J Grid Computing (2012) 10:769–790
DOI 10.1007/s10723-012-9247-y

A Single Sign-On Infrastructure for Science Gateways on a Use Case for Structural Bioinformatics

Sandra Gesing · Richard Grunzke · Jens Krüger · Georg Birkenheuer · Martin Wewior · Patrick Schäfer · Bernd Schuller · Johannes Schuster · Sonja Herres-Pawlis · Sebastian Breuers · Ákos Balaskó · Miklos Kozlovszky · Anna Szikszay Fabri · Lars Packschies · Peter Kacsuk · Dirk Blunk · Thomas Steinke · André Brinkmann · Gregor Fels · Ralph Müller-Pfefferkorn · René Jäkel · Oliver Kohlbacher

Received: 14 December 2011 / Accepted: 25 July 2012 / Published online: 23 November 2012

Abstract Structural bioinformatics applies computational methods to analyze and model three-dimensional molecular structures. There is a huge number of applications available to work with structural data on large scale. Using these tools on distributed computing infrastructures (DCIs), however, is often complicated due to a lack of suitable interfaces. The MoSGrid (Molecular Simulation Grid) science gateway provides an intuitive user interface to several widely-used applications for structural bioinformatics, molecular modeling, and quantum chemistry. It ensures the confidentiality, integrity, and availability of data via a granular security concept, which covers all layers of the infrastructure. The security concept applies SAML (Security Assertion Markup Language) and allows trust delegation from the user interface layer across the high-level middleware layer and the Grid middleware layer down to the HPC facilities. SAML assertions had to be integrated into the MoSGrid infrastructure in several places: the workflow-enabled Grid portal WS-PGRADE (Web Services Parallel Grid Runtime

This work is supported by the German Ministry of Education and Research under project grant #01IG09006 (MoSGrid) and by the European Commission's 7th Framework Programme under grant agreement #RI-261556 (EDGI), #RI-261323 (EGI-InSPIRE), #261585 (SHIWA), and #RI-283481 (SCI-BUS).

S. Gesing and R. Grunzke contributed equally in this work.

S. Gesing (✉) · J. Krüger · O. Kohlbacher
University of Tübingen,
Tübingen, Germany
e-mail: sandra.gesing@uni-tuebingen.de

R. Grunzke (✉) · R. Müller-Pfefferkorn · R. Jäkel
Technische Universität Dresden,
Dresden, Germany
e-mail: richard.grunzke@tu-dresden.de

G. Birkenheuer · J. Schuster · G. Fels
Universität Paderborn, Paderborn,
Germany

M. Wewior · S. Breuers · L. Packschies · D. Blunk
Universität zu Köln, Köln, Germany

P. Schäfer · T. Steinke
Konrad-Zuse-Zentrum für Informationstechnik
Berlin, Berlin, Germany

B. Schuller
Forschungszentrum Jülich, Jülich, Germany

S. Herres-Pawlis
Ludwig-Maximilians-Universität München,
München, Germany

Á. Balaskó · M. Kozlovszky · A. S. Fabri · P. Kacsuk
MTA SZTAKI, Budapest, Hungary

A. Brinkmann
Johannes Gutenberg-Universität Mainz, Mainz, Germany

and Developer Environment), the gUSE (Grid User Support Environment) DCI services, and the cloud file system XtreemFS. The presented security infrastructure allows a single sign-on process to all involved DCI components and, therefore, lowers the hurdle for users to utilize large HPC infrastructures for structural bioinformatics.

Keywords Single sign-on · Science gateway · Security · DCIs · Structural bioinformatics

1 Introduction

Structural bioinformatics and computational chemistry have become indispensable tools in many fields of biomedical research. Molecular dynamics methods, quantum chemical methods, and protein-ligand docking provide deep insights into the structure of biomolecules and their interactions and are essential tools in such diverse areas as materials science and drug design. While very powerful, most of the tools and applications used for computational chemistry calculations reflect the complexity of the underlying scientific theories. Using these tools thus requires a lot of experience. Their usability is often limited and frequently deters novice users.

The computational complexity of these methods makes them ideal candidates for high-performance computing infrastructures [38]. DCIs, however, have usability issues of their own, which limit their acceptance in the scientific community.

Overcoming these usability issues and thus enabling the use of DCIs for a broader user community was the key goal when the MoSGrid (Molecular Simulation Grid) project [16] was conceived. It is part of the German Grid Initiative (D-Grid) and is designed to address the requirements of both commercial and academic users.

MoSGrid offers a science gateway for the computational chemistry community, providing easy access to tools from the field of quantum chemistry, molecular dynamics, and docking. Currently, the MoSGrid community consists of about 110 users or working groups, respectively. At this stage, the science gateway is opened for beta testing to about 20 users from academia and industry whose feedback and demands are invaluable for further development. It is planned to offer the science gateway to the whole community in the near future. Novice and advanced users are enabled to run their computations on Grid resources. They are assisted by graphical user interfaces with different levels of sophistication to accommodate both user groups. Additionally, standard methods for specific problem classes are offered. MoSGrid provides a framework for developing, storing and providing simple and complex workflows. Furthermore, users are enabled to archive the results of their calculations in a repository and share them with other users.

Having left the first prototypic state, developments in MoSGrid continue to focus on security requirements of different environments. Distributed computing infrastructures are accessible by a number of users from different locations at the same time. The broad user community has to be provided with an infrastructure that efficiently protects their know how and molecular data.

The MoSGrid science gateway lowers the barrier of utilizing HPC infrastructures by allowing secure access to UNICORE infrastructures [47] via a single sign-on concept based on SAML [45]. This paper describes the recent developments in the MoSGrid security infrastructure. Considering both the demands of academic and commercial users, the paper focuses on the integration and interoperability of the employed components with respect to user authentication and authorization and data security.

The paper is structured as follows: Section 2 introduces to the application domain, the MoSGrid infrastructure, and related work. The developments for the MoSGrid security infrastructure are presented in Sections 3 and 4 demonstrates domain specific workflows utilizing the security infrastructure.

2 Background

Some of the application cases of structural bioinformatics and computational chemistry, in particular applications in pharmaceutical industry, impose strict requirements on data security in order to protect potential intellectual properties. We will discuss these issues briefly and then examine how a good level of security can be obtained, while still providing a convenient single sign-on access.

2.1 Application Domain

Structural bioinformatics deals with the prediction and analysis of the structure, and the mechanisms of function of biological macromolecules [8], including proteins, nucleic acids [26], lipids, and sugars. Major issues handled by this field are for example the improvement of drug targeting [10], the derivation of enzymatic design principles, or the development of computational models that describe structure function relations. Knowledge is gained by both, experimentally derived structures, as well as computational models. Regarding the computational methods, we focus on three selected areas:

1. quantum chemical calculations (QC) dealing with the electronic structure of molecules
2. molecular dynamics (MD) employing classical mechanics approaches
3. docking estimating ligand-receptor interactions

Within the QC domain, density functional theory, coupled-cluster methods as well as perturbation methods are used to calculate electronic structures. Due to the restrictions of common codes like Gaussian [13], Turbomole [49], and Orca [36], long-running simulation jobs (duration up to weeks) with only a small degree of parallelization are the typical use case. The scientific community using quantum chemistry encompasses inorganic, organic, and physical chemists. The majority of calculations deals with small structures (200 atoms or less).

Significantly larger molecules of hundreds of thousands atoms are computed in the MD domain. In order to access relevant time scales a classical mechanics approach is followed, treating molecules as interacting point masses. The computation of force fields are typically parallelized and scale well even for a large number of CPUs. This scientific community is driven by organic and bio-organic chemistry, while inorganic systems with metal interactions and delocalized electrons are difficult to handle by these methods.

Within the docking domain the evaluation of ligand-receptor interactions is the main focus and by means of that enhancing modern drug design. A multitude of possible ligand structures in numerous conformations and directions is fit into an active site. Moreover, large molecule ensembles of potentially active drugs have to be managed, enabling linear scaling as the individual docking simulations are independent of each other. Key application for this type of calculation are found in the area of computer-aided drug design, both in academia and industry.

All of these methods, molecular dynamics in particular [39], potentially produce large amounts of data.

2.1.1 Sensitive Data in Research and Science

Both input and output data of these simulations have to be stored securely and reliably. In particular, for commercial applications already the structures that enter a simulation represent valuable intellectual property and need to be secured accordingly. Similarly, the results of simulations, which often take months to compute, are valuable and need to be protected from data loss. Keeping that in mind, it is essential that the scientist has full control over the access policies to all of his simulation data. With respect to a collaborative work strategy, the option to share selected data with co-workers is also an essential feature. One has to differentiate what kind of data should be shared. The pure simulation data, such as intermediate molecular structures, raw trajectories, or unanalyzed energies are usually only of interest for closest collaborators. In contrast, access should be granted to a broader community after the scientific results have been published.

Within an academic environment, to publish in a scientific journal is a prerequisite before analyzed and approved data is shared with third parties. In collaboration with industry partners the focus shifts to other priorities. Publications are out of question before a patent application is filed to protect the data from competitors' interests.

In both areas, users of different experience want to use the DCI, ranging from expert users to absolute beginners. In all cases, they face the common problem to get access to computational facilities. This usually includes lengthy proposals and application processes as well as non-standardized usage policies. The latter often involves various forms of shell scripting, different schedulers, and limited monitoring abilities. Another critical obstacle is the secure data transfer and storage of simulation data. In any cas,e a highly secure exchange of data including robust encryption and authentication techniques is immanent. MoSGrid offers an approach to overcome the aforementioned obstacle and fulfill essential requirements. These consist of industrial and academic environments.

The security demands in an industrial context comprise a multitude of details;

- the data shall be transferred with robust encryption,
- the data shall not be visible or modifiable by third parties, and
- the jobs and even their existence shall not be disclosed to third parties.

In academic environments, the demands are different due to the more open and collaborative approach to work. This kind of approach generates a different set of challenges.

- A large degree of transparency in terms of versions and changes for all contributors is desired because the project data is handled like a "living" document.
- When a project is highly distributed, simultaneous access to data can cause problems with naming schemes and versions as well as concurrency issues.
- During a long-term project, a mass of preliminary data is produced, which cannot be stored forever.

Hence, strategies for secure long-time archiving and reliable removing of data have to be implemented. Science gateways furthermore have to ensure a high degree of data persistence, i.e. the availability of data in the future and protection from data loss. This is facilitated by unique identifiers and suitable replication policies in the MoSGrid data repository.

2.2 The MoSGrid Infrastructure

In general, a *science gateway* can be defined as a single point of entry giving access to scientific software of a specific application domain operating across organizational boundaries. We characterize a *Grid portal* as a web-based science gateway utilizing Grid infrastructures and demanding solely a web browser on the user's side.

The workflow-enabled Grid portal WS-PGRADE [9] is the basis for the MoSGrid science gateway and the highly flexible graphical user interface for gUSE [27]. The latter provides a large set of services for the management of workflows in DCIs (see Section 2.2.1).

