Maintaining a Science Gateway – Lessons Learned from MoSGrid

Lukas Zimmermann
Applied Bioinformatics,
University of Tübingen,
Tübingen, Germany
lukas.zimmermann@
student.uni-tuebingen.de

Richard Grunzke
Center for Information Services
and High Performance
Computing, Technische
Universität Dresden, Dresden,
Germany
richard.grunzke@tu-dresden.de

Jens Krüger*
Applied Bioinformatics,
University of Tübingen,
Tübingen, Germany
krueger@
informatik.uni-tuebingen.de

Abstract

We here present the experiences collected maintaining and updating the MoSGrid science gateway over the past years. Insights are provided on a technical and organizational level useful for the design and operation of science gateways in general. The specific challenges faced and solved are considered to be valuable for other communities.

1. Introduction

The Molecular Simulation Grid (MoSGrid) is being operated for over six years now [1]. The scientific gateway provides workflows, access to compute infrastructure and data management capabilities for computational chemists and scientists from related fields. Molecular simulations are resource-demanding by their very nature. Ab initio quantum chemistry calculations are decelerated by solving the Schrödinger equation, molecular dynamics require adequate representation of potential large systems and emerging thermodynamic properties, docking approaches rely on large underlying screening libraries. These molecular simulations need to be run on high performance computing (HPC) resources to enable scientific projects to perform cutting edge research.

Regardless of their scientific background, all users wishing to conduct studies of that kind should have the opportunity to do so. Having steadily recurring requirements and necessities already at their fingertips, these users should be relieved from the need to deploy these themselves, for example grid engines, storage systems, and involved software.

Those specifications mentioned here clearly point towards the direction of science gateways. MoSGrid, whose maintenance over the past six years shall be discussed here, bridges the gap between complex software[1] in molecular simulations and user convenience. It is discussed which challenges need to be met for a science gateway to be maintained and scientific protocols to be curated.

MoSGrid consists of a sophisticated stack of technologies providing access to German compute infrastructures. Maintaining these technologies and providing a high-level service to the community is a challenging task. The experiences made are shared to provide helpful insight for other communities. Figure 1 shows the landing page of the MoSGrid science gateway, representing the entry point to the services discussed in the following sections. We will provide insights going beyond the update process of individual components of the gateway framework, providing a broader picture of the whole venture.

2. Related Work

Extensive literature search yielded no results on related work focusing on the specific topic of updating and maintaining a complete science gateway. Thus, to the best of our knowledge, the challenges of systematic and reliable procedures for updating science gateways seem to be an unaddressed issue.

For some specific frameworks used in science gateways such as gUSE/WS-PGRADE [2,3] procedures exists to extend underlying databases, functionalities and portlets consequently upgrading the gUSE/WS-PGRADE version. Science gateways like the AMC Neuroscience Gateway [4], the VisIVO Gateway [5] and the Statistical Seismology Gateway [6], but also others, undergo this process regularly.

A similar approach is available for updating Galaxy [7] instances keeping features, configuration files, databases, tools, and other major functionalities up to date.

* corresponding author

URI: http://hdl.handle.net/10125/41918
ISBN: 978-0-9981331-0-2

For the Catania Science Gateway [8] no particular upgrade documentation could be obtained. But as this framework was designed to be very modular, updating individual components appears feasible. The Catania Science Gateway Marketplace offers the possibility to virtual research communities (VRCs) to request new applications and functionalities.

HUBzero [9] which is the underlying content management system for e.g. nanohub.org [10] offers ready-to-use packages available for Debian 6/7 and RHEL 6.

Apache Taverna is a popular software suite for designing workflows. Webservices can be attached to workflows as WSDL descriptors. The project is now merged into the Apache Incubator stage [11,12], so efforts for maintenance of both suites can be unified.

Apache Airavata is a software framework enabling the composition, management, execution and monitoring of applications and workflows on distributed computing resources [13]. Apache Airavata is a fairly new technology and can be considered to be still under development.

Vine Toolkit [14] offers an API for developers which is particularly focused on plugins, each adapted to solving a well-defined problem. Thus, Vine Toolkit is modular from its very design. It is also a reasonable candidate when it comes to the integration with Liferay and supports many operating systems due to the employment of the Apache Flex Framework [15].

KNIME, the Konstanz information miner [16], is an open source tool suite for the assembly of data analysis pipelines. It offers its deployment as server application with the TomEE Application server. KNIME's capabilities in terms of High-Performance Computing (HPC) are represented by the cluster execution plugin, which enables individual nodes of the workflow being submitted to an Oracle grid engine, and a recent development of integrating KNIME with HPC using UNICORE. KNIME heavily relies on community-curated nodes in order to delegate support to the user base.

