IWPLS'09

Building Science Gateways with EnginFrame: a Life Science example

Livia Torterolo[1,2*], Ivan Porro[1,2], Marco Fato[2], Maurizio Melato[3], Antonio Calanducci[4], and Roberto Barbera[4,5]

[1]Nextage srl, Italy
[2]Department of Communication, Computer and System Sciences of the University of Genoa, Italy
[3]Nice Srl, Italy
[4]Italian National Institute of Nuclear Physics, Division of Catania, Italy
[5]Department of Physics and Astronomy of the University of Catania, Italy

[*]Co-responding Author

ABSTRACT

Given their ability to hide the complexities of the underlying computational environment, Science gateways, Grid portals and Web Service interfaces are crucial in enhancing user adoption of e-Infrastructures.
Much effort has been invested in this field to create virtual environments that allow researchers to focus on their research carried out within the Virtual Organisations they belong to. The approach presented in the paper follows the Teragrid vision, according to which gateways enable entire communities of users associated with a common scientific goal to use the resources of a cyberinfrastructure through a common interface.
The paper describes the EnginFrame framework developed to easily build Science Gateway prototypes and presents a use case from the Life Sciences domain.

1 INTRODUCTION

One of the most rapidly evolving areas of Grid and Distributed Computing is the construction of Grid Portals, high level graphic interfaces that greatly simplify the execution of scientific applications on e-Infrastructure exploiting the various middleware services in a transparent and ubiquitous way.
The main goal of Grid Portals is to hide the details and the complexity of the Grid infrastructure from the users in order to improve usability and utilization of the Grid. Usability can be enhanced by lowering end-user requirements for accessing the Grid and verticalizing the service offering. In parallel utilization can be improved making the Grid evolution transparent to the end-user and enforcing Grid utilization policies [1].
A number of Grid portals have become popular in the last few years in the scientific research community [2] but most of them are designed on the basis of requirements coming from a specific user community thus appearing like "vertical" web-based applications (or collection of services) rather than "horizontal" portals. The DEGREE project[1], for example, has particularly been both efficient and effective in the exploitation and dissemination of Grid technology in the field of Earth Science identifying a short list of requirements for Grid portals for Earth Science (ES) applications. In order to achieve its aims of making the Grid easier to use for the ES community, DEGREE has successfully integrated ES applications and middleware functionalities but only within specific web Portals[2].
Notable exceptions are the EnginFrame portal[3], that boasts relevant references in the industrial world (Ferrari, Schlumberger, ENI, Partners Healthcare, etc.) and it has been used in various research projects such as A-WARE [3], DEISA [4], BEinGRID[4], and EGEE [5], and the P-Grade Portal [6] used as a Multi-Grid portal serving several EGEE Virtual Organizations and based on GridSphere [7].
However, at present the most limiting aspects of existing solutions are related to how users authenticate themselves to the portal and the underlying infrastructure and to how many and what kind of resources they can access. The first great distinction should be done between EGEE-like and non EGEE-like portals, since the first ones often require the user to have an existing account on Grid middleware resources too (certificate and VO membership). The second classification takes into account the available underlying middleware. Most portals are tailored to a specific middleware while others may launch jobs on different kind of resources. Finally, data storage also plays a part when comparing portals, as access to the storage via the portal can vary from case to case. Most EGEE-like portals, for example, can easily access only storage services controlled by gLite middleware [8].

[1] Dissemination and Exploitation of GRids in Earth sciencE, http://www.eu-degree.eu
[2]Deliverable 4.1 and Deliverable 4.3, http://www.eu-degree.eu/DEGREE/internal-section/wp4/
[3] EnginFrame Portal, http://www.enginframe.com
[4] BEinGRID project, http://www.beingrid.eu/

Those issues may be approached with a paradigm shift: considering that a portal should be kept as generic as possible, in order to be used by the largest possible number of communities, and considering that scientific communities require specific services, tools and interfaces on top of the general purpose ones, researchers are migrating towards the Science Gateway concept. This is explained in the next section. The EnginFrame framework is outlined in section 3 in the context of a Science Gateway for Life Sciences and an exemplar use case is shown in section 4. Conclusions are then drawn in section 5.

2 THE SCIENCE GATEWAY VISION

According to TeraGrid[5], a Science Gateway is a community-developed set of tools, applications, and data that is integrated in graphical user interface via a portal or a suite of applications and that is customized to meet the needs of the targeted community. Some gateways expose customized sets of community codes so that scientists or students can run them. Others bring new services and applications to the community that would not be accessible otherwise. Depending on the needs of the specific community, any of the capabilities listed below should be provided by a typical Science Gateway:

(1) **Job execution services:** the classical preparation, submission, monitoring and output retrieval;

(2) **Access to data collections:** the ability to access and query data collections, intended as collection of files and describing metadata;

(3) **Workflows:** the possibility to design and run virtual experiments in the form of organized sequence of tasks with inter-dependencies;

(4) **Visualization software and hardware:** the availability of high-end visualization tools and systems to visualize complex and large datasets;

(5) **Data analysis and movement tools:** the capability to provision data to a specific location accordingly to user, network, and performance requirements;

(6) **Resource discovery:** an indexed collection of resources that applications and user-services will rely on;

(7) **Domain-specific computational applications:** specific services that may collect several of those aspects resulting in end-user applications that address a specific, complex issue.