XtreemFS [21] was chosen as distributed file system for MoSGrid to safeguard data and provide each resource with secure access. It is described in detail in Section 2.2.2. The Grid middleware UNICORE is used to access DCIs and manage jobs as well as data transfer. Its integration into the project is described in Section 2.2.3.

2.2.1 gUSE and WS-PGRADE

gUSE provides a collaborative, community-oriented application development environment, where developers and end-users can share sophisticated (layered and parameter sweep enabled) workflows, workflow graphs, workflow templates, and ready-to-run workflow applications. The workflows are able to use a large set of virtualized, high-level DCI services. This environment is capable of providing interoperation among classical service and desktop Grids, clouds and clusters, and unique web services in a scalable way.

Internally, gUSE is implemented as a set of web services that bind together the different components, like a workflow storage, an application repository, an information system, and a monitoring system. The architecture (see Fig. 1) contains a workflow engine (called Zen), which provides seamless enactment of DAG (Directed Acyclic Graph) based workflows and supports also extra features such as embedded workflows, timed workflows, web service type inputs and database access. The workflow engine manages the workflows and hands over single jobs of the

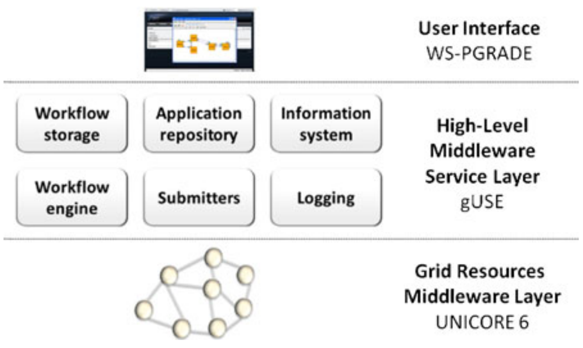


Fig. 1 gUSE architecture

workflows to so-called submitters. Submitters are responsible for the authentication mechanism to the underlying DCI and the management of the jobs. For each supported middleware for underlying DCIs (e.g., UNICORE, Globus Toolkit [11], gLite [29]) a specialized submitter has been developed to provide access. Currently, the various submitters allow two different kinds of authentication depending on the DCI: authentication via SAML assertions and authentication via proxy certificates.

WS-PGRADE is an easily usable, highly flexible, co-operative, graphical user interface of gUSE. It uses the client APIs (Application Programming Interface) of gUSE services to turn user requests into sequences of gUSE specific web service calls. Users can access WS-PGRADE via HTTP and HTTPS. A graph editor component is offered, which is a JAVA Webstart based application and can be downloaded from WS-PGRADE via the browser. The editor can be used to define the static skeleton of workflows, while the HTML pages of WS-PGRADE provide interfaces to add content to graphs, to generate complete Grid and web service applications.

Nowadays, WS-PGRADE portals are operating and serving numerous user communities and international projects, providing access to multi-institutional Grids and Grid based virtual organizations as generic DCI portals or eScience gateways within Europe. gUSE and WS-PGRADE have become open-source in 2011 and the code is available under the GPL license at Sourceforge.

2.2.2 XtreemFS

XtreemFS is an object-based distributed file system, i.e., file data and metadata are stored on different servers. Its architecture can be seen in Fig. 2. The object storage devices (OSDs) store the content of a file split into fixed-size chunks of data (the objects). The metadata and replica catalogs (MRCs) contain for example the filename, DN of the owner, and the directory tree. These servers are connected by a FUSE client or a client using the Java API. The FUSE client mounts the volumes of the MRC and translates file system calls into Remote Procedure Calls (RPCs), which are sent to the respective servers.

The advantages of using a globally distributed and replicated file system such as XtreemFS are a global namespace, good scalability, and increased data integrity through replication.

Authentication in XtreemFS is handled by both OSD and MRC, while authorization is handled by the MRC only. There is no communication between MRC and OSDs when authorizing file access to a client. Instead the client requests a lease on a file from the MRC. A lease represents the authorization level and is valid for a limited timespan, and as such has to be renewed periodically. To access a file stored on an OSD, the client sends a lease confirming that it is allowed to access the file.

2.2.3 UNICORE

UNICORE is a four-layered system, comprising the client, proxy, services, and target system layer (see Fig. 3).

The target system layer comprises the interface to the local operating system, file system, and

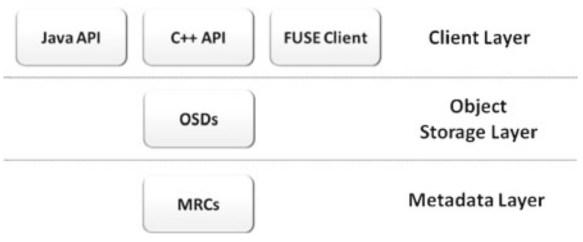


Fig. 2 XtreemFS architecture

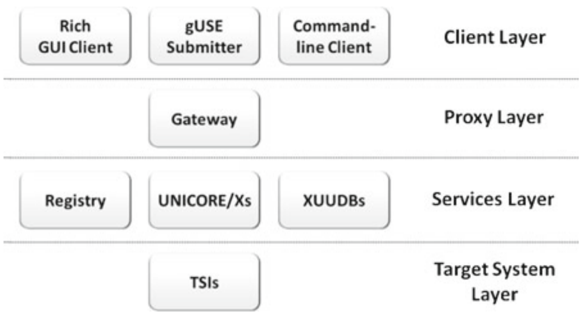


Fig. 3 UNICORE architecture

resource management (batch) system. A component named *target system interface* (TSI) is responsible for submitting jobs, performing file I/O, and checking job status.

The services layer comprise one or more UNICORE/X servers hosting the UNICORE services. By default, UNICORE offers services for resource discovery (Registry service), job execution (Target System Factory and Target System Services), job management, and file access (Storage and File Transfer Services). Additionally, workflow execution services are provided. The UNICORE services are SOAP Web Services that follow the WSRF paradigm [41], which means that entities like jobs, systems, and storages are modelled as individually addressable resources.

A UNICORE client invokes operations on SOAP Web Services over HTTPS connections. Usually, connections are made through the gateway, which acts as a proxy, meaning that sites need to open only a single port in their firewall. Client-authenticated SSL is used, i.e., clients need a certificate issued by a trusted authority.

Communication between the gateway and the UNICORE/X server and between UNICORE/X servers is again using client-authenticated SSL. This raises the issue of trust delegation, i.e., how a server can perform actions on the user's behalf. UNICORE uses trust delegation assertions as described in Section 2.2.

In a UNICORE job, expressed in JSDL (Job Submission Description Language) [2], data staging is used to download data into the job directory before the job is executed, and to upload result files to a permanent storage when the job has finished. This data staging mechanism is extensible. The characteristic has been exploited to provide support for the XtreemFS file system. If XtreemFS is mounted locally at a site, downloading or uploading files from XtreemFS can be done via a simple local copy command. The user is enabled to use URLs with the schema "xtreemfs://" to work directly with an XtreemFS storage. A UNICORE/X server can be configured to either copy such files locally or to access a remote storage depending on how XtreemFS is available.

2.3 Related Work

Security is a key aspect for science gateways [27] on top of DCIs. Currently, the established basis for authentication in Grid middlewares (e.g., UNICORE, Globus Toolkit, gLite) are X.509 certificates. The basic security concept includes offering *single sign-on* to users. It is a principle for access control in distributed systems. The user has to authenticate himself just once and gains access to all connected systems without the need for further authentication procedures. Another main advantage is that the user does not have to maintain several means of authentication, meaning no multiple passwords or certificates for multiple systems are required.

Single sign-on relies on the principle of *trust delegation* by which systems can be allowed to act on behalf of the user. It is used, for instance, in workflow systems, where a whole workflow consists of multiple jobs. Using trust delegation, a workflow engine acting in the name of the user, can submit the individual jobs to suitable resources without further user interaction. This approach decouples job submission and user interaction.

To support single sign-on and thus trust delegation UNICORE uses the approach of *explicit trust delegation* (ETD) [46] in its dynamic style [6]. It allows the dynamic creation of jobs in the name of the user, though the trust relationships are still static. ETD advanced to its dynamic style offers increased flexibility, while maintaining robust security properties. In UNICORE signed trust delegation assertions, encoded in SAML, are used to provide these properties. SAML is an XML-based framework developed by the Security

Services Technical Committee of the Organization for the Advancement of Structured Information Standards (OASIS) [42]. It supports assertions, protocols, bindings, and profiles and is widely used in different ways, like handling web single sign-on, attribute-based authorization, and securing web services. For checking whether users are allowed to perform a particular action, UNICORE uses access control policies expressed in XACML 2.0 [40]. Each user identity is mapped to a set of attributes, which are then checked against a set of XACML policies.

Several standard groups use SAML as a basic concept for security and identity in their work. SAML assertions in UNICORE are used to express that a client delegates trust to a specific UNICORE server instance and it can contain several additional statements specifying the assertion in more detail. These can also be chained, meaning that an entity acting on the user's behalf can delegate trust to yet another entity, which is then also able to act on the user's behalf. SAML trust delegation assertions offer important security characteristics. They can be limited to one entity, to a specific validity time span, and to a trust chain of a maximum length. Furthermore, SAML is supported by various single sign-on infrastructures, which allow mapping of local accounts to federated identities. One is the widespread Shibboleth system [31] managed by the Internet2 Middleware Initiative (I2MI) [22]. The open-source software Shibboleth supports federated access control and allows scalable use of the technology. Users are enabled to securely access web-based resources. Service providers are enabled to interconnect their identity management systems and use attribute-based authorization to provide privacy control of the exchanged personal information. Shibboleth is production ready and is continuously evolving.

Other Grid middlewares like the Globus Toolkit or gLite implement trust delegation via GSI [12] (Grid Security Infrastructure) proxy certificates. GSI is a specification for secure communication in a Grid environment and is based on public key cryptography using certification authorities (CAs) and X.509 certificates. These proxy certificates have several disadvantages compared to trust delegation based on SAML. The proxy certificate is always transferred along with its private key, which is extremely sensitive since anyone, who possesses it, can impersonate the user. To mitigate this problem, the validity span is often severely limited, which creates new problems. Furthermore, it is impossible to reconstruct each step of a trust chain build with proxy certificates. To lessen the problem of a short validity time users can upload their certificate to MyProxy [48] servers and periodically generate proxy certificates valid for a certain period of time. A MyProxy server also lowers certain security risks, since the real user certificates with the private keys do not have to be stored on every machine used. However, it also creates new risks, because the central servers have to be very well secured. Furthermore, it does not improve the security of GSI proxy certificates, since they are still used and vulnerable.

Both approaches for trust delegation are based on X.509 certificates, which require that users go through a multistage application process to receive their user certificates. Additionally, based on their certificates, they have to create essential files like SAML assertions or proxy certificates to enable the trust delegation. These procedures are time-consuming and may discourage users to utilize DCIs. Therefore, several approaches are on the way to simplify the application process or to automatically generate the essential credential files.

