GES 2007 German e-Science

Available online at http://www.ges2007.de

Interoperable job submission and management with GridSAM, JMEA, and UNICORE

D. Frank and T. Soddemann

Rechenzentrum Garching der MPG (RZG),
Max-Planck-Institut für Plasmaphysik
Boltzmann Str. 2, 85748 Garching b. München, Germany
Email: thomas.soddemann@rzg.mpg.de
phone: (+49 89) 3299 2694, *fax:* (+49 89) 3299 1301

Abstract

Achieving interoperability between Grid infrastructures is required by all users consuming computing time for projects spanning across Grid domain boundaries. Standards naturally evolve slowly and on Grid level only a few have been proposed and widely accepted so far, among them JSDL. This paper describes how GridSAM which supports JSDL in combination with JMEA can be used to submit jobs to a UNICORE infrastructure and hence how the number of Grid projects accessible via GridSAM can be increased right now.

1 Introduction

While interoperability for Grid-Users seems like a natural requirement, it is a complex problem for middleware deployers. Causes are the different approaches undertaken in the past to provide a useful infrastructure to the user. E.g., while the Globus Toolkit [1] with the version 2 releases focused on a toolkit to be used and extended, UNICORE [2] tried to deploy a vertical solution where everything was tailored to fit together for the draw back that limitations on the server side cannot be overcome without a detailed knowledge of the NJS [3] component.
The Open Grid Forum [4] tries to address the problems of interoperability by recommending standards to the grid community. The most prominent recommendation is certainly the Open Grid Services Architecture [5] which tries to combine standardization efforts in the realms of compute job handling as well as in the realm of access to information and data.
Although the current efforts of Grid middleware projects such as the Globus Toolkit and UNICORE are to become OGSA compliant with their upcoming releases, there is a large pressure from the end user community towards providing interoperability. In order to find out what is possible right now, an activity has been founded under the cloak of the OGF to demonstrate interoperability in a multi-grid scenario. This activity is backed by all major grid projects world wide. The experiences gained from realizing interoperability this way, will certainly have an impact on the development of grid middleware and influence the next generation of specifications of the Open Grid

Forum. Nevertheless, all achievements present just showcases which have no immediate effect on the current end-user community which likes to be able to harvest CPU hours from several grid projects.
In order to overcome this drawback the OMII-UK (Open Middleware Infrastructure Institute - UK) initiative [6] (which now expanded into a European project OMII-Europe [7]) supported the development of middleware which is intended to allow cross grid submission and handling. Among them is a job management component GridSAM (Grid Submission And Monitoring web service) [8] which gives users the possibility to submit and manage job requests formulated in JSDL [9] (including some of its extensions e.g. for supporting MPI [10] jobs) to supported Grid middleware infrastructures. The focus of GridSAM is to provide services via the Web Service interface across site boundaries. The GridSAM development was lead by the London e-Science Center [11].
Another way of achieving interoperability is certainly followed by the concept of so-called Science Gateways. Here, users make use of web applications which present the functionality of underlying applications via HTML and allow users to formulate tasks and submit jobs using their web-browsers. Especially with the emergence of the JSR-168 specification, the so-called portlet specification [12], it becomes much simpler to deploy the same web application (or a part of it – the portlet) in different context for different underlying grid infrastructures. Hence, the user employs the same web interface regardless of the underlying infrastructure. In order to connect such a web application to UNICORE, within the DEISA project [13] JMEA [14,15] was developed by the RZG [17]. JMEA is a Java enterprise application which allows clients such as web applications or web services to submit jobs to and manage them with UNICORE [16].
Both, GridSAM as a web service and JMEA as a Java enterprise application implement interfaces which are defined by the SAGA-RG [18] in the Open Grid Forum. GridSAM employs so called connectors to communicate with the underlying distributed resource management (DRM) systems. DRM Connectors did originally exist for Globus, Condor-G [19], and some other local batch scheduling systems, but not for UNICORE.

This paper describes the efforts undertaken to enable GridSAM to submit jobs to UNICORE and monitor/manage them. For this purpose a connector to JMEA was build rather than connecting to UNICORE directly.

