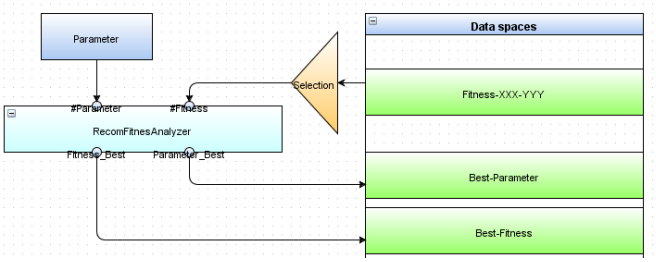


Figure 2. Data flow level

For each computational block in the control flow level the user has to describe data flow model.

The data flow level (Figure 2) description contains detail information about the computational job that will be started on the HPC resource. Also this information includes descriptions of input and output data sets, variables that describe different parallel data sets, parameter values for the different data sets and expressions to select and group the data sets.

GriCoL provides a module approach [9]. Jobs that will be started on the HPC resource are described in the system as executable modules. A module is an abstract description of the computational job. It includes information about input and output data sets. The module description is used to create a computational experiment. Each executable module has one or more implementations. Different implementations for different resources can be used. In runtime the system selects the most suitable resource for each job. Then system runs appropriate implementation of the executable module on the selected resource.

The system keeps a library of described executable modules and a number of implementations for each module. This allows to reuse an executable module in different experiments. The set of the module implementations can be changed in the case of changes in the set of computational resources.

Once described a computational experiment can be run many times. Description of the input and output data sets does not refer to file and directory names. This allows to run experiments with different input data sets, which have different file names and different numbers of data sets. GriCoL offers its own language elements to describe data communication in the experiment. Such description weakly depends on real runtime data and number of data sets. This approach allows to run a once described experiment with an undefined number of data sets. Also data sets can be parametrized with a set of defined variables.

GriCoL supports the description of nested experiments. The number of nesting levels is unlimited. This allows to

use simple experiments as a part of a more complicated experiment.

III. WORKFLOW ENGINE IMPLEMENTATION

The workflow engine is a part of a SEGL system server. The system server is installed in the local network of the organization. It has the following tasks: running of the workflow engine, communication with system agents to run computational jobs and handling of client requests. The server keeps all required information in a relational database. This is information about such entities as computational resources of the organization, executable modules, implementations of the executable modules, experiment models and system users (Figure 3).

Detail descriptions of the computational resources are kept in a format of GLUE 2 [10]. Descriptions of the executable modules and the module implementations are kept in a modified format of JSDL [11].

Selection of the computational resource to run a job depends on the following factors: whether there is an appropriate module implementation, whether the resources are accessible for the user and whether these resources are operable at the moment. It also depends on the current load of the resource and the optimal value of the different benchmarks for the given job [12]. Each resource has a set of the predefined benchmarks values. These values indicate efficiency of the resource for different tasks. And the implementation of the executable module keeps information about the most important benchmarks. This set of the most important benchmarks is ordered in relevance to the computational task. Correlation between resource benchmark values and the list of task benchmarks allows to select the best resource from the set of available resources.

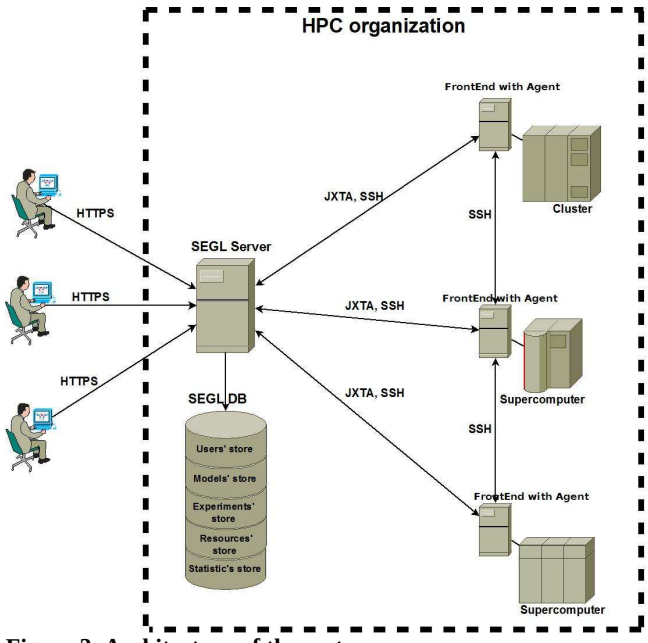


Figure 3. Architecture of the system

The system stores all runtime information about the started experiment in the database. The following information is stored: the selected resource for each job, the location of the input and output data on the resource, if the job finished with success or failure and the cause of the failure. This information helps to analyze the experiment later. In addition there is the possibility to continue an experiment which was interrupted.

Jobs are started on the resource by the system agents. The system agent is a service that is started on the front end machine (Figure 3). The agent starts and monitors jobs. After job completion the agent analyzes the output data of this job. The agent is controlled by commands of the server. The agent is also responsible for data transfer between resources.

The system has an authorization mechanism, a so called Access Control List (ACL). Users can have different access rights for such system objects as executable modules, experiment models, experiments.

Use cases of the system can be as follows:

- Some user creates an executable module and a set of its implementations. Then this user can grant access to edit and/or use this module.
- Some user creates a model of an experiment with the help of before described modules. Then this user can grant access to edit this model or to create new experiment using this model.
- Some user starts an experiment. Then this user can grant access to monitor this experiment or to get output data.

A graphical client of the system is a desktop application. It is installed, started and updated by Java Web Start framework [13].

User is authenticated in the system by login and password or by X.509 certificate. The system keeps information about allowed resources for the user and about the user accounts on these resources.

Using the client the user describes executable modules, creates experiment models, starts experiments and gets results of the experiments. The client comunicate with the server via HTTPS.

IV. EXECUTING OF THE EXPERIMENT AND RUNNING OF THE JOBS

Before starting the experiment the user has to specify the location of the input data. The data can be located on the user workstation or on the computational resource. In the last case the user has to have read access to these data. If the data is located on the user workstation, it is transferred from the client to the required resources through the server. Also the result data can be stored at the resources or can be transferred to the user workstation with the help of the client. The first case is more preferred when the data has a large size, for example several terabytes.

The system supports working with parallel file systems, such as Lustre [14]. User can allocate named large disk

spaces at the resources, so-called workspaces. The SEGL system allocates only one workspace for each experiment on each resource. As a rule, workspaces have a limited life time. This lifetime can be extended by mechanisms of the parallel file system (limited or unlimited number of extending times, it depends on the file system and its settings). Also extending can be implemented as a system function.

During workflow execution, the server selects a computational resource for each job and then the server transfers the input data of the job to the selected resource. The data is stored on the resource under the account of the user who starts the experiment. This is implemented in the system as described below.

Each user of the system has to add a server public key into a list of the authorized keys (~/.ssh/authorized_keys). After that the server can connect to some front end via SSH [3] under the account of the user (Figure 4). The server transfers the input data to the resource and gets the output data of the job via SFTP protocol.

To submit the job to the queue, to monitor the job state and to analyze the result data of the job the agent is used. The agent is started on the front end (Figure 4). To execute some commands, the agent creates a SSH connection from the given front end to the given front end under the account of the user. To do it, agent certificate must be added into the list of the authorized certificates. This will be executed by the server automatically.

Communication between the server and the agents is based on a P2P protocol JXTA [15]. HTTP is used as transport layer for JXTA. If necessary, the TLS (SSL) protocol with a mutual authentication can be used.

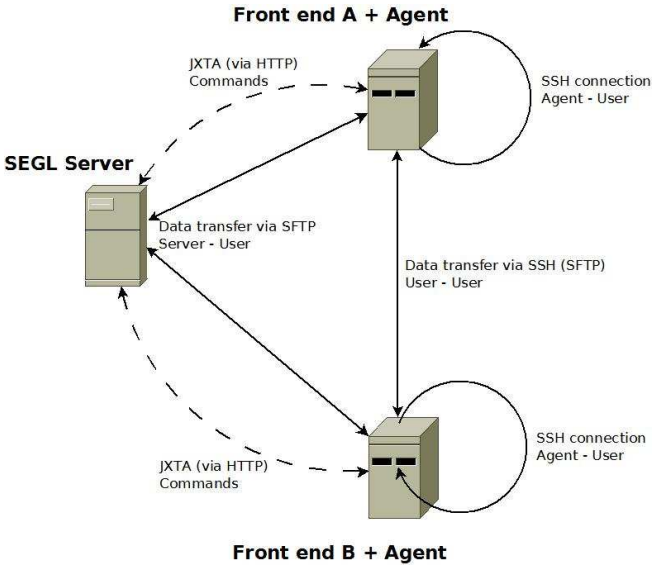


Figure 4. System communications

A data transfer from one resource to another is done via SFTP. The agent of the destination resource receives a command from the server. This command contains such information as the source resource, the required data, the user name on the source resource and the user name on the destination resource. The agent of the destination resource creates a local SSH connection under the user's account on the resource. We have selected this approach, because we have some limitations or special conditions which will be described below (see V). Then this agent starts the data transfer from the source resource via SFTP.

To start a job, the batch system of the resource [1] is used. The job is submitted to the queue as a script. Then the job will be started under control of the batch system.

The maximum number of jobs in the queue per user is limited. However, many jobs must be started simultaneously in the scope of the experiment. Also each user can start a several experiments simultaneously. To solve this problem the SEGL system has its own internal queues. These queues are part of the workflow engine. There is one internal queue per user and resource in the system. This queue is used for all experiments of the given user on the given resource.

At the development stage of the system we have found one more possibility to solve the problem of the resource queue limitation. HPC resources have a lot of nodes and processors. Most jobs use only part of these processors. The user has to specify the number of cores or processors for the job. This number can be defined as a special parameter of the job submitting script. However, the maximum number of jobs per user in the queue is independent on the number of used cores/processors. If you make a common script for a few jobs, you will be able to start multiple real jobs as one from the viewpoint of the queue. In this case the number of real jobs can be larger. If a user starts jobs manually, he or she has to make such scripts manually. In the SEGL system this action can be done automatically. The system will be able to generate such scripts merging jobs of different experiments of the same user.

V. Special Conditions

A significant requirement to the SEGL system is not to require changes to the infrastructure of the HPC organization. The agents which are started on the front end machines do not have any specific privileges in comparison with other users. This means that a simple user account for the agent will be created identical to the accounts of all other users. This user account does not have root privileges and it does not belong to any privileged groups. It cannot use SUDO command [16]. Note that using root privileges or SUDO command often is impossible, because it may affect security of the organization.

The system stores the input data and starts the jobs on the resources under the account of the end user. Therefore all limitations and policies of the organization will be applied to this user. A billing of used resources and disk spaces will be made for this end user too.

Since the system performs all actions on the resources under the account of the end user, it is not necessary to make any changes in the user accounting system of the organization. All security policies, billing principles and user group policies are remaining the same.

To start working with the system you need to have your own user account in the system. All your resource accounts must be added to (described in) the SEGL system. Then the server public key must be added into the list of your authorized keys. That is all. If user wants to stop working with the system, he or she has to remove server certificate from the list of the authorized certificates.

VI. Conclusions and Future Work

This paper describes a workflow system called SEGL that was developed to work in a real HPC organization without any infrastructure changes in such organization.

The SEGL is a high-performance and modular system that provides on the one hand, a convenient way to start complex computational experiments and on the other hand, possibilities to increase quality of HPC services. The implementation of the system is based on the well-known technologies, which are often used in HPC organizations.

There are several directions for future research. First of all, we want to improve system performance in handling large data files and large number of the data sets. Also we would like to improve the user interface of the client.

The system is installed at the High Performance Computing Center Stuttgart (HLRS) and is used to start engineering, biological and physical simulation experiments.

Within the scope of the bwGRiD Portal Project [17] the system will be integrated as a part of the HPC Web Portal. This portal will work within the scope of a VO that has a number of the HPC resources with different middleware systems. The workflow engine will be integrated with the Grid Application Toolkit (GAT) [18]. Also the system will be changed to work under conditions of the VO. Possibly the client will be implemented as a Rich Web Application.

We plan to implement automatic generation of merged scripts for multiple jobs to solve the problem of queue limitations.

In the future we would aim to develop complex planning of the experiment execution. To plan the experiment execution, the system will collect, keep and analyze detail statistical information about all resources of the organization.

References

[1] TOP 500, http://www.top500.org

[2] Byun, C., Duncan, C., Burks, S., “A Comparison of Job Management Systems in Supporting HPC ClusterTools”, Presentation for SUPerG, Vancouver, Canada, 2000.

[3] OpenSSH, http://www.openssh.com/

[4] Taylor, I., Deelman, E., Gannon, D., Shields, M., “Workflows for e-Science”, Springer Press, 2007 (ISBN: 1-84628-519-4).

[5] Foster, I., Kesselman, I., Tuecke, S., “The Anatomy of the Grid”, 2001.

[6] Baker, R., Yu, D., Wlodek, T., “A Model for Grid User Management, Computing in High Energy and Nuclear Physics”, La Jolla, California, USA, 2003.

[7] Antoni, T., Bühler, W., Dres, H., Grein, G., Roth, M., “Global grid user support – building a worldwide distributed user support infrastructure”, Journal of Physics: Conference Series, 2008.

[8] Dorozhko, Y., Krasikova, T., Yudin, Y., Currle-Linde, N., Resch, M., “An Abstract Language and Environment for the Creation and Execution of Experiments over Distributed Computers”, International Scientific Conference Simulation-2010, Kiev, Ukraine, 2010.

[9] Bouziane, H., Currle-Linde, N., Perez, C., Resch, M., “Analysis of Component Model Extensions to support the GriCoL Language”, In; Making grids Work, pp 45-59, Springer, 2008.

[10] GLUE Specification v. 2.0, http://www.ogf.org/documents/GFD.147.pdf

[11] Job Submission Description Language (JSDL) Specification, Version 1.0, http://www.gridforum.org/documents/GFD.56.pdf

[12] Armstrong, B., Bae, H., Eigenmann, R., Saied, F., Sayeed, M., Zheng, Y., “HPC Benchmarking and Performance valuation with Realistic Applications”, 2006 SPEC Benchmark Workshop (spec), 2006.

[13] Java WebStart Overview, http://www.oracle.com/technetwork/java/javase/overview-137531.html

[14] Lustre Cluster FS, http://www.lustre.org/

[15] JXTA Project, https://jxta.dev.java.net/

[16] sudo command, http://www.gratisoft.us/sudo/sudo.man.html

[17] bwGRiD Portal, http://www.bw-grid.de/portal/

[18] JavaGAT Project, https://gforge.cs.vu.nl/gf/project/javagat/

Nano-Science Gateway development with Vine Toolkit and Adobe Flex

Piotr Dziubecki[1], Piotr Grabowski[1], Michał Krysiński[1],
Tomasz Kuczyński[1], Krzysztof Kurowski[1] and Dawid Szejnfeld[1]
[1]*Poznań Supercomputing and Networking Center, Poznań, Poland,*
[deepres, piotrg, mich, docentt, krzysztof.kurowski, dejw]@man.poznan.pl

***Abstract*—In this paper we present the example domain-specific Nano-Science Gateway created using Java/Flex based framework called Vine Toolkit. Vine Toolkit, or simply Vine, is a modular and extensible Java and Adobe Flex library provided together with the high-level Application Programming Interface (API). The main aim of Vine is to accelerate the development of advanced web-based graphical user interfaces (GUI) and applications connected to underlying third-party services and Grid infrastructures managed by middleware, such as gLite, UNICORE, Globus, GRIA or QosCosGrid. Vine can be easily deployed in Java Web Start, Java Servlet 2.3 and Java Portlet 1.0 containers or used as a desktop application. It has been successfully integrated with well known portal frameworks, e.g. GridSphere and Liferay. We also compare in this paper leading web development technologies for sophisticated and interactive interfaces that can be used for rich web applications, namely Adobe Flex and Microsoft Silverlight. Currently, Vine supports advanced BlazeDS data services and takes advantage of Adobe Flex technology for interactive and dynamic client-server web applications. Main motivations of using Adobe Flex technology are also presented. Finally, we briefly describe our experiences with the development of a computational web portal for nanotechnology and nanoscience. Using Vine we successfully created a set of web-based graphical tools, data analysis, and visualization application around the ABINIT modeling and simulation software to meet scientists' requirements, in particular the access to collaborative large-scale simulation studies of systems made of electrons and nuclei on the basis of Density Functional Theory (DFT) and Many-Body Perturbation Theory.**

***Index Terms*—Science Gateway, Web2.0, ABINIT, Vine Toolkit, Liferay, GridSphere, Adobe Flex, Microsoft Silverlight, Material Science, Nanotechnology**

I. INTRODUCTION

VINE Toolkit was designed as an environment to facilitate the development and integration of web-based applications with HPC resources, Grid services and various existing large-scale computing infrastructures managed by Grid middleware, such as gLite, UNICORE, Globus, QosCosGrid and GRIA. In this paper we show how easily a computational web-based Science Gateway can be created using a modular structure and existing Vine components. Consequently, an easy-to-use presentation layer can be deployed together with various collaborative and visualization tools to simplify the way researchers, in this case representing the nanotechnology domain, perform intensive computing studies and share data between simulations via lightweight web interfaces. Vine together with a set of built-in modular components is an excellent solution to establish web gateways for advanced scientific and engineering applications with grid-enabled resources in the backend. Moreover, the heterogeneity of Grid services and HPC resources can be unified thanks to Vine APIs and built-in capabilities for remote job submission, monitoring and control as well as data and workflow management, security and user management. Thus, integrating existing Vine modules and adding application-specific extensions based on ABINIT software packages we were able to create a sophisticated Nano-Science gateway to support collaborative nanotechnology research. The architectural overview of Vine Toolkit has been presented in [1]. However, in this paper we focus on new Vine Toolkit functionalities that are relevant for advanced web-based applications.