According to the Science Gateway definition, in the next section we present a Science Gateway for Life Sciences based on EnginFrame, a Grid framework through which it is possible to integrate these functional elements in a stable and powerful environment. Missing key aspects in the above check-list, such as **authentication and authorization,** will be addressed in detail.

[5] Teragrid project, http://www.teragrid.org/gateways/

3 THE ENGINFRAME FRAMEWORK

The Science Gateway described in this section belongs to Life Science and has its focus on biomedical informatics, in which the authors have a significant experience [9].

In this section we will concentrate on the architecture and general functionalities of the portal, based on EnginFrame, while a use case will be presented in section 4.

The work benefits from hardware infrastructure donated as a SUR Grant from IBM to the University of Genoa in 2008. It is composed by a Blade Centre architecture with one visualization blade, four Intel based computing nodes and three IBM PowerCell/BE based computing nodes. The computing cluster is accessed by users using the portal described in this paper.

Moreover, thanks to the collaboration with RENCI[6], North Carolina, and the INFN GILDA team [5] in Catania, Italy, two different Grid middleware have been tested in our prototype: iRDOS [10] and gLite [8].

The portal is based on EnginFrame, a framework that allows virtual organizations to provide application-oriented computing and data services both to users (via Web browsers) and in-house applications (via SOAP/WSDL based Web services), hiding all the complexity of the underlying Cluster, Grid, or Cloud infrastructure.

EnginFrame as Grid Portal framework offers a wide range of facilities to IT developers and provides end users with application-oriented vertical services.

In particular, it simplifies the development of Web portals exposing computing services that can run on different computational Grid systems such as Platform LSF[7], Sun Grid Engine[8], Altair PBS[9], EGEE gLite[10], etc.

EnginFrame supports several open and vendor neutral standards, distributed file systems, GUI virtualization tools and different kinds of authentication systems.

The EnginFrame Grid portal receives incoming requests from the Web, authenticates and authorizes the requests, and then executes the required actions in the underlying Grid computational environment. Then, it gathers the results and transforms them into a suitable format before sending the response to the client. Transformation of results is performed according to the nature of the client: HTML for Web browsers and XML for Web-services based client applications.

EnginFrame architecture is layered in three tiers, as shown in Figure 1.

[6] Renaissance Computing Institute, http://www.renci.org/

[7] Platform Computing, http://www.platform.com/

[8] Open source Grid Engine, http://gridengine.sunsource.net

[9] Altair PBSpro, http://www.altair.com/software/pbspro.htm

[10] EGEE gLite, http://glite.web.cern.ch/glite

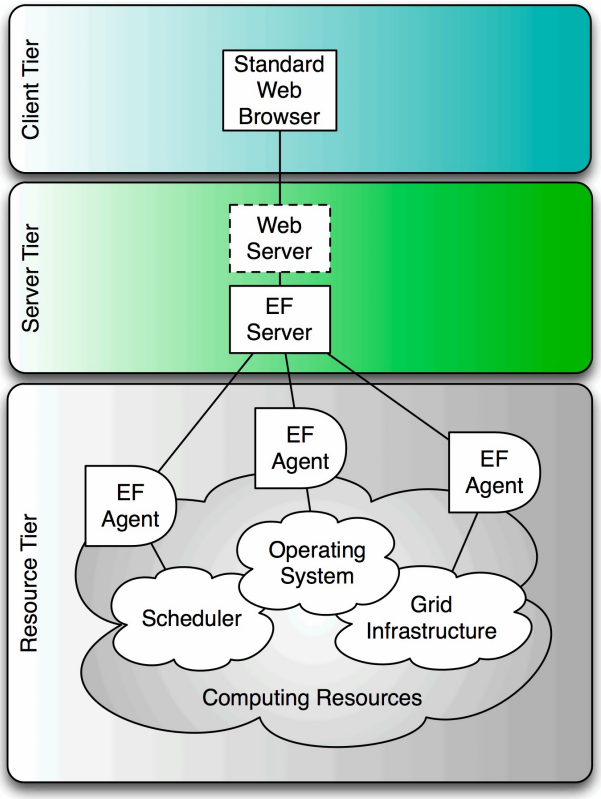

Figure 1: EnginFrame Grid Portal Architecture

The *Client Tier* consists of the user's Web browser. In EF, this interface is based on well established Web standards like XHTML and JavaScript. This tier is independent of the specific software and hardware environment used by the end user. The Client Tier also integrates remote visualization technologies.