The Java library GridCertLib [35] supports users of web-based science gateways by automatically obtaining X.509 certificates and using proxy certificates. The prerequisite is that the science gateway has access to a SAML assertion of a previous successful Shibboleth authentication. This library could be adapted for the use of SAML assertions and employed in the MoSGrid science gateway in case the German national Grid infrastructure will be extended for offering federated identities based on Shibboleth.

A similar concept has been implemented by the UK project SARoNGS [50]. However, the generation of a MyProxy certificate in the portal still needs the interaction of the users and a web service, which demands Shibboleth authentication. This mechanism is analogously used in WS-PGRADE and therefore the MoSGrid science gateway. In both solutions, the users are provided with an intuitive user interface to create their credentials without the need to use any command line invocations.

To further ease the authentication process, MoSGrid plans to integrate tools developed in the projects GAP-SLC [15] and GridShib [5]. The goal of GAP-SLC has been to produce concepts and tools for the use of short-lived credentials in portal-based Grids. One result is a plugin for the web browser Firefox which allows to utilize imported certificates for the login to portal frameworks. MoSGrid aims to offer the same login procedure based on the results of GAP-SLC additionally to the existing one based on username and password. In the GridShib project the gap between Shibboleth with SAML and PKI with X.509 is bridged for campus and Grid interoperability. Tools have been developed to bind SAML assertions to proxy certificates and to perform SAML-based authorization for Globus Toolkit-based web services. MoSGrid plans to use parts of the technologies to map X.509 certificates to the portal accounts and automatically create suitable SAML assertions in the science gateway.

The GENIUS portal supports the concept of X.509-based robot certificates [4]. These are not associated with specific users but with communities, applications, or science gateways. The certificates are handed over to the users on smart cards, which requires card readers connected to the users' computers. Users are authenticated via login and password in the GENIUS portal and are allowed afterwards to use DCIs via the robot certificate on the smart card. This solution has two major drawbacks. The first one is the need for additional hardware on the users' side. The second one is the duplicated additional effort for already implemented processes in Grid security infrastructures, like mapping user distinguished names (DN) to local accounts on HPC facilities.

In October 2010 the EU project EGI (European Grid Infrastructure) [7] presented the result of a questionnaire about requirements for authentication and authorization infrastructures for DCIs. It was answered by a number of projects from different domains. One result was that the major technologies currently in use include SAML and X.509 certificates and that the goal should be to bridge these different security domains by using for example Shibboleth. Since the MoSGrid science gateway already uses SAML, its security infrastructure can be easily adapted to rely on Shibboleth for user authentication instead of certificates.

3 The MoSGrid Security Infrastructure

The MoSGrid security infrastructure consists of four layers: the science gateway as an intuitive user interface, the high-level middleware service layer including gUSE and XtreemFS, the Grid middleware layer with UNICORE, and suitable HPC facilities in the D-Grid infrastructure (see Fig. 4). The science gateway infrastructure used in MoSGrid are described in Section 2.2.

MoSGrid employs security features of the open-source portal framework Liferay [23] and has extended WS-PGRADE, gUSE, and XtreemFS for the use of UNICORE and SAML assertions. These extensions affect three major domains of security in DCIs: user and credential management, workflow and job management, and distributed data management. These are described in the following sections.

3.1 User and Credential Management

The chosen WS-PGRADE version employs Liferay, which supports the JSR168 [1] standard and its successor JSR286 [37]. The authentication process to the portal is handled with the security features of Liferay. Liferay supports various authentication standards and protocols, e.g., LDAP (Lightweight Directory Access Protocol), single sign-on with CAS (Central Authentication Service), Open ID, and OpenSSO. Furthermore, it offers the option for reserving credentials so that the users are prevented from registering with special screen names and email addresses.

Liferay consists of a portlet container with default applications and a portal interface, which is deployed inside an application server. MoSGrid has chosen Apache Tomcat 6 [3] as the underlying application server. It implements the Servlet 2.5 [24] and JavaServer Pages 2.1 [25] specifications and their security models.

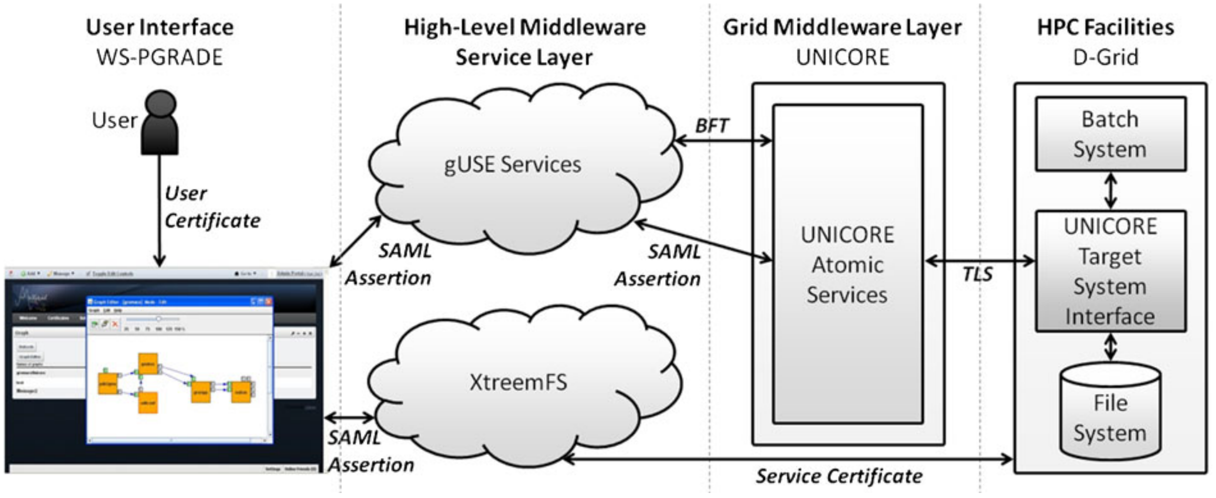


Fig. 4 The MoSGrid security infrastructure

Apache Tomcat handles the access control of users and programs to resources and the integrity of data during transfers via HTTP or HTTPS. Furthermore, the application server offers role-based authorization modules and supports the login with user name and password. Liferay utilizes these modules and extends the role-based authorization with more granular security mechanisms in the user management by providing organization, community, and group management. Organizations may present various divisions or various locations of a company and offer privately and publicly accessible pages of the portal. In contrast, communities are designed to allow access across organizational boundaries or to pages, which should be seen by all users of a portal.

To meet the needs of the computational chemistry community, the organization, and the community management are used in the MoSGrid science gateway. Hence, we implemented six hierarchical roles via Liferay: guests, novice users, advanced users, consortium users, developers, and administrators.

Guests are characterized by the absence of an account for the science gateway. However, they can obtain information about the project and about essential steps for getting access to the MoSGrid science gateway and its features. Liferay offers the option that an account can be created by an unknown user. As soon as users have a login created, they can apply for the MoSGrid community membership via email and their accounts will be assigned to the novice or advanced user role.

Novice users, within the meaning of being novice to structural bioinformatics tools, are classified as MoSGrid users. This role enables to choose pre-defined workflows to become acquainted to the tools and domain specific workflows. The latter are offered via intuitive graphical user interfaces, which lower the barrier for utilizing the tools as well as using them on high-performance computing facilities. Novice users are allowed to change the input and parameters and to invoke and monitor workflows. The access rights are implemented as a community role for MoSGrid users.

The access to additional features for creating and changing workflows is granted via the *advanced user* role.

Consortium users and *developers* are enabled to change gUSE and Grid settings in the portal, e.g., disk quotas for each user and available DCIs. Both roles are implemented as organization roles, which provides them with a further workspace in the portal. This workspace only includes the additional management features and the developer wiki. Consortium users may insert feature requests and developers add documentations about features and applications. Furthermore, developers are allowed to add new implemented features and portlets.

Finally, the *administrators* are additionally able to manage all credentials, users, organizations, and communities.

The user management is the first part of the single sign-on infrastructure. It only implements the access to the MoSGrid features in WS-PGRADE (e.g., creation of workflows), but not the access to the underlying DCI infrastructures. The login procedure to the portal is independent of the authentication procedure to the underlying DCIs. WS-PGRADE offers a certificate portlet which allows the users to create proxy certificates and SAML assertions based on their user certificates. The created credentials are stored on the portal server and are available to the submitters. This access is granted through a security token generated via the certificate portlet of WS-PGRADE. It represents the second part the SSO infrastructure.

3.1.1 Managing Security Tokens with a Certificate Portlet

Every user who wants to utilize UNICORE infrastructures has to obtain an X.509 user certificate from an appropriate certificate authority (CA). To protect the user's certificate it is fundamental to minimize its necessary transfers in the authentication process and the locations where it has to be stored.

WS-PGRADE achieves this goal by offering a certificate portlet for credential management without the need to upload personal certificates to the portal server. The original version of the certificate portlet only supported proxy certificate-based authentication via MyProxy servers. A new certificate portlet was thus created that provides features for various credential formats including SAML.

Applets ensure that processed data remains on the user's computer and signed applets additionally use policy files to ensure the integrity of the processed data. A signed applet has been integrated into the certificate portlet for generating SAML assertion files locally on the user's computer. The X.509 certificate of the MoSGrid science gateway is taken as a policy file and fulfills the supplemental purpose to be the recipient of the users trust delegation. The original applet, which is provided as open source by the UNICORE developers, allows to set the following options;

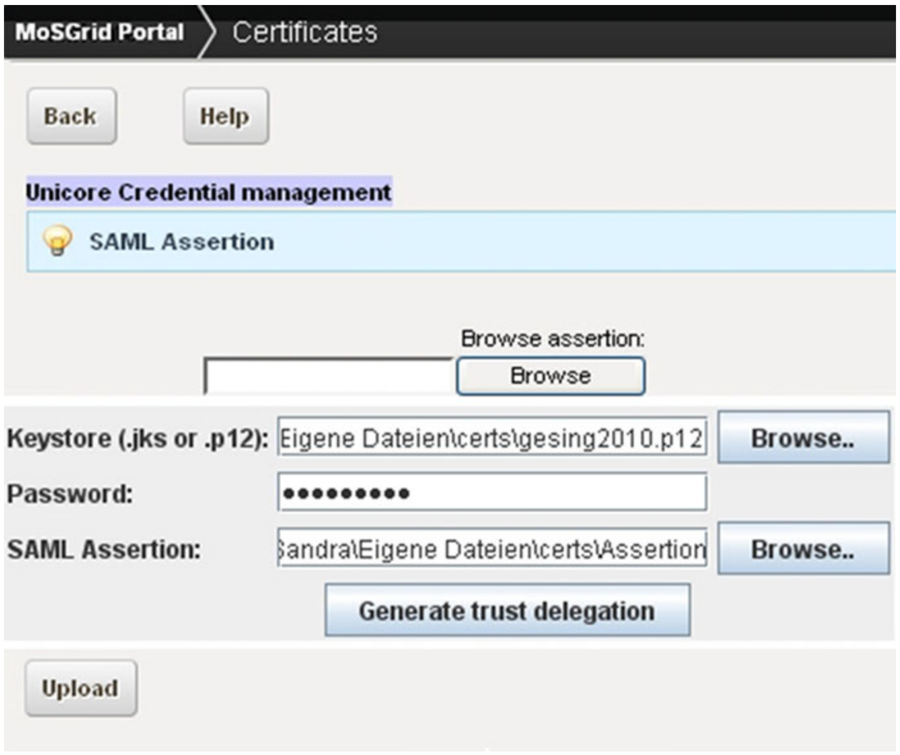
– location of the keystore
– password of the keystore
– name of the subject for the SAML assertion
– location of the SAML assertion
– alias for the keystore
– validity of the SAML assertion
– length of the trust chain

In the MoSGrid portal, the user only has to specify the location of his certificate on his computer, the corresponding password, and the location on his computer where the generated assertion file should be stored (see Fig. 5). The applet then locally and automatically generates an SAML trust delegation assertion file with the same validity as the user's certificate. The portlet can be easily adapted to produce credentials with a shorter validity period. The assertion file is uploaded to the portal to be available as the users security token.