A further step in the direction of interoperability can certainly be seen in application service provisioning. E.g., the Application Hosting Environment (AHE) [20] is an application which allows acting as a service provider and allows a user to have everything managed from job preparation, data staging, submission, monitoring, and gathering of outputs as well as automatic post-processing results. The AHE makes use of GridSAM.

With the GridSAM DRM connector to UNICORE, it was demonstrated at Supercomputing 06 in Tampa, FL, that users employing the AHE can submit jobs to DEISA (UNICORE based infrastructure), TeraGrid [21] (Globus based infrastructure), and several UK (Globus and Condor-G based infrastructure) sites without the need of adjusting to the differences of their underlying infrastructure. The same setup is currently in production use in the DEISA grid infrastructure.

2 Involved Technologies

2.1 UNICORE

UNICORE is a Grid-middleware suite which targets the job management in a Grid Infrastructure. In its version 5 it offers capabilities of manually discovering resources, submitting compute jobs to those, and managing such

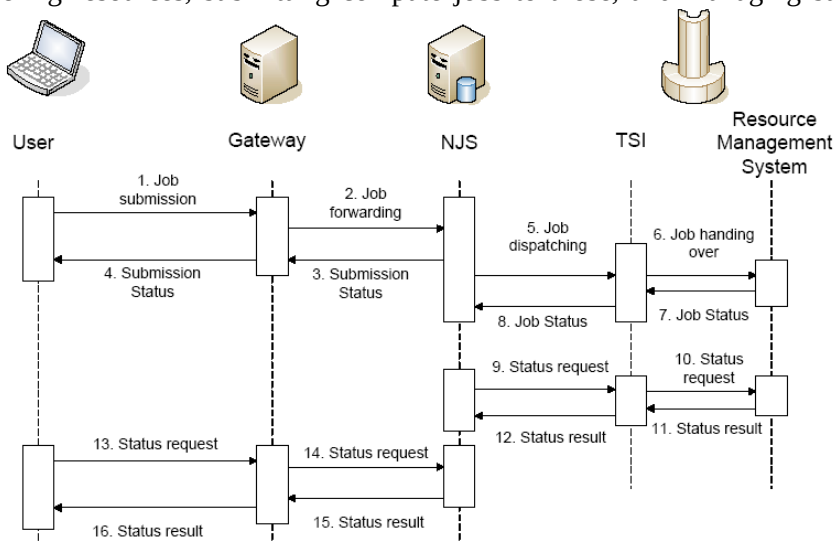

Figure 1: Job submission and management sequences in UNICORE.

jobs. Along with the job submission and result retrieval files can be send along either included in the submission or employing alternate transport methods manageable by UNICORE (e.g. via GridFTP [22]). Figure 1 sketches in a simplified sequence diagram a job submission sequence and a job status request sequence in order to facilitate a general understanding of a UNICORE based Grid infrastructure.

The user uses a UNICORE client to submit a job to a previously selected UNICORE enabled site. During the job submission first the Gateway of the target site is contacted. It performs an authentication and a part of the authorization by forwarding to the NJS only submission requests of authenticated users. NJS is an acronym for the Network Job Scheduler, the UNICORE core component on the server side. It receives a Job submission request from the gateway and performs an authorization employing a user database. Furthermore, it caches the job request in database in order to be able to retrieve job information after restart. Once the job request has suc-

cessfully been handled by the NJS, it replies with an according successful submission message to the user which is routed via the gateway. In subsequent step (5) the NJS tries with the help of the so-called target system interface (TSI) which acts as a binding component between NJS and the local batch scheduling system to dispatch the user's job to the local compute resources. The status is then reported back to the NJS and cached. If the job has successfully been dispatched to the local resource management system, the NJS queries periodically this system via the TSI and updates the job status in its database (9-12). User requests on job stati are answered by cache lookups rather than contacting the local batch scheduling system (13-16). In the same manner job results are retrieved once the job has been completed.
The protocol used for the communication between client, gateway, and NJS is proprietary and called UPL. It consists in principle of java objects send serialized over an encrypted channel (SSL/TLS). Actual requests are formulated as instances of subclasses of an AbstractJob and form the so-called Abstract Job Objects (AJO).
As reported in [15] and [16] it is not straight forward to integrate UNICORE into a multi thread and multi user environment, since the available client library Arcon [23], was developed with having a single user in mind. Hence, an enterprise application (JMEA) was developed within the DEISA project, which hides proprietary details of UNICORE, adds functionality and increases performance in the interaction with the user.