MyExperiment [17] is a repository with which scientists can share their experiments with their community. These workflows can be of several different supported formats to solve diverse challenges from different scientific areas. Also, workflows can be combined in so-called packs, which gives their distribution more structural integrity.

There are also commercial solutions available for data mining. One instance is RapidMiner [18], which was also selected to be leader in the 2016 Gartner Magic Quadrant for Advanced Analytics Platform [19]. Here, support is provided directly by the developers and included in the license fee.

All the individual approaches for the different frameworks are viable and can be reasonable answers to infrastructure-related questions. But science gateways usually consist of considerably more, for example customized user interfaces, connections to compute and data resources, authentication and security concepts. These challenges also need to be handled on the long run.

3. Operational Strategy

Before describing the infrastructural and technological details of the MoSGrid science gateway we focus on the operational concept behind MoSGrid. When MoSGrid was conceived and funded by the German education ministry, and subsequently by the EU in the SCI-BUS and ER-flow projects, it was anticipated that the operation and maintenance of portal, storage, and compute infrastructure is being shared among a few core project members. Furthermore, it is notable that MoSGrid is one of the last major D-Grid projects being maintained in Germany. These two main factors had some considerable impact on design decisions which had direct influence on maintenance and operational considerations. The security concept based on personal user certificates is one major example, the federated storage concept based on XtreemFS is another one. Although such decisions were reasonable when they were made about six years ago, some exterior developments turned the operation of MoSGrid into a challenging enterprise. In 2012 D-Grid was decommissioned, which had many implications for MoSGrid. From an operational point of view the most relevant one was that compute centers only reluctantly support a certificate-based authentication mechanism for reasons that have been hard to conceive. It also turned out to be quite challenging as the Zuse Institute Berlin, which is an important MoSGrid partner, had to limit its resources dedicated to XtreemFS development and operation of the MoSGrid storage instance. Fortunately, the remaining partners were able to handle these and similar challenges. The main strategy can be condensed in a single sentence: How can we keep the portal and associated services running with personnel and financial restraints? In essence, with limited resources at hand, security and operation related issues have the highest priority, which is accompanied by some shortcomings. Software updates, which merely introduce new features are considered optional and are at most applied with an unavoidable delay. Extensions and further developments are limited to essential improvements.

4. Infrastructure

The following chapters encompass topics more related to 'low-level' infrastructural matters. The discrimination from portal related topics is by no means strict due to the tightly integrated multilayer structure of the gateway (see Figure 2). The individual components have been described in detail earlier (please refer to [1]).

4.1. Storage

XtreemFS is a distributed data management system [20]. It is integrated with MoSGrid in a frictionless way as previously described [1]. In order to ease the maintenance of XtreemFS, the installations at different locations (Berlin, Dresden, Tübingen, and Cologne) were consolidated to Dresden. Although the resilience is theoretically lowered, this step safes considerable maintenance effort. It turned out that the coordination of slight setup changes or updates among four compute centers required more effort than the updates themselves. Now the data management system is exposed via the UNICORE system at Dresden to be utilized by the other UNICORE installations as the central entry point, instead of the previously local XtreemFS installations. The MyData Portlet still utilizes the XtreemFS API in the same way, but now directly connects to the installation in Dresden.

Figure 1 Landing Page of the MoSGrid science gateway serving as entry point and providing basic information.

Another option to decrease the maintenance effort was evaluated as well, which involved to phase out XtreemFS completely. This would have significantly decreased the complexity further as the effort of updating and maintaining XtreemFS would have been avoided. This would be feasible since XtreemFS is just running locally in Dresden and UNICORE is utilized to expose it for access. XtreemFS is then not strictly needed anymore. This was not done as XtreemFS is also accessed via its API independently of UNICORE from within the application domain portlets and the MyData portlet on the science gateway. Hence, it would have involved too much of an effort to partly re-develop the portlets in order to access the storage via other means. Instead of a direct XtreemFS connection, UNICORE could serve as an uniform entry point. Anyhow, re-deploying XtreemFS in Dresden offered the most sustainable effect for the lowest required effort for the time being.

4.2. UNICORE

UNICORE [21] is a middleware to abstract from complex computing resources and expose them in various ways such as a multitude of clients and APIs with various high-level services being available. The integration of UNICORE with the MoSGrid science gateway was previously described [1].

A part of the continuous maintenance involves considerable efforts in updating UNICORE. When configuration files need to be adapted, which seldom is the case, the effort is even more elevated. Updates are a necessity to counter security issues in either the underlying libraries that UNICORE employs or the UNICORE libraries themselves.