Furthermore, the original certificate portlet is adapted to simplify its use. In MoSGrid, users do not have to distinguish between diverse options but are still able to use all relevant options regarding a SAML assertion file and its management, e.g., generating, uploading, and deleting the assertion file. The uploaded SAML assertion file sets the stage for the authentication processes in UNICORE, XtreemFS, and the domain specific portlets.

3.2 Workflow and Job Management

The collaborative, community-oriented application development environment of WS-PGRADE offers a graphical workflow editor and enables the users to create, change, invoke, and monitor workflows. The latter may contain jobs on local resources and distributed resources in Grid and cloud infrastructures. Existing workflows, workflow graphs, workflow templates, and sophisticated workflow applications can be shared via a local repository.



Keystore (.jks or .p12): Eigene Dateien\certs\gesing2010.p12 Browse..
Password: ●●●●●●●●●●
SAML Assertion: ;andra\Eigene Dateien\certs\Assertion Browse..
Generate trust delegation

Fig. 5 Trust delegation generation with integrated certificate portlet

WS-PGRADE allows to configure settings for various Grid middlewares and corresponding resources intuitively. In the case of UNICORE, MoSGrid advanced users are able to add UNICORE registries, which provide access to a number of infrastructures. Afterwards, the preferred UNICORE registry can be chosen by all users out of a list of configured registries. However, the whole integration process of UNICORE in WS-PGRADE additionally required the development of a submitter plug-in in gUSE as further described below.

gUSE provides a set of services for the management of workflows in DCIs including the data-driven workflow engine and submitters. Jobs within the same workflow may be configured for diverse DCIs and the workflow engine invokes each with an appropriate submitter. In general, gUSE submitters are Java-based applications developed to provide authentication mechanisms and the management of single jobs for a specific DCI. They implement the interface GridService of the workflow engine with methods for the management of jobs including authentication, authorization, and data-staging.

gUSE offers various submitters for Grid and cloud infrastructures, desktop Grids, and web services. In MoSGrid we have additionally developed the submitter for UNICORE [17] to enable submitting of jobs to UNICORE via WS-PGRADE. The submitter utilizes the UCC (UNICORE Commandline Client) libraries, implements authentication with SAML assertions, manages data-staging and invokes jobs.

To authenticate users with SAML assertions against a UNICORE infrastructure, the submitter requires access to three files: the SAML assertion file created via the certificate portlet, the X.509 certificate of the gateway to which the trust delegation are issued by users, and a truststore, which includes the public keys of the CAs used in the UNICORE infrastructure. The first file is unique for each user, the second and the third are the same for all users of the MoSGrid science gateway.

As soon as users upload their SAML assertion file via the certificate portlet to the portal server, the submitter is able to access the file. The public key of the MoSGrid science gateway is used by the certificate portlet to create the SAML assertion file. An administrator of the science gateway ensures that the X.509 certificate used for the trust delegation as well as the truststore is available for the submitter. Accordingly, the submitter employs these essential files to authenticate the user against the selected UNICORE middleware installation, which then checks whether the credentials are valid and authorizes the user based on the XUUDB or returns an error.

Once users are authenticated, the submitter creates a job on the targeted UNICORE resource. As a result, UNICORE automatically provides a job working directory on an HPC facility (USpace), which is solely accessible for the user who invokes the job. To invoke a job, the submitter defines its characteristics via JSDL. The parameters for the JSDL job description are set by the user in WS-PGRADE (e.g., the executable, filenames of input and output, number of MPI nodes). WS-PGRADE has been extended to offer automatically a list of available tools which are defined in the incarnation database (IDB) of UNICORE. The IDB of an UNICORE instance contains the installed software of the underlying DCI resources. Thus, WS-PGRADE provides up-to-date information to users about which tools are accessible to them by selecting a UNICORE registry.

The user has two alternatives to provide an application to the submitter; the first one is to choose an available tool on the selected registry, the second one is to upload an executable (see Fig. 6). As soon as users select a registry, WS-PGRADE connects to the UNICORE registry with the SAML assertion of the user and the list of tools is updated via AJAX (Asynchronous JavaScript and XML) [44]. When a tool is selected, its name is handed over to the submitter and the submitter inserts the tool name in the JSDL job description. Since the tool is available on the underlying DCI resource, this step is sufficient for the definition of the application in the job. In case the user fills in the name of an executable in WS-PGRADE, it is uploaded to the portal server and the binary is handed over to the submitter. Afterwards, the submitter uploads the executable to the USPACE and extends the JSDL job description accordingly.

Besides the executable or tool, JSDL also allows to define various parameters for the application. For standard settings (e.g., the expected wall time, number of required nodes), the MoSGrid science gateway offers single parameters which can be filled in by the users. Additionally, users can provide command line parameters. The JSDL job description is extended accordingly by the submitter for the different kinds of parameters. Input and output file names are also added to the JSDL job description.

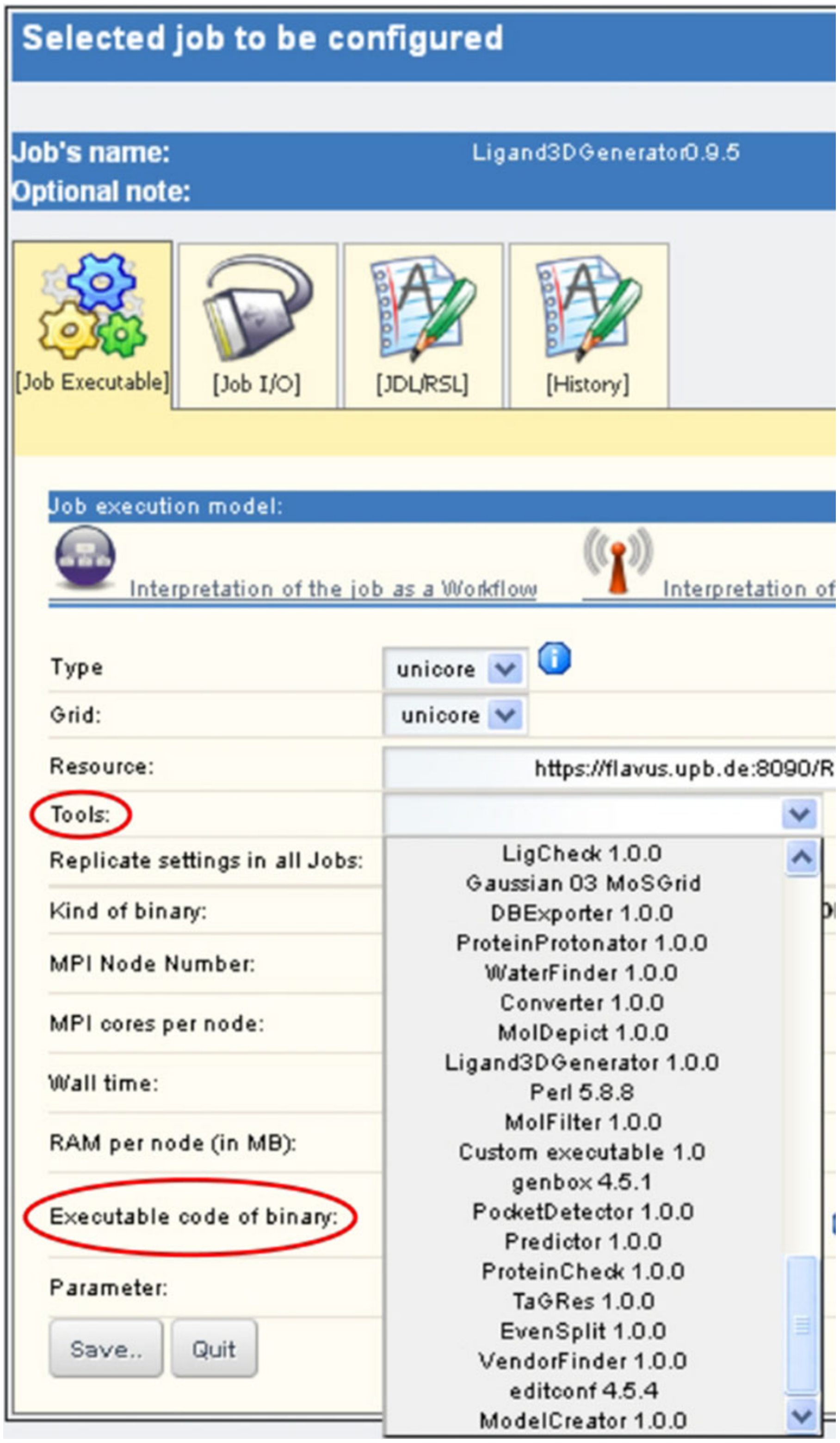
In the MoSGrid science gateway, two main principles are implemented; management of local files and of remote files. The submitter utilizes the BFT (Basic File Transfer) protocol for uploading or downloading local files belonging to a job to or from the USpace. Thus, only the names of the local files have to be added to the JSDL job description. In contrast, remote files are defined by a URL and the file handling is taken over by UNICORE and XtreemFS (see Section 3.3 for further details). The submitter adds the URL to the JSDL job description, but does not handle the files. Standard output files like STDOUT and STDERR are treated like local files and are downloaded by the submitter to the portal server.

3.2.1 Application Specific Module

gUSE provides a sophisticated web-based way to create, configure, and execute Grid applications on various types of DCIs. However, portal developers demand to use features and functionalities of gUSE from portlet codes. Thus, developers can focus on creating domain specific portlets that are tailored especially for the applications and for the users' needs e.g., the possibility to set inputs in a user-friendly way or make some post-process after the execution, such as picture or chart generation from the results. Authentication on Grid and cloud infrastructures, submission, and monitoring of workflows are handled jointly by various web-service components of gUSE.