The *Server Tier* consists of a Server that interacts with the EF Agents and manages the interaction with the users. EF Server is a Java Web application and must be deployed inside a Java Servlet Container (e.g., Apache Tomcat). It takes care of exposing services to users.

The *Resource Tier* consists of one or more EF Agents deployed on the back-end infrastructure. EnginFrame Agents are stand/alone Java applications which manage computing resources on user's behalf and interact with the underlying operating system, job scheduler, and Grid infrastructure to execute the portal services (e.g., starting jobs, moving data, retrieving cluster load, etc.).

EnginFrame allows exposure of computing services solutions on the Web. This is done by developing XML-based descriptions of the services and scripts representing the actual services implementations. Most of the information managed by EnginFrame is described by dynamically generated XML documents. The source of such information is typically the service execution environment: an XML abstraction layer aims to submit service actions and translate raw results (from the computational environment) into XML structures. The XML abstraction layer is designed to decouple the portal from the actual Grid environment. This characteristic allows to extend the functionalities of the portal by developing ad hoc plug-ins for specific Grid middleware and legacy applications.
A plug-in is a self-contained software bundle that encapsulates XML service descriptions and the scripts or executables involved with the services actions. In other words, a plug-in is a piece of software that extends the EnginFrame portal functionalities in many different areas:

- **Bundle** - full featured package containing other plug-ins, e.g. Grid Portal for LSF;
- **Kernel** - extension that enriches the portal's core system, e.g. Enterprise Portal;
- **Auth** - extension that authenticates users towards an authoritative source (e.g., Operating system, NIS, LDAP, etc.);
- **Data** - extension that helps display user data inside the portal, e.g. a Remote Spooler Plug-in;
- **Grid** - extension that connects the Grid portal with a Grid manager, e.g. LSF Power Pack;
- **SOA** - extension that transforms the Grid portal into a SOA provider, e.g. the EnginFrame Web Service Integration Kit;
- **Interactive** - extension that allows clients to interactively use the applications launched by the Grid portal, e.g. VNC Plug-in;
- **Application** – extension providing a set of services for integrated back-end industry applications, e.g. EDA[11] or MCAE[12] applications.

The plug-in mechanism provided by EnginFrame allows developers to easily and dynamically extend its set of functionalities and services offering a full set of plug-ins targeting each of the domains and capabilities covered by the Science Gateway model.

3.1 Authentication and authorization

A flexible authentication delegation offers a wide set of pre-configured authentication mechanisms: Linux Operating system, NIS, PAM, LDAP, Microsoft Active Directory, MyProxy, Globus, etc. It can also be extended throughout the plug-in mechanism.
Besides authentication, EnginFrame provides an authorization framework that allows to define or inherit from the underlying system group of users, access control lists (ACLs) and to bind ACLs to resources, services, service parameters and service results. The Web interface of the services provided by the Portal can be authorized and thus tailored to the specific users' roles or access rights.

[11] EDA: Electronic Design Automation
[12] MCAE: Mechanical Computer-Aided Engineering

3.2 Job execution services

EnginFrame supports a wide variety of compute environments like the MAUI and Torque scheduler, PBS (both OpenPBS and PBS Pro), the Platform LSF product suite, Sun Grid Engine, and Grid middleware like Globus, gLite, etc. An XML virtualization layer invokes specific middleware commands and translates results, jobs and Grid resources descriptions into a portable XML format called GridML that abstracts from the actual underlying Grid technology in use.

3.3 Access to data collections

Concerning data management, the EnginFrame framework allows users to browse and handle data on the client side or remotely archived in the Grid and then to host service working environment in filesystem areas called Spoolers. On the Grid side, the approach we followed is twofold: both SRB/iRODS and gLite Grid middleware can be integrated as DataGrid layers with EnginFrame.

SRB and iRODS

The integration of EnginFrame with SRB is provided through a plug-in. The set of functionalities includes the possibility to get and send data to SRB and to browse the SRB structures directly from the user's browser. Moreover, the plug-in also extends the Spoolers view in order to link to SRB areas and see files and collections together with the metadata associated to them.
The plug-in contains a collection of Unix scripts that interact with SRB commands, a set of XML services, XSL and Javascript code [11].
Another plug-in integrates EnginFrame with iRODS, a data-Grid technology that is an evolution of SRB. The rich set of functionalities includes the ability to get and send data to iRODS and to browse and manage the iRODS data Grid structures directly from the user's browser. It is also possible to obtain audit data for every object in the data Grid and to retrieve and search for associated metadata, ACL, quota information and so on in a very comfortable way through a web browser.
Moreover, the plug-in provides a unique feature for managing iRODS rules, in order to give the user the ability to visually create, edit, save and run rules.
The plug-in contains a collection of Unix scripts which interact with iRODS commands, rules and microservices, a set of XML services, XSL and Javascript/Ajax code.