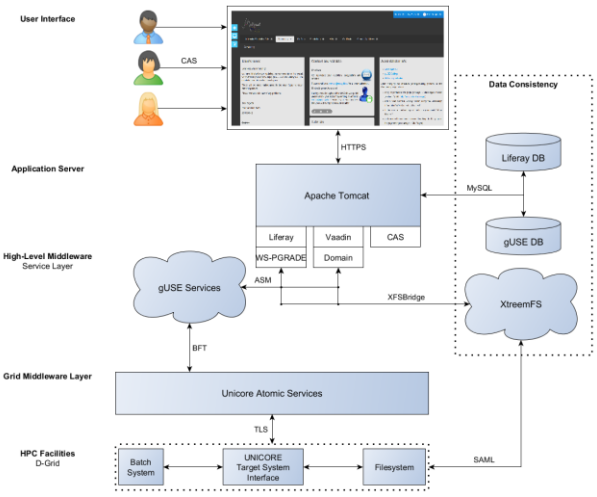

Figure 2 Multilayered architecture of MoSGrid including the underlying infrastructure. In particular, the individual services and corresponding technologies are highlighted.

Since the beginning of MoSGrid, the Atlas cluster at Dresden was a major part of the computing infrastructure underpinning MoSGrid. Atlas was decommissioned at the end of 2015. This had no further consequences on the MoSGrid capabilities but the computing resource capacity was reduced. Atlas' replacement is the Taurus cluster. Since expertise with

the UNICORE middleware has been gathered since the first days of MoSGrid, it was installed also for Taurus and is at the time given fully operational. To integrate Taurus with MoSGrid, two options are taken into consideration.

First, the new UNICORE registry in Dresden could be employed for all workflows in MoSGrid; however, this would require all workflows to be adapted accordingly. To complicate issues even further, each individual node of each workflow must be reconfigured. In addition, UNICORE version compatibility problems arise in such cases. The UNICORE submitter for gUSE was developed in the context of the MoSGrid project in 2010. The current UNICORE libraries at that time were versioned to 6.4 and deployed consequently. This installation had functioned without problems since then. Recently, with UNICORE 7, problems appeared, affecting the data staging between nodes of a workflow.

Second, the current MoSGrid registry could be used to also incorporate the Taurus cluster. This is currently not possible, as the registry is of an older version and cannot be updated, since the newest UNICORE version requires Java version 8. This in turn cannot be installed on the respective server as the Linux operating system concerned is outdated. We consider this issue to be completely solvable given enough time.

For all efforts described in this section a simple four (or more) eyes strategy is followed. The two or more people involved in each aspect discuss and coordinate the effort. For reoccurring tasks, like certificate renewals, a simple note about the accomplishment is sufficient, for more complex challenges like registry migration, extended discussions are required.

Currently, we are evaluating the following course of action. First, efforts would be undertaken to update the gUSE submitter to the newest UNICORE version to avoid library compatibility problems. Second, a migration to the new UNICORE registry should prevent incompatibilities. Third, an alternative DNS entry for the new UNICORE Registry could be created to avoid the need to adjust every workflow manually.

4.3. Authentication and Security

The whole security concept of MoSGrid relies on grid certificates issued by trusted certification authorities (CA). All interacting bodies need to have a valid certificate including a valid chain of trust down to the root of the CA [22]. In practice this means that all users need to obtain a personal certificate as well as all portal, storage, and compute resources need to have a server certificate. These certificates usually need to be renewed once a year which can be considered as slight inconvenience for the users that are often IT-novices. On the portal administrative side the reoccurring renewal of approximately a dozen server certificates e.g., for the portal itself, each UNICORE instance, XtreemFS storage, etc. becomes an organizational challenge. Another item on the maintenance checklist is the renewal of the public keys of the root certificates. Each public key from an integrated certificate authority needs to be up-to-date. In the end the communication between all entities of the MoSGrid science gateway is then handled by SAML trust delegation assertions relying on mutual trust.

Currently, the problem arose that modern browser started to drop their support for Java applets. This is a problem for MoSGrid as the secure and user-friendly client-side creation of the SAML trust delegation assertions for users depends on such a Java plugin. We evaluate the intermediate solution of creating a documentation for users to utilize the UNICORE Commandline Client (UCC) to create assertions and then upload them through the portal. The command looks like the following:

```
"ucc issue-delegation
-S "CN=mosgrid.informatik.uni-
tuebingen.de, OU=Universitaet
Tuebingen, O=GridGermany, C=DE"
-s "CN=othello.zih.tu-dresden.de,
OU=Technische Universitaet Dresden,
O=GridGermany, C=DE"
-t assertion -V 60"
```

The SAML trust delegation assertion is issued to a subject (-S) to access computing a site (-s) in the name of the issuer (configured via the UCC preferences file). The name of the delegation is set with the option "-t" and the validity in days with "-V". We are currently working on a new security portlet to enable this procedure for the users in a convenient way. From a technological perspective, this portlet will rely on JavaServer Pages, which seems to be best supported by the gUSE software stack and thus a sustainable solution.