To hide the complex architecture from developers, a new component called ASM (Application Specific Module) has been developed that can be used as an API. Every application included in the local repository of gUSE can be reused via a portlet using the ASM libraries. Applications in gUSE consist of workflows and corresponding parameters, input and output files. ASM provides various

Fig. 6 Configuration of a job in WS-PGRADE



Selected job to be configured

Job's name: Ligand3DGenerator0.9.5

Optional note:

Job execution model:

Type	unicore
Grid:	unicore
Resource:	https://flavus.upb.de:8090/R
Tools:	
Replicate settings in all Jobs:	
Kind of binary:	
MPI Node Number:	
MPI cores per node:	
Wall time:	
RAM per node (in MB):	
Executable code of binary:	
Parameter:	

LigCheck 1.0.0
Gaussian 03 MoSGrid
DBExporter 1.0.0
ProteinProtonator 1.0.0
WaterFinder 1.0.0
Converter 1.0.0
MolDepict 1.0.0
Ligand3DGenerator 1.0.0
Perl 5.8.8
MolFilter 1.0.0
Custom executable 1.0
genbox 4.5.1
PocketDetector 1.0.0
Predictor 1.0.0
ProteinCheck 1.0.0
TaGRes 1.0.0
EvenSplit 1.0.0
VendorFinder 1.0.0
editconf 4.5.4
ModelCreator 1.0.0

interfaces for the management of applications and contains functions to be able to manage the whole execution lifecycle. These functions include the following;

- list the users who have exported applications to the local repository
- list the exported applications for a specified user
- import an application to the user space
- manage input and output files for upload and download
- manage remote files
- set and get the command line arguments of a specified job
- submit the application
- get the status of the application
- abort an application
- delete the application

In this way, the original but generic solutions for creating and managing applications are hidden from the users, and they can use well-tested applications via interfaces that were tailored to their needs. To assure that the users can set and execute an exact copy of tested and shared applications, ASM does not provide methods to make structural modifications on the workflows, for instance adding or deleting jobs.

ASM has been developed in Java and supports AJAX actions. The Java methods can be called from portlets, which in turn can use any technology and visualization method fitting to the applications needs, independently from the underlying solution. The security mechanisms rely on the implemented submitters in gUSE. Hence, portlets developed for the MoSGrid science gateway can use the submitter for UNICORE via the configured applications and are unaffected if the implementation of the underlying security infrastructure changes.

3.3 Distributed Data Management

The distributed data management uses XtreemFS as its basis. XtreemFS supports two modes of authentication. The first mode is using X.509 service certificates, which identifies a service instead of a person. Here, the MRC regards XtreemFS clients as a trusted system component, meaning that the MRC will accept any DN sent by the client for authorization. This setup is useful for multiuser environments on trusted machines and may be applied on the D-Grid infrastructure. The second mode is the use of X.509 user (proxy) certificates or SAML assertions. Here the MRC will accept the DN contained in the certificate or SAML assertion for authorization. This setup is useful if users want to mount XtreemFS directly on an otherwise untrusted machine to gain access to their data.

XtreemFS handles POSIX access rights and ACLs based on the DN entries of the X.509 certificates or SAML assertions, which is regarded as a user ID. Files in XtreemFS are indentified by a DN, which in turn belongs to a user. The XtreemFS release 1.3 supports GSI and provides a Java, C++, and FUSE client for MoSGrid. Currently, MoSGrid is operating 16 OSDs with a capacity of 96 TiByte and one MRC service.

XtreemFS had to be integrated in the overall infrastructure at various places, including the Grid middleware UNICORE. XtreemFS also fulfills two key requirements of MoSGrid: The first requirement is the support of security based on X.509 certificate infrastructures using UNICORE XUUDB mapping information and SAML assertions. The second requirement is the seamless integration into existing services by use of a FUSE client and UNICORE TSI on the D-Grid infrastructure and a Java client within WS-PGRADE portlets.

3.3.1 Integration of XtreemFS in the Science Gateway

One way to access XtreemFS volumes from WS-PGRADE is to use a dedicated portlet, which provides simple access to the Grid file system. The user can upload input data and download results using this portlet. When the portlet is initialized, the volume is accessed using the XtreemFS Java API and the user's SAML assertion. Both the Java API and support for SAML assertions had to be implemented. The XtreemFS servers communicate using a protocol based on Google Protocol Buffers [18]. Its use eased the development of the XtreemFS Java API since both provide a similar interface. SAML assertion support was implemented using the OpenSAML libraries [30]. Authentication between the Java API and the XtreemFS servers is done using an X.509 service certificate. When a file is accessed the user's SAML assertion is verified by the Java API. The user's DN is then used to authenticate against the XtreemFS MRC server.

The other way is to use XtreemFS references in WS-PGRADE gUSE workflows, meaning that input files can be downloaded into the job's working directory and output files uploaded back into XtreemFS. Note that this relieves the WS-PGRADE portal from handling the files itself. It now just handles references, which does not burden the portal server with data transfer and avoids unnecessary transfers. To allow references to files of an XtreemFS volume to be included in UNICORE jobs, changes to components in WS-PGRADE were made. First, WS-PGRADE was extended to support the XtreemFS URL schema in UNICORE. The UNICORE submitter was modified to allow the conversion of XtreemFS file definitions to UNICORE staging definitions. Second, an extension for the ASM module in WS-PGRADE was developed to support the use of XtreemFS file definitions from within application portlets. These file definitions can be set for a gUSE workflow using the ASM module and are passed to the UNICORE submitter for inclusion in UNICORE jobs.

3.3.2 Integration of XtreemFS in UNICORE

The integration of XtreemFS into UNICORE means that files in an XtreemFS volume are accessible from within UNICORE. This was realised in the following way.

A URL schema for easily referencing files in an XtreemFS volume from within UNICORE was developed (see Section 3.1.3. for a description). An XtreemFS volume is mounted on a frontend node of a DCI where a UNICORE TSI is running.

Mounting the filesystem is accomplished by the XtreemFS FUSE [14] client and a service certificate. It is accessible by a global mount point (i.e. /mnt/mosgrid/). This mount point is integrated into UNICORE using a UNICORE Storage Management Service (SMS). Together with the XtreemFS URL schema, files in XtreemFS are referenced in the following way; “xtreemfs://path/filename”. When a file staging is defined in a UNICORE job description the file will be tranfered from XtreemFS into the job working directory by the UNICORE TSI component. This is either done by a simple UNIX copy command using the global mount point or a call to a remote UNICORE storage service. This component acts as the local user on the cluster and copies the file from XtreemFS into the job's working directory. When the mount point is accessed, the XtreemFS FUSE client maps the local user to a distinguished name (DN) based on UNICORE User Database (XUUDB) information, which contains the mapping between the user's DN and a login on an HPC facility. This DN is passed to XtreemFS for authorisation.

To summarize, an XtreemFS volume is mounted via an XtreemFS FUSE client, integrated via an UNICORE SMS, and made available easily via the XtreemFS URL schema. This way of integration offers important advantages. First of all, the integration is transparent in regard to XtreemFS, meaning that independent of the available storage resources, XtreemFS provides one global namespace. Furthermore, XtreemFS handles the transfer of data between the science gateway and the HPC facilities.

The UNICORE middleware will take care of transferring the data from the mounted XtreemFS volume on the HPC machine to the USPACE of the simulation job. This way, UNICORE is not responsible for extensive data transfers over long distances since it is less efficient in this regard. The simulation jobs can directly access the MoSGrid volume via UNICORE that provides a technically mature and proven way for this feature.

4 Domain Specific Workflows

In addition to the WS-PGRADE based workflow oriented instruments to use Grid resources, MoSGrid aims to provide intuitive ways for novice users to run chemical simulations. To serve this purpose, the chemical simulation codes, workflows, and IT infrastructures are hidden. The user accesses portlets that directly offer instruments to start and manage simulations for different subjects of structural bioinformatics.

Currently, MoSGrid offers specific portlets for molecular dynamics and quantum chemistry, and protein-ligand docking. The portlets utilize the ASM library together with the gUSE services like the UNICORE submitter. This enables the developers to focus on the domain-related features to further improve the user experience. The design and functionality of the domain specific portlets are described in the following.

4.1 The Molecular Dynamics Portlet

The Molecular Dynamics (MD) portlet enables chemists to easily access molecular simulation codes. The portlet allows chemists to simulate the processes of frequently used standard *recipes*. These recipes are mapped to UNICORE workflows and are available in the portal. This way, the portal should on the one hand ease the work of experienced users and on the other hand lower the hurdle for novice users. The scientists can submit molecular simulations without knowledge of the DCI. The MD portlet is organized in two main sections. The first section covers the recipe configuration and submission and the second section covers the monitoring of running simulations.

Submission The MD submission section is designed to provide a molecular dynamics service on multiple levels. Figure 7 shows the design.

When the user selects the MD portlet, he is welcomed with a short description of the portlet's abilities. In addition, the number of available cluster systems is displayed. To get this information, the portlet has taken the user's credentials to connect to the underlying DCI and detect how many computing facilities can be accessed. If the user cannot connect to any facility, maybe due to missing SAML assertions, he is informed about the problem.

The MD simulation allows the user an easy use of standard chemical recipes. In the current state, the user is enabled to submit a single simulation using a directly uploaded job description. Alternatively, the user can run a complex recipe that includes an energy minimization and a subsequent equilibration. This recipe is an indispensable prerequisite for all kinds of production runs. The portal supports the user with a description of the purpose and requirements of the selected simulation.

For each simulation the user has to upload a file containing either the job description (Gromacs TPR format) or the structural information (PDB format). In the background the portlet automatically checks the correctness of the job description. Unnecessary input information is automatically removed. The user is notified about obvious problems like missing residues in the input structure.