gLite

Concerning the data management with the EGEE middleware, gLite provides three kinds of services to deal with data on the Grid: Storage Elements (SE), the LCG File Catalog (LFC) and the AMGA Metadata Catalog. Files are stored into Storage Elements (SE, the Grid service that takes care of data persistence). Files can be replicated on several geographically distributed SEs for ubiquity, security, redundancy and load-balancing purposes. LFC provides a global hierarchical namespace of Logical File Names (LFNs) of stored files, keeping track of the mapping between these LFNs and the physical replica paths of the storage containing the actual data. Finally, the AMGA Metadata Catalog can include descriptive information (metadata) in the content of each file. Using this metadata, users and applications can make queries to the service to quickly individuate and later retrieve the files they are interested in.
Sometimes the direct usage of the three previous services can be difficult because of the complexity and fragmentation of those service's APIs. In order to cope with this problem, two higher level services have been developed to provide a transparent and easy interface to the underlying services.
The first is a Java Object Oriented Framework, named Grid Storage Access Framework (GSAF) [12]. This has been designed for developers of applications that adopt data Grids as repository infrastructure, and allows applications to manage files in a coherent way among the tree main data management services offered by gLite. Looking at the design viewpoint, GSAF identifies a kind of Grid software engineering pattern trying to offer a standard solution for common problems. GSAF wraps each gLite data services through a corresponding software module built on top of the related API, providing functions to interact with Storage Elements, the File and Metadata Catalog. It has already been integrated in EnginFrame and successfully tested within a neuroinformatics application for the early diagnosis of the Alzheimer disease [13].
The second service easily integrable in our portal, is gLibrary [14], a system to manage digital libraries on Grid infrastructures. With gLibrary all the complexity of the underlying Grid system is hidden to users and applications. They only deal with concepts like repositories, digital objects types and collections: once a repository has been created by the administrator, a list of types and collections for the objected that can be stored in the repository has to be defined with their attributes. Those attributes represent the metadata of the digital objects that will be actually stored into some Grid Storage Elements. Additionally, filter attributes need to be defined per each type, allowing to find quickly the desired object. At present, a web interface has been developed that implements cascading filters values, in a manner similar to the Apple iTunes software, to browse multimedia collections. It allows users to find and download a given object with few mouse clicks. In the next release of gLibrary, a REST API is planned that will allow to issue queries and manage uploads and downloads using the nowadays very popular and easy-to-use web 2.0 paradigm of the HTTP(s) GET, PUT, POST, DELETE commands supporting using X.509 certificates to ensure the compliance with the Grid Security Infrastructure requirements.

3.4 Workflows

Several workflow technologies have been integrated in the EnginFrame framework and can use through it. In the context of the A-WARE [2]– project, a workflow designer application, a workflow orchestrator service and a workflow repository service have been developed to implement in EnginFrame the main tasks of designing, storing, executing, monitoring and orchestrating workflows and the involved resources. The Grid middleware adopted by A-WARE was Unicore [15], but the services developed could be interfaced to other middlewares as well, such as gLite.
The MOTEUR workflow engine [16] has also been tested to enact the execution of workflows designed with the Taverna Workbench [17] in a scalable way. Since MOTEUR is a Java application, its execution has been easily integrated in EnginFrame providing

users with the opportunity to submit workflow as if they were normal jobs. Workflows were based on simple WSDL web services available on the Internet.

3.5 Visualization software and hardware

A client with pluggable image processing and visualization tools is available through the Web portals powered by EnginFrame. Traditional 2D visualization of graphics and GUI-based applications is possible thanks to the VNC Plug-in for EnginFrame. VNC[13] is an industry standard technology for Desktop remotization, since it provides the ability to view a remote computer Desktop on a network connected client with minimal bandwidth overhead, thus making possible to control a computer using DSL and low-level broadband home connections. The VNC plug-in available in EnginFrame covers only 2D applications. To obtain unmodified 3D performances on clients that are not capable of accelerated 3D, an extension to the classical architecture is required. Basically, it consists of compressing the 3D component of the scene (OpenGL) and send it to clients as a separate stream. This is achieved using a visualization plug-in implemented with IBM Deep Computing Visualization (DCV)[14], SUN VirtualGL[15] and HP RGS[16]. These are advanced 2D/3D GUI virtualization solutions that can provide 3D content from unmodified 3D OpenGL applications to clients. The approach may benefit from the availability, on the server side, of server nodes equipped with computer graphics hardware.

EnginFrame plays a central role in the remote visualization architecture: it allows users to run and control their interactive applications just using a common Web browser and to share sessions with other users (e.g., for teleconsulting, training, etc.).

A screen-shot of the Portal, running two interactive services (a 3D brain viewer and a computational chemistry software) is reported in Figure 2.