The mid-term plan is to incorporate an identity federation such as eduGAIN [23]. This way, users do not need to apply for and regularly renew certificates and they do not have to configure them anymore. They could use their usual home institutional login. Technically, the Unity service [21] shall act as the identity proxy as it integrates well with UNICORE as well as Shibboleth [24] which is the underlying technology of eduGAIN. Then, UNICORE will be configured to trust the Unity instance and accept SAML assertions that are issued by this Unity instance. This is necessary as users do not have a certificate anymore and can, thus, not issue assertions anymore by themselves.

VAVID [25] is a science gateway that is based on MoSGrid and tailored to use cases related to wind energy power stations and car crash simulations. In VAVID an integration with Unity is currently being developed to be used with a LDAP authentication source. It is anticipated to adapt this for MoSGrid with eduGAIN. As this approach still requires considerable effort, it is not possible to implement this for MoSGrid at this point in time.

Another security related aspect regarding safe communication also had to be addressed when the science gateway instances were migrated. For the production instance of MoSGrid SSL certificates need to be transferred from the keystore of the previously employed JVM, where one must ensure not to break the certificate chain between Root CA and the certificate issued by the University of Cologne for encrypted communication. Consequently, the HTTPS protocol could be enabled again, which is not used for the development instance.

4.4. Portal Instance

The gUSE Liferay bundle is currently deployed on a Virtual Machine (VM) at the WSI of the University of Tübingen. The choice of the OS respects the recommendations of the developers of gUSE and currently is Scientific Linux 6.7; the underpinning Java Virtual Machine is 1.7. Software packages are maintained in a conservative manner: new packages are exclusively installed via the package manager of SE6, yum. MySQL in server version 5.1.73 on the same machine is used to store the Liferay and gUSE databases. No foreign software repositories are integrated to avoid incompatibilities between OS user space programs and deployed gUSE infrastructure.

For the most recent updates – instead of iteratively patching from gUSE 3.6.3 to the most recent version 3.7.4 – the gUSE install wizard, which internally downloads Liferay 6.2 GA2 and Apache Tomcat 7.0.55, was executed to install the gUSE instance from scratch. For production purposes, the Apache Tomcat Native Library support was supplemented via yum. The advantage is the access to the APR connector including the support of non-blocking I/O and OpenSSL. In addition, the Tomcat server classpath was supplemented with more recent version of already present libraries, such as the cryptography library Bouncycastle and javax.mail. Additionally, the JavaEE API was updated from version 5 to 7 for an enhanced compatibility with Servlet 3.

Detailed description of the Portal setup will be given in the following section.

5. Portal

The MoSGrid science gateway spans multiple layers of tightly integrated technologies (see Figure 2). The following section will cope with topics related to the portal itself.

5.1. Liferay

The Liferay portal framework developed by Liferay Inc. is probably the most sophisticated portal engine available, even regarded as the leading product among software covering the same market [19]. The spectrum of out-of-the-box features ranges from LDAP support to message boards and with Liferay 6.1 encompasses about 180 portlets. All major Java application servers and databases are supported [26].

The employment of Liferay in MoSGrid offers a consistent framework for both administrative and scientific components and, in conjunction with Vaadin [27], user-friendly GUI elements, in particular Google Web Toolkit (GWT) widgets. Vaadin is heavily utilized in MoSGrid's simulation portlets to keep the layout consistent and to harmonize the usage of different simulation portlets. For example, the selection of a workflow, assigning a descriptive name to the new workflow instance, and specifying parameters in a uniform input mask, is all presented with the same look-and-feel.

MoSGrid modifies the user view of the portlet with a custom theme. For the new portal installation, a new theme was written which (a) remains close to the color scheme of the previous MoSGrid instance and (b) incorporates many elements of Liferays 'classic' theme to make the navigation elements look slim and modern. Previous bulky MoSGrid portlet styles were essentially discarded. Themes are one possibility of creating Liferay Plugin Projects, which is best accomplished with support of the Liferay Plugins SDK and the Liferay IDE offered for Eclipse IDE.

Liferay 6.2 brings new capabilities of portal administration, for instance a revised control panel for application management deployment, a marketplace for the acquisition of new apps, and an improved web content management system, which for instance allows organizing web content in folders, in the same fashion as media objects.

5.2. gUSE/WS-PGRADE

The Grid User Support Environment (gUSE) enables distributed computing over a range of supported middlewares and backend execution environments [2]. The frontend is driven by an Apache

Tomcat (version 7.0.55) application server with Liferay (version 6.2) already fully incorporated. Recent developments in MoSGrid comprised exhaustive updates of the frontend node and gUSE itself (going from version 3.6.3 to 3.7.4), which was accompanied by the migration from Liferay 6.1 to version 6.2 [26]. In addition to the usual upgrade routine, the operating system deploying Tomcat was also updated to Scientific Linux 6.7.