II. RELATED WORK

Currently, there are several grid portal frameworks available that help users to create advanced and easy-to-use science gateways. P-GRADE is a good example of highly integrated parallel application development web-based system for Grid and clusters [2]. It uses Globus, Condor-G, ARC, BOINC and MPICH-G2 as grid-aware middleware to conduct computations**.** Another example is a collaborative environment where scientists can safely publish their workflows and experiment plans, share them with groups and find those of others, called myExperiment.org [3]. In this approach, workflows, other digital objects and bundles (called Packs) can now be swapped, sorted and searched

like photos and videos on the Web. Unlike Facebook or MySpace, myExperiment fully understands the needs of the researcher and makes it really easy for the next generation of scientists to contribute to a pool of scientific methods, build communities and form relationships - reducing time-to experiment, sharing expertise and avoiding reinvention. EnginFrame is a good example of a web-based front-end for simple job submission, tracking and integrated data management for HPC applications and other services [4]. EnginFrame can be easily plugged on several different schedulers or grid middlewares like: Platform LSF, Sun Grid Engine, PBS, or gLite middleware. Another approach to build an API that provides the basic functionality required to build distributed applications, tools and frameworks is SAGA [5] – it offers however only easy to use programming interface without any GUI support needed to create easy-to-use science gateway (see the Vine Toolkit and SAGA comparison table in figure 1). Moreover, thanks to Vine Toolkit native JSDL and BES support (both OGF standards[6]), the used in the prototype portal QosCosGrid middleware stack can be easily changed to any type of supported by Vine middlewares without changes in codes or even need to restart the portal. All changes can be done by simply changing the Vine configuration file, what can be done on-line by each user acting in administrator role. Moreover, the monitoring and control of tasks in the application is also working with other grid middlewares than the QosCosGrid if such functionality is provided in those cases.

Middleware	Vine Toolkit	SAGA – Java adaptors
gLite 3 - Cream	**Yes**	**Yes - JSAGA**
gLite 3 - WMS	**Yes**	**Yes - JSAGA**
gLite 3 - JDL	**Yes**	**under development - JSAGA**
Globus Toolkit	**Yes (4.0.x, 4.2.1)**	**Yes (up to 4.2) - JSAGA/JavaGAT**
Globus Toolkit – MyProxy	**Yes**	**Yes - JSAGA**
Globus Toolkit – gsiftp	**Yes**	**Yes - JSAGA**
Globus Toolkit - WS-GRAM	**Yes**	**Yes - JSAGA**
BES	**Yes**	**Yes - JSAGA**
JSDL	**Yes**	**Yes - JSAGA**
GRIA	**Yes (5.3)**	**No**
Unicore 6	**Yes**	**Yes - JSAGA**
Active Directory	**Yes**	**No**
Java Keystore	**Yes**	**Yes - JSAGA**
X509 Certificates	**Yes**	**Yes - JSAGA**
Storage Resource Manager	**Yes**	**Yes - JSAGA**
Storage Resource Broker	**(S)FTP, SSH, HTTP(S), ZIP**	**Partly (http, SSH applet)**	**Yes - JSAGA/JavaGAT**
local data management	**Yes**	**Yes - JSAGA**
WebDav	**Yes**	**No**
VOMS	**Yes**	**Yes - JSAGA**
iRODS	**Yes**	**Yes - JSAGA**
NAREGI (Super Scheduler)	**No**	**Yes - JSAGA**
OGSA-DAI	**Yes (2.2)**	**No**
RUS	**Yes**	**No**
QosCosGrid	**Yes**	**No**
GRMS,GRMS3	**Yes**	**No**

Figure 1. Vine Toolkit and SAGA comparison table.

In mentioned in this section approaches, access mechanisms to underlying services and computing infrastructures are hidden behind high-level web interfaces. However, they offer relatively simple graphical and visual interfaces as well as limited access control and management features. Moreover, they have been tailored to support specific Grid services and HPC systems, whereas Vine Toolkit is the most generic and advanced web-based framework integrated with Adobe Flex technology available to best of our knowledge.

III. ARCHITECTURE

Starting from the top of the Vine Toolkit software stack, it provides an efficient and robust user interface framework based on the Adobe Flex and BlazeDs software. Vine allows the integration of the rich internet application standard directly to a browser and enables applications to act and look exactly as their stand-alone versions. Thus, it is possible to create advanced portal applications like science gateways where developers can create web-based versions of many legacy applications and their GUIs. One of the key new requirements for Vine Toolkit regarding the integration with existing portal frameworks was to enable web application developers to create rich and advanced user interfaces as quickly as possible. Initially, we tried to use JS/AJAX-based frameworks within Vine Toolkit. Various problems related to the software portability in different web browsers encouraged us to migrate to other frameworks for the development and deployment of cross-platform rich Internet applications. Today, there are two main web application frameworks available that integrate visual interfaces, computer graphics, animation and interactivity into a single runtime environment: Adobe Flex [7] and Microsoft Silverlight [8]. Both frameworks provide runtime environments for most popular operating systems and web browsers. They offer a large set of user interface components and support various ways of communicating with web servers. However, there are many differences between them in terms of functionalities they provide and the way they can be used by developers. For example, in case of Adobe Flex the web application development is done using MXML and ActionScript, while a Silverlight developer has to use any of the .NET family languages. This

makes Silverlight a natural choice for .NET developers. One the other hand, there are only two main development tools supporting Microsoft Silverlight, either Microsoft Visual Studio (proprietary) or Eclipse4SL (open-source). More development tools supporting are available for Adobe Flex, including: Adobe Flash Builder, FlashDevelop, FlexBean, Intellij IDEA, etc. Another important criterion is a license model, especially for open scientific applications and research environments. Both Adobe Flex SDK and BlazeDS were released under open source licenses (MPL and LGPLv3 respectively), whereas Microsoft Silverlight was released as a commercial software under the MS-EULA license. In Figure 2 we compare main technical features and key capabilities of both mentioned solutions.

Examined Feature	Adobe Flex	Microsoft Silverlight
Charts suport	Yes	Yes
CSS styles	Yes	No
Integration with JavaScript	Yes	Yes
Printing	Yes	Not directly
SDK availability	Yes, all platforms, open source (Flex SDK). Flash Builder (Windows, Mac Os), paid.	Yes, Windows (.NET), paid, Unix (Mono not compatible with the latest Silverlight)
Licence	Adobe Flex SDK: open-source (Mozilla Public License) BlazeDS: open-source (LGPL v3)	Proprietary MS-EULA
Languages	ActionScript, Mxml	C#, Visual Basic, XAML
Multi-threading	No	Yes (SL4)
Data Services	Yes (LifeCycle, BlazeDs)	Yes

Figure 2. Adobe Flex and Microsoft Silverlight comparison table.

Eventually, Adobe Flex was chosen for developing rich and advanced user interfaces in Vine Toolkit. The main reason was the fact that at the time of the project's inception, Microsoft Silverlight was far behind Flex in terms of functionality. Also, the licensing favoured Flex. Obviously, the presentation layer is a front end to various components and services provided by Vine Toolkit, but it is getting more and more important once advanced web applications are available. Thanks to a pluggable Vine architecture it is possible to extend its base functionality in a uniform way. For instance, at the beginning Vine Toolkit offered only a support for the Globus Toolkit middleware. Currently, it is possible to use the majority of leading middleware stacks: GRIA, gLite, UNICORE, QosCos middleware [9] and many other well-known standards, such as OGF JSDL, OGF OGSA-BES or OGF-HPC Profile. Technically speaking, a new service in Vine can be added by creating a separate project and implementing a set of predefined APIs. Then, after a proper configuration, it can be used transparently by the end user without any additional changes in the application code. Finally, Vine offers various deployment configurations including standalone mode, web service mode and more importantly a ready to use integration with portal environments and portlet containers, e.g. Gridsphere [10] or Liferay [11]. Therefore, with a single software stack it is possible to build a complex solution consisting of the services, portal and set of user-customized applications at once available as a web gateway. Vine was designed to work with well-known JSR-168 open standard and its reference implementation [12] and Tomcat web application container. Since version 1.1 Vine Toolkit also supports Liferay JSR-286 enterprise portal [13]. Consequently, Vine Toolkit gives its users a great opportunity for creating and delivering production-quality web environments as it covers major web-based development aspects, especially for scientific and computing portals.

IV. NANO-SCINCE GATEWAY

In this section we describe an example science gateway we have recently developed and deployed under the PL-Grid infrastructure project [14]. In fact, it was a joint research and development effort with researchers interested in collaborative, Web2.0 and large-scale simulation studies based on Density Functional Theory (DFT) and Many-Body Perturbation Theory. Thus, we selected a key software package called ABINIT [15] and created from scratch many new and advanced web-based applications around it. In a nutshell, the ABINIT simulation software package allows for solving problems like: finding the total energy, charge density and electronic structure of systems made of electrons and nuclei within Density Functional Theory (DFT), using pseudo-potentials and a planewave basis. ABINIT also includes options to optimize the geometry according to the DFT forces and stresses, or to perform molecular dynamics simulations using these forces, or to generate dynamical matrices, Born effective charges, and dielectric tensors. Excited states can be computed within the Time-Dependent Density Functional Theory (for molecules), or within Many-Body Perturbation Theory.

Despite of its many capabilities, ABINIT provides only command-line tools. Moreover, it requires from its users not only domain-specific knowledge, but also a lot of expertise in computer science, and experiences with specific data formats and structures. To hide the complexity and provide a web-based collaborative access to ABINIT we created many new rich web applications using Vine Toolkit and

Adobe Flex. Consequently, we are able to support the transparent web access for sequential and parallel execution of DFT codes deployed on HPC computing clusters available for users in the PL-Grid infrastructure. By providing basic and advanced modes we are able to support both experts and beginners during their simulation studies. Moreover, Nano-Science Gateway was successfully presented at the NANO 2010 workshop attached to the 4th National Conference on Nanotechnology [16]. During the workshop scientists could test our new portal solution solving on-line ABINIT tutorial examples on the real HPC testbed without the need to access remote computing clusters using SSH or file transfer protocols. Users were using web browsers to perform all the computing simulations on remote machines. It was also a great opportunity to gather new requirements, e.g. tools to parse input/output parameters in the form-like panel for ABINIT and other DFT code, e.g. Quantum Espresso. Some parameters that were not reflected in the form could be now easily added using a new rich editor in the advanced mode. One should note that inputs needed for ABINIT job submission are pseudo-potentials files. Using the built-in Vine Toolkit file manager users can easily access, copy and assign appropriate files. It is also possible to store users files by taking the advantage of Vine Toolkit files repository as well as advanced access control mechanisms to share them according to defined policies.

After preparing a set of parameters and selecting steering parameters for DFT calculations, a user can add another set of parameters to be solved in parallel. When the whole experiment is ready (typically, it consists of several parameter sets) the user can send the experiment to a Grid middleware controlling remote computing clusters. In this approach, we used the QosCosGrid middleware stack together with a meta-scheduler services called Grid Resource Management System (GRMS). Therefore, we were able to make the underlying HPC resources fully transparent for end-users, so they could focus only on domain-specific interfaces and problems instead of struggling with computing infrastructure details, such as available processing power, deployed libraries or memory limits.

After the job submission the user can monitor all his/her simulations by simply checking the progress bars. Additional monitoring tool is a chart presenting the relative difference between subsequent computation iterations. If the user notes that the difference does not converge during experiment, he/she may decide to cancel this task in order to save the computational power. It is also possible to cancel the whole experiment – in this case all pending tasks (for corresponding parameter sets) are cancelled.

After each job completion the user can see generated results immediately in the portal. As it was mentioned above, all the ABINIT output files are stored in a Vine repository or can be transferred to remote file servers, e.g. GridFTP or more advanced Data Management Services, e.g. iRODS or DMS. The calculated functions of the total density of electronic states (DOS) line charts can be also displayed as visualization results. It is possible to view multiple series from different tasks on the same chart. The chart can be dynamically cropped and zoomed and special values like the Fermi energy can be also marked on the chart (if it was calculated in given case). For those applications we used Vine extensions based on Adobe Flex. The figure below shows the example experiment and it's visualization results.

Figure 3. Nano-Science Gateway – Abinit web client: Total energy calculation and calculated functions of the total density of electronic states (DOS) line charts.

All parameter sets and corresponding computing jobs together with their results are stored as cases for further use. This allows users to start or re-submit long term calculations from the portal, observe their progress after some time (possibly with another browser and session), view the results of completed experiments and reuse the previous experiments altering some parameters. To enable quick search of a certain example all parameter sets and previous cases can be annotated with keywords by users. Thus, it is possible to look for similar experiments descriptions and results using these tags by other users working on similar data sets. In this case, a special module for web searching is used with two data sources available: Google and ScienceDirect. The Nano-Science Gateway also consists of an additional general part showing and monitoring computing resources characteristics to provide a useful feedback for end-users, developers and administrators in different administrative domains where computing clusters or other HPC resources are located. To gather all presented metrics, various monitoring tests are invoked periodically

and their results are presented as graphical maps or diagrams in the portal using Adobe Flex applications. For example, thanks to the QosCosGrid middleware, results of bi-directional tests of cross-domain QCG-OMPI0 and QCG-ProActive0 applications measuring bandwidth and latency between clusters located in different administrative domains can be visualised. One of the most useful features of this part of the portal are Gantt charts showing local and cross-domain job executions as well as advance reservation and co-allocation of computing resources in time. Both monitoring tools are presented in Figure 4.

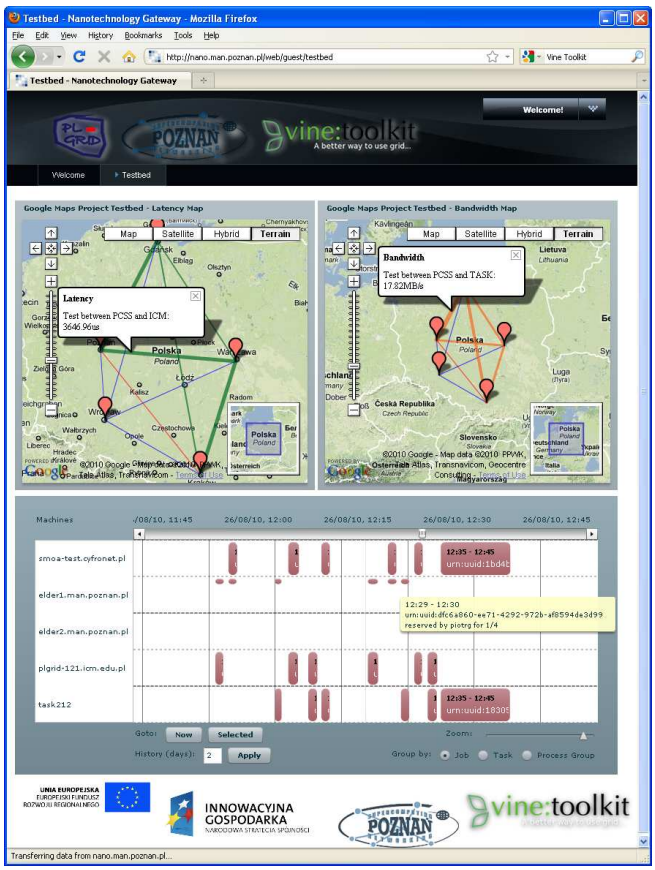


Figure 4. Nano-Science Gateway – Testbed information page showing network latency and bandwidth together with active advance reservations in HPC clusters available in the PL-Grid infrastructure.

V. FUTURE WORK

The future works on the Nano-Scince Gateway is divided into two parts: various extensions to the existing ABINIT web package client and support for another applications and packages. Many existing users identified several new applications that can be easily added to the gateway, e.g. Quantum Espresso or NAMD, and their integration will be a next step. Then, we would like to extend the gateway to support all major applications used for large-scale parallel simulation in nanotechnology, thus giving scientists possibility to make their advanced calculations in one collaborative web space. Some additional features will be added to incorporate popular social networking capabilities; some of them are currently available in Vine Toolkit and Liferay repositories. Social networking features will be used to improve data sharing and semantic descriptions of performed experiments and obtained results. Finally, we plan to add new visual components for interactive editing of nano scale structures and molecules using accelerated client-side libraries using graphical cards. Those new visualization features will support on-line rendering and management of complex structures displayed within our Nano-Science Gateway.

VI. CONCLUSION

In this paper, we briefly presented our approach to build Science Gateways using our Vine Toolkit integrated with the Adobe Flex software development kit. The presented web-based gateway supports various nanotechnology scientists in execution and management of large-scale simulations on computational grids using DFT code like ABINIT or Quantum Espresso. The described framework offers an effective way to start, monitor and control scientific applications via web based user-friendly graphical interfaces. Other Vine Toolkit based gateways were successfully developed and adopted in several other communities and R&D projects like QosCosGrid, BEinGrid, Omii-Europe or HPC-Europa. Currently Vine Toolkit is being enhanced and tested under the national Polish Grid Infrastructure (PL-Grid) project and is planned to be deployed in production environments.

ACKNOWLEDGMENTS

The authors would like to thank all people whose work allowed us to provide the Nano-Science Gateway for the scientific community. The special thanks goes to Mr. Michał Hermanowicz and people at the Institute of Physics, Poznan University of Technology who provided a lot of useful feedback to user interfaces and generated main requirements.

This work has been funded by the PL-Grid0 project: contract number: POIG.02.03.00-00-007/08-00, website: www.plgrid.pl. The PL-Grid project is co-funded by the European Regional Development Fund as part of the Innovative Economy program.