Client-server paradigm allows for lightweight, noiseless, low powered and small form factor clients, enabling the provision of complex 3D and 4D data on small client, remote displays and mobile devices. From a computational point of view, computing intensive tasks, such as integration among medical imaging modalities (CT, PET, MRI, fMRI), brain surface extraction, voxel based analysis, are performed on server side and can benefit from powerful hardware.

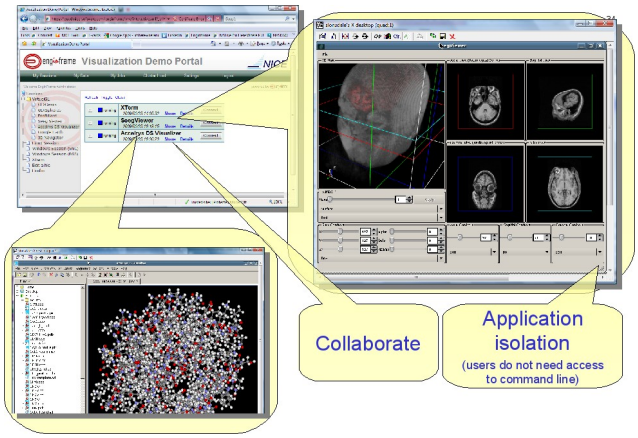

Figure 2: High-level and high-quality visualization services in EnginFrame. Two examples are shown: a brain 3D viewer and Accelerys Discovery Studio Visualizer software for chemical computing.

3.6 Data analysis and movement tools

Different sites can be integrated into a distributed architecture which enables users to access their data transparently and run complex analyses.

Data are usually stored in different locations - in most cases there is no shared file system or common namespace - and they need to be moved between locations.

For these reasons, components such as File and Replica Catalogs (to keep track where data are stored) and File transfer services (for scheduled, reliable file transfer) represent important layers in every Grid middleware stack.

Depending on the middleware, EnginFrame "speaks" the right language to interact with the appropriate component: e.g., LFC[17], SRM[18] and gridFTP[19] for gLite middleware or iCAT[20] and iCommands[21] for iRODS.

From the user perspective, distributed data are accessible in an easy and transparent way, regardless their physical location, in line with the concept of global virtual file system(s).

3.7 Resource discovery

Resource discovery systems are essential components in every Grid environment because they provide information about Grid resources and their status, with the possibility to use this information for monitoring and accounting purposes. gLite Information System and iRODS Resource Monitoring System are just two examples of solutions we have tested.

EnginFrame has the capability to internally monitor new resources availability. Every time the administrator of the portal publishes a new resource (plugins, computing and storage nodes, services,

[13]http://www.realvnc.com/vnc/index.html

[14]http://www-03.ibm.com/systems/deepcomputing

[15]http://www.virtualgl.org

[16]http://h20331.www2.hp.com/hpsub/cache/286504-0-0-225-121.html

[17]LCG File catalog, http://www.gridpp.ac.uk/wiki/LCG_File_Catalog

[18]Storage Resource Manager, https://sdm.lbl.gov/srm/

[19]GridFTP, http://www.globus.org/grid_software/data/gridftp.php

[20]ICAT, https://www.irods.org/index.php/iCAT

[21]iCommands, https://www.irods.org/index.php/icommands

features, etc.), it can be automatically visualized and used by the other users entitled to do so.
An example of high-level resource discovery system has been implemented in the context of the A-WARE project. The A-WARE architecture basically involves four layers, from the top to the bottom:

(1) Client layer (the Web browser);
(2) Front-end layer (the Grid portal, e.g. EnginFrame);
(3) Middle-layer: an ESB (Enterprise Service Bus) called ASB (A-WARE Service Bus), an integration layer providing many services and functionalities. E.g. applicative services, Grid integration, Workflow orchestrator, etc;
(4) Grid layer, Unicore.

Grid services are defined in the back-end Grid environment and automatically exposed on the ASB.
The portal, interfacing the ASB, was thus able to dynamically expose EF services, ASB services, and Grid services (through the ASB).

3.8 Domain-specific computational applications

This aspect is widely described in the next section reporting the use case from Life Sciences.

4 A USE CASE

4.1 Application design

We present in this section an example of bioinformatics application integrated in a portal based on EnginFrame and successfully tested and used in production mode by the Functional Genomics laboratory of the National Cancer Research Institute of Genoa.
The service developed, Survival Online (available at http://ada.dist.unige.it:8080/enginframe/bioinf/bioinf.xml), is a novel method to perform correlations between microarray gene expression data and clinic-pathological data through a combination of available and newly developed processing tools.
Microarray data analyses procedures are usually made of several steps ranging from preprocessing (background correction, normalization, summarization) to the statistical analysis of differential gene expression. This may involve different computer programs, ranging from specialized tools and statistical software environments to simple spreadsheets. It is therefore difficult for single researchers to analyze large datasets just using their local resources and, even if it is possible, it requires time, costs and skills.
The service becomes a good solution to the challenge of translating a complex manual procedure into an automated and consolidated Web-based process which can benefit from a back-end Grid infrastructure.
The automated procedure is shown in Figure 3. It represents the workflow of the processing steps required by the analysis.