2.2 The Job Management Enterprise Application – JMEA

JMEA's main purpose is to act as a kind of conduit between a Java based application (such as a Java EE web application) and resource management systems. In particular, the current available version of JMEA [14] supports UNICORE as a resource management system. Figure 2 sketches a typical

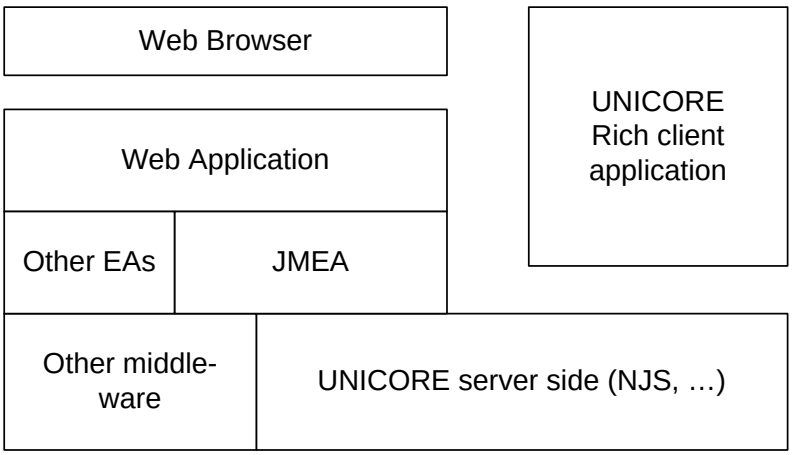

Figure 2: Typical use of JMEA and UNICORE in a Grid Environment

deployment of JMEA in a Grid infrastructure such as DEISA.
JMEA plays the center role in managing jobs seen from a web applications (or equally possibly a web service) point of view. It almost completely hides

the proprietary interface of UNICORE by offering a common interface following the suggestions of the SAGA working group of the Open Grid Forum. Furthermore, it also supports staging of data completely independent of UNICORE itself.
JMEA itself consists of five major components as sketched in Figure 3. The UNICORE implementation of JobManager Library provides all functionality needed to talk to the UNICORE infrastructure. It implements a SAGA like interface and additionally offers methods for natively dealing with a UNICORE infrastructure, since in a simple API not everything is reflected what is offered by an underlying resource management system. The JobManager stateless Enterprise Java Bean (EJB) implements the same interface. As an EJB it is a remote component managed by an enterprise application container. Hence, with this EJB a single point of entry to a UNICORE infrastructure can be created for each organization. The JobManager Service MBean is a management bean which takes care of housekeeping tasks and e.g. manages the timer tasks for querying the resource management system on updates of job stati. For that purpose it makes use of other components such as EJBs.

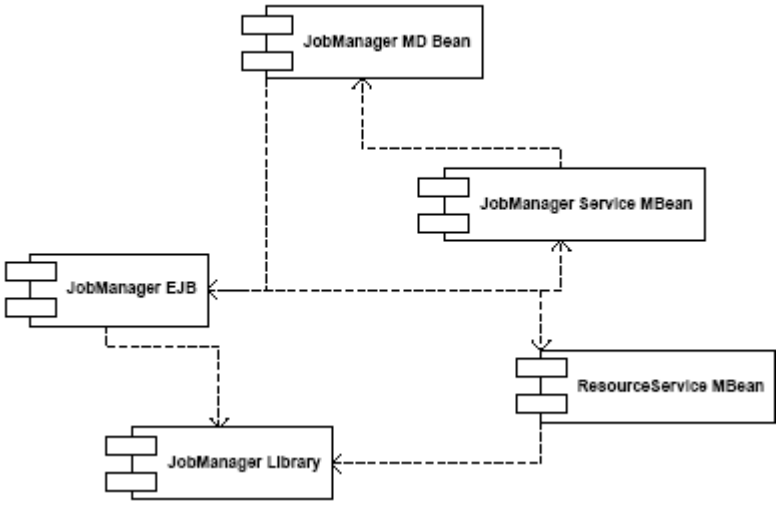

Figure 3: JMEA components and their dependencies

The JobManager Message Driven EJB implements a subset of the JobManager stateless EJB's functionality. As a message driven been the interface is designed for asynchronous interaction. Such beans are e.g. employed by the JobManager MBean to update Job information asynchronously and in parallel. The ResourceService MBean is mainly responsible for resource discovery and resource information provisioning. In the UNICORE implementation it makes use of the NJS capability to report resource information statically configured in via the TSI configuration. It could easily be extended to use information available from outside a UNICORE context such as LDAP systems [24], Globus MDS4 [25], and Inca [26].