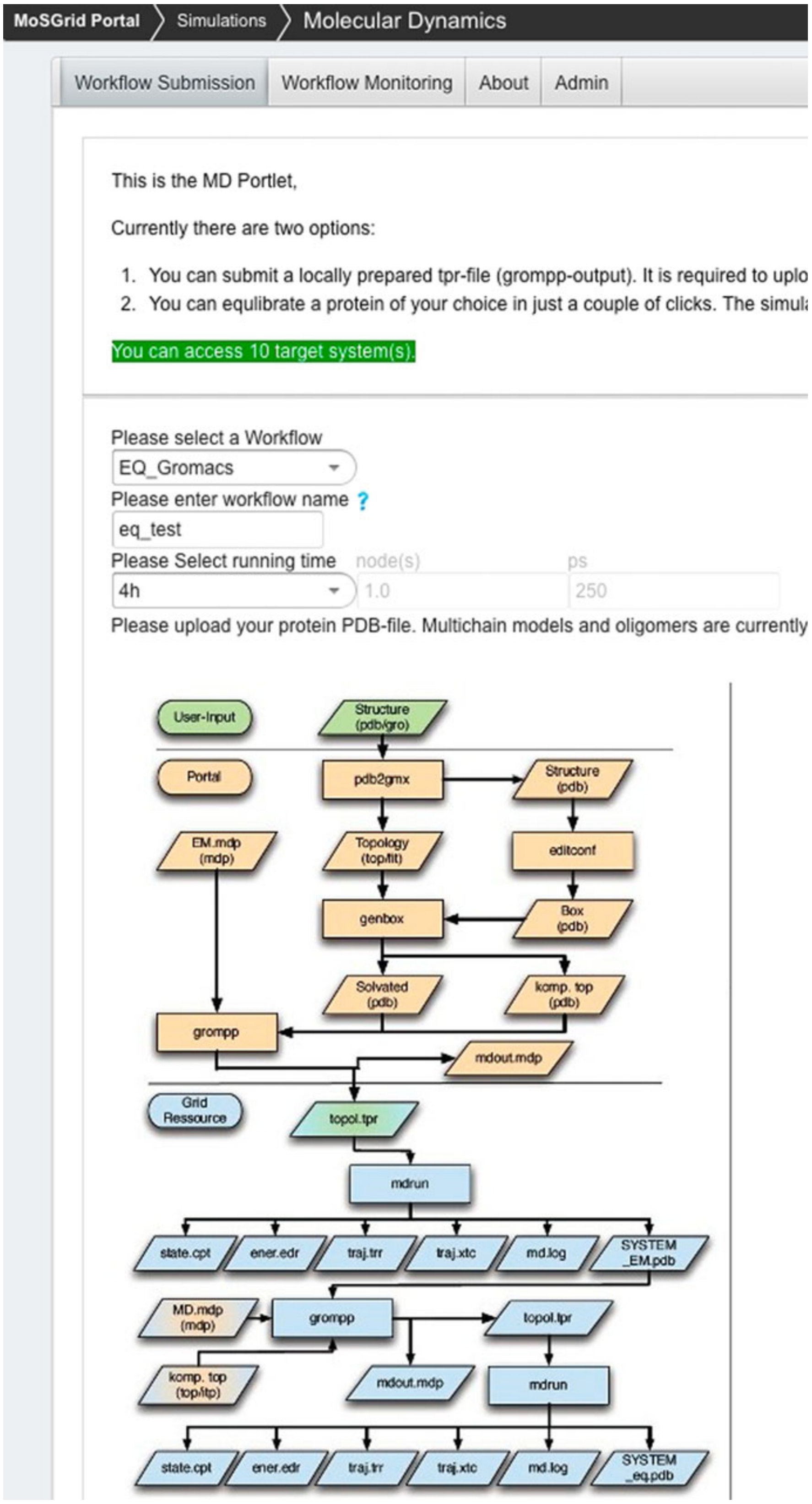
In the next development stage, the MD portlet will detect topological features of the input structure, e.g., if the protein is a monomer or a multimer and adapt the simulation to the different input files.

Even if the portlet minimizes the necessary user input as far as possible, it still requires further information. First of all, it is hard to guess how long a chemical process should be simulated [28]. Therefore, the user has to define the simulation length. Secondly, the user has to define the resources for the simulation. This includes the number of parallel nodes and the maximum duration of the simulation (wall time). When all information is given and checked, the user can submit the job to the MoSGrid infrastructure.

The job submission utilizes the Grid connection through the WS-PGRADE and gUSE infrastructure and its underlying MoSGrid extension, the UNICORE submitter. The MD portlet uses the user's SAML assertion for the authentication and accesses the submitter through the ASM module. Alternatively, the connection can be established to the UNICORE Grid middleware directly.

The UNICORE submitter checks the IDB of the connected Grid clusters. The IDB contains meta information about the simulation codes, available in its cluster system. This allows to select the appropriate cluster for executing the recipe's substeps, even if uncommon code is used.

Monitoring After job submission, the user can monitor the job process in the second main section. The jobs are either named after a user given identifier, or if the user did not specify a job name, the jobs are named after the user login on the



MoSGrid Portal
Simulations
Molecular Dynamics

Workflow Submission | Workflow Monitoring | About | Admin

This is the MD Portlet,

Currently there are two options:

1. You can submit a locally prepared tpr-file (grompp-output). It is required to uplo
2. You can equlibrate a protein of your choice in just a couple of clicks. The simula

You can access 10 target system(s).

Please select a Workflow
EQ_Gromacs
Please enter workflow name ?
eq_test
Please Select running time node(s) ps
4h 1.0 250
Please upload your protein PDB-file. Multichain models and oligomers are currently

Fig. 7 Preparing a recipe with the MD portlet—at the *top* parameters can be adjusted and a recipe chosen and in the *bottom* the whole recipe is shown

portal, combined with submit time, and name of the workflow recipe. A traffic light for each simulation entry shows the status of the simulation.

Yellow	indicates that the job is still executing or queued.
Green	means that the job has been successful.
Red	is used for a failed job.

Further information, e.g., about the underlying HPC facility, which the job utilizes, is hidden.

For each recipe the user can query the output files, even for an ongoing simulation. The files are displayed in a tree like shape, reflecting the substeps of the running workflow. The resulting or intermediate file of the workflow can be downloaded or displayed in the portal. The MD portlet shows either plain text, pictures or figures, and in case a molecule file is selected, a 3D view in Jmol [19] (see Fig. 8).

In the future the MD portlet will be enhanced with more simulation codes, additionally to the currently supported Gromacs [20].

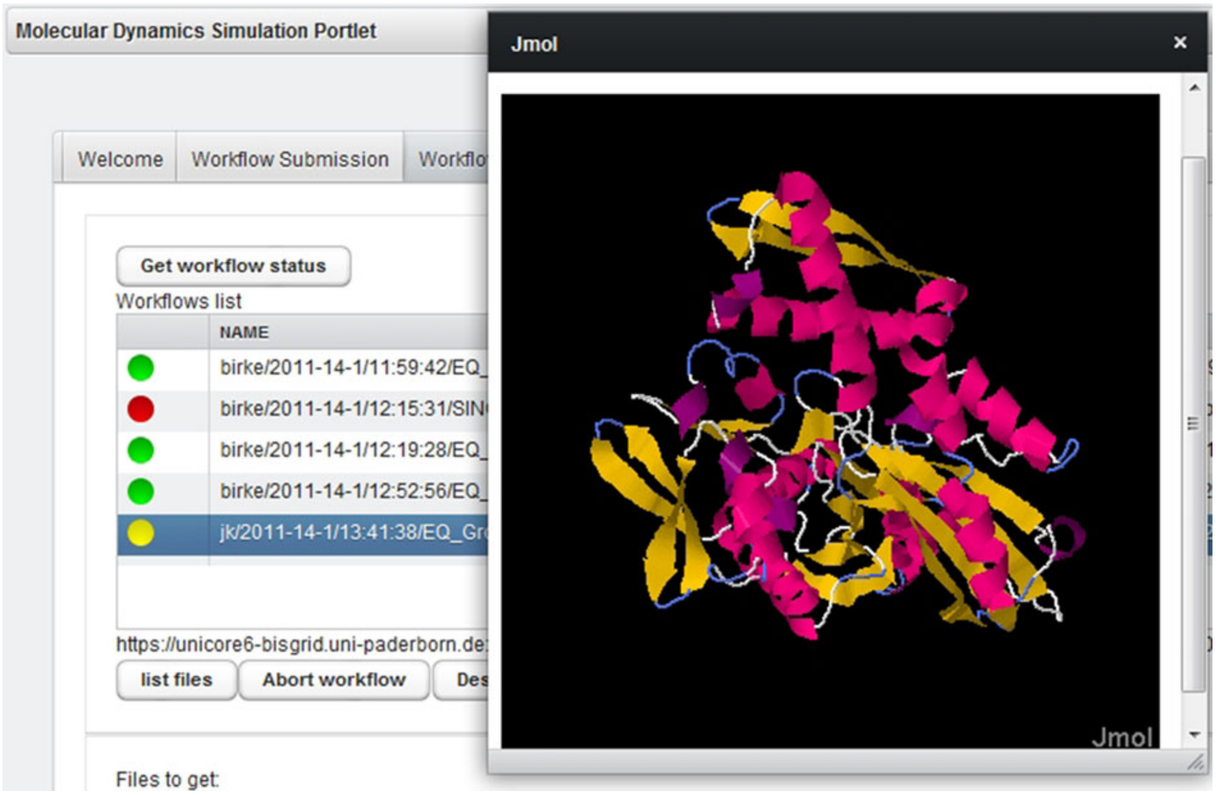
4.2 The Quantum Chemistry Portlet

The Quantum Chemistry (QC) portlet implements a complete quantum chemical workflow [51].

The platform enables researchers to submit their molecular simulations, monitor the progress, and retrieve the results. It is designed for users both experienced and inexperienced with computational chemistry. Therefore, pre- and post-processing routines are available. Amongst others, these can be used to export the output of the simulation tools in a standardized format.

The chosen software design is open for extensions. Instead of directly accessing the UNICORE Commandline client, the portlet is implemented to make use of ASM. With the use of ASM, the authentication via SAML is processed by the UNICORE submitter.

To be able to use the ASM it was required to create WS-PGRADE workflows. First simple workflows include the execution of the specific



	NAME
●	birke/2011-14-1/11:59:42/EQ_
●	birke/2011-14-1/12:15:31/SIN
●	birke/2011-14-1/12:19:28/EQ_
●	birke/2011-14-1/12:52:56/EQ_
●	jk/2011-14-1/13:41:38/EQ_Gr

https://unicore6-bisgrid.uni-paderborn.de:

Fig. 8 MD portlet—monitoring and view of a molecule file in Jmol

tool and parsers for data evaluation. The support for MSML/CML [32–34] is fully functional, but relying on preliminary definitions, which are not yet standardized. So, while being able to generate valid MSML, the input files are still generated and submitted in tool specific formats.

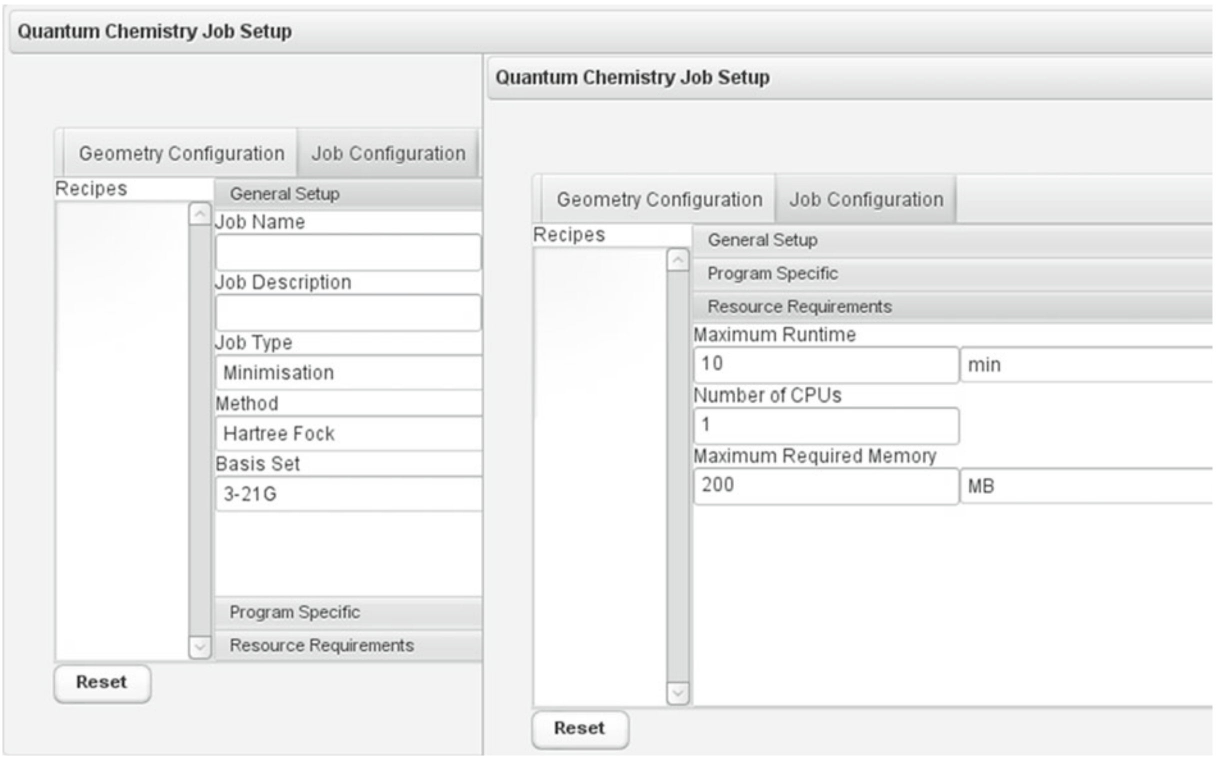
Taking the users' feedback into account, the user interface was split up in two portlets; the first is specialized for creating and submitting the workflows, the second portlet monitors the workflows and visualizes the results.