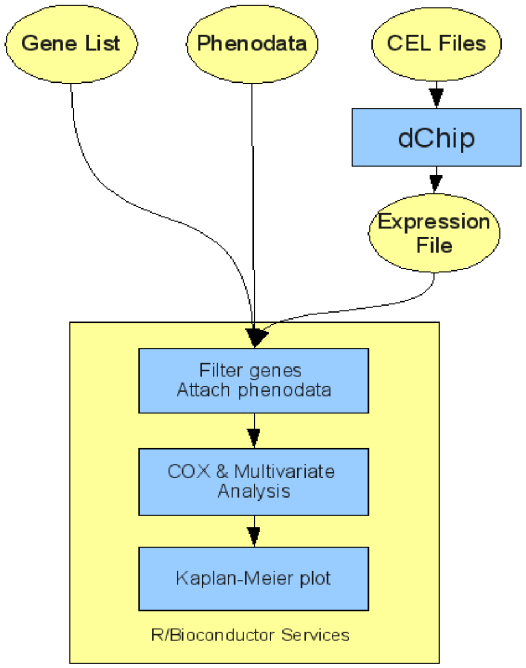

Figure 3: Survival analysis workflow

As first step, microarray data are preprocessed (using background correction and normalization methods) and gene expression values are obtained by running a parallel version of dChip [18].
dChip software, which had been ported on Linux platforms and parallelized, is able to run on both cluster environments and distributed Grid infrastructures, like EGEE. In this scenario, the added value of using Grid technologies is the possibility to analyze a large number of arrays in parallel thus reducing computational times and data transfers.
The following processing steps further evaluate expression values by means of several R/Bioconductor methods [19].
First, under/over-expressed genes are filtered so that only a user-defined set of genes is retained. Then, specific phenodata are associated to samples. Finally, a Cox regression analysis, possibly combined with a multivariate analysis, is performed and the Kaplan-Meier survival plot is computed [20].The Kaplan-Meier model graphically describes a step function with sudden changes in the estimated probability whenever an event is observed. It estimates the survival function from life-time data. In this context, it is used to measure the fraction of patients living for a certain amount of time after treatment.
An example of resulting Kaplan-Meier plot is shown in Figure 4.

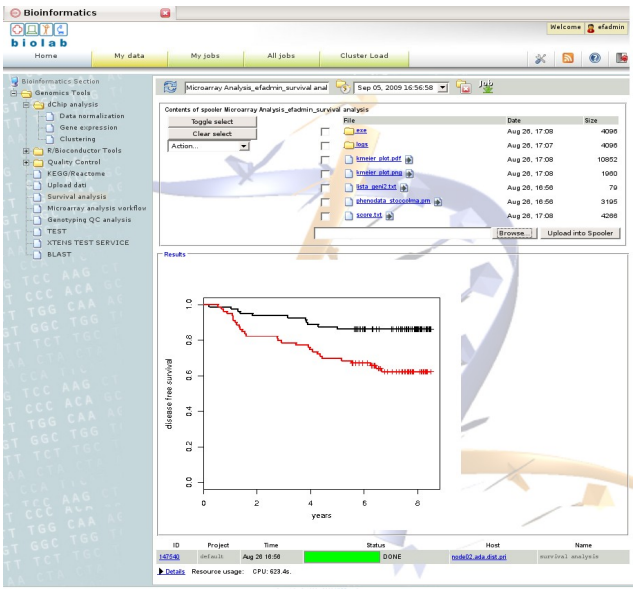

Figure 4: Results visualization in the Spooler area of the portal.

4.2 Service development

The application presented in the above subsection has been integrated in a Science Gateway based on EnginFrame. Thanks to EnginFrame, it has been possible to develop new portal services with very little effort.
EnginFrame services are made of an XML description, input parameters and the action to be executed. Normally, all the services exported by EnginFrame are described in a XML file called Services Definition File (SDF). For the sake of completeness and clearness, a fragment of SDF file for Survival analysis service is reported in Figure 5.

```
<ef:service id="ma_analysis">
<ef:name>Survival analysis</ef:name>
        <ef:info>
        <p> The service is able to analyze gene expression microarray data [...] </p>
        </ef:info>

        <ef:option id="analysis_name" label="Analysis name" type="text"></ef:option>
        <ef:option id="chip_type" label="Chip type: " type="list">
          <ef:option label="HG-U133A">hgu133a</ef:option>
          <ef:option label="HG-U133Plus2">hgu133plus2</ef:option>
        </ef:option>

        <ef:option id="FILE" label="Upload files .CEL" multi="true" target="file" type="rfb" />

        <!-- code here -->

    <ef:action id="submit" label="Submit job" result="text/xml"><![CDATA[
        EF_COMMAND="${EF_PLUGIN_PATH}/bin/ma_serv.sh"
        export EF_COMMAND
        ${EF_PLUGIN_PATH}/bin/job.submit
    ]]></ef:action>
</ef:service>
```