REFERENCES

[1] Russell, M., P. Dziubecki, P. Grabowski, M. Krysiński, T. Kuczyński, D. Szjenfeld, D. Tarnawczyk, G. Wolniewicz, and J. Nabrzyski (2008). The vine toolkit: A java framework for developing grid applications. Parallel Processing and Applied Mathematics (2008), pp. 331-340.
[2] P-GRADE http://www.p-grade.hu/
[3] myExperiment http://www.myexperiment.org/
[4] EnginFrame http://www.nice-software.com/web/nice/products/enginframe
[5] SAGA, http:// http://saga.cct.lsu.edu/
[6] OGF, http://ogf.org/
[7] Adobe, Flex / BlazeDs software http://www.adobe.com/products/flex/
[8] Microsoft, Silverlight, http://www.silverlight.net/

[9] Kravtsov, V., Schuster, A., Carmeli, D., Kurowski, K. & Dubitzky, W. (2008), Grid-enabling complex system applications with QosCosGrid: An architectural perspective, in Proc. of The Int'l Conference on Grid Computing and Applications (GCA'08), Las-Vegas, USA.
[10] Gridsphere, http://www.gridsphere.org/
[11] Liferay, http://www.liferay.com/
[12] JSR-168, http://jcp.org/en/jsr/detail?id=168
[13] JSR-286 http://jcp.org/en/jsr/detail?id=286
[14] PlGrid Project, http://www.plgrid.pl/en
[15] ABINIT, http://www.abinit.org/
[16] The 4th National Conference on Nanotechnology, http://www.nano2010.put.poznan.pl/
[17] QCG OMPI and QCG ProActive, http://node2.qoscosgrid.man.poznan.pl/gridsphere/gridsphere/guest/complex/

Krzysztof Kurowski holds the PhD degree in Computer Science and he is leading now Applications Department at Poznan Supercomputing and Networking Center, Poland. He was involved in many EU-funded R&D projects in the following areas Distributed and High Performance Computing and Grids over the last few years, such as GridLab, inteliGrid, HPC-Europa, or QosCosGrid. He was a research visitor at University of Queensland, Argonne National Lab, and CCT/Louisiana University. His research activities are focused on the modeling of advanced applications, scheduling and resource management in networked environments. Results of his research efforts have been successfully presented at many international conferences and workshops.

Piotr Dziubecki received his M.Sc. degree in Computer Science from Czestochowa University of Technology in 2006. The same year he joined Poznan Supercomputing and Networking Centre, Poland where he works as a software analyst. Before joining PSNC Mr. Dziubecki was involved in the development and benchmarking of the parallel multiple sequence alignment software. Currently his main area of interest is interface design for grid–aware portals. Since 2006, until now Piotr has participated in a number of EU funded projects such as inteliGrid, OMII-Europe, and BeInGrid. He is an author or co-author of several papers in professional journals and conference proceedings.

Piotr Grabowski obtained his M.Sc. in Computer Science in 1998 at Poznań University of Technology, Distributed Computer Systems. After his M.Sc., he joined the programmers group at PUT and worked on mobile network protocol analyzers software for Siemens A.G. and Tektronix, Inc. Since 2002 he has been working in Poznań Supercomputing And Networking Center, Application Department. In 2002 he joined GridLab project and worked on access for mobile users. Since 2005, until now Piotr has participated in a number of EU funded projects such as inteliGrid, OMII-Europe, and BeInGrid and local initiatives on enabling access from Mobile Devices and Portal technologies. The main thrust of his current research work are Mobile Devices and e-Mobility, Web Services and Web technology. He is an author or co-author of several papers in professional journals and conference proceedings. Since 2010 he leads User Interface Laboratory at PSNC.

Tomasz Kuczyński holds MSc Eng degree in Computer Science (Software Engineering) from Czestochowa University of Technology. Since 2004 employed at PSNC. He was involved in many, both EU-funded and national research projects, such as GridLab, HPC-Europa, OMII-Europe, BEinGRID, ACGT to name a few. Led users' interface WP in the ClusterIX national project. Contributed to the GridSphere portal project. Currently he is taking part in PL-Grid national project. Main areas of expertise are web technologies and security. He is credited with discovery of numerous security vulnerabilities in portal software including Liferay, Apache Tomcat, Mortbay Jetty, Caucho Resin and others. All vulnerabilities discovered by Tomasz was verified and published by the US-CERT (Department of Homeland Security of the USA).

Michał Krysiński received his M.Sc. degree in Computer Science from Poznań University of Technology in 2008. Since 2006, when he joined PSNC, Michał has participated in a number of EU funded projects such as OMII-Europe, and ACGT. He is an author or co-author of several papers in professional journals and conference proceedings.

Dawid Szejnfeld Head of Laboratory of Portal Applications at PSNC, holds a MSc in Computer Science. He has been involved in many EU funded projects in the area of web technologies and web security like HPC-Europa, OMII-Europe, BEinGRID. Currently involved in HPC-Europa2 project and polish infrastructural project called PL-Grid.

A Grid Portal as an e-Collaboration environment powered by Liferay and EnginFrame

R. Barbera roberto.barbera@ct.infn.it[1,2], G. La Rocca giuseppe.larocca@ct.infn.it[2], R. Rotondo riccardo.rotondo@ct.infn.it[2], A. Falzone alberto.falzone@nice-software.com[3], L. Carrogu luca.carrogu@nice-software.com[3], E. Usai enrico.usai@nice-software.com[3] and N. Venuti nicola.venuti@nice-software.com[3]

[1]*Department of Physics and Astronomy of the University of Catania, Italy*

[2]*Italian National Institute of Nuclear Phyisics division of Catania Via S. Sofia 64, Catania, I-95123, Italy*

[3]*NICE srl, Via Milliavacca, 9, I-14100 Asti, Italy*

***Abstract*— Nowadays, e-Science uses sophisticated high level tools that allow scientists separated by long distances and belonging to different administrative domains to work on the same project. In other to let researchers to easily access all the services related to the new paradigm, Grid Portal are spreading and becoming very popular. An implementation case using Liferay and GENIUS/Enginframe technologies will be shown.**

***Index Terms*—Science Gateway, Grid Portal, e-Collaboration, e-Science, Grid Computing, GENIUS, EnginFrame, Liferay.**

I. INTRODUCTION

GRID technology and ICT (Information Communication Technology) based infrastructures have been instrumental for the adoption of the e-Science paradigm, allowing scientists and researchers to work together to solve complex interdisciplinary problems using geographically distributed computational and data resources which are referred to as e-Infrastructures.

However, one of the weak points of this new way of doing scientific research is related to its complexity and difficulty of use make the learning curve too steep. Many efforts have then to be made to hide the complexity embedded in "the Grid". Even the World Wide Web (WWW) was not so popular in the first times until Mosaic, father of present browsers, and the first versions of Netscape were developed and made available to the community of users. Their clean and easily understandable user interfaces really made the WWW accessible to everybody.

Similarly, Grid Portals are developed to allow users to access Grid services in the easiest way with the assistance of intuitive graphic user interfaces. Along the same line, the demand of a web portal where users belonging to the same Virtual Organization can collaborate paved the way to the creation of the so called Science Gateways.

In this contribution we will show the new solution proposed by INFN and NICE [4] to create a new e-Collaboration environment based on the re-engineering of GENIUS with EnginFrame 2010 and Liferay as underlying frameworks. In section 2 GENIUS, EnginFrame 2010 and Liferay technologies are presented. Section 3 shows the architectures of the new e-Collaboration environment being developed. Conclusions are drawn and future plans are outlined in section 4.

II. STATE OF THE ART

This paragraph will introduce all the elements that are involved in the design of the new e-Collaboration environments: EnginFrame and GENIUS (subsection A) and Liferay (subsection B).

A. EnginFrame framework and functionalities

Grid Portals are developed to provide an easy and intuitive way to access services offered by a Grid infrastructure. Researchers do not have to waste their time in learning new programming languages or command line tools in order to access Grid infrastructures, but use its services to address and resolve new scientific challenges. GENIUS [1][2], the Grid Portal jointly developed since 2001 by INFN [3] and the Italian web technology company NICE srl [4], has achieved these goals hiding the complexity of command line interfaces and fulfilling requirements of security referred both to user's data and transactions. The GENIUS portal is a general-purpose portal; this means that it basically exposes some services to address the needs of generic users and communities. With GENIUS users can securely access the Grid, submit single jobs, job collections and parametric jobs (including entire workflows that can defined by Directed Acyclic Graphs). Users can also interact, in a seamless way, both with remote files located on the Grid User Interface or on a remote File Catalog. Due to the modularity and flexibility of EnginFrame[5], which acts as a general purpose framework "behind the scenes", GENIUS can also be easily customized/adapted in order to address the

requirements coming from a specific application or VRC. From a technical point of view, the portal customization can be done by editing some SDF (Service Description File) based on the EnginFrame XML dialect. Each user can create his/her personal SDL files for the application that he/she wants to run on the Grid. Since 2001, when the first version was released, the portal has been tailored to allow users of different scientific domains to access the Grid and run their applications, monitor the status of their jobs, and retrieve their results seamlessly using a conventional web browser.
The current implementation of the GENIUS portal is based on the gLite middleware developed within the European EGEE project [6], but it can adapted to interact with other Grids and/or new VRCs using any kind of middleware. Acting as a simple and intuitive "gateway" to access the Grid, the GENIUS portal holds a tremendous dissemination power. This is why the portal is the official portal of the GILDA [7] t-Infrastructure running since 2004 to disseminate the Grid paradigm through tutorials and other diverse training events. The 3-tier architecture of the GENIUS portal is shown in Figure 1.

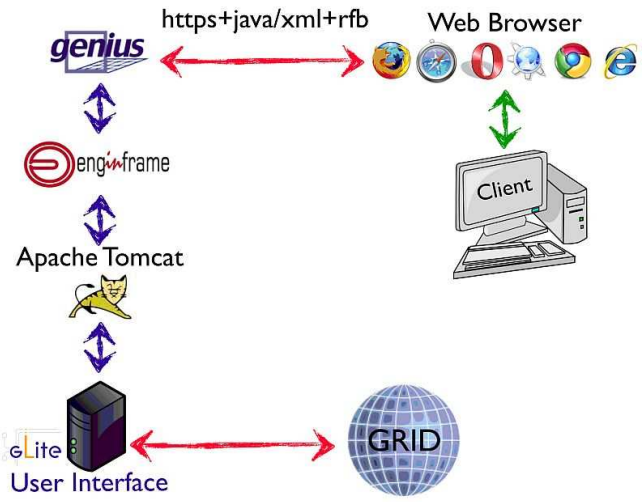


Figure 1. Three-tier model of the GENIUS Portal.

- The client (top right in the figure): it can be either a workstation (PC, notebook) or a hand-held device;
- The access protocols (centre top in the figure): users can use different protocols through their web browsers to access the presentation engine over web services;
- The server (bottom left in the figure): it is a User Interface (UI) machine (equipped with the gLite middleware [8] services to submit jobs and manage data on the Grid) which runs the Apache Web Server, the EnginFrame framework and the GENIUS services. The server block is made of:
    - The presentation engine for layouts and XSL/XML streams rendering, based on leading web standards, which provides access to underlying services via HTTPS, including HTML, SOAP and RSS;
    - The layer for the Authentication and ACL (Access Control List) enforcement responsible for the restriction of the view of services by different types of users;
    - The Data Management and Virtualization modules which provide an abstraction layer to access remote data sources and supports complete data life-cycles;
- The Application kits (centre left): they are part of the abstraction layer that hides the business logic of specific end-user applications and (centre right) provide other transversal services that allow VOMS Proxy authentication and the remote access to interactive graphic applications; applications are developed as EnginFrame plug-ins as well as GENIUS services themselves;
- The remote resources (bottom right in the figure): the Grid with its computational and storages resources.

The EnginFrame framework accepts the requests submitted by the user through the GENIUS portal and transmits them to a gLite user interface that can access directly the grid services.

B. Liferay portal framework

Close to Grid Portals develop, as GENIUS, Science gateway [9] are gaining popularity. A science gateway, according to Teragrid [10] definition, is a framework of tools that allows scientists to run applications with little concern for where the computation actually takes place. This is similar to cloud computing in which applications run as Web services on remote resources in a manner that is not visible to the end user. However, a science gateway is usually more than a collection of applications. Gateways often let users store, manage, catalog, and share large data collections or rapidly evolving novel applications they cannot find anywhere else. Training and education are also a significant part of some science gateways [11][12].
Many portal frameworks have been used in the recent past to create science gateways and one that have recently gained great popularity is Liferay [13]. It contains several portal management tools together with a content management system and a web content management, an enterprise service bus, all organized in a way following a service-oriented architecture. A Liferay portal is highly customizable thanks to the adoption of portlet technology defined in the Java Specification Request (JSR 168 and 286) and it is compatible with the most modern web applications.
Science gateways powered by Liferay implement high collaborative environments where users have many tools to share their works and results with their collaborators. It integrates social networking technologies that can aid

students and enable a community of scientist to share tools and ideas. Unfortunately, to analyze data, users often need to access grid infrastructures based on a given middleware and this is not currently possible out of the box with Liferay that is not equipped with tools to interact with grid services as EnginFrame framework.

III. A NEW E-COLLABORATION ENVIRONMENT

In this section we will show the solution proposed by INFN and NICE to create a new e-Collaboration environment [14] where the advantages of science gateway are merged with the benefits of grid portals. In subsection A we will explain environment's requirements, while in subsection B we will tell about environment's architecture and implementation.

A. Requirements

Although highly configurable, the current version of GENIUS implemented with EnginFrame is only a Grid Portal and lacks all the collaborative services provided by a standard Science Gateway.
An "ex ante" study of a new environment has produced the following list of requirements:

- Security and simplicity: Single Sign On (see Figure 2). Once user accesses the portal, he/she gains access to all his/her services and resources. System recognizes him/her automatically;
- Scalability: Portal performances are not influenced by numbers of user accessing the system;
- Web applications share the same environment. The new portal is realized merging two or more web applications functionalities. It is important to have a mechanism to share data or events with top security;

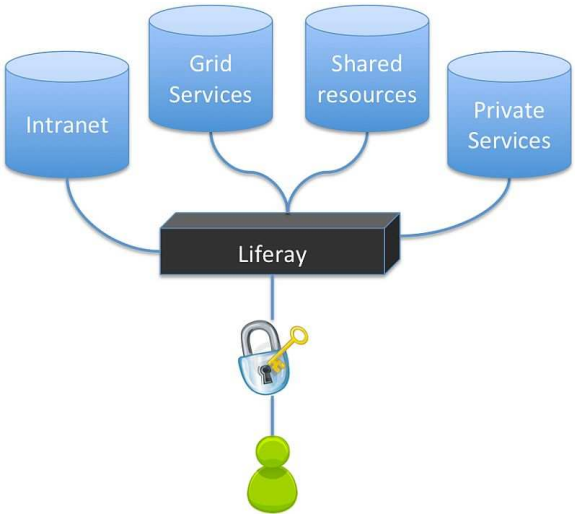


Figure 2. Single sign-on: users use their credentials only once to access all Grid service.

B. Architecture and Implementation

Currently, GENIUS is based on EnginFrame 4.1 and does not support Web 2.0 technologies. Its update with the new EnginFrame 2010's functionalities is under way. The new version will support portlets very much like Liferay or GridSphere [15] do and will ease the interactions among users belonging to a workgroup through the adoption of CMS/WCM technologies
In order to satisfy the new identified needs and allow users to share different kind of information offering, at the same time, the access to the grid resources of an e-Infrastructure, a new architecture is proposed and described in the following.
Referring to Figure 1, GENIUS is being replaced with a Science Gateway based on Liferay. In order to keep offering Grid infrastructure access, INFN and NICE have developed portlets, deployable not only into Liferay but also into any portlet container, such as GridSphere. Moreover, since portlets themselves are very versatile tools, they can be mixed together to realize complex custom applications based on simple Grid services. They can also simplify Grid usage too: a user can organize its personal page with only those portlets he/she needs for his/her study or research or project, without knowing how all available services work in the background. This solution surpasses the Grid Portal limitations but there is still a problem to solve, i.e. the interaction with the gLite User Interface and the fact that it is mandatory in the current GENIUS architecture (see Figure 1). In Figure 3 we show the solution we are working on. A new Grid Layer manages EnginFrame-Grid interaction hiding Grid services complexity even to EnginFrame and acting as an abstraction layer. This layer will be implemented using the APIs that are released with the middleware user interface and that let to create Grid applications without the need of a dedicated machine. The API functions will be organized in an abstract class so to have a very modular structure where only necessary methods will be implemented. This layer will also provide the correct environment that which is needed to the API to work properly.

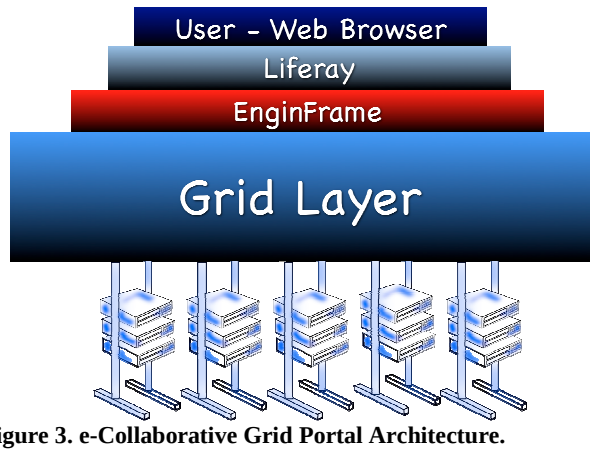


Figure 3. e-Collaborative Grid Portal Architecture.

Open new scenarios:

- At a higher level, designing a standard interface to the Grid Layer will allow third party applications to access the Grid services even without the need of Liferay and GENIUS/EnginFrame;
- At a lower layer, the new Grid Layer could be customized for middleware different from gLite such as VDT [16], used by Teragrid, or others.

Recently, the architecture of the EnginFrame Java/XML framework, on which the GENIUS portal is built, has been enhanced in order to simplify user authentication and authorization. The current state-of-the-art of the Grid security is based on the Public Key Infrastructure (PKI) of X.509 certificates and the procedure to deal with these certificates are unfortunately not straightforward especially for non-expert users. This is why, the high security policy requested to access the Grid has been a rather big limiting factor when trying to broaden the usage of Grids into a wide community of users. In order to address this kind of requirements, the EnginFrame framework has been enhanced with a transparent support to robot certificates[17][18]. Robot certificates have been introduced to allow users, who are not familiar with personal certificates and do not belong to any Virtual Organization (VO), to adopt the Grid paradigm in their research activity. The robot certificate is usually associated to a specific application (function) the application developer/provider wants to share with all the Grid community and can be installed in a smart card, or a USB key and used through a web portal. In order to strongly reduce the risks that the portal certificate can be compromised or accessed by non-authorized users, several CAs decided to issue these new certificates on board of the Aladdin eToken USB key. The additional features introduced in GENIUS are sketched in Figure 4.