Creating and submitting a workflow follows the structure centric approach. First, the user selects the geometry/geometries of a molecule. Currently, there is support for uploading files and entering the information directly. OpenBabel [43] is used to enable support for various geometry file formats.

The second step, represented to the user in a second tab, is the configuration of the workflow (see Fig. 9). Here, parameters can be selected in three categories. The first category summarizes the generic parameters of QC calculations. This includes the job type specification (e. g., optimization, energy minimization) and the selection of the simulation method. The second category allows the user to select a specific tool for the simulation and to set up tool specific parameters. The third and last tab offers access to hardware specific settings like runtime, number of processors, and the amount of memory needed.

The monitoring and display of results was transferred to a second, independent portlet. Originally it was designed for the monitoring of all tools from all domains. With the ASM integration the listing of workflows will use a tree-like structure. For the current single-step-workflows there is a list entry for each workflow. The status is represented by the well known items *queued*, *running*, *successful*, or *failed*. On the right, the most important information for each job is summarized. In case of a successful execution, the exit code of the tool is shown as well as the information that result data is available.

This data can be viewed on the second tab of the monitoring portlet, where detailed informa-


Quantum Chemistry Job Setup

Geometry Configuration | Job Configuration

Recipes

General Setup

Job Name

Job Description

Job Type: Minimisation

Method: Hartree Fock

Basis Set: 3-21G

Program Specific

Resource Requirements

Reset

Quantum Chemistry Job Setup

Geometry Configuration | Job Configuration

Recipes

General Setup

Program Specific

Resource Requirements

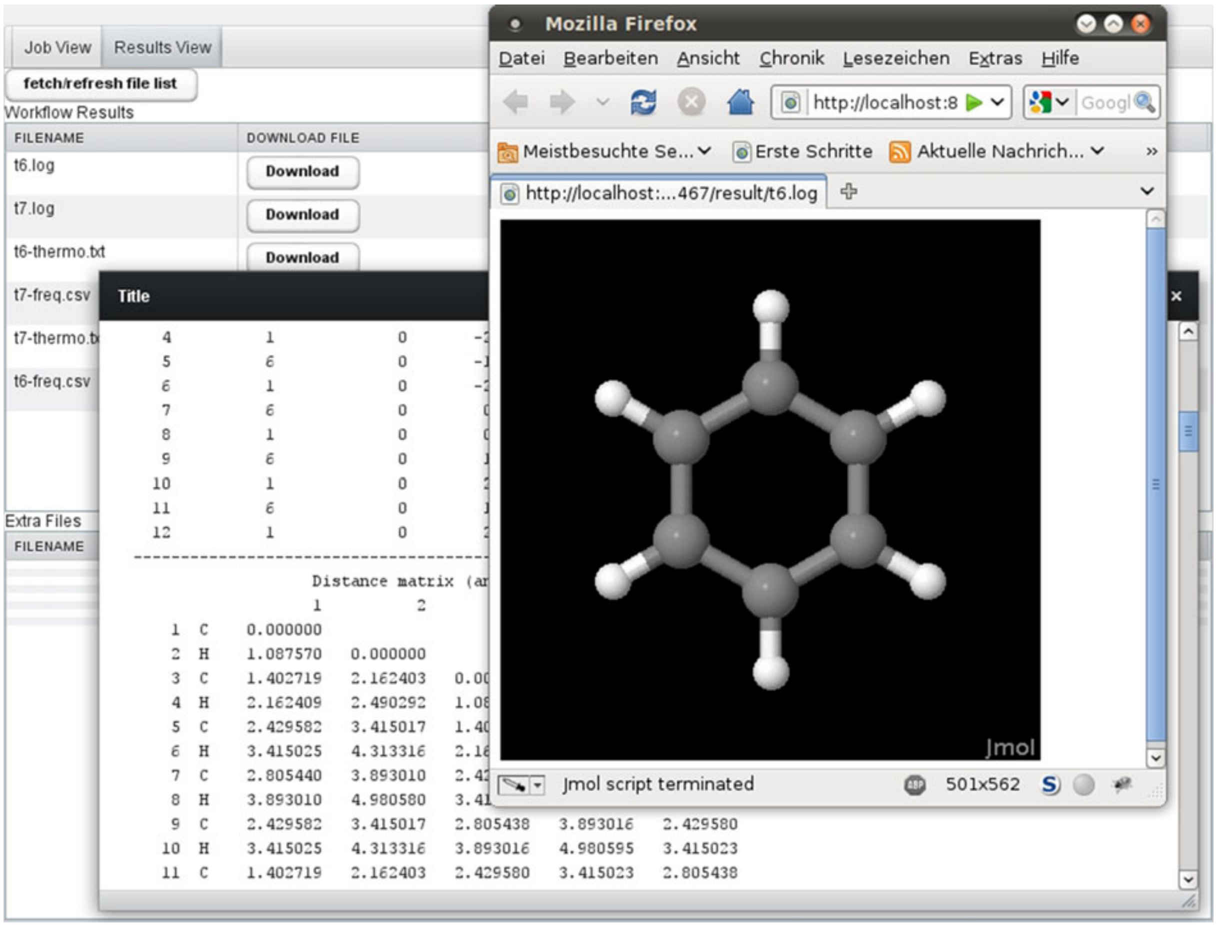
Maximum Runtime: 10 min

Number of CPUs: 1

Maximum Required Memory: 200 MB

Reset

Fig. 9 QC portlet—configuration of a workflow



```
     4          1          0      -
     5          6          0      -
     6          1          0      -
     7          6          0
     8          1          0
     9          6          0
    10          1          0
    11          6          0
    12          1          0
 ---------------------------------
                  Distance matrix (an
                   1          2
   1  C    0.000000
   2  H    1.087570   0.000000
   3  C    1.402719   2.162403   0.00
   4  H    2.162409   2.490292   1.08
   5  C    2.429582   3.415017   1.40
   6  H    3.415025   4.313316   2.16
   7  C    2.805440   3.893010   2.42
   8  H    3.893010   4.980580   3.41
   9  C    2.429582   3.415017   2.805438   3.893016   2.429580
  10  H    3.415025   4.313316   3.893016   4.980595   3.415023
  11  C    1.402719   2.162403   2.429580   3.415023   2.805438
```

Jmol script terminated

Fig. 10 QC portlet—result view with Jmol

tion and visual representation is provided (see Fig. 10).

Depending on the input data, the native output format of the simulation tool, specific values and the trend of these values are plotted to files, which can be viewed, downloaded, and processed in common spreadsheet applications. Visualization of the data is provided by utilizing Jmol.

5 Summary and Outlook

We presented the security infrastructure of the MoSGrid science gateway offering single sign-on to HPC facilities via SAML assertions. Users are enabled to intuitively create SAML assertions and are provided with domain specific workflows and portlets. Furthermore, WS-PGRADE offers the ASM API, which allows developers to focus on domain specific workflows and portlets without the need to become acquainted to the security infrastructure in detail. On the high-level middleware service layer, gUSE and the cloud file system XtreemFS were extended for the use of SAML assertions.

Our next steps regarding the security infrastructure will enhance the usability of the authentication mechanism. Therefore, we will utilize the user certificate embedded in the browser. This embedded user certificate will be employed for two purposes: first, for an automatic login of the user based on membership in the MoSGrid virtual organization, second, for an automatic creation of SAML assertions. The latter step will eliminate the user interaction for creating a SAML assertion and for choosing a certificate from the local hard drive. Since personal certificates expire annually, information about how to renew the certificate will be presented when this case occurs. Together with the previous mentioned measures, this will further aid the user in smoothly using the MoS-Grid science gateway.

Acknowledgement We would like to thank Valentina Huber for the basic version of the applet for generating SAML assertions.