Figure 5: Fragment of SDF code

The fragment shows the basic elements for the description of any service. The *ef:option* tags represent the input parameters for the service and are transformed to HTML form fields by the EnginFrame Server according to their type. When a user requests the service execution, pressing the "submit" button, the script contained in the *ef:action* (in this specific case *EF_COMMAND="${EF_PLUGIN_PATH}/bin/ma_serv.sh"*) is submitted to the EnginFrame Agent which executes it on the back-end resource tier. The input parameters of the service provided by the user in the HTML form fields are passed to the script as environment variables. Looking at the specific example in Figure 4, the option with *id="FILE"* is rendered as a HTML file widget. When the user presses the *"Submit job"* button, the selected file is uploaded (via HTTP) from the user's machine to the spooler area created for the submitted service and then a job is submitted to the back-end infrastructure (gLite, SRB, LSF, etc., depending on the plug-in used). The path of the transferred file is stored in the environment variable *FILE*. Layout and graphical aspects of the portal are kept separated from the services definition and involve XSL programming. EnginFrame system XSL provides libraries coping with the translation of XML into HTML and this makes the task of configuring the graphical layout straightforward.
The whole development process is very rapid thanks to the possibility of re-using existing scripts and to the framework ability to dynamically catch any changes in services and layout avoiding in this way complex and tedious deployments iterations.

A graphical overview of the developed service is given by the portal screen-shots shown in Figures 4 and 6.
Figure 6 represents the user interface for application parameters and data upload.

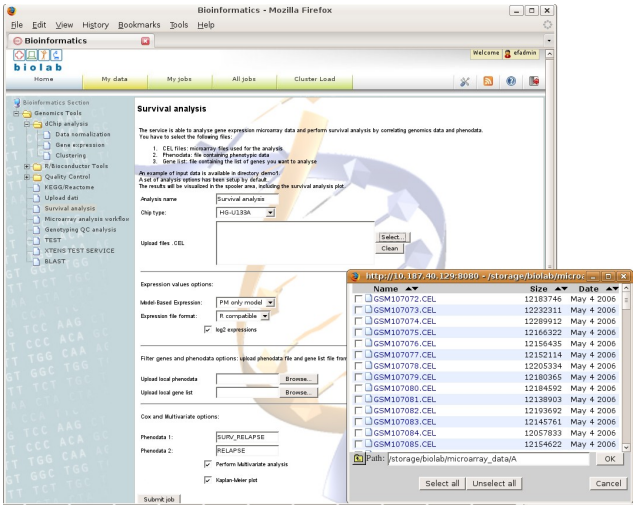

Figure 6: Service interface for data upload and analysis options selection

The user is first enabled to select - remotely or locally - datasets or single samples thereof, as well as single genes or lists of genes.
Expression values of selected genes are then correlated with sample annotation data by running mono- or multi-variate Cox regression, depending on user preferences.
Survival analyses results, together with the Kaplan Meier plot, are shown in the spooler area of the portal (see Figure 4). Execution status of the analysis (*DONE*) is visible as well.

The system was tested using publicly available breast cancer datasets and GO (Gene Ontology) derived gene lists or single genes for survival analyses.

5 CONCLUSIONS

In this paper we present a framework to integrate Grid resources and application tools according to the Science Gateway paradigm.
The goal is to deliver customized graphical user interfaces for those applications which require many computational resources, so they need to be performed either on Grid infrastructures or on High-Performance Computing facilities.
Based on a service-oriented framework, EnginFrame provides a platform for many kinds of Science Gateways belonging to many domains
A bioinformatics application for the analysis of Affymetrix GeneChip microarrays, successfully deployed at IST, National Cancer Research Institute of Genoa, has been presented as an example of EnginFrame usage and benefits.
As future work, special attention will be given to Web 2.0 technologies as a new way to enrich user interface and allow scientists to run their analysis through social web sites.

ACKNOWLEDGEMENTS

This work is developed within the Italian FIRB project ITALBIONET ("Rete italiana di bioinformatica"). We would like to thank Carla Milani from IBM Italian University Relation Office who made the competition of the University of Genoa in the 2008 edition of IBM SUR Grant possible, and prof. Enrico Giunchiglia, formal winner of the grant.
Thanks also go to Dott. Ulrich Pfeffer from National Cancer Research Institute, Genoa, for his support in experiment design and validation phases.
Finally we would like to acknowledge the support of the e-Science Institute in Edinburgh.