In essence, gUSE consists of a series of web applications that provide all functionality required to offer a modern science gateway. The DCI-bridge allows a standardized connection to distributed computing infrastructures (DCIs) by introducing a new layer of abstraction between workflow systems and the computing resources. Assembly and presentation of statistics is accomplished with the Stataggregator and Statvisualizer modules, respectively. Most importantly, WS-PGRADE offers portlets for creating and modifying workflows, as well as the possibility to download their templates from the gUSE repository.

The latest recent update encompassed redeployment of gUSE web applications in the context of the portal reinstallation. The DCI-bridge descriptor was the most important configuration to update, to ensure that it complies with the new XML schema specifications and contains up-to-date configuration for the UNICORE middleware.

5.3. Portal Databases

Data persistence is guaranteed by a MySQL database management system with two separate databases for gUSE and Liferay [2,26]. Among others, the tables contain information about workflows, statistics, and user credentials. We employ the gUSE database for having user submitted workflows in a persistent state and to keep their work safe. The Liferay database keeps track of user-specific attributes like portlet configurations on private pages. Table schemas are automatically updated when migrating from Liferay 6.1 to Liferay 6.2. It is required to dump tables from the previous installation and submitting encompassed queries to the newly established MySQL DBMS, the update process then commences on the next start of Liferay and the integrity of the tables and the document library is then validated.

5.4. Portlet API

Integrating new components (e.g. simulation or visualization portlets) should be possible for different developers in a highly consistent way. This need gave rise for establishing an API for portlets specifically adapted to the requirements of MoSGrid [1], which are: (a) portlets with a consistent layout for the submission of scientific workflows, (b) a well-documented way to connect new portlets to XtreemFS, and (c) the monitoring of intermediate and output files with subsequent data visualization. All MoSGrid-specific components extend the Portlet API.

From a developer's perspective, Java is the language of choice for MoSGrid extensions. Bindings for gUSE features are provided by its Application-Specific Module (ASM), Liferay access is provided by the respective Liferay SDK, and server-side components are easily created with the Vaadin Framework. For the simulation portlets, ASM provides methods to query the repository of user workflows when the respective user credentials are provided. Combining the Liferay SDK with the Liferay IDE plugin for Eclipse provides a powerful way to create new portlets plugins, hooks, layouts, and themes. Especially the last plugin type was employed to give MoSGrid its design; the corresponding theme is developed with the help of the Apache Velocity project.

The Portlet API is currently characterized via a strong dependence on other components, e.g. the XFSbridge, which is not necessarily required for the principal function of the particular domain portlet. This currently has the effect that simulation portlets will fail to start as soon as the XtreemFS installation in Dresden becomes unreachable. This constitutes an important point of improvement in the near future.

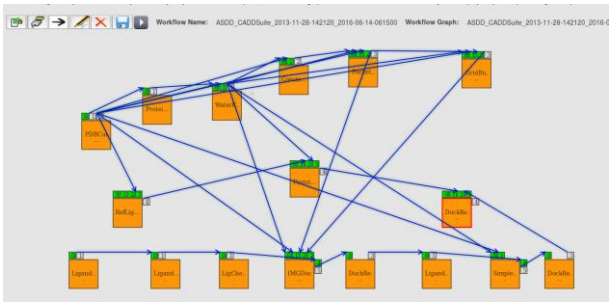

Figure 3 Docking workflow created with the graph editor of WS-PGRADE. It is used within the Docking portlet and available to all MoSGrid users.

Another challenge arises due to inherent incompatibilities between Vaadin 6.8.12 and Liferay 6.2. The Vaadin plugin for the Liferay control panel for this version combination is not functional and any attempts to make it work failed so far. As a consequence, the compilation of widgetsets with Google Web Toolkit is aggravated and one needs to do this manually. All attempts to do so have also failed

due to problems of the application referring to the correct widgetset.

Javascript libraries offer a large array of opportunities for data visualization and intuitive human-computer interaction. For these purposes, the ChemDoodle [28] and Dygraphs [29] modules are on their verge of being made available again on the updated MoSGrid portal to enable simplified input of chemical structures and visualization with dynamic graphs, respectively. Current work on the gateway heavily focuses on re-establishing widgetset compilation to offer a variety of new portlets in the future.

5.5. Simulation Portlets and Workflows

A new user proceeds as follows to submit a new workflow instance: First, in the input phase, a toolsuite is selected according to the needs of the experiment. Then, a workflow is to be chosen which encompasses a well-defined series of individual processing steps, which can be influenced by parameters provided by the user. The user then switches to the submission phase where individual steps of the workflow are to be examined and adjusted to individual needs with optional default values being preset. Finally, the workflow (see Figure 3) is submitted and the progress can be monitored in the monitoring tab. Intermediate and output files are also made available there. Submitted workflow instances will be stored without a time limit and all related files can be accessed from the XtreemFS portlet, which users can employ to view and download their workflow-related files.