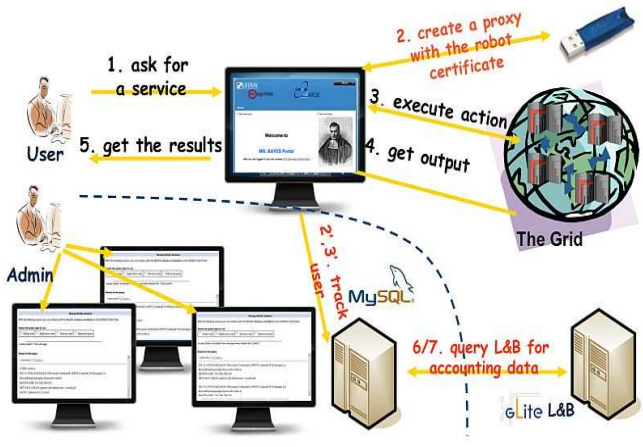


Figure 4. The new scenario of accessing a Grid infrastructure with the GENIUS Grid portal and robot certificate.

With these new features, every time users try to access the portal and the USB token is plugged on the UI the GENIUS server is running on, an automatic service drives them through the creation of a temporary proxy certificate. This user proxy is generated reading the credentials stored on the robot certificate installed in the USB key. Once the proxy has been successfully generated by the portal, users are automatically redirected to the home page of the application associated with the robot's function. Thanks to the new approach, users can access the Grid infrastructure without having a personal digital certificate or belonging to any specific VO. GENIUS Grid Portal.

EnginFrame2010 latest version presents a completely revamped user interface based on the latest Web 2.0 and AJAX technologies.

Currently, EnginFrame manage grid services so that portlets deployed in Liferay allow users to access grid functionalities sending commands to a gLite user interface through EnginFrame. Users, inside any instance of a Liferay portal, can submit job (see Figure 5) or upload/download file to a storage element (see Figure 6); in this case users access file on local machine, on user interface and on storage element in the same way.

Figure 5 . Portlet to submit a job for the MrBayes experiment.

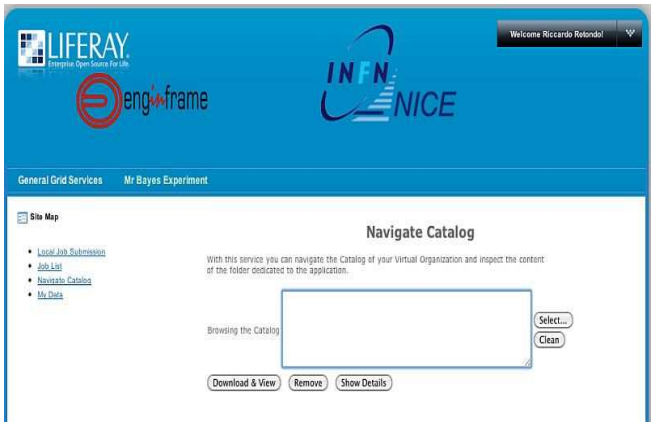


Figure 6. Portlet pre-configured to download a file from a storage element.

IV. Conclusion and Future Works

This work presents, through a concrete examples, advantages of using Grid Portals that enhance and ease the way to access and use the grid services provided by the e-Infrastructures.

A description of a e-Collaboration environment where a Grid Portal, implemented using a portal framework as Liferay that encourages working group and social network activities, has been made with the explanation of both the architecture and implementation. A Grid Layer is proposed as the solution to generalize the concept of Science Gateway decoupling it from the particular middleware deployed in the Grid infrastructure. Inside the portal, all out of the box tools embedded in Liferay will be available for user in other to let them to share their workflow and documents. Moreover collaboration will be provided too thanks to social utilities like instant messaging, blog and wiki available in any portal instance.

A realistic estimation of the time needed to design and implement the Grid Layer to access the gLite middleware services is of approximately two months. However, other middleware, such as VDT used by other grid users, could be also supported in a very easy way.

Other important features that had been implemented on the GENIUS portal will be provided as remote visualization using tightVNC [19].

Finally this framework will be used in two EU-funded Grid projects: DECIDE (life sciences) and INDICATE (cultural heritage).

References

[1] Andronico G, Barbera R, Falzone A, Lo Re G, GENIUS: a web portal for grid. Available at: https://genius.ct.infn.it Nucl. Instrument and Methods in Phy., 2003.
[2] Barbera R, Falzone A, Ardizzone V, Scardaci D. The GENIUS Grid Portal: Its Architecture, Improvements of Features, and New Implelemtations about Authentication and Authorization. Enabling Technologies: Infrastructure for Collaborative Enterprises, 2007.
[3] The INFN home page. Available at: http://www.infn.it/
[4] The NICE home page. Available at: http://www.nice-italy.com/
[5] The EnginFrame framework home page. Available at: http://www.enginframe.com/
[6] http://public.eu-egee.org/
[7] https://gilda.ct.infn.it/
[8] The gLite middleare. Available at: http://www.glite.org/
[9] Science Gateway home page. Available at: http://www.sciencegateway.org/
[10] Teragrid Science Gateway home page. Available at: https://www.teragrid.org/web/science-gateways/home
[11] Nancy Wilkins-Diehr, Dennis Gannon, Gerhard Klimeck, Scott Oster, Sudhakar Pamidighantam: TeraGrid Science Gateways and Their Impact on Science. IEEE Computer 41(11): 32-41 (2008).
[12] Nancy Wilkins-Diehr: Special Issue: Science Gateways - Common Community Interfaces to Grid Resources. Concurrency and Computation: Practice and Experience 19(6): 743-749 (2007).
[13] Liferay framework home page. Available at: http://www.liferay.com/
[14] R. Barbera, G. La Rocca, R. Rotondo, A. Falzone, P. Maggi, N. Venuti: Conjugating Science Gateways and Grid Portals into e-Collaboration environments: the Liferay and GENIUS/EnginFrame use case. http://doi.acm.org/10.1145/1838574.1838575
[15] GridSphere framework home page. Available at: http://www.gridsphere.org/gridsphere/gridsphere
[16] The Virtual Data Toolkit middleware. Available at: http://vdt.cs.wisc.edu/components/vdt.html
[17] Barbera R, Donvito G, Falzone A, La Rocca G, Milanesi L, Maggi G, Vicario S. The GENIUS Grid Portal and robot certificates: a new tool for e-Science. BMC Bioinformatics 2009, 10(Suppl 6):S21 doi:10.1186/1471-2105-10-S6-S21.
[18] "GENIUS Grid Portal and robot certificates to perform phylogenetic analysis on large scale: an experience within the Italian LIBI Project" – Prof. BARBERA Roberto, Dr. DONVITO Giacinto, Dr. FALZONE Alberto, Dr. LA ROCCA Giuseppe, Prof. MAGGI Giorgio Pietro, Dr. MILANESI Luciano – Managed Grids and Cloud Systems in the Asia-Pacific Research Community – Lin, Simon C., Yen, Eric (Eds.), 2010, XII, ISBN 978-1-4419-6468-7
[19] TightVNC homepage http://www.tightvnc.com/

The MoSGrid Gaussian Portlet – Technologies for the Implementation of Portlets for Molecular Simulations

Martin Wewior[1], Lars Packschies[1], Dirk Blunk[1], Daniel Wickeroth[1], Klaus-Dieter Warzecha[1], Sonja Herres-Pawlis[2], Sandra Gesing[4], Sebastian Breuers[1], Jens Krüger[3], Georg Birkenheuer[3] and Ulrich Lang[1]

[1]*University of Cologne, Germany*
{wevior|packschies|d.blunk|wickeroth|breuerss|lang}@uni-koeln.de
[2]*TU Dortmund, Germany, sonja.herres-pawlis@tu-dortmund.de*
[3]*University of Paderborn, Germany, {mercutio|birke}@uni-paderborn.de*
[4]*Eberhard-Karls-Universität Tübingen, Germany, sandra.gesing@tu-tuebingen.de*

***Abstract*—The development of a portlet for the MoSGrid (Molecular Simulation Grid) web portal is described. This portlet enables scientists in the field of Computational Chemistry to perform quantum chemical simulations on remote High Performance Computing platforms through a standards-compliant web browser. Since this portlet has a prototype character for further developments in MoSGrid and possibly beyond, the technologies used for its implementation were thoroughly evaluated and establish a common platform for future work.**

I. INTRODUCTION

COMPUTATIONAL CHEMISTRY has established itself as a new discipline in the natural sciences. Akin to other branches in chemical research, e.g. synthesis or analytics, the community has agreed upon particular standards and techniques. Consequently, the number of well recognized computation suites is relatively small, but their practical algorithmic correctness is commonly accepted.

Chemistry *in silico* can, for example, predict the exact geometry of small molecules and their interaction with electromagnetic fields. It helps to understand the misfolding of proteins (large, complex molecules comprised of amino acids) as the possible cause of diseases such as BSE (mad cow disease) or Alzheimer's disease. Furthermore, it allows to inspect the interaction of potential pharmacophores with proteins in order to develop a more efficient treatment for various human diseases.

With High Performance Computing (HPC) infrastructures available at most universities and research facilities, the number of researchers tempted to support their experimental results with further suitable calculations rapidly increases.

The enormous capabilities of the available software packages, however, come with a price. Written by specialists for their likes, the programmes were optimized for performance and versatility, while user experience hardly played any role. As a result, computational suites mostly are pure command-line tools or exhibit graphical user interfaces which were developed focussing on functionality and less on design guidelines.

The MoSGrid[2] (Molecular Simulation Grid) research project1 intends to provide a standards-compliant, functional and extensible environment for the execution of molecular simulations on remote HPC facilities.

Ideally, such an environment provides guidance for the inexperienced user without restraining the experienced researcher aware of all advanced options of molecular simulation packages.

Preferably, the user interface has the look and feel of modern desktop applications, is platform independent, does not involve any local installation, and would only require a modern web browser.

In principle, all these needs are met by web applications. However, MoSGrid intends to provide the same look and feel for the submission of compute jobs, the monitoring of progress, and the retrieval of (post-processed) results for of several different molecular simulation codes computed on various different clusters.

While the latter challenge is addressed by the introduction of an additional abstraction layer, namely the UNICORE [1] middleware, the further requirements call for a modular approach, in which the interfaces for the different molecular simulation codes are realized as independent portlets on a common portal server.

II. RELATED WORK

The MoSGrid project aims to establish a platform which can be used by both experienced and inexperienced researchers to submit their molecular simulations, monitor the progress, and retrieve the results. Moreover, it is projected as an extensible framework that allows for the stepwise implementation of further molecular simulations as well as pre- and post-processing routines, if requested.

At the present day, the submission of compute jobs to HPC facilities is typically performed through other pathways: Most frequently, resources are directly accessed via remote shells. The input files, together with specific job

[2] jointly funded by the German Federal Ministry of Education and Research (BMBF, reference 01IG09006) and the German Grid Initiative (D-Grid); http://www.mosgrid.de

scripts are submitted to batch systems, e.g. the TORQUE Resource Manager [2] and the MAUI Cluster Scheduler [3]. These allow job monitoring and fine-tuning, such as requesting of memory resources. Alternatively graphical interfaces can be found, either in the form of applications, which have to be installed to the end users' PCs, or as web based front ends.

When the submission of compute jobs to different clusters in a heterogeneous grid is intended, middlewares, which establish an abstraction layer to account for all peculiarities of the compute clusters involved, become a necessity. UNICORE (Uniform Interface to Computing Resources) [1] is a typical representative of these middlewares. It provides access either through a command line client (UCC – UNICORE command line client) or through different graphical interfaces, such as the Eclipse-based [4] UNICORE rich client. The clients allow to monitor and manage jobs and storages on the grid, and facilitate access to application data, e.g. the results of molecular simulations.

The UNICORE rich client was also utilized as the user interface in the EUROGRID project [5]; a plug-in written in Java allowed for the submission of Gaussian [6] jobs to a grid. In the past, several projects were devised to provide web based access to grid resources.

The CASPUR[3] reported an experimental setup for the submission of Gaussian jobs to a grid [7]. The ChemPo [8] web portal and the InSilicoLab [9] represent two other approaches and were developed in the context of the EGEE [10] project.

The Chemomentum [11] project, strongly related to the *REACH*[4] legislation framework of the EC [12], aimed to provide specific tools based on the UNICORE rich client.

A portal for computational biochemistry was developed within the G-Fluxo project [13], hosted at the Supercomputing Center of Galicia (CESGA). The P-GRADE portal container [14] and its embedded workflow support is utilised. JSR 168-compliant [15] portlets, together with the integration of the Java-based Jmol [16] applet allowed for a complete GROMACS workflow with subsequent visualization of the results.

Another implementation for GROMACS[17] was reported from the EELA2 project [18]. This project uses the GENIUS portal [19] which was developed to be the standard graphical interface for the EGEE [10] project. Plans were announced to integrate with this web portal for easier access.

III. MoSGrid Portlet

A. Basic Technologies

Web portals offer a very convenient way for users to access services. There is no requirement to install software and the service is available wherever a web browser can be connected to the internet. Furthermore, portals accumulate functions for different, but connected tasks.

Portlets are small Java based web applications. These application can be easily combined with other portlets inside a portal – a graphical user interface container like Liferay [20]. The flexibility gained by this approach allows writing small portlets for specialised tasks and combining them to larger applications. Additionally, the full integration with the Java language allows developing sophisticated applications and backend logic.

Currently, there are two standards connected with the development of portlets, their interfaces and interoperability. JSR 168 [15] is the basic portlet standard which is extended in JSR 286 [21] amongst others by an improved inter-portlet communication.

A user action, such as clicking a button, triggers a request being sent to the server. The server's response is rendered on the user's browser after receiving it. This follows the request-response-scheme of static web pages. To improve the visual appearance and especially to overcome this restriction of HTML, Javascript [22] can be used. Javascript is a scripting language which is executed in the local web browser. It is used to manipulate the Document Object Model (DOM), which basically represents the website and the browser's state. This allows design elements which dynamically change their appearance without requiring communication with the web server. Furthermore, the communication with the server can be done independently from user actions. This technology is known as AJAX (formerly asynchronous JavaScript and XML). Information can be retrieved from the server and inserted into the webpage without requiring the user to wait for complete page swaps.

A typical web-based email application may be used in the following way: The user sees a list of mails and an empty text area. As soon as the user clicks on one mail, the content is loaded from the server and inserted into the text area. Meanwhile, the folder structure or the list of mails, which the user sorted by date before stays untouched. No page refresh is required.

Portlets generate HTML fragments from Java code. These fragments are embedded and displayed in the portal. However this method lacks separation of logic and layout. Apart from that it requires designing the look of the interface and the components from scratch.

Using the Java Server Faces (JSF) technology [23] eases designing the websites using standardised components. The layout and design is defined in separate files. The output of the components is generated by backing (managed) JavaBeans running on an application server. Java Server Faces would improve and speed up the development process, but the results would still be quite static web pages unless Javascript libraries would be integrated.

Another Framework is the Struts Portlet Framework [24]. It supports the Model-View-Controller

[3] Inter-University Consortium for the Application of Super-Computing for Universities and Research

[4] Registration, Evaluation, Authorisation and Restriction of Chemicals

design pattern and therefore the desired separation of user interface and programme logic. It extends the JSF approach with support for the portlet standard. So JSF can be used to describe the design, while the framework supports the generation of the portlet code. Javascript or AJAX are not directly supported.

In contrast, ICEfaces [25] is an application framework based on JSF and AJAX technologies. It supports creating portlets and servlets with extended JSF syntax and therefore the Model-View-Controller design pattern. AJAX technologies allow creating portlets which have a look and feel similar to desktop applications used by scientists over the last years.

The connection to the computational infrastructure is realised by using the UNICORE GRID-middleware [1] in version 6. This middleware provides means for transparently submitting simulation jobs to different clusters and collecting the results using consistent means. These services are provided by the client layer of UNICOREs three layered architecture. The second layer, the service layer, amongst others transfers files, manages jobs, sites and registrations, and authenticates users. The system layer contains the target system, i.e. the computing resource and the target system interface (TSI).

For the integration within MoSGrid, interfaces with the client layer or a library implementation of the client layer has to be found or implemented.

B. First Prototype

Modelling the electronic structure of molecules is one of the major domains in Computational Chemistry; the Gaussian [6] quantum chemical program package is one of the de facto standards in this field.

The Gaussian Portlet presented here was developed using a user-centric approach in collaboration with the chemical community (see Section IV).

The prototype is fully functional and maps a complete quantum chemical workflow. The use of web technologies allows easy access and there is no need for installing additional software. The architecture of the first prototype is depicted in Figure 1.

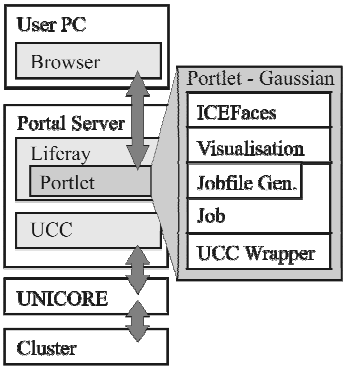


Figure 1. Architecture of the Gaussian Portlet.

The Gaussian Portlet resides in a Liferay portlet container on a portal server. Submission, monitoring and retrieval of finished simulations proceeds through a wrapper via the UNICORE command line client (UCC).

The user interface of the portlet is generated with ICEFaces. Single pages, called views, are described in separate files. Each view is backed by a JavaBean, which controls the visualization and the user input. The latter is handled by the "jobfile manager" Bean, and stored in an instance of the "Job" class. Information is processed inside the portlet and sent to the computational hardware using a wrapper class library.

Suitable parser programmes, currently launched by the portlet, extract relevant information from the simulation results. Their development, completely independent from the portlet, is beyond the scope of the article.

Further development and independent optimization of the project's modules will benefit from the layered architecture outlined above, e.g. the wrapper library currently used to utilise the UCC will be replaced by an improved connection layer to the UNICORE middleware without additional changes to the system.