While JMEA's development goal was to create an enterprise application which could be interfaces by Java web applications, GridSAM was created to offer a web service based access to a grid infrastructure. How GridSAM works and what had to be added in order to make it interoperable with JMEA will be discussed in the next section.

3 GridSAM and the New Connector Plugin to JMEA

GridSAM can be seen a framework for creating a job submission and monitoring service. It was developed within the OMII-UK project and is already shipped with support for various underlying resources management systems. It provides a set of web service methods for submission and monitoring of jobs. These are implemented in Java employing the Apache Axis 1.x web service stack [27] and the Hivemind microkernel [28]. Job submissions have to be formulated in the Job Submission Description Language (JSDL), a standard proposed by the Open Grid Forum. This ensures interoperability at least on the level of the exchanged job description.

As a framework GridSAM offers extension points which allow a customization for a given infrastructure or environment. In particular, that is realized by supporting plugins which can be made available by configuration. GridSAM is already shipped with support for CondorG and Globus, but lacked support for UNICORE. In the following, we briefly describe the UNICORE integration, and thereby have a glance at GridSAM's internal architecture.

GridSAM plugins have to extend an abstract class in order to define the sequence of stages is defined which need to be executed once a job arrives at GridSAM. Figure 4 sketches the different stages a job traverses from submission to its end as states in an UML like state diagram. In a first validation stage the general validity of the submitted JSDL document is checked. Next, in the identification stage, a job request is created within GridSAM and the

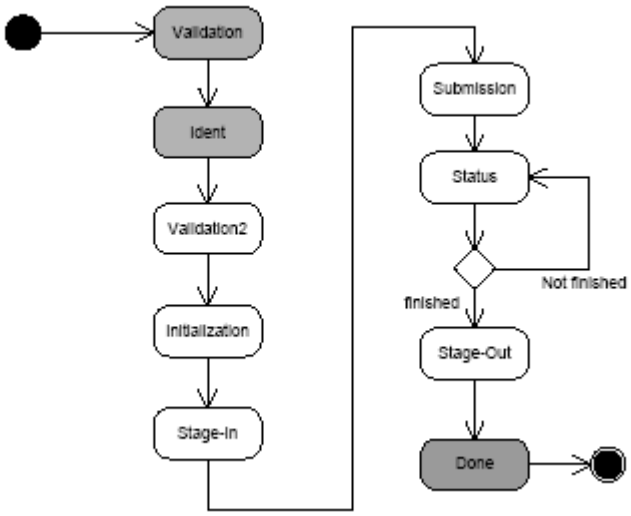

Figure 4: Sketch of a state diagram of the GridSAM resource management connector. White states indicated UNICORE extension points.

job gets an identifier. Then it will encounter the first UNICORE specific

stage (Validation2) where a check on the JSDL document is performed to ensure that it contains all fields necessary to submit a job to a UNICORE infrastructure. Then the UNICORE job is initialized by creating a temporary job staging area at the submission site. All files defined in the JSDL document to be staged are then transferred to that staging area by employing GridFTP. In the submission stage the remainder of the JSDL document is translated into the UNICORE proprietary abstract job object (AJO) format. The resulting AJO is then submitted via JMEA to UNICORE. The job within GridSAM enters now the status stage where the status of the job is periodically monitored. Since JMEA takes care of the monitoring, GridSAM's monitoring information is directly retrieved from there. Once the job has finished, results are staged out to the initially specified locations and the done stage will be entered.

For the interaction with UNICORE, the so-called Explicit Trust Delegation model [] was employed, which allows an agent (JMEA in this case) to submit jobs on behalf of a user, which is trusted to be authenticated by the agent.

In order to be able to identify a user, GridSAM had to be extended to make the information of the authenticated user available at the submission stage for every supported security model. Since UNICORE requires a certificate of the user to be sent along, a certificate pool has to be employed by the GridSAM connector to JMEA for the case that only point to point security (HTTPS) is employed and that the user sent along e.g. myproxy credentials within the JSDL document. Some myproxy servers will only create proxy credentials and do not send a complete certificate chain. UNICORE on the other hand needs the normal x.509 user certificate in order to be satisfied.

4 Demonstration of Interoperability of Three Major Grids

The combination of GridSAM, JMEA, and UNICORE offers users a great opportunity to submit jobs in a coherent way to different Grid infrastructures offering GridSAM as their gateway. This was actually exploited by a DEISA extreme computing initiative (DECI) project.

The DECI project LIAMS [30] was carried out in 2006 by the Centre for Computational Science, UCL, London, UK. In two studies of biological macromolecules large-scale molecular dynamics (MD) simulations were performed on resources provided by DEISA, TeraGrid, and the National Grid Service (NGS) in the UK. In order to increase the ease of use, LIAMS simulation relied on the use of the Applications Hosting Environment [31] developed within the RealityGrid project [32] in the UK. AHE's focus is to provide scientists with application specific services to utilize grid resources in a quick, transparent manner with the scientific objective as the main driver of the activity. The job submission and management backend employed by AHE is GridSAM.

Figure 5 sketches the deployment of the different components grouped by organization. The AHE is deployed at the UCL and used by several users from the department of chemistry. These users interface the AHE with a rich client application. With its help they can make simulation datasets available and formulate job submissions interactively, which are upon submission converted into JSDL. The AHE takes care of submitting a job request to the

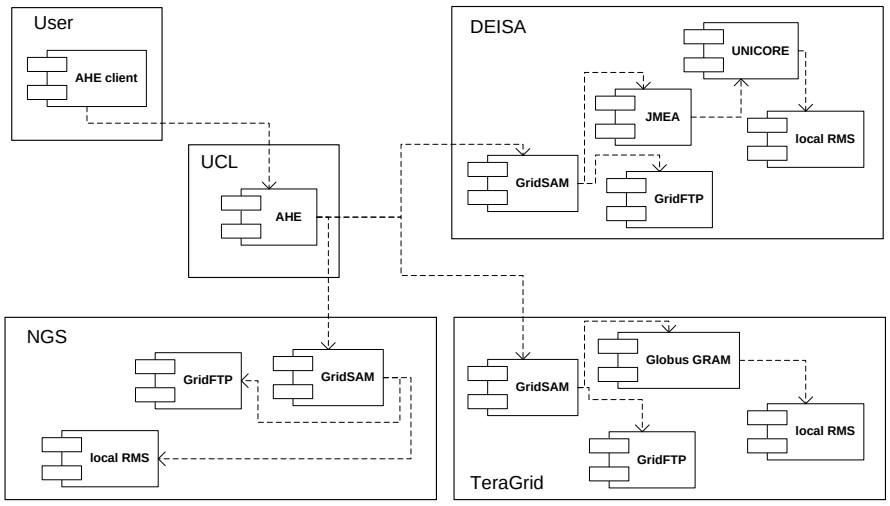

Figure 5: AHE making use of GridSAM in a Multi Grid Environment

selected computing infrastructure. For that purpose, the AHE interfaces GridSAM in the various deployment scenarios at NGS, TeraGrid, and DEISA.

As a result interoperation is achieved by providing access to these Grid infrastructures by giving the users the opportunity to submit and manage jobs with the same components.

This multi Grid infrastructure could be demonstrated at the SC|06 conference in Tampa, FL, USA [33]. For that purpose dedicated resources within DEISA, TeraGrid, and the NGS were accessed during several presentations held at various booths of participating partners in these Grid projects on the exhibition floor.

5 Conclusions and Outlook

The described work is certainly a good example of how interoperation on a small scale can already be achieved today by employing only a minimum number of standards. In this case in principle only JSDL standard proposal and the SAGA-WG proposal of a job submission and monitoring application interface have been applied on Grid level. And of course, there is a whole set of standards working silently below that level.

It is obvious that we will see more and more researchers harvesting computing time offered by multiple grid projects in the future. Virtual organizations of researchers will no longer exist as organizational parts of a single Grid project. Rather, they will be created outside of the established Grid infrastructures we know today and request computing time as needed. Those

demands will not necessarily be met by a single Grid infrastructure but more likely by the combined strength of an alliance of infrastructures formed as needed.

Hence, Grid interoperation is an extremely important topic. Standardization is crucial to minimize the effort in dealing with otherwise diverse infrastructures. Blueprinting a complete architecture for a standardized ideal interface to a Grid is carried out by the working groups of the Open Grid Forum. But such an effort takes time. Achieving Grid interoperation now may only be possible on small scale by concentrating on tiny parts of the big picture. Nevertheless, these tiny bits and pieces, could they not make a difference, too?

This work has shown that already today for the given purpose the major players of the Grid middleware, Globus and UNICORE, can be accessed in the same way by utilizing a GridSAM enabled infrastructure. Naturally, many aspects have not been addressed. Among them the still unsolved puzzle on how to choose the best matching computing resource (and how to define it in first place). While here all resource have been selected manually, automated resource discovery and meta scheduling are and will be an important topic not only from a computational scientist's point of view.

Acknowledgement

The authors thank Peter Coveney, Stefan Zazada, Stephen McGough, and Hermann Lederer for their efforts in supporting the presented work.

References

1. Globus Toolkit, http://www.globus.org/toolkit/
2. UNICORE, http://www.unicore.eu/
3. NJS, Network Job Scheduler, see 2
4. Open Grid Forum, OGF, http://www.ogf.org/
5. Open Grid Service Architecture, OGSA, http://www.gridforum.org/documents/GWD-I-E/GFD-I.030.pdf, http://www.ggf.org/documents/GFD.80.pdf
6. Open Middleware Initiative Institute, OMII-UK, http://www.omii.ac.uk/
7. OMII-Europe, http://www.omii-europe.com/
8. Grid Job Submission and Monitoring Web Service, GridSAM, http://gridsam.sourceforge.net/
9. Job Submission Description Language, JSDL, http://www.gridforum.org/documents/GFD.56.pdf
10. Message Passing Interface, MPI, http://www.mpi-forum.org/
11. London e-Science Center, http://www.lesc.ic.ac.uk/
12. Java Specification Request 168, JSR-168 (Portlet Specification), http://jcp.org/en/jsr/detail?id=168
13. Distributed European Infrastructure for Supercomputing Applications, DEISA, http://www.deisa.eu/
14. Job Management Enterprise Application, JMEA, http://sourceforge.net/projects/jmea
15. T. Soddemann, Job Management Enterprise Application, JMEA, in W. Lehner et al. (Eds.): Euro-Par 2006 Workshops, LNCS 4375, pp. 253–262, 2007.

16. T. Soddemann, Science gateways to DEISA: user requirements, technologies, and the material science and plasma physics gateway, Concurrency and Computation: Practise and Experience, DOI 10.1002/cpe: 1082
17. Rechenzentrum der Max Planck Gesellschaft, Garching, RZG, http://www.rzg.mpg.de/
18. Simple API for Grid Applications – Research Group, SAGA-RG, https://forge.gridforum.org/projects/saga-rg/
19. Condor-G, http://www.cs.wisc.edu/condor/condorg/
20. Application Hosting Environment, AHE, http://www.realitygrid.org/AHE/
21. TeraGrid, http://www.teragrid.org/
22. GridFTP, http://www.globus.org/toolkit/docs/4.0/data/gridftp/
23. Arcon, http://www.unicore.eu/download/unicore5/
24. LDAP, RFC 2251, http://www.ietf.org/rfc/rfc2251.txt
25. WS-MDS/MDS4, http://www.globus.org/toolkit/docs/4.0/info/
26. Inca, http://inca.sdsc.edu/
27. Axis, http://ws.apache.org/axis
28. hivemind, http://hivemind.apache.org/
29. D. Snelling, et al., Explicit Trust Delegation, FUJITSU Sci. Tech. J., 40,2,p.282-294(December 2004)
30. LIAMS, http://www.deisa.org/applications/projects2005-2006/liams.php
31. AHE, http://www.realitygrid.org/AHE/index.shtml
32. RealityGrid, http://www.realitygrid.org/
33. SC|06, http://sc06.supercomp.org