Table 1 Currently available applications and number of curated workflows available through the individual simulation portlets.

Simulation Domain	Toolsuite	Workflows
Quantum Chemistry	Gaussian 09	7
	NWChem	1
Molecular Dynamics	Gromacs	4
	eSBM Tools	1
Docking	Autodock Vina	2
	CADDSuite	1
	FlexX	2

Overall more than 100 scientific workflows have been created and deployed via MoSGrid over the past years. But as there is the need to check each concrete workflow in detail after each (partial) update only a much smaller number is made available through the domain specific portlets (see Table 1). This set of workflows represents well curated simulation protocols, robust enough to ensure that also novice users can obtain meaningful results. A representative example is shown in Figure 4.

Critical from a maintenance point of view are changes to the computing infrastructure accessible through UNICORE. The use of UNICORE registries orchestrating the access to individual compute resources allows to replace particular resources by alternatives e.g., a replacement of a HPC cluster without the need to change workflows. But this requires that everything is perfectly consistent with the previous configuration. Furthermore, the capabilities of the UNICORE incarnation database to map the call of a particular application from a workflow to an installation of that application on a remote compute resource facilitate migration and updates. There are no standards or automatisms for handling e.g., a version upgrade of an application. Such a change can affect the configuration of workflow nodes, the MSML template [30] describing the workflow and sometimes even the structure of the workflow graph. If for example a new version of a particular application provides a modified output scheme, it might be necessary to rework the whole workflow including a parsing step to ensure consistent results.

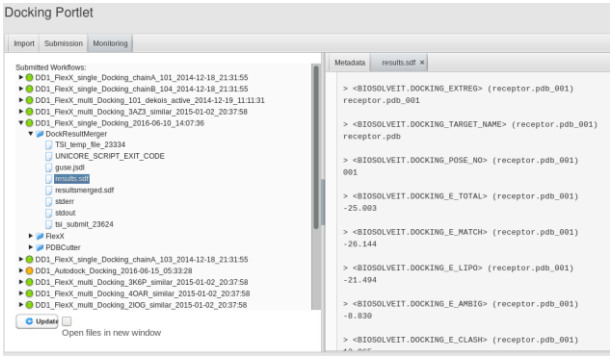

Figure 4 Docking workflow created with the graph editor of WS-PGRADE. It is used within the Docking portlet and available to all MoSGrid users.

6. Stability and Resilience

Providing a stable and resilient service for the users of MoSGrid has the highest priority. Only satisfied users return and use the service regularly or provide constructive feedback which is essential to improve a science gateway. If there would be frequent outages, disappearing jobs or lost data, user satisfaction can not be retained.

MoSGrid is an academic portal relying on the support of various university compute centers distributed over multiple states of federal Germany.

We are truly grateful for the close collaboration over the past years. Such a distributed infrastructure poses some challenges. One, which might be considered as trivial on the first glance, are scheduled maintenances at compute centers. To communicate such an event beginning from the technician being in charge of the work, over the compute center management to the MoSGrid operators and consequently to the user base, remains difficult as all these layers are involved.

Although all services used by MoSGrid are closely monitored, it may happen, in particular when there are unexpected events, that problems and failures are not detected correctly. This can lead to situations where services appear to be operational which in fact are not. Hence workflows can be submitted to compute instances which are not working correctly, leading to failures with error messages that are difficult to trace. The distribution over multiple sites can amplify this problem.

Although plenty of infrastructural challenges exist, MoSGrid was continuously operated over the last six years without extended periods of downtime. On average each year one or two scheduled maintenances of less than a week were performed. Approximately four to twelve technical failures happen every year which are fixed immediately or latest on the next work day. So far no security breach was observed.

The *modus operandi* has to be considered not optimal from a technical perspective, alas the possibly most efficient way to run things regarding the academics involved in MoSGrid. Consequently, we consider the level of service for the users of MoSGrid to be very high, in particular for an academic science gateway, alas not comparable to any industrial environment.

7. Lessons Learned

The maintenance and operation of the MoSGrid science gateway over the past six years brought some unexpected challenges well beyond home-made issues one could eventually plan for. With the end of D-Grid quite a few external services such as virtual organization management or certificate-based authentication mechanisms were not available anymore in the way they originally were. In close collaboration with the compute centres, solutions could be obtained to ensure the service to the users. In particular, the principle of mutual trust and open communication turned out to be the key aspects for successful collaboration.