IV. User Experience and Evaluation

THE described Gaussian Portlet convinces by its adaption to the demands of chemists who have no or few knowledge of informatics. The portlet offers two modes for the job submission: i) direct submission of a pre-defined Gaussian input file (.gjf or .com) or ii) assembling all required information in a graphical user interface (GUI) which is easy to use due to its clarity and resemblance to other popular GUIs. No special programme or shell has to be installed, the Gaussian Portlet can be accessed directly by standard web browsers which is a great advantage for users.

The direct submission is a fast and easy way to submit jobfiles into the grid. An input file conversion takes place behind the scenes. Thus, Windows users can directly submit their job files to GNU/Linux clusters without manual conversion. For the novice user, the GUI allows to select several computational methods, basis sets, functionals, charge, multiplicity, and many more options (e.g. mixing of HOMO and LUMO). After the definition of job type and details, molecular coordinates can be uploaded before the portlet generates the job file. The user may inspect and modify it, e.g. with additional keywords, before the job is submitted. The user is guided intuitively through the steps of the Gaussian job file generation.

After submission, the jobs appear with name and submission ID in a panel. The job status ("running", "failure" and "finished") is displayed as well.

After successful computation of the job, a button at the right hand side allows displaying and downloading the output file. As additional feature, the generation of adapted output files by targeted extraction of desired data is enabled. At the moment, several parsers are implemented which automatically extract relevant information of the Gaussian

output. In case of a frequency calculation, several parts of the output are presented to the user: i) the full output file, ii) vibration analysis, iii) thermo-chemical data like zero-point energy correction, and correction to enthalpies and Gibbs free energies. By this automated output extraction, the user gets an extremely fast overview of the most important results of the calculation, e.g. if the calculated structure is a minimum structure. Furthermore, the extraction of thermo-chemical data facilitates the further process of transforming the calculated electronic energies into ZPE corrected enthalpies of Gibbs energies. It is also possible to download the checkpoint files, which is sometimes desired by users.

Overall, this novel portlet provides a very comfortable user interaction and represents a promising development in the field of Grid applications. Novice users are guided through the jobfile preparation and submission process. Advanced users considerably profit from the targeted output extraction.

V. Conclusion and Future Work

MOSGRID will provide a web based portal for easily setting up, running, and evaluating calculations in the field of molecular simulations carried out on the German Grid (D-Grid) resources. In addition, MoSGrid will present data repositories containing the results of calculations, as well as repositories of "recipes" or workflows, that can be set up, used, improved and distributed by the users.

In this paper we present an advanced Gaussian Portlet prototype running in a Liferay portal environment.

The portlet enables novice and advanced Gaussian users to easily set up simple and even complicated Gaussian tasks and submit them to associated grids by the click of a button. The results are processed and information is extracted. Depending on users' requirements these post processing steps can be extended. Summarizing, the user does not necessarily i) have to know the input file format of the Gaussian quantum chemical software suite or ii) be aware of the computing infrastructure behind the scenes or iii) know the output format of the software suite.

The prototype was devised with the intention to evaluate technologies that will make further development of the Gaussian Portlet itself as well as the generation of portlets for other software products easier, providing a positive user experience at the same time.

The Gaussian Portlet has been shown to provide exactly that.

Future work on the portlet will include the addition of features contained in the Gaussian software suite, improving the user experience, e.g. by reducing the negative sensation of waiting cycles. In terms of the general goal of MoSGrid, the development of a GROMACS portlet has been started, based on the technologies presented here.

Making use of WS-PGRADE in the future will improve both the user experience as well as the scientific impact of the MoSGrid by enabling the user to develop, share and reuse workflows, which might include various pre-and post-processing options, as well as advanced schemes, involving sequences of grid-distributed calculations, merging of the results and resubmission of the data to other molecular simulation codes.

Acknowledgement

The generous financial support of the MoSGrid project by the BMBF (German Federal Ministry of Education and Research) – promotional reference 01IG09006 – and the German Grid Initiative (D-Grid) is gratefully acknowledged.

References

[1] UNICORE. [Online]. Available: http://www.unicore.eu/
[2] TORQUE Resource Manager. Cluster Resources Inc. [Online]. Available: http://www.clusterresources.com/products/torque-resource-manager.php
[3] MAUI Cluster Scheduler. Cluster Resources Inc. [Online]. Available: http://www.clusterresources.com/products/torque-resource-manager.php
[4] Eclipse. [Online]. Available: www.eclipse.org/
[5] B. Lesyng, P. Bała, and D. Erwin, "EUROGRID–European computational grid testbed," J. Parallel Distrib. Comput., vol. 63, no. 5, pp. 590–596, 2003.
[6] M. J. Frisch, G. Trucks, E. Frisch et al., Gaussian 03, Revision E.01. Gaussian, Inc., Wallingford CT, 2004.
[7] N. Sanna, T. Castrignano, P. D. De Meo, D. Carrabino, A. Grandi, G. Morelli, P. Caruso, and V. Barone, "Gaussian grid: a computational chemistry experiment over a web service-oriented grid," Theor. Chem. Acc., vol. 117, no. 5-6, pp. 1145–1152, MAY 2007.
[8] Mariusz Sterzel and Tomasz Szepieniec and Daniel Har e¸ ˙zlak. Current Status of Chempo Web Portal. [Online]. Available: http://indico.cern.ch/getFile.py/access?contribId=341&resId=1&materialId=slides&confId=55893
[9] [Online]. Available: https://insilicolab.grid.cyfronet.pl/
[10] EGEE. [Online]. Available: http://www.eu-egee.org/
[11] B. Schuller, B. Demuth, H. Mix, K. Rasch, M. Romberg, S. Sild, U. Maran, P. Bała, E. Del Grosso, M. Casalegno et al., "Chemomentum-UNICORE 6 based infrastructure for complex applications in science and technology," in Proceedings of the 2007 conference on Parallel processing. Springer-Verlag, 2007, pp. 82–93.
[12] REACH. European Commission. [Online]. Available: http://ec.europa.eu/enterprise/sectors/chemicals/reach/index en.htm
[13] E. Guti´errez, A. Costantini, J. L. Cacheiro, and A. Rodr´ıguez, "G-FLUXO: A Workflow Portal Specialized in Computational BioChemistry," in Proceedings of the International Workshop on Portals for Life Sciences, S. Gesing and J. van Hemert, Eds., Oct. 2009. [Online]. Available: http: //ceur-ws.org/Vol-513
[14] P-GRADE. [Online]. Available: http://www.p-grade.hu/
[15] JSR 168: Portlet Specification. [Online]. Available: http://jcp.org/en/jsr/detail?id=168
[16] Jmol: an open-source Java viewer for chemical structures in 3D. [Online]. Available: http://jmol.sourceforge.net/
[17] B. Hess, C. Kutzner, D. van der Spoel, and E. Lindahl, "Gromacs 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation," J. Chem. Theory Comput, vol. 4, no. 3, pp. 435–447, 2008.
[18] A. Ribeiro, "GROMACS," in Proceedings of the First EELA-2 Conference, R. Mayo et al., Eds., 2009.
[19] EGEE Genius Portal. [Online]. Available: http: //egee.cesnet.cz/en/user/genius.html
[20] Liferay. [Online]. Available: http://www.liferay.com/
[21] JSR 286: Portlet Specification 2.0. [Online]. Available: http://jcp.org/en/jsr/summary?id=286
[22] ECMAScript Language Specification. [Online]. Available: http://www.ecma-international.org/publications/standards/Ecma-262.htm
[23] Java Server Faces - JSR-314. [Online]. Available: http://jcp.org/en/jsr/detail?id=314
[24] Apache Struts. [Online]. Available: http://struts.apache.org/
[25] ICEFaces. [Online]. Available: http://www.icefaces.org/

Martin Wewior graduated from the University of Magdeburg with a degree in computer science (Diplom-Informatiker). He is a research associate at the Regional Computing Centre of the University of Cologne working in the fields of distributed computing and user interfaces. Having worked for the EC-funded project CoSpaces he now is one of the major developers of the MoSGrid Project.

Dr. Lars Packschies is one of the initiators of the MoSGrid Project. He is a chemist and holds a doctorate degree from the University of Dortmund (Max Planck Institute of Molecular Physiology). Since the year 2000, he works at the Regional Computing Centre of the University of Cologne and is in charge of the suites and programmes in the field of computational chemistry on grid resources and high performance computing environments.

Workflow Interoperability in a Grid Portal for Molecular Simulations

Sandra Gesing[1], István Márton[2], Georg Birkenheuer[3], Bernd Schuller[4], Richard Grunzke[5], Jens Krüger[6], Sebastian Breuers[7], Dirk Blunk[7], Gregor Fels[6], Lars Packschies[7], André Brinkmann[3], Oliver Kohlbacher[1] and Miklos Kozlowsky[2]

[1]*Ebarhard-Karls-Unisersität Tübingen, Germany,*
{sandra.gesing|oliver.kolhbacher}@uni-tuebingen.de
[2]*MTA-SZTAKI, Budapest, Hungary {imarton|m.kozlowsky}@sztaki.hu*
[3]*Paderborn Center for Parallel Computing, Germany {birke|brinkmann}@uni-paderborn.de*
[4]*FZ Jülich, Germany, b.schller@fz-juelich.de*
[5]*Technische Universität Dresden, Germany, richard.grunzke@tu-dresden.de*
[6]*University of Paderborn, Germany, {mercutio|fels}@uni-paderborn.de*
[7]*University of Cologne, Germany,*
{breuerss|blunk|packschies}@uni-koeln.de

***Abstract*—Molecular simulations are an invaluable tool in multiple research areas like chemistry, biology, and physics. The emerging MoSGrid (Molecular Simulation Grid) portal intends to integrate various molecular simulation tools in WSPGRADE, a workflow-enabled grid portal. The portal will therefore also support the execution of workflows using these simulation codes. UNICORE is a grid middleware with the additional feature of an integrated workflow engine. Here we present a tool to invoke subtasks of a WSPGRADE workflow in UNICORE and vice versa, to integrate existing UNICORE workflows in WSPGRADE workflows. Researchers are enabled to create and use both kinds of grid workflows without the need of becoming acquainted to the grid infrastructure or the workflow languages.**

I. INTRODUCTION

GAINING new knowledge about the behavior of new substances, understanding chemical reactions, and the design of effective drugs is complex and time-consuming work. Today's chemists are supported by molecular simulation tools that gather information about properties of molecules based on theoretical procedures. In cooperation with powerful computing infrastructures, these programs allow to gather information about increasingly complex chemical structures. This information saves a lot of time in research and drug design, as only the most promising substances have to be synthesized and analyzed.

However, computational chemistry has two major drawbacks. Firstly, the number of electronic structure methods and molecular mechanics and dynamics codes and their usability is often limited. Secondly, the complexity of the methods as well as the missing graphical user interfaces complicates their use.

Therefore, MoSGrid, the Molecular Simulation Grid project, develops a problem-driven portal to provide molecular simulation tools on the resources of the German D-Grid initiative. This portal supports new users in the process of creating chemical simulation jobs and allows advanced users to just import all molecular information they need and start powerful workflows following chemical recipes. For the creation and use of complex workflows, MoSGrid integrates the grid middleware UNICORE 6 and the WS-PGRADE portal.

The remainder of the paper is organized as follows. Sections II and Section III give a brief introduction to WS-PGRADE and UNICORE. Section IV describes the integration of UNICORE into the MoSGrid portal. Section V shows an exemplary use case and Section VI presents related work.

II. WS-PGRADE

The MoSGrid portal is designed as workflow-enabled grid portal developed on top of WS-PGRADE. In general, a portal can be defined as a web-based framework for integrating information and applications. It operates across organisational boundaries and as a single entry point for a community. Users can customise their tools and views and do not need to deal with the details of software installation and hardware configuration. A grid portal is a specific portal that utilizes grid infrastructures.

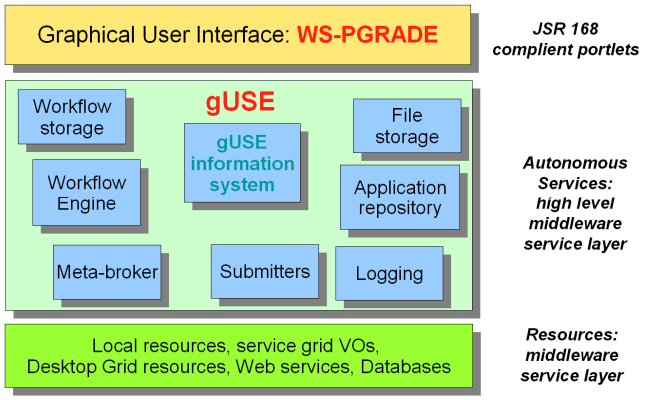


Fig. 1. WS-PGRADE and gUSE internal architecture.

WS-PGRADE is an easily usable, highly flexible, co-operative, graphical user interface for the grid User Support Environment (gUSE) [1]. It provides a collaborative, community-oriented application development environment, where developers and end users can share sophisticated (layered and parameter sweep enabled) workflows [2], workflow graphs, workflow templates, and ready-to-run workflow applications via a repository.

gUSE is a virtualization environment providing a large set of high-level Distributed Computing Infrastructures (DCI) services by which interoperation among classical service and desktop grids, clouds and clusters, unique web services, and user communities can be achieved in a scalable way. Internally, gUSE is implemented as set of Web services, which dynamically provide user services in DCI and/or Web services environments (see Figure 1).

WS-PGRADE uses the client APIs of gUSE services to turn user requests into sequences of gUSE specific Web service calls. WS-PGRADE hides the communication protocols and sequences behind JSR168 [3] compliant portlets and its GUI can be accessed via Web browsers.

III. UNICORE

UNICORE [4] is an integrated grid middleware system, that includes a full software stack from clients to several server components down to components for accessing the actual compute or data resources. Its fundamental ideas are abstraction of site-specific details, openness, interoperability, operating system independence, security, and autonomy of resource providers.

Furthermore, the software is easy to install, configure, and administrate. UNICORE is being deployed and used in a variety of use cases, ranging from small projects to large (multisite) infrastructures involving high-performance computing resources.

UNICORE has been developed in several German and European projects since 1997 [5]. Its current version UNICORE 6 is based on Web services, using many XML based standards such as JSDL, XACML and SAML.

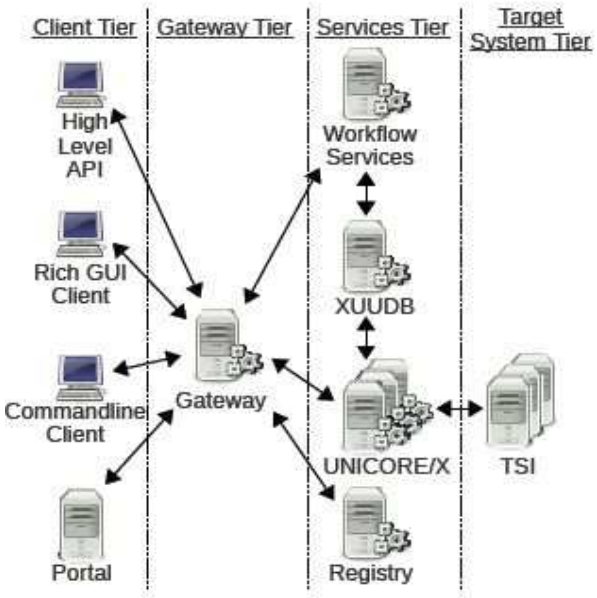


Fig. 2. UNICORE 6 architecture.

As can be seen in Figure 2, UNICORE consists of four tiers, namely the client, gateway, services, and target system tier.

A variety of clients exist, from programming interfaces[6], a command-line client[7], a powerful graphical client based on the Eclipse framework to portals as described in this paper.

The gateway is a thin authentication and routing component protecting the grid services behind it. The underlying services tier provides basic services such as resource discovery, job execution and storage access. as well as higher level services such as the workflow system.

The target system tier connects the services from the upper tier to the specific resources. It consists of the Target System Interface to communicate with the batch system, file-system, and local operating system.

The UNICORE workflow system consists of two major services: the workflow engine deals with workflow processing, while the service orchestrator combines a resource broker with a single job execution manager. The workflow engine offers powerful constructs such as sequences, loops, conditions and workflow variables. The default workflow description is a custom XML dialect, though the engine supports multiple workflow description

dialects through a plug-in architecture. Single jobs in the workflow are passed to a service orchestrator for brokering and execution.

IV. Integration of UNICORE into the MoSGrid Portal

The integration of UNICORE into the MoSGrid portal consists of two parts. One part is to add a so-called submitter (a Java-based application) to gUSE, the other part is to make the submitter and corresponding infrastructures available to the users in WS-PGRADE. Latter offers authentication with proxy certificates, which fits to the supported grid middlewares like Globus Toolkit and gLite. Currently the prototype of the MoSGrid portal also relies on X.509 proxy certificates converted into PKCS#12 keystores. The aim is to use SAML (Security Assertion Markup Language) avoiding that the user has to download a proxy certificate from a MyProxy server into the portal for the invocation of UNICORE jobs.

A. gUSE submitter for UNICORE

gUSE offers a grid service interface to integrate various high performance computing (HPC) facilities through grid middlewares via submitters. Submitters are used for single tasks of a WS-PGRADE workflow; the workflow itself and its dependencies are handled by the workflow engine of gUSE. This encapsulation of tasks allows to submit different tasks of one workflow to the local portal server, to diverse HPC facilities, and grid infrastructures. A submitter has to provide the following methods.

- actionJobSubmit - submission of a job including the data staging
- actionJobAbort - cancel a job
- actionJobOutput - get the output of a job
- actionJobStatus - query the status of a job
- actionJobResource - return the resource the job was submitted to

The gUSE submitter developed for the MoSGrid portal (submitter unicore) provides these methods based on the UCC libraries (Figure 3).

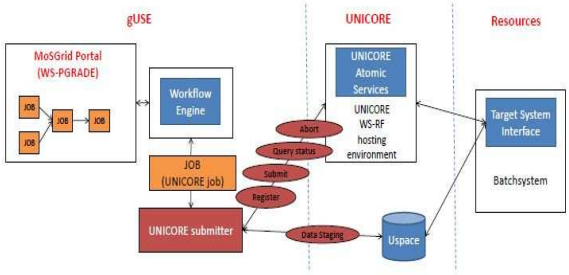


Fig. 3. Involved components for handling a UNICORE job.