REFERENCES

[1] Yang, X., Dove, M.T., Hayes, M., Calleja, M., He, L., Murray-Rust, P. (2006) Survey of Major Tools and Technologies for Grid-enabled Portal Development, Proceedings of the UK e-Science All Hands Meeting 2006

[2] Gentzsch, W., (2009) Grid and Cloud Portals for Design, Simulation, and Collaboration, Parallel, Distributed and Grid Computing for Engineering, Saxe-Coburg Publications, pp. 83-10

[3] Menday, R., Hagemeier, B., Schuller, B., Snelling, D., van den Berghe, S., Cacciari, C., Melato, M. (2007) A One-Stop, Fire-and-(almost)Forget, Dropping-off and Rendezvous Point. Springer, 2007.Lecture Notes in Computer Science;4375), doi: 10.1007/978-3-540-72337-0_22

[4] Lederer, H. and Alessandrini, V. (2008) DEISA: Enabling Cooperative Extreme Computing in Europe, Parallel Computing: Architectures, Algorithms and Applications. Volume 15 Advances in Parallel Computing; eds. C. Bischof et al. (ISBN: 978-1-58603-796-3), IOS press, p. 689.

[5] Andronico, G., Ardizzone, V., Barbera, R., Catania, R., Carrieri, A., Falzone,A., Giorgio, E., La Rocca, G., Monforte, S., Pappalardo, M., Passaro, G., Platania, G. (2005) GILDA: The Grid INFN Virtual Laboratory for Dissemination Activities, TRIDENTCOM 2005: 304-305.

[6] Kacsuk, P. and Sipos, G., (2005) Multi-Grid, Multi-User Workflows in the P-GRADE Portal Journal of Grid Computing, Vol. 3, No. 3-4, Springer Publishers, pp. 221-238

[7] Novotny, J., Russell, M., and Wehrens, O. (2004) GridSphere: an advanced portal framework. Euromicro Conference, 412–419.

[8] Scardaci, D., Scuderi,G., (2007) Managing Confidential Data in the gLite Middleware, Enabling Technologies, IEEE International Workshops on, pp. 298-299, 16th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2007).

[9] Torterolo, L., Porro, I., Venuti, N., Gatti, S., Calanducci, A., Scifo, S., and Fato, M. (2007) BMPortal: A Bio Medical Informatics Framework EGEE User Forum 2007, Manchester, UK, 176.

[10] Zhu, B., Marciano, R., and Moore, R., (2009) Enabling Inter-repository Access Management between iRODS and Fedora". Accepted by the 4th International Conference on Open Repositories. Atlanta, Georgia, USA. May 18-21.

[11] Beltrame F, Maggi P, Melato M, Molinari E, Sisto R, and Torterolo L. (2006) SRB Data Grid and Computer Grid Integration via the EnginFrame Grid Portal, SRB Workshop San Diego Supercomputing Center.

[12] Scifo, S. (2007) GSAF Grid Storage Access Framework. Enabling Technologies: Infrastructure for Collaborative Enterprises, 296–297.

[13] Torterolo, L., Corradi, L., Canesi, B., Fato, M., Barbera, R., Scifo, S., Calanducci, A., Scardaci, D., Parisi, S., and Scuderi, G. (2008) A new paradigm to design, implement and deply Grid oriented application: a biomedical use case". Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and Healthcare.

[14] Calanducci, A., Castrillo, F.P., Pollàn, R.R., del Solar, (2008) Enabling Digital Repositories on the Grid, Advanced Engineering Computing and Applications in Sciences, International Conference on, pp. 45-50.

[15] Clementi, L., Cacciari, C., Melato, M., Menday, R., Hagemeier, B., (2007) A Business-Oriented Grid Workflow Management System Proceedings of 3rd UNICORE Summit 2007, Springer, LNCS 4854, pp.131-140.

[16] Glatard, T., Montagnat, J., Lingrand, D., and Pennac, X. (2007) Flexible and efficient workflow deployment of data-intensive applications on grids with MOTEUR, International Journal of High Performance Computing and Applications.

[17] Oinn, T. and et al.(2000) Taverna: Lessons in creating a workflow environment0 for the life sciences, Concurrency Computat: Pract Exper, 1–36.

[18] Corradi, L., Fato, M., Porro, I., Scaglione, S., and Torterolo, L. (2008) A Web-based and Grid-enabled dChip version for the analysis of large sets of gene expression data. BMC Bioinformatics, 9:480.

[19] Gentleman, R, Carey, J. V., Huber, W., Irizarry, A.R, and Dudoit, S. (2005) Bioinformatics and Computational Biology Solutions Using R and Bioconductor. Statistics for Biology and Health, Springer.

[20] Kaplan, E and Meier, P. (1958) Nonparametric estimation from incomplete observations. J Am Stat Assoc, 53:457–481.