Another more technical aspect we faced is the migration of a science gateway instance. Migrating an existing production environment to a new machine with an updated operating system and also updating essential components of the software infrastructure introduces challenges, which were not realized at the time the project was concluded. Hence, a careful planning and generous provision of manpower, time and further resources are essential to ensure a successful update. In order to prepare for this endeavour an almost identical development instance resembling a twin of the actual MoSGrid instance was used to evaluate and practice update steps. This proved to be highly useful but cannot prepare for all details which might arise, complicating matters.

The actual effort spend on the maintenance of individual components is difficult to quantify. Updating a solitaire configuration file or workflow node is a matter of minutes. To organize such a step ensuring consistency over multiple sites easily extends to an effort distributed over several weeks. Hence, providing generally applicable numbers for the challenges depicted in this paper is not possible in a meaningful way.

To generalize, the experience gained over the past years but in particular throughout the update: Choose a robust and actively maintained science gateway framework suiting the needs of developers and community. Enrich such a framework with features and services which truly add an extra value. Keep the number of used technologies, interdependencies and overall complexity as low as possible. Ensure that direct access to remote compute and data instances is possible. Following these advices already in the design phase for a new science gateway improves the chances for its robustness and consequently a positive user experience.

8. Future Work

Quite a few new technological approaches have emerged since the original design of MoSGrid was conceived. Most notably is the containerization of applications. This approach is not new by itself, but since Docker [31] made its appearance and became useable on a production ready level, it should be considered to contain small-scale but complex scientific applications [32]. It eases version management and deployment and also facilitates the usage of cloud resources for high throughput workflows. For high scaling MPI parallel applications this approach remains suboptimal. It also should be considered that the (academic) compute centers connected to the science gateway need to support Docker. Furthermore, quite some effort has to be made to handle user mapping, data staging and security issues for Docker instances. We anticipate to port most

of our molecular docking applications and workflows to such an environment to increase their flexibility and sustainability.

Another trend is the usage of mobile devices for almost everything, so why should science gateways not be used with such devices [5,33]? For MoSGrid this means that all webpage content is designed in a simple, readable and scalable way. Wherever visual content like graphs, plots or molecular structures are displayed, we rely on WebGL enabled solutions, like ChemDoodle or Dygraphs [28,29]. We intend to go further into this direction, especially for the submission of scientific workflows. For the thorough analysis of simulation data, we see some natural limits as the processing of these large amounts of data or the visualization of complex simulation systems on a mobile device clearly has its limits.

We are looking forward to see further virtualization possibilities to appear. The employment of so called microservices promises to enable scientists to conveniently move compute resources to their research data and not vice versa.

9. Summary

MoSGrid is now successfully operated for more than six years, well over the original funding period. Several hundred users have used MoSGrid successfully throughout this time period and it is also regularly used in teaching higher education classes. We hope that sharing our experiences in providing this service to the computational chemistry community will provide some insight useful to other communities. We suggest to consider a lean, robust and open-source technology stack when making design decisions for future science gateways.

10. Acknowledgement

The excellent collaboration with the Zuse Institute Berlin (ZIB), the Regional Computer Center Cologne (RRZK), the Paderborn Center for Parallel Computing (PC^2), and the Center for Information Services and High Performance Computing Dresden (ZIH) is gratefully acknowledged. Special thanks go to S. Gesing, P. Thiel, P. Schäfer, S. Herres-Pawlis and L. Packschies for the long and fruitful collaboration. Furthermore, financial support by the German Research Foundation for the MASi project (NA711/9-1) is gratefully acknowledged.

11. References

[1] Krüger, J., Grunzke, R., Gesing, S., et al. The MoSGrid Science Gateway – A Complete Solution for Molecular Simulations. *Journal of Chemical Theory and Computation 10*, 6 (2014), 2232–2245.

[2] Kacsuk, P., Farkas, Z., Kozlovszky, M., et al. WS-PGRADE/gUSE Generic DCI Gateway Framework for a Large Variety of User Communities. *J. Grid Comput. 10*, 4 (2012), 601–630.

[3] Gottdank, T. Introduction to the WS-PGRADE/gUSE Science Gateway Framework. In *Science Gateways for Distributed Computing Infrastructures*. Springer International Publishing, Cham, 2014, 19–32.

[4] Shahand, S., Jaghoori, M.M., Benabdelkader, A., et al. Computational Neuroscience Gateway: A Science Gateway Based on the WS-PGRADE/gUSE. In *Science Gateways for Distributed Computing Infrastructures*. Springer International Publishing, Cham, 2014, 139–149.

[5] Sciacca, E., Vitello, F., Becciani, U., Costa, A., and Massimino, P. VisIVO Gateway and VisIVO Mobile for the Astrophysics Community. In *Science Gateways for Distributed Computing Infrastructures*. Springer International Publishing, Cham, 2014, 181–194.