References

1. Abdelnur, A., Hepper, S.: JSR 168: Portlet Specification. http://www.jcp.org/en/jsr/detail?id=168 (2003)
2. Anjomshoaa, A., Drescher, M., Fellows, D., Ly, A., McGough, S., Pulsipher, D., Savva, A.: Job Submission Description Language (JSDL) specification, version 1.0. http://www.gridforum.org/documents/GFD.56.pdf (2005)
3. The Apache Software Foundation: Apache Tomcat. http://tomcat.apache.org/tomcat-6.0-doc/ (2012)
4. Barbera, R., Andronico, G., Donvito, G., Falzone, A., Keijser, J., Rocca, G.L., Milanesi, L., Maggi, G.P., Vicario, S.: A Grid portal with robot certificates for bioinformatics phylogenetic analyses. Concurrency Computat.: Pract. Exper. **23**(3), 246–255 (2011)
5. Basney, J., Martin, S., Navarro, J., Pierce, M., Scavo, T., Strand, L., Uram, T., Wilkins-Diehr, N., Wu, W., Youn, C.: The problem solving environments of ter-aGrid, science gateways, and the intersection of the two. In: IEEE International Conference on eScience, pp. 725–734 (2008)
6. Benedyczak, K., Bała, P., van den Berghe, S., Menday, R., Schuller, B.: Key aspects of the UNICORE 6 security model. Future Gener. Comput. Syst. **27**(2), 195–201 (2011)
7. Brewer, S., Sipos, G.: Benefits and vision for the VRC community model. In: EGI User Forum 2011, Book of Abstracts (2011)
8. Chandra, N., Anand, P., Yeturu, K.: Structural bioinformatics: deriving biological insights from protein structures. Interdisciplinary Sciences: Computational Life Sciences **2**(4), 347–366 (2010). doi:10.1007/s12539-010-0045-6
9. Farkas, Z., Kacsuk, P.: P-GRADE portal: a generic workflow system to support user communities. Future Gener. Comput. Syst. **27**(5), 454–465 (2011)
10. Fauman, E.B., Hopkins, A.L., Groom, C.R.: Structural Bioinformatics in Drug Discovery, chapter 23, pp. 477–497. Wiley-Liss Inc., Hoboken, New Jersey (2003)
11. Foster, I.: Globus toolkit, version 4: software for service-oriented systems. In: IFIP International Conference on Network and Parallel Computing, (LNCS 3779), pp. 2–13. Springer-Verlag (2006)
12. Foster, I., Kesselman, C., Tsudik, G., Tuecke, S.: A security Infrastructure for computational Grids. In: CCS '98 Proceedings of the 5th ACM Conference on Computer and Communications Security (1998)
13. Frisch, M.J., Trucks, G.W., Schlegel, H.B., Scuseria, G.E., Robb, M.A., Cheeseman, J.R., Montgomery, J.J.A., Vreven, T., Kudin, K.N., Burant, J.C., Millam, J.M., Iyengar, S.S., Tomasi, J., Barone, V., Mennucci, B., Cossi, M., Scalmani, G., Rega, N., Petersson, G.A., Nakatsuji, H., Hada, M., Ehara, M., Toyota, K., Fukuda, R., Hasegawa, J., Ishida, M., Nakajima, T., Honda, Y., Kitao, O., Nakai, H., Klene, M., Li, X., Knox, J.E., Hratchian, H.P., Cross, J.B., Bakken, V., Adamo, C., Jaramillo, J., Gomperts, R., Stratmann, R.E., Yazyev, O., Austin, A.J., Cammi, R., Pomelli, C., Ochterski, J.W., Ayala, P.Y., Morokuma, K., Voth, G.A., Salvador, P., Dannenberg, J.J., Zakrzewski, V.G., Dapprich, S., Daniels, A.D., Strain, M.C., Farkas, O., Malick, D.K., Rabuck, A.D., Raghavachari, K., Foresman, J.B., Ortiz, J.V., Cui, Q., Baboul, A.G., Clifford, S., Cioslowski, J., Stefanov, B.B., Liu, G., Liashenko, A., Piskorz, P., Komaromi, I., Martin, R.L., Fox, D.J., Keith, T., Al-Laham, M.A., Peng, C.Y., Nanayakkara, A., Challacombe, M., Gill, P.M.W., Johnson, B., Chen, W., Wong, M.W., Gonzalez, C., Pople, J.A.: Gaussian 03, revision C.02. Gaussian, Inc., Wallingford CT (2004)
14. FUSE: http://fuse.sourceforge.net (2012)
15. GAP-SLC: http://gap-slc.awi.de/ (2009)
16. Gesing, S., Kacsuk, P., Kozlovszky, M., Birkenheuer, G., Blunk, D., Breuers, S., Brinkmann, A., Fels, G., Grunzke, R., Herres-Pawlis, S., Krger, J., Packschies, L., Mller-Pfefferkorn, R., Schfer, P., Steinke, T., Fabri, A.S., Warzecha, K., Wewior, M., Kohlbacher, O.: A science gateway for molecular simulations. In: EGI User Forum 2011, Book of Abstracts, pp. 94–95 (2011)
17. Gesing, S., Marton, I., Birkenheuer, G., Schuller, B., Grunzke, R., Krüger, J., Breuers, S., Blunk, D., Fels, G., Packschies, L., Brinkmann, A., Kohlbacher, O., Kozlovszky, M.: Workflow interoperability in a Grid portal for molecular simulations. In: Barbera, R., Andronico, G., La Rocca, G. (eds.) Proceedings of the International Workshop on Science Gateways (IWSG10), pp. 44–48. Consorzio COMETA (2010). http://documents.ct.infn.it/record/474/files/iwsg10-proceedings.pdf
18. Google: Protocol Buffers. https://code.google.com/p/protobuf/ (2011)
19. Herraez, A.: How to Use Jmol to Study and Present Molecular Structures. Lulu Enterprises, Morrisville, NC, USA (2007)
20. Hess, B., Kutzner, C., van der Spoel, D., Lindahl, E.: GROMACS: algorithms for highly efficient, load-balanced, and scalable molecular simulation. Journal of Chemical Theory and Computation **4**(3), 435–447 (2008)
21. Hupfeld, F., Cortes, T., Kolbeck, B., Stender, J., Focht, E., Hess, M., Malo, J., Marti, J., Cesario, E.: The XtreemFS architecture—a case for object-based file systems in Grids. Concurrency Computat.: Pract. Exper. **20**(17), 2049–2060 (2008). doi:10.1002/cpe.1304

22. I2MI: Internet2 Middleware Initiative. http://www.internet2.edu/middleware/ (2011)
23. Inc. Liferay: Liferay. http://www.liferay.com (2012)
24. Java Community Process: Java Servlet 2.5 Specification. http://jcp.org/aboutJava/communityprocess/mrel/jsr154/index.html (2003)
25. Java Community Process: Java Server Pages 2.1. http://jcp.org/aboutJava/communityprocess/final/jsr245/index.html (2006)
26. Jonikas, M.A., Laederach, A., Altman, R.B.: RNA Structural Bioinformatics. Wiley-Liss Inc. (2003)
27. Kacsuk, P.: P-GRADE portal family for Grid infrastructures. Concurrency Computat.: Pract. Exper. **23**(3), 235–245 (2011)
28. Krüger, J., Fels, G.: Ion permeation simulations by Gromacs – an example of high performance molecular dynamics. Concurrency Computat.: Pract. Exper. **23**(3), 279–291 (2011)
29. Laure, E., Gr, C., Fisher, S., Frohner, A., Kunszt, P., Krenek, A., Mulmo, O., Pacini, F., Prelz, F., White, J., Barroso, M., Buncic, P., Byrom, R., Cornwall, L., Craig, M., Meglio, A.D., Djaoui, A., Giacomini, F., Hahkala, J., Hemmer, F., Hicks, S., Edlund, A., Maraschini, A., Middleton, R., Sgaravatto, M., Steenbakkers, M., Walk, J., Wilson, A.: Programming the Grid with gLite. In: Computational Methods in Science and Technology, vol. 12, pp. 33–45 (2006)
30. Lewis, K.D., Lewis, J.E.: Web single sign-on authentication using SAML. IAENG International Journal of Computer Science **2**, 41–48 (2009)
31. Morgan, R.L., Cantor, S., Carmody, S., Hoehn, W., Klingenstein, K.: Federated security: the shibboleth approach. EDUCAUSE Quarterly **27**(4), 12–17 (2004)
32. Murray-Rust, P., Rzepa, H.S.: Chemical markup, XML, and the world wide web 1. Basic principles. J. Chem. Inf. Comput. Sci. **39**(6), 928–942 (1999). doi:10.1021/ci990052b
33. Murray-Rust, P., Rzepa, H.S.: Chemical markup, XML and the world wide web 2. Information objects and the CMLDOM. J. Chem. Inf. Comput. Sci. **41**(5), 1113–1123 (2001). doi:10.1021/ci000404a
34. Murray-Rust, P., Rzepa, H.S.: Chemical markup, XML, and the world wide web 4. CML schema. J. Chem. Inf. Comput. Sci. **43**(3), 757–772 (2003). doi:10.1021/ci0256541
35. Murri, R., Kunszt, P., Maffioletti, S., Tschopp, V.: GridCertLib: a single sign-on solution for Grid web applications and portals. Journal of Grid Computing **9**(4), 441–453 (2011)
36. Neese, F.: The ORCA program system. WIREs Comput. Mol. Sci. **2**(1), 73–78 (2012)
37. Nicklous, M., Hepper, S.: JSR 286: Portlet Specification 2.0. http://www.jcp.org/en/jsr/detail?id=286 (2008)
38. Niehörster, O., Birkenheuer, G., Brinkmann, A., Elsässer, B., Blunk, D., Herres-Pawlis, S., Krüger, J., Niehörster, J., Packschies, L., Fels, G.: Providing scientific Software as a service in consideration of service level agreements. In: Proceedings of the Cracow Grid Workshop (CGW) (2009)
39. Niehörster, O., Brinkmann, A., Fels, G., Krüger, J., Simon, J.: Enforcing SLAs in scientific clouds. In: IEEE International Conference on Cluster Computing 2010 (Cluster) (2010)
40. OASIS: eXtensible Access Control Markup Language (XACML) Version 2.0. http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf (2005)
41. OASIS: Web Services Resource Framework (WSRF)—Primer v1.2. http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf (2006)
42. OASIS: Organization for the Advancement of Structured Information Standards. http://www.oasis-open.org (2011)
43. O'Boyle, N., Banck, M., James, C.A., Morley, C., Vandermeersch, T., Hutchison, G.R.: Open babel: an open chemical toolbox. J. Cheminf. **3**, 33 (2011)
44. Riordan, R.M.: Head 1st Ajax. O'Reilly (2008)
45. Security Assertion Markup Language (SAML) V2.0: http://docs.oasis-open.org/security/saml/v2.0/saml-2.0-os.zip (2002)
46. Snelling, D., van den Berghe, S., Li, V.: Explicit trust delegation: security for dynamic Grids. Fujitsu Sci. Tech. J. **40**(2), 282–294 (2004)
47. Streit, A., Bala, P., Beck-Ratzka, A., Benedyczak, K., Bergmann, S., Breu, R., Daivandy, J.M., Demuth, B., Eifer, A., Giesler, A., Hagemeier, B., Holl, S., Huber, V., Lamla, N., Mallmann, D., Memon, A.S., Memon, M.S., Rambadt, M., Riedel, M., Romberg, M., Schuller, B., Schlauch, T., Schreiber, A., Soddemann, T., Ziegler, W.: Unicore 6—Recent and Future Advancements. JUEL-4319 (2010). http://hdl.handle.net/2128/3695
48. Tuecke, S., Welch, V., Novotny, J.: An online credential repository for the Grid: MyProxy. In: Proceedings of the 10th International Symposium on High Performance Distributed Computing (HPDC-10), IEEE press, pp. 104–111 (2001)
49. Turbomole v6.2 2010: A Development of University of Karlsruhe and Forschungszentrum Karlsruhe Gmbh, 1989-2007, Turbomole Gmbh (2007). http://www.turbomole.com
50. Wang, X.D., Jones, M., Jensen, J., Richards, A., Wallom, D., Ma, T., Frank, R., Spence, D., Young, S., Devereux, C., Geddes, N.: Shibboleth Access for Resources on the National Grid Service (SARoNGS). In: 5th International Conference on Information Assurance and Security, vol. 2, pp. 338–341 (2009)
51. Wewior, M., Packschies, L., Blunk, D., Wickeroth, D., Warzecha, K.D., Herres-Pawlis, S., Gesing, S., Breuers, S., Krüger, J., Birkenheuer, G., Lang, U.: The MoSGrid Gaussian portlet—technologies for the implementation of portlets for molecular simulations. In: Barbera, R., Andronico, G., La Rocca, G. (eds.) Proceedings of the International Workshop on Science Gateways (IWSG10), pp. 39–43. Consorzio COMETA (2010). http://documents.ct.infn.it/record/474/files/iwsg10-proceedings.pdf