A Science Driven Production Cyberinfrastructure - the Open Science Grid

The Open Science Grid Executive Board: Mine Altunay[1], Paul Avery[2], Kent Blackburn[3], Michael Ernst[4], Dan Fraser[5], Rob Quick[6], Rob Gardner[7], Sebastien Goasguen[8], Tanya Levshina[1], Miron Livny[9], John McGee[10], Doug Olson[11], Ruth Pordes[1], Maxim Potekhin[4], Abhishek Rana[12], Alain Roy[9], Chander Sehgal[1], Igor Sfiligoi[12], Frank Wuerthwein[12]
Contact: Ruth Pordes, OSG Executive Director, Fermilab, PO Box 500, Batavia, IL
E-mail: ruth@fnal.gov

Abstract. This article describes the Open Science Grid, a large distributed computational infrastructure in the United States which supports many different high-throughput scientific applications, and partners (federates) with other infrastructures nationally and internationally to form multi-domain integrated distributed systems for science. The Open Science Grid consortium not only provides services and software to an increasingly diverse set of scientific communities, but also fosters a collaborative team of practitioners and researchers who use, support and advance the state of the art in large-scale distributed computing. The scale of the infrastructure can be expressed by the daily throughput of around seven hundred thousand jobs, just under a million hours of computing, a million file transfers, and half a petabyte of data movement. In this paper we introduce and reflect on some of the OSG capabilities, usage and activities.

Keywords: Scientific Computing, Distributed Computing, Grids, High Throughput Computing, Data Intensive Computing

Acknowledgements: We acknowledge the prior, current and hoped for future many contributors and sponsors of the OSG use and work.

1. Overview

The Open Science Grid (OSG) [1] provides a *collaborative environment for communities of scientists and researchers* to work together on both common and user specific distributed computing problems and solutions. The collaboration includes a broad, multi-disciplinary community of scientists and researchers, IT providers, software developers, educators and computing administrators. The OSG partners with peer organizations in the US and abroad to provide integrated solutions for its users and also supports groups to build and operate their own distributed systems to meet their local needs. The OSG project is funded jointly by the Department of Energy SciDAC-2 program and the National Science Foundation.

In this paper we: Introduce the OSG, including the underlying goals, fundamental concepts, principles and organization; Explain the operational services that the OSG delivers to its users in support of their computing activities, including security; Tell you about the software that OSG packages, releases and supports—and that is used by both the OSG sites and users as well as other communities and projects; Summarize the application usage modes through the OSG and compare alternative methods that we support for a user to submit their jobs and accomplish their work; Cover what a site and resource owner needs to do to make their farms and/or disks accessible to users of the OSG, including how the idea of community (Virtual Organization or VO) relates to the (end)-user and (owned)-site; Give some challenges of interoperation of heterogeneous infrastructures; And finally, touch on some new technology directions we are currently working on.

2. Introduction

The communities contributing to the OSG Consortium drive the capabilities and evolution of the infrastructure, software and activities. The multi-agency sponsorship of the OSG, mentioned above, provides a unique opportunity for participation at all scales, from individual research PIs to several thousand member global scientific collaborations and small university campus groups to large DOE laboratory facilities.

There is active participation in and use of the OSG by groups from molecular dynamics, protein structure prediction, biology, climate, text mining, and computer science. However, the user communities with the most

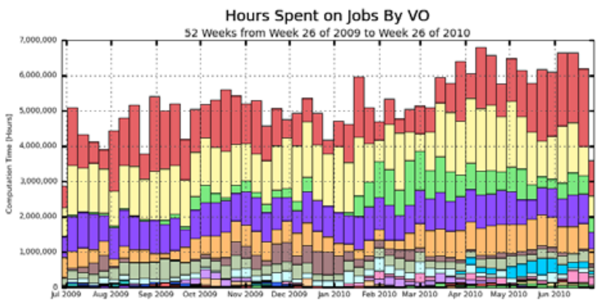
[1] Fermilab, [2]University of Florida, [3]Caltech, [4]Brookhaven National Laboratory, [5] Argonne National Laboratory,[6]Indiana University, [7]University of Chicago, [8]Clemson University, [9]University of Wisconsin Madison, [10] RENCI, [11] Lawrence Berkeley National Laboratory, [12]University of California, San Diego.

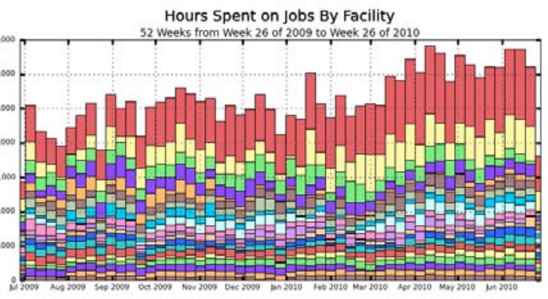
challenging needs, scale and sustained use, are the large physics collaborations in the United States. The OSG provides the computing infrastructure in the United States for the Large Hadron Collider (LHC) ATLAS [2] and CMS [3] experiments. Other major users are the Laser Interferometer Gravitational Wave Observatory (LIGO) [4], the Tevatron experiments (D0 and CDF) and the STAR Relativistic Heavy Ion Experiment. The software used on the OSG is based on the Condor [5] and Globus [6] technologies. Today several thousand users have accessed the OSG infrastructure and services.

The *OSG* provides (research and science) communities the means to build *vertically integrated distributed computational systems to their specific needs*. Many of the larger user communities have "thick" middleware and