[6] Kocair, Ç., Şener, C., and Akkaya, A.D. Statistical Seismology Science Gateway. In *Science Gateways for Distributed Computing Infrastructures*. Springer International Publishing, Cham, 2014, 167–180.

[7] Afgan, E., Baker, D., van den Beek, M., et al. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2016 update. *Nucleic acids research*, (2016), gkw343.

[8] Ardizzone, V., Barbera, R., Calanducci, A., et al. The DECIDE Science Gateway. *Journal of Grid Computing 10*, 4 (2012), 689–707.

[9] McLennan, M. and Kennell, R. HUBzero: A Platform for Dissemination and Collaboration in Computational Science and Engineering. *Computing in Science and Engineering 12*, 2 (2010), 48–52.

[10] Klimeck, G., McLennan, M., Brophy, S.P., Adams III, G.B., and Lundstrom, M.S. nanoHUB.org: Advancing Education and Research in Nanotechnology. *Computing in Science & Engineering 10*, 5 (2008), 17–23.

[11] Hull, D., Wolstencroft, K., Stevens, R., et al. Taverna: a tool for building and running workflows of services. *Nucleic Acids Research 34*, suppl 2 (2006), W729–W732.

[12] Apache Taverna. https://taverna.incubator.apache.org/.

[13] Apache Airavata. https://airavata.apache.org/.

[14] Russell, M., Dziubecki, P., Grabowski, P., et al. The Vine Toolkit: A Java Framework for Developing Grid Applications. In *Parallel Processing and Applied Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, 331–340.

[15] Apache Flex. http://flex.apache.org/.

[16] Berthold, M.R., Cebron, N., Dill, F., et al. KNIME - The Konstanz Information Miner. *SIGKDD Explorations 11*, 1 (2009), 26–31.

[17] Roure, D. De, Goble, C., and Stevens, R. Designing the myExperiment Virtual Research Environment for the social sharing of workflows. *Future Generation Computer Systems*, IEEE Computer Society (2007), 603–610.

[18] Hofmann, M. and Klinkenberg, R. *RapidMiner: Data Mining Use Cases and Business Analytics Applications*. Chapman & Hall/CRC, 2013.

[19] Murphy, J., Valdes, R., Phifer, G., Tay, G., and Revang, M. Gartner: Magic Quadrant for Horizontal Portals. 2014. https://www.gartner.com/doc/2861117/magic-quadrant-horizontal-portals.

[20] Hupfeld, F., Cortes, T.., Kolbeck, B., et al. The XtreemFS Architecture - A Case for Object-based File Systems in Grids. *Concurrency and Computation: Practice and Experience 20*, 17 (2008), 2049–2060.

[21] Benedyczak, K., Schuller, B., Petrova, M., Rybicki, J., and Grunzke, R. UNICORE 7 - Middleware Services for Distributed and Federated Computing,. *International Conference on High Performance Computing Simulation (HPCS)*, (2016), (accepted).

[22] Gesing, S., Grunzke, R., Krüger, J., et al. A Single Sign-On Infrastructure for Science Gateways on a Use Case for Structural Bioinformatics. *Journal of Grid Computing 10*, 4 (2012), 769–790.

[23] Geant Project: eduGAIN - Interconnecting Federations to Link Services and Users Worldwide. http://www.geant.net/service/eduGAIN/Pages/home.aspx.

[24] Morgan, R.L., Cantor, S., Carmody, S.T., Hoehn, W., and Klingenstein, K.J. Federated Security: The Shibboleth Approach. *Educause Quarterly 27*, (2004), 12–17.

[25] Aguilera, A., Grunzke, R., Markwardt, U., Habich, D., Schollbach, D., and Garcke, J. Towards an Industry Data Gateway: An Integrated Platform for the Analysis of Wind Turbine Data. *International Workshop on Science Gateways (IWSG)*, (2015), 62–66.

[26] Liferay. https://www.liferay.com/.

[27] Grönroos, M. Book of Vaadin. 2010. https://vaadin.com/book.

[28] ChemDoodle. http://www.chemdoodle.com/.

[29] Dygraphs. http://dygraphs.com/.

[30] Grunzke, R., Breuers, S., Gesing, S., et al. Standards-based metadata management for molecular simulations. *Concurrency and Computation: Practice and Experience*, (2013), http://doi.wiley.com/10.1002/cpe.3116.

[31] Docker. https://www.docker.com/.

[32] Haydel, N., Gesing, S., Taylor, I., et al. Enhancing the Usability and Utilization of Accelerated Architectures via Docker. *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*, (2015), 361–367.

[33] Rodriguez, J.M., Zunino, A., and Campo, M. Introducing mobile devices into Grid systems: a survey. *International Journal of Web and Grid Services 7*, 1 (2011), 1.