The advantage of using the UCC libraries compared to the programming interface HiLA is that also UNICORE workflows can be processed via the UNICORE submitter. Instead of starting solely a single job, a workflow is invoked which is managed by the UNICORE workflow engine. UNICORE workflows appear to gUSE solely like a single job (Figure 4).

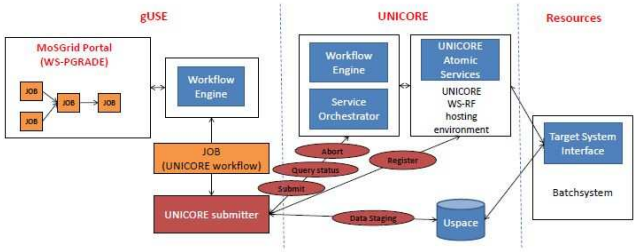


Fig. 4. Involved components for handling a UNICORE workflow.

Thus, users are enabled to re-use existing UNICORE workflows via the MoSGrid portal within a WS-PGRADE workflow. The portal presents information about the status of each task of an invoked WSPGRADE workflow. This also includes the status of UNICORE workflows.

B. Integration of UNICORE into WS-PGRADE

The integration of an existing submitter and corresponding VOs can be intuitively processed in WS-PGRADE. The portal enables administrators to easily add new VOs, grid infrastructures, and high-performance facilities as target systems for workflows via a portlet. Once these settings are stored in the portal, end users can choose from a list, which submitters they want to use for a task in a workflow. Their credentials are checked as soon as they want to submit a workflow to the chosen resources.

V. Use Case

Scientists are dealing with a broad range of questions and challenges when simulating molecular systems. One of the frequently repeating tasks is the preparation, minimization, and equilibration of a globular protein as prerequisite for production runs. This task (Figure 5) is presented as use case for Gromacs [8] in the following section.

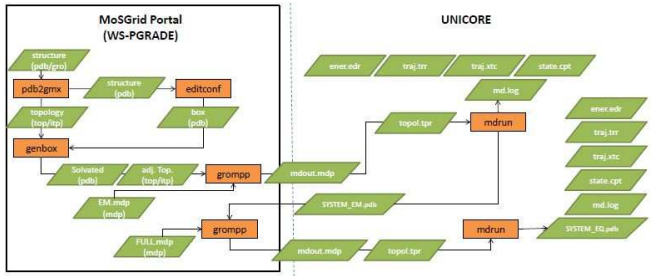


Fig. 5. Preparation, minimization, and equilibration of a globular protein.

The first task is to identify all atoms and bonds in order to generate a correct description of the atomistic interactions, a

so called topology (topol.top). This is accomplished using the Gromacs tool pdb2gmx.
All hydrogen atoms on the protein are adjusted according to the requirements of the force field (here a Gromos96 derivative 45a3) being used for the later simulations in order to achieve a reasonable protonation state. The resulting structure with force field consistent atom names is put into a simulation box using editconf.
The size of the box is chosen just large enough to avoid mirror effects by neighboring PBCcells. Afterwards the box is filled with SPC water using genbox. The topology file is updated accordingly. The precompiler grompp is executed in conjunction with a simulation description (EM.mdp) yielding the file topol.tpr including all coordinates, bonds, forces, and further information to carry out an energy minimization. This calculation is carried out by calling a MPI-parallel version of mdrun. The output of the minimization is directly taken as starting point for a brief equilibration run. Again the precompiler grompp is executed using a different simulation description (FULL.mdp). After 250 ps the last frame of the simulation is written to SYSTEM_EQ.pdb serving as starting point for production runs, docking studies, and/or further scientific analysis (Figure 6). This scenario is meant as a basis that will

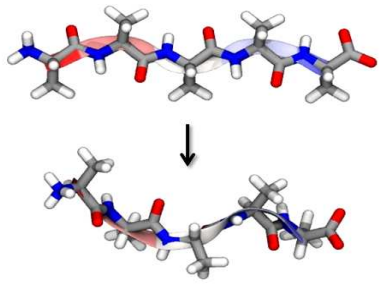


Fig. 6. A small penta-alanine was equilibrated by a 200 ps molecular dynamics simulation.

be enhanced in subsequent generations of the MoSGrid portal. By adding further options to the input masks that guide the user through the simulation setup process the (advanced) user will gain more control to adapt the workflow to his/her specific problem. The modularity that is achieved by subdividing the workflow into different tasks ensures the reusability of the subtasks and by means of this provides more flexibility for the user.

VI. Related Work

EnginFrame [9] is, similar to WS-PGRADE, a workflow-enabled portal which supports various grid infrastructures. The EU project AWARE (An easy Way to Access grid REsources) [10] already integrated UNICORE 5 into EnginFrame. The drawback of this portal for the MoSGrid project is the commercial license model for non-academic users, since the MoSGrid consortium consists of academic and industrial partners. Latter shall use the same portal as the academic partners under an open source license. The ProSim [11] project offers workflows and workflow templates for carbohydrate recognition in a portal on top of WSPGRADE with access to several grid infrastructures.
The projects MoSGrid and ProSim can benefit from each other. Users could re-use workflows of ProSim for sub-areas of molecular simulations in the MoSGrid portal and the ProSim portal could offer carbohydrate recognition in UNICORE 6 infrastructures. Another related project is G-FLUXO [12], which provides Gromacs workflows and visualization of the results with Jmol in a portal based on PGRADE, the first generation of WS-PGRADE. Since Gromacs is one of the molecular simulation tools MoSGrid will support, the workflows of G-FLUXO could be integrated into the MoSGrid portal. Galaxy [13] is an intuitive science gateway for creating and sharing workflows, especially for life sciences. However, Galaxy does not enable users to invoke workflows and to send them to grid infrastructures.

VII. Conclusion and Future Work

We presented MoSGrid's support for UNICORE 6 jobs and workflows in WS-PGRADE and demonstrated a use case with Gromacs. Users are enabled to create, to re-use, and to submit WS-PGRADE workflows to UNICORE 6, and vice versa, to invoke their existing UNICORE workflows via WS-PGRADE. At this stage, a UNICORE workflow forms one task in a WS-PGRADE workflow and is treated as a single job. To visualize the structure of a UNICORE workflow and to monitor the single tasks, we intend to offer a conversion of UNICORE workflows to WS-PGRADE workflows. Hence, the users have the same features in the portal for both kinds of workflows and do not need to distinguish between them.

Acknowledgement

The authors would like to thank the BMBF (German Federal Ministry of Education and Research) for the opportunity to do research in the MoSGrid project (reference 01IG09006).

References

[1] gUSE - grid User Support Environment. [Online]. Available: http://www.guse.hu/

[2] P. Kacsuk, K. Karoczkai, G. Hermann, G. Sipos, and J. Kovacs, "WS-PGRADE: Supporting parameter sweep applications in workflows," in Proc. of 3rd Workshop on Workflows in Support of Large-Scale Science, In conjunction with SC 2008, Nov.17 2008, pp. 1–10.

[3] JSR 168: Portlet Specification. [Online]. Available: http://jcp.org/en/jsr/detail?id=168

[4] UNICORE SourceForge project. [Online]. Available: http://sourceforge.net/projects/unicore

[5] UNICORE. [Online]. Available: http://www.unicore.eu

[6] R. Menday and B. Hagemeier. HiLA 1.0. [Online]. Available: http://www.unicore.eu/community/development/hila-reference.pdf

[7] UNICORE commandline client. [Online]. Available: http://www.unicore.eu/documentation/unicore6/manuals/ucc

[8] B. Hess, C. Kutzner, D. van der Spoel, and E. Lindahl, “GROMACS 4: Algorithms for highly efficient, loadbalanced, and scalable molecular simulation,” JCTC, vol. 4, pp. 435–447, 2008.

[9] L. Torterolo, I. Porro, M. Fato, M. Melato, A. Calanducci, and R. Barbera, “Building Science Gateways with EnginFrame: a Life Science example,” in Proceedings of the International Workshop on Portals for Life Sciences, S. Gesing and J. van Hemert, Eds., Oct. 2009. [Online]. Available: http: //ceur-ws.org/Vol-513

[10] A-WARE. [Online]. Available: http://www.a-ware-project.eu/

[11] T. Kiss and et al., “Parameter Sweep Workflows for Modelling Carbohydrate Recognition,” submitted to Journal of Grid Computing, 2010.

[12] E. Guti´errez, A. Costantini, J. L. Cacheiro, and A. Rodr´ıguez, “G-FLUXO: A Workflow Portal Specialized in Computational Biochemistry,” in Proceedings of the International Workshop on Portals for Life Sciences, S. Gesing and J. van Hemert, Eds., Oct. 2009. [Online]. Available: http://ceur-ws.org/Vol-513

[13] Galaxy. [Online]. Available: http://galaxy.psu.edu/

Sandra Gesing is a PhD candidate and research associate in the area of Grid Computing and Bioinformatics at the Eberhard-Karls-Universität Tübingen since 2006. She holds a diploma in computer science from her extramural studies at the FernUniversit at Hagen and has perennial experience as a head of a systems programmer group and systems developer in industry. She leads the work package ‘Portal’ in MoSGrid.

István Márton is working as a Research Fellow at the Laboratory of Parallel and Distributed Systems at the Computer and Automation Research Institute of the Hungarian Academy of Sciences from 2005, where he is a main developer of the P-GRADE and WS-PGRADE Portals. He received his BSc degree at College of Dunaujvaros in 2003. He has been involved in several European grid projects such as ETICS and CANCERGRID.

Recipes for Success in New Science Gateway Development

Rion Dooley[1]
[1]*Texas Advanced Computing Center, Austin, TX, USA, Dooley@tacc.utexas.edu*

***Abstract*—This paper describes how one can use lightweight middleware solutions as a rapid prototyping technique in the development of a new science gateway. Much attention and funding has been given to the development of large, comprehensive cyber-infrastructures (CI). While these play a critical role in facilitating partnerships, bridging institutions, and exposing core services to a broad audience, they can slow down the development time for new gateways. In this paper, we provide a cookbook of alternative approaches to provide services commonly provided by CI without the heavyweight software stack. These recipes can be used to rapidly prototype new science gateways. The first section of this paper contains the cookbook, with recipes grouped into categories of job submission, monitoring, workflows, and file management. The second section presents a gateways using these techniques in practice.**

***Index Terms*—Gateways, cyber-infrastructure development, web service.**

I. INTRODUCTION

A Science Gateway is defined as, "a community developed set of tools, applications, and data that is integrated via a portal or a suite of applications, usually in a graphical user interface, that is further customized to meet the needs of a targeted community." [1] Note that the authors say nothing about the size of the targeted community, or the overall usage of the gateway. This is because gateways vary greatly in their user base and their use cases. NanoHub (http://www.nanohub.org), the Longhorn Visualization Portal (https://portal.longhorn.tacc.utexas.edu/), and the UltraScan portal (http://www.ultrascan.uthscsa.edu/) are all examples of heavily used gateways, however their focus, functionality, and number of users vary dramatically. Despite their differences, these gateways all share a common characteristic: they make the process of conducting science easier for their users. They do this by adding value to the existing resources, software, and services provided by their underlying resource providers.

Successful gateways do not appear overnight. They do not start out on day one with a large user base and heavy usage. Successful gateways evolve over time. They grow through birth, childhood, and adolescent stages on their way to maturity. Each stage is different and carries with it its own set of needs. Things that are good for a gateway later in life are not necessarily good for it in childhood. One example of this is complex service-oriented architecture.

Much attention and funding has been given to the development of large, comprehensive service-oriented cyber-infrastructure (CI) solutions. While these play a critical role in facilitating partnerships, bridging institutions, and exposing core services to a broad audience, they can slow down the development time for new gateways. Robust CI is ideal for long-term production runs, but carries with it a strong learning curve, a large set of dependencies, and dozens of features that will never be used by most gateways throughout their lifetime. While such a service stack is something that a heavily used, production gateway would want to move towards as it matured over time, it is prohibitive for prototyping new gateways and for use by gateways with small usage and user bases.

In this paper we present simple, lightweight solutions for obtaining desirable features of a CI without using a heavyweight software stack. These are straightforward, proven approaches that can help birth a gateway and carry it into its childhood stage. The paper is broken down into a cookbook section and a case study section. The cookbook section is organized into groups of recipes for job submission, monitoring, workflows, and file management. The case study section looks at one gateway using recipes from the cookbook in practice, PetroApp. We conclude with a short summary and tips on moving from the cookbook to CI.

II. THE COOKBOOK

This section serves as a cookbook for building out simple service solutions to common needs when developing new science gateways. These techniques are not necessarily appropriate long-term solutions, nor should they replace the role of a mature middleware infrastructure for large-scale production gateways. They are, however, very useful techniques for getting a gateway off the ground and into the hands of users much, much faster.

A. Job Submission

The core functionality provided by many gateways is running end-user applications. Running applications on HPC systems requires interaction with a batch queuing

system such as PBS, LSF, SGE, etc. Every scheduler has a slightly different syntax for defining a job. There are several solutions for abstracting these platform-specific details using robust middleware solutions, however they require several other dependencies at the resource and service layer. Additionally, when initially building out a science gateway prototype, the number of machines on which your gateway will run jobs is usually 1, so adding middleware does little more than add complexity and points of failure early on in the development process. To get up and running quickly, it is sufficient to deal with the queuing system directly. Chances are that this experience will be valuable down the road when running on other machines regardless of whether you use heaver solutions at that time.

Submitting a job directly to a batch queuing system requires the creation of a batch submit script. If you're building a gateway, you are already familiar with what should go in this script from your experience running your applications at the command line. For all of our job submission recipes, generating this file is a pre-requisite.

Recipe 1 – Job Submission By Direct Invocation: Copy all your required files to the remote machine. Generate a batch submit script, then SSH into the machine to submit the script to the job queue. You can parse the output of the submit command to get the local job id.

Recipe 2 – Job Submission By Automated Invocation: Create a new Action Folder [2] on the remote machine and configure it to submit its contents to the remote queue. This requires no direct interaction with the machine other than a file copy. Depending on the needs of your particular situation, the ActionFolder can send the local job id back to your application, write it to your database, or save it in a file.

Recipe 3 – Job Submission Using the Scheduler API: Some schedulers come with web service APIs installed by default. In the case of SGE, it uses the Distributed Resource Management Application API (DRMAA) [3]. DRMAA compliant services are also available for LSF and PBS/Torque, though you may need to set those up separately. If they are already installed, this approach simply requires you to copy the data to the remote machine and invoke the service. The main difference here being that your batch submit script would be writing in the XMLbased Job Submission Descript Language syntax [4] rather than the textual local submit script syntax.

B. Job Monitoring

In the author's experience, polling is bad and should be avoided whenever possible. It puts unnecessary load on system resources, creates excessive chatter on a network, scales horribly, and still never achieves real time feedback. If that is not enough, system administrators highly discourage polling of their systems.

That being said, people still poll HPC due to the dearth of an eventing system provided by most resource providers. If an arbitrary publish/subscribe system were available, polling would be neither needed nor desirable as a monitoring technique. The following recipes describe several ways to implement lightweight event-driven solutions using asynchronous callbacks.

Recipe 4 – Monitoring With A Trigger Service: A trigger service is a simple service whose only purpose is to update the status of an event when invoked. In the context of this recipe, a trigger service used for job monitoring would be a REST service that reads the local job id, a unique job identifier, and a status from the URL and updates the relevant record in your database with the new status. You invoke the service by adding several curl commands to your batch submit script at the points where a job starts running, stops running, starts staging data, finishes staging data, or fails. Advantages of this approach are its simplicity (the service can be a simple php page), ease of deployment (just adding a couple lines to the batch submit script you're creating), and scalability (you can use this job monitoring recipe no matter how large your usage grows).

Recipe 5 – Monitoring With Email Notifications: This is an old but effective solution. Every batch scheduler allows you to subscribe for email updates when jobs start, stop, or fail. Some schedulers allow you to subscribe for even more changes of status. These emails can be used as a monitoring tool. Simply set up an email account for your gateway and subscribe for notifications with that address. When an email arrives, use the local job id in the subject line to lookup your record of that job in the jobs table of your gateway's accounting database and update the status to whatever status is given in the email. Depending on the scheduler, you can also obtain information about the start time, end time, wall time, memory, nodes used, exit code, etc. This can be helpful when debugging the underlying application using your gateway.

Recipe 6 – Monitoring With A Database: This approach is pretty straightforward. In your batch submit script simply update your database at different points in the file to indicate the progress of a job. Advantages of this approach are that it is fast and scalable. Disadvantages are that compute nodes may not be able to access your db from their internal network. You would also have to include your db connection parameters in the submit file, which may not be desirable for security reasons.

C. Workflows

Workflows are inherently complicated. There is rarely an easy solution that works in multiple situations. As a result, if your gateway's use case is such that you need to create arbitrary, user-defined workflows, you are better off starting out using an existing workflow solution such as XBaya, Taverna, Kepler, Triana, etc. If, however, you have just a

handful of predefined workflows, or you are limiting your use cases to just a few possible workflows for the prototype, the following recipes may be just what you need.

Recipe 7 – Running Workflows Using ActionFolders: We already described how ActionFolders can be used to submit jobs. They can also be chained together to execute workflows in lock-step. For example, use one ActionFolder to submit a job. Use a second to hold the output of the job and watch for the job to complete. When the first job completes, the second ActionFolder can use the output of the first job to submit another job, post-process data, archive the output, start a visualization routine, or anything else you may need. This process can continue as many times and across as many resources as you need.

Recipe 8 – Managing Workflows Internally With Triggers: Using the concept of trigger services above, you can wire together multiple events into a workflow using callbacks and your own internal logic. When one job finishes, it will callout to a trigger service that updates the status of a job and optionally starts another task. For well-defined, sequential, independent tasks, this can be a very clean, simple solution. However, as workflow complexity increases, or the need for parallel tasks arises, this approach can quickly degrade into what essentially becomes you writing a custom workflow system.

D. File Management

File management is a critical, but ignoble task in most gateways. Many times it is viewed as a necessary evil; something to be dealt with, but never embraced. The good news is that there is no need to reinvent the wheel. There are already many good file management solutions available as CLI [5][6][7], API [8][9], and web services [10][11][12]. They each have their own advantages and disadvantages as well as relevant usage scenarios. Fortunately, they are all well documented and widely used on many existing projects for points of reference.

Thus, in this section, we include recipes for some common file-related tasks rather than basic file management. The following recipes describe approaches to virtualize a user file space and implement collaborative permissions.

Recipe 9 – Virtualizing A User File Space: Your gateway will have a user account under which it runs on the remote HPC systems. To provide your users with their own disk allocation, you can virtualize a user file space for each of your gateway users within your gateway account's file space by resolving all file requests relative to the path to a file you create for each of your users. When a file request is made, generate the canonical path of the requested path to resolve parent file references, and then append the resulting string to the path of the user's folder in your gateway home directory. With the resolved relative path, you can then use any of the file management tools above to manage user's data. This approach inherently handles privacy issues without requiring you to implement access control lists, permissions, or write your own file system. This approach works well for organizing job output data as well as virtualizing online storage.

Recipe 10 – Implementing a Collaborative File Space: You do not need to deploy a full-blown document management system to provide basic collaborative file sharing to your users. User-defined sharing relationships can be implemented simply using shadow files or a small database. Given a server with sufficient disk space, stand up a service using Recipe 9 to create and map user folders. Next, add a permissions interface to your service to allow users to specify other people with whom they would like to share particular items. Store this relationship in a shadowed lookup file in both the owner's home folder and their colleague's home folder. Lastly, implement logic in your file management service that implements a smart folder in every user's home folder. These smart folders will appear as another folder, but instead of listing physical files, they will read the authenticated user's lookup file and resolve the requested path against its entries.

III. PETROAPP CASE STUDY

The PetroApp is a gateway used to support the BlackOil Reservoir Simulator [13]. The BlackOil reservoir simulator solves the equations for multiphase fluid flow through porous media, allowing researchers to simulate the movement of oil and gas in subsurface formations. BlackOil is a computationally intensive application. Simulations generally require 100 iterations, each comprised of roughly 300 highly independent tasks that are run using a task farming approach. Total execution time for the runs can vary between 2 and 95 days depending on the size of the underlying compute resource. The user was running their simulations by hand, but had trouble tracking the progress of all the tasks and visualizing the output. PetroApp was a pilot project to explore how a science gateway could increase his productivity and speed up his time from proposal to publication.

The main goals of the PetroApp project were to give the researcher a way to track the progress of his jobs, easily determine where his simulations were running, and visually locate results of interest by looking through an aggregate photo album. To accomplish these goals, we built in a tabbed view to provide each of the above features in a separate view. Figure 1 shows the monitoring panel with results from the current iteration loaded in the right pane. Figure 2 shows a Google Map overlay of where the current iteration is running. Figure 3 shows an image gallery of the output images from the current iteration.

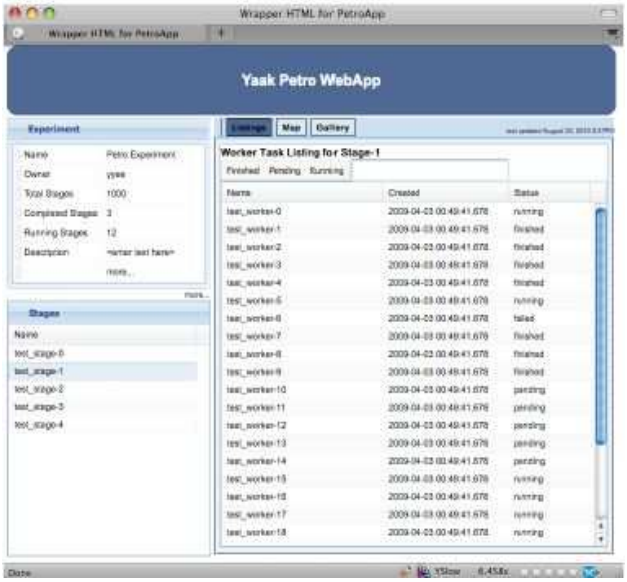


Fig. 1. Screenshot of the real-time monitoring available in the PetroApp.

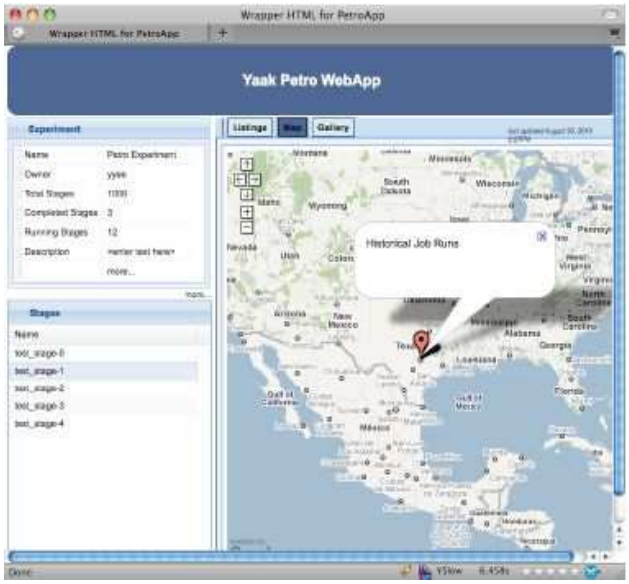


Fig. 2. Screenshot of job geo-tracking available in the PetroApp.

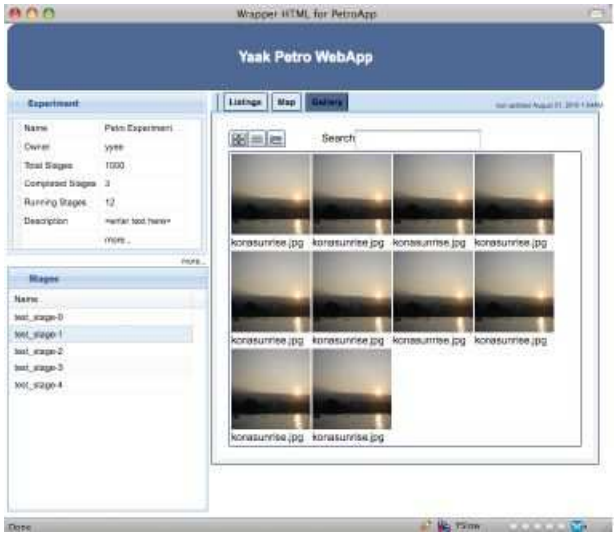


Fig. 3. Screenshot of the aggregated image gallery in the PetroApp.

The front end of PetroApp was written using the Google Web Toolkit [14]. We used Apache Tomcat and a PostgreSQL database to run the services. Since the user was already utilizing the SAGA Toolkit to manage the individual tasks, we took advantage of SAGA's persistence module to implement monitoring using Recipe 6. To detect the end of long runs and start the archiving and post-processing tasks, we used Recipe 4.

The output of interest from each BlackOil task is a summary file and an image file. We used the IRODS [15] CLI and API to move data from the job output folders to archive. Once the files were archived, they were instantly available in the PetroApp image gallery shown in Figure 3. After the initial requirements conversation, the total development time to get an initial working version of the PetroApp up and running was less than 5 days.

IV. Conclusions

There is no replacement for a well designed, robust, production service infrastructure to support mature gateways. However, on the way from proposal to production, new gateways must iterate over features, architecture, and user interfaces. Building in support for a production middleware stack early in the development process can add unnecessary complexity and slow down the project as a whole. In this paper, we have provided 10 recipes for success to help you add functionality similar to that provided by production middleware stacks, without all the moving parts.

Once a gateway prototype is stabilized and ready to move towards production, attention must shift from simply getting something in place, to getting scalable infrastructure in place. The good news is that the recipes given in this paper can be swapped out one-for-one in a modular way and replaced with more heavyweight solutions. When doing so,

you can be confident that your gateway logic is in place, and focus solely on the idiosyncrasies of the CI you choose to adopt.

ACKNOWLEDGEMENT

We'd like to acknowledge Steve Mock, Matthew Hanlon, and Praveen Nuthulapati for critical review of this paper. The writing of this paper was supported by the Open Grid Computing Environments (OGCE) project. OGCE is funded through National Middleware Initiative award number 0721656 by the National Science Foundation.

REFERENCES

[1] TeraGrid Science Gateways, https://www.teragrid.org/web/sciencegateways/
[2] Action Folders, http://www.tacc.utexas.edu/tacc-projects/actionfolders/
[3] "DRMAA and GridRPC Documents Achieve "Grid Recommendation" Status". Open Grid Forum. 2008-01-07. http://www.ogf.org/News/newscal_enews.php?oct07#LINK3.
[4] "Job Submission Description Language (JSDL) Specification, Version 1.0". Global Grid Forum. December 2005. http://www.gridforum.org/documents/GFD.56.pdf.
[5] BBFTP, http://doc.in2p3.fr/bbftp/
[6] The Globus Striped GridFTP Framework and Server. W. Allcock, J. Bresnahan, R. Kettimuthu, M. Link, C. Dumitrescu, I. Raicu, I. Foster. Proceedings of Super Computing 2005 (SC05), November 2005.
[7] SSH File Transfer Protocol, http://en.wikipedia.org/wiki/SSH_File_Transfer_Protocol
[8] Features of the Java Commodity Grid Kit. Gregor von Laszewski, Jarek Gawor, Peter Lane, Nell Rehn, Mike Russell, and Keith Jackson. Concurrency and Computation: Practice and Experience, 14:1045-1055, 2002.
[9] The SAGA C++ Reference Implementation ,"Lessons Learnt from Juggling with Seemingly Contradictory Goals",OOPSLA/LCSD 2006, Authors: Hartmut Kaiser, Andre Merzky, Stephan Hirmer, Gabrielle Allen
[10] Globus.org, http://www.globus.org/service/
[11] TeraGrid Virtual File Space, ftp://vfs.teragrid.org.
[12] T. Kosar and Miron Livny, "Stork: Making Data Placement a First Class Citizen in the Grid", In the Proceedings of 24th IEEE International Conference on Distributed Computing Systems (ICDCS 2004), Tokyo, Japan, March 2004.
[13] "Developing Autonomic Distributed Scientific Applications: A Case Study From History Matching Using Ensemble Kalman-Filters," Yaakoub El Khamra and Shantenu Jha. Sixth International Conference on Autonomic Computing. Barcelona, 2009.
[14] Google Web Toolkit, http://code.google.com/webtoolkit/
[15] iRODS: integrated Rule Oriented Data System, https://www.irods.org/index.php.

Rion Dooley Rion joined TACC as a RESA III in the Distributed and Grid Computing Group in March 2006. His responsibilities include architectural design and implementation of the iPlant Collaborative API amd TeraGrid Mobile User Portal. He is also lead developer on the TeraGrid's comprehensive file management services. Prior to joining TACC, Rion was an IT Analyst at the Center for Computation & Technology at LSU where he served as principal coordinator for CCT on the GridChem project as well as the Cactus Task Farming Infrastructure

A web portal for management of biological data and applications

Matteo Gnocchi[1], Alessandro Orro[1], Davide Di Pasquale[1], Luciano Milanesi[1]
[1]*Institute for Biomedical Technologies, Segrate, Italy, luciano.milanesi@itb.cnr.it*

***Abstract*—Recent advances in methods and technologies for molecular biology and medicine are going to produce an increasing amount of data related to DNA samples but also to phenotypes believed crucial in the disease under study. Presently medical research area is characterized by heterogeneous data that are difficult to manage in a complete and consistent way. In fact the use of different complementary technological platforms is often necessary in order to validate experimental results and these platforms are often located in different laboratories.**

We propose the implementation of a web portal to data management in bioinformatics laboratories in which analysis results can be annotated and integrated. The portal is based on the integration of a set of web services that have been implemented in order to manage genomic data coming from the same institution.

***Index Terms*—Genome analysis, bioinformatics, web portals.**

I. INTRODUCTION

GENOME wide search for genes underlying common diseases is facilitated by the use of high throughput genotyping. Nowadays, huge amount of molecular markers are available for the human genome and laboratories equipped with recent genotyping technologies can use them to quickly generate hundreds of thousands of genotypes for each DNA under study.

In particular, Single Nucleotide Polymorphisms (SNPs) are one of the most common forms of human genetic variation that can be used to discover the sequence variants affecting common diseases by examining them for statistically significant association with measurable phenotypes. On the other hand microarray data can be used to confirm the results obtained by analyzing the expression profiles of the genes involved in the disease that have been indentified in the genetic analysis.

In a typical molecular biology laboratory, genome data are managed with LIMS software (Laboratory Information Management Systems) that implements several useful functions. Some genotype management systems have been implemented in last years with different features and supporting different genotyping and gene expression technologies 000000. Even though these are useful tools, unfortunately, none of the available systems seem to be easy to customize or integrate in pre-existent infrastructures. In particular the integration in a unique database of genotype, phenotype and demographic data coming from different laboratories facilitates the generation of reports for both visualization and data input for further analysis.

In this work we present a web portal that allows developers to integrate the computational resources and information usually produced and managed in a multidisciplinary laboratory. The portal is mainly devoted to the management of genomics data although it has been used to integrate other types of information. Presently the main features of the system are:

- Management of all the information (user, groups, research activities, …)
- Functional Genomics Applications:
  - Genotypes data management and analysis
  - Workflow for Linkage Analysis

The web user interface based on Liferay allows user of the system to easily interact with the developed functionality.

II. MATERIAL AND METHODS

In this chapter the main features of the system are described focusing mainly on the genomics application and the implementation of the web portal.

A. Web portal technology

The Liferay framework system is the world's leading open source java enterprise portal solution; the system architecture supports a high availability, robustness and safety middleware for mission-critical web applications [1]; it ships in two different editions: Liferay Portal Standard Edition (SE) and Liferay Portal Enterprise Edition (EE), the standard edition of Liferay Portal is offered for free under the business-friendly open source license; the Portal EE version is a supported version of Liferay Portal for the enterprise. Hardened for security and designed to be very stable, EE is offered with a subscription and support package, allowing organizations to build their portals on a stable version of the product that is offered over an extended

period of time.

Liferay Portal by default is configured to sit at the root (i.e., /) of the application server used (Such as Tomcat or Glassfish), this method allows the separation between the portal environment and the web applications being developed. With this method the running instance of Liferay is likely to be hosting many other web applications, and even integrating several of them together on a single page on the portal.

Liferay is a portal server [2]. This means that it is designed to be a single environment where all of the applications a user needs can run, and these are integrated together in a consistent and systematic way. If an application lives outside of the portal, the portal should be able to consume some resource of the application (such as an RSS feed) so that the end user can see everything he or she interacts with at a glance.

To achieve this, all of the application functionality within Liferay Portal is in fragments of the page called portlets. Portlets are web applications that run in a portion of a web page. The heart of any portal-based web site is its portlets, because portlets are where all of the functionality is implemented. Liferay's core is a portlet container, and the container's job is to aggregate the set of portlets that are to appear on any particular page and display them properly to the user.

The portal server uses the Service Oriented Architecture (SOA) design principles throughout and provides the tools and framework to extend SOA to other applications. This is designed to create, support and deploy portlets, implemented by means of many type of technologies (Such as Java, Php, Python) that adhere to the portlet API compliant with both JSR-168 and JSR-286. A set of more important portlets (such as document Library, Calendar, Wikis, and so on) are included with the standard portal installation package [3]; when an application needs to be replaced, it can easily be disconnected and subsequently reinstalled from the running system at a single point.

It also uses Hibernate framework and JDBC API to manage the connection between the application and the databases throughout persistent connection and transaction (there are many database management systems supported, such as MySql, Postrgresql or Oracle).

To allow a high security level, Liferay middleware uses industry-standard, government-grade encryption technologies, including advanced algorithms such as DES, MD5, and RSA. Liferay was benchmarked as one of the most secure portal platforms using LogicLibrary's Logiscan suite; in order to improve flexibility and usability, Liferay allow the integration of external single sign-on protocol system, such as LDAP, Microsoft Exchange or Open ID.

Liferay uses standard ways to communicate with other softwares [4]: There are various standards used or supported by the system: AJAX, iCalendar, and Microformat, JSR-168, JSR-127, JSR-170, JSR-286 (Portlet 2.0), JSR-314 (JSF 2.0), OpenSearch, Open platform with support for web services (including JSON, Hessian, Burlap, REST, RMI, and WSRP), WebDAV, and CalDAV. It follows and implements the new W3C Recommendation WCAG 2.0 (Web Content Accessibility Guidelines), to make all the web content accessible to a wide range of people with disabilities, including blindness and low vision, deafness and hearing loss, learning disabilities, cognitive limitations, limited movement, speech disabilities, photosensitivity, and combinations of these.

In particular, the developed Portal has been developed by using the LIFERAY 6.X Version and the MySql RDBMS system for the data storage. The structure of the developed web portal, can be divided in two different parts:

1) The first part is represented by the integration of the standard liferay modules on the portal. Each module delivers different service to the users in according to their permission (such as insert or modify documentation, events and information related to the project).

2) The second part is composed of a set of interfaces developed for the management of biological information. In depending on the user account permission, each page can access to different services and interfaces.

The interfaces implemented in the portal were developed with the Java Vaadin web Framework [5]. Each service is independent from the other and the navigation between them is possible by a link menu located on the left side of the portal web page. Each service retrieves and shows the data for an external resource; indeed, the information showed in each service are stored in a external database, and the communication between them is possible by means of RPC (Remote Procedure Call) API procedure.

B. User management

Access policy is managed with a mixed approach based on liferay users that map the users that are specific for the wrapped application.

To ensure that the right people control the right information, portal administrators can assign individual users or groups of users to different "roles" or "Permission"; A Permission is an action on a resource. Portal-level permissions can be assigned to the portal (for example, users, user groups, communities, and organizations) through roles. Group-level permissions can be assigned to groups (for example, organization and communities). Page-level permissions can be assigned to page layouts. Model permissions can be assigned to model resources, (for example, blogs entries, web content, and so on). Portlet permission can be assigned to portlets (for example, view, configuration, and so on).

A Role is a collection of permissions. Roles can be assigned to a user, user group, community, location, or organization. If a role is assigned to a user group, community, organization, or location, then all users who are members of that entity receive permissions of the role.

Besides roles and permission, Liferay users can be intuitively grouped into a hierarchy of "organizations" or cross-organizational "communities," providing flexibility and ease of administration. An Organization represents the enterprise-department-location hierarchy. Organizations can contain other organizations as sub-organizations. Moreover, an organization acting as a child organization of a top-level organization can also represent departments of a parent corporation. A Community is a special group with a flat structure. It may hold a number of users who share common interests. Thus we can say that a community is a collection of users who have a common interest. Both roles and users can be assigned to a community.

C. Functional Genomic applications

The “functional genomic applications” module is an independent software system based on Python-Zope that provides several functionalities for the management of genomics data. Although the system is mainly devoted to the management of SNP data produced with the Illumina platform [12], this is not a strict requirement. Other types of SNP genotyping technologies and microarray (such Affymetrix [13]) can be added in the system using suitable XML descriptors.

The main features of the system are

1. Automatic import of raw genotype and expression data from the genomics platforms;
2. Definition and assignment of phenotypes to the subjects,
3. quality control of the data in order to select markers with high genotyping score;
4. statistical descriptive analysis that provides information about basic features and quality of data;
5. analysis of the genetic population structure to identify stratification;
6. statistical descriptive analysis that provides information about basic features and quality of data;
7. single-point analysis of association between genotype and quantitative or qualitative traits;
8. multi-locus analysis to combine genotypes of adjacent markers and find associations between haplotypes and phenotypes.

A particular type of phenotypes is the Demographic attributes. They are related to the parental relationship between the subjects and the race of the subjects. They are managed like the phenotype attributes but it is not possible to define acquisition session in this case because they are strictly related to the subject and not estimated.

The majority of developed features have been exported through a web service interfaces (XML-RPC) and can be used by any compatible client. In particular, the Liferay web interface implements similar tasks simply by interfacing with these web services. An example of this is shown in Fig. 1 where the web interface obtains biological information (such as chromosome, gene, and position) about the markers in the database by querying the corresponding web service implemented in the “functional genomics” module. A similar example in Fig. 2 shows the visualization of the statistics about a particular chip. The chip can be considered as a set of markers that are used together in the acquisition of genomics data.

As a further example, we have ported to the liferay interface a workflow for the linkage analysis. Linkage Analysis is a genetic analysis that permits the discovery of genetic correlations in complex diseases following their transmission through family generations. Fig. 3 shows the linkage workflow in which it is possible to set all the parameters of the analysis and to monitor the overall computation executed in a glite Grid environment [14].

The “functional genomic module” until now has been used in many national and international projects by many partners. In particular, in projects related to the study of the genetic determinant of cardiovascular disease and in the study of mental disorders (such as autism and schizophrenia) with the integration of genotypes and complex brain phenotypes.

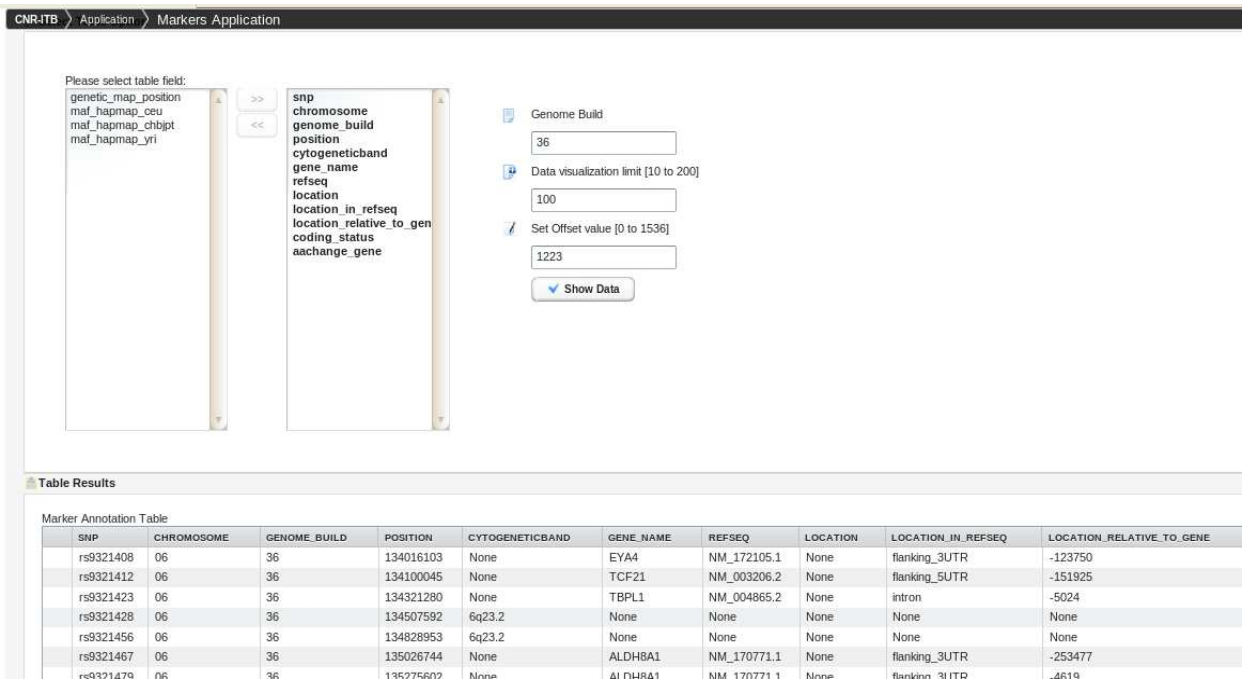


Fig. 1 Web portlet showing the biological annotation of the markers

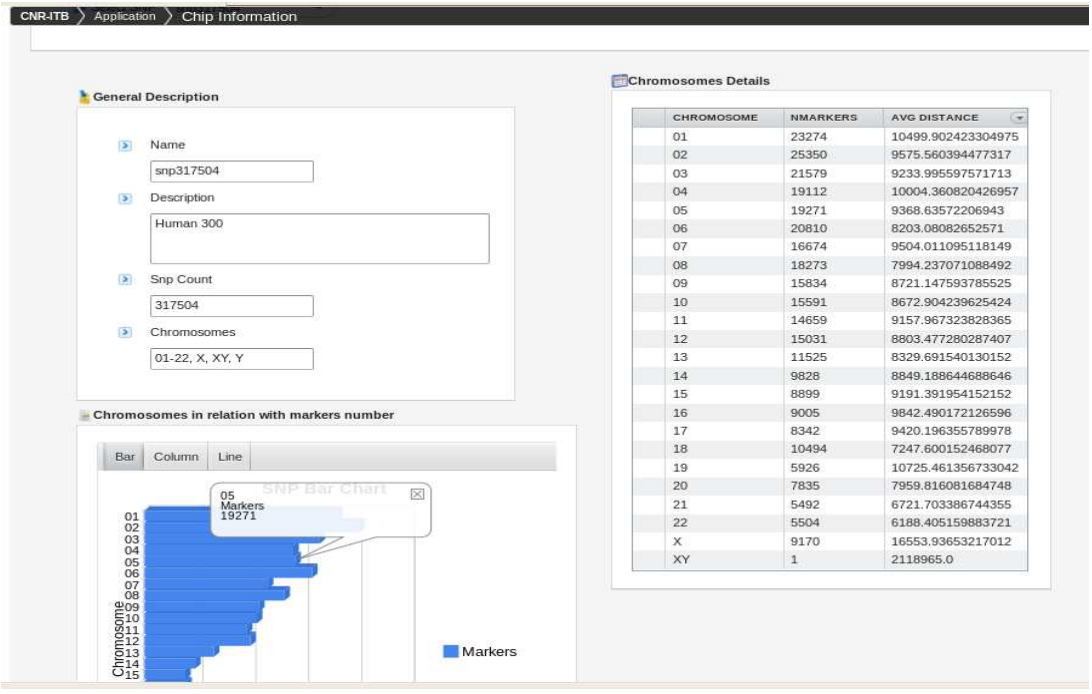


Fig. 2 Web Portlet showing the information available for the marker in a given Chip

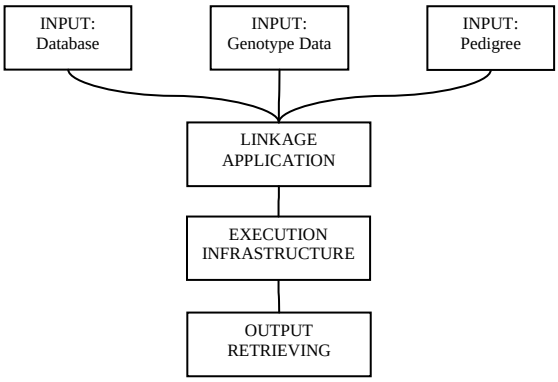


Fig. 3 Workflow of the Genetic Linkage Analysis

III. Conclusion

The application developed represents an easy way to interact with the various biological data and information; The portal created with the Liferay technologies has a unique access point to the different services and provides an advanced system for security and user management.

The Liferay Portal provides various types of interfaces and services, which allow users to add and view the data related to the Illumina chip, the creation of documentation and the managing of project tasks.

The interaction with the services forms represents a reliable software layer that hides all the complexity related to the maintenance of the portal and ensures at the same time security and integrity of the data management. The use of JAVA technology increases the portability of the system on different platforms, allowing an easier integration of the application and ensuring high scalability.

Acknowledgments

This work has been supported by the CNR Italian Bioinformatics Network, MIUR FIRB ITALBIONET (RBPR05ZK2Z), Bioinformatics analysis applied to Populations Genetics (RBIN064YAT_003), SHIWA SHaring Interoperable Workflows for large-scale scientific simulations on Available DCIs (Contract N. 261585). Authors gratefully thank John Hatton for reviewing the manuscript.

References

[1] Liferay enterprise portal site: http://www.liferay.com
[2] Richard L. Sezov, Jr.; Liferay Administrator's Guide; 2009; Liferay, Inc.
[3] Liferay bundle download url: http://www.liferay.com/downloads
[4] Jonas X. Yuan; Liferay Portal 6 Enterprise Intranets - Build and maintain impressive corporate intranets with Liferay; Packt Publishing; April 2010
[5] Vaadin web Framework site: http.//www.vaadin.com
[6] Li JL, Deng H, Lai DB, Xu F, Chen J, Gao G, Recker RR, Deng HW. Toward high-throughput genotyping: dynamic and automatic software for manipulating large-scale genotype data using fluorescently labeled dinucleotide markers. Genome Res. 2001;11:1304–1314.
[7] Donofrio N, Rajagopalan R, Brown D, Diener S, Windham D, Nolin S, Floyd A, Mitchell T, Galadima N, Tucker S, Orbach MJ, Patel G, Farman M, Pampanwar V, Soderlund C, Lee YH, Dean RA. PACLIMS: A component LIM system for high throughput functional genomic analysis. BMC Bioinformatics. 2005;6:94.
[8] Zhao LJ, Li MX, Guo YF, Xu FH, Li JL, Deng HW. SNPP: automating large-scale SNP genotype data management. Bioinformatics. 2005;21:266–268.
[9] Monnier S, Cox DG, Albion T, Canzian F. T.I.M.S: Taqman Information Management System, tools to organize data flow in a genotyping laboratory. BMC Bioinformatics. 2005;6:246.
[10] Hampe J, Wollstein A, Lu T, Frevel HJ, Will M, Manaster C, Schreiber S. An integrated system for high throughput TaqMan™ based SNP genotyping. Bioinformatics. 2001;17:654–655.
[11] Wang L, Liu S, Niu T, Xu X. SNPHunter: a bioinformatic software for single nucleotide polymorphism data acquisition and management. BMC Bioinformatics. 2005;6:60.
[12] Illumina http://www.illumina.com
[13] Affymetrix http://www.affymetrix.com
[14] gLite middleware for grid computing http://glite.web.cern.ch

Matteo Gnocchi received a BS degree in Information Technologies and Communication. Since 2008 he is staff scientist at the Italian National Research Council at the Institute of Biomedical Technologies (CNR-ITB). His main research interests are in the field of the Bioinformatics data integration; in particular the development of Informatics System and service for Biological data integration and manipulation through Internet.

Alessandro Orro received his Ph.D. degree in electronics and computer engineering in February 2005, after a three-year course at the University of Cagliari, Italy, under the supervision of Professor G. Armano. He is currently working at Italian National Research Council at the Institute of Biomedical Technologies (CNR-ITB). His main research interests are in the field of Bioinformatics; in particular he is investigating multiple alignment algorithms and techniques for protein secondary structure prediction. The underlying techniques and tools, such as genetic algorithms and artificial neural networks, fall into the category of soft computing.

Davide Di Pasquale graduated in Physics at the Università degli Studi di Milano, Italy in 2001. He currently works at the Italian National Research Council at the Institute of Biomedical Technologies (CNR-ITB) where he develops bioinformatics web

applications. His main activities are in the field of Information Technologies and Bioinformatics with special interest in distributed computing technologies and algorithms.

Luciano Milanesi received a BS degree in Atomic Physics and the specialisation degree in Health and Hospital Physics. Since 1987 he is staff scientist at the Italian National Research Council at the Institute of Biomedical Technologies (CNR-ITB). He is the coordinator of the CNR Interdepartmental Bioinformatics Research Network in Life science, Medicine and ICT and in several National and European projects. He is author in more than 300 publications in the field of Bioinformatics, Systems Biology and Medical Informatics.

.

FARO - The Web portal to access ENEA-GRID Computational Infrastructure

A. Rocchi, S. Pierattini, G. Bracco, S. Migliori, F. Beone, A. Santoro, C. Sciò, S. Podda
ENEA – Centro Ricerche Frascati – V. Enrico Fermi 45, Frascati (ROMA)

***Abstract*—**The need to centralize the access to resources, services and systems via ubiquitous tools such as a Web browser has led to the creation of FARO (Fast Access to Remote Objects). FARO provides intuitive access to the entire ICT infrastructure of ENEA-GRID, giving researchers a user-friendly way to perform tasks as HPC job submission and remote graphics processing, and allowing them to be instantly connected with a wide range of software and hardware architectures. In order to take advantage of such features, only a common Web Browser is needed (given that it is enough recent to support Java), so that almost every platform (PCs, netbooks, PDAs) is supported. The developed technology enables the realization of custom interfaces for the applications and the complex services available on the ICT infrastructure and can be customized for the special needs of user groups, making immediate and effective the use of the available resources.

FARO is built on top of the NX architecture and protocol, developed by NoMachine company. In particular, it relies on the FreeNX open-source server, which has been customized in order to implement some advanced features as load balancing and session distribution over a cluster of servers. This customization, taking into account also the session resuming capability, results in a rather complex setup which can be affected by problems of sessions inconsistencies. For this reason, the FARO suite has been extended with a new application, "NX Watchdog", which monitors the status of user sessions and automatically deletes them, if they are recognized to be stale and impossible to resume.

A GUI, written in Java, effectively implements the user-friendly portion of FARO. It can be easily extended with additional modules which also integrate seamlessly with the resources of the ENEA-GRID infrastructure (Authentication, Distributed File System, Cluster Manager, Job Scheduler, heterogeneous operating systems and hardware platform and instruments, etc.). This ability allows the creation of custom interfaces for groups of researchers needing to use special applications and services.

FARO integrates also the "Remote Visualization Tool", which allows 3D applications to execute on the graphics cards available locally on the CRESCO HPC system (the main computational resource of ENEA-GRID infrastructure) and provides the remote 3D rendering on the user workstation display. In this way the user can operate easily on complex 3D models with minimal local resources as he does not require any advanced GPU device or/and special 3D applications, nor the download of big 3D model. FARO is currently available in more than one research organization and is used in many fields as computational chemistry, fluid dynamics and plasma physics.

Workflows and Analysis Approaches for Molecular Dynamics Simulations

Jens Krüger*[1], Georg Birkenheuer[2], Sebastian Breuers[3], Sandra Gesing[4], Martin Wewior[5], André Brinkmann[2], Dirk Blunk[3], Oliver Kohlbacher[4], Lars Packschies[5] and Gregor Fels[1]

[1]*Department Chemie – Universität Paderborn,*
[2]*PC² – Universität Paderborn,*
[3]*Department für Chemie – Universität zu Köln,*
[4]*Bioinformatik – Eberhard-Karls-Universität Tübingen,*
[5]*RRZK – Universität zu Köln*

*dr.jens.krueger@gmail.com

***Abstract*—**Molecular dynamics simulations are an extremely powerful tool to evaluate a broad range of questions in biomedical and life sciences. Within the context of the MoSGrid-Project (www.mosgrid.de), an interdisciplinary consortium with the goal to make simulation codes easily available within the German D-Grid, is focusing on molecular dynamics as basic technique employing codes such as Gromacs, GLAT, NAMD, NWChem or Desmond [1]. Our main motivation is to offer a portal solution, which is a real aid for the scientist in his daily research work in terms of ease of use, availability, and efficiency. All the aforementioned programs offer the opportunity to generate excellent simulation data. However, all of them suffer from the problem that their usage requires text-based input and modification of complex configuration files. This raises the hurdle for new users who often have only a limited background about modern information technology. We will present scientific standard approaches for system preparation, e.g. for the solvation and equilibration of globular proteins or the setup of a box filled with a bulk material for free energy calculations. These recipes are represented as workflows within UNICORE-6, a widely used grid middleware enabling access to clusters within the grid.

The service is completed by offering a comprehensive analysis of simulation data. The evolvement of potential energy, RMSD, density and other parameters of the simulation are computed after completion of the calculation. In order to evaluate if a running simulation is still sane, selected parameters will be made available while the simulation is still running.

REFERENCES

[1] G. Birkenheuer, S. Breuers, A. Brinkmann, D. Blunk, G. Fels, S. Gesing, S. Herres-Pawlis, O. Kohlbacher, J. Krüger, and L. Packschies: Grid-Workflows in Molecular Science Software Engineering 2010, Grid Workflow Workshop, pp. 177-184, GI-Edition - Lecture Notes in Informatics (LNI), P-160, ISSN 1617-5468, 2010.

ISBN 978-88-95892-03-0

International Workshop on Science Gateways

Consorzio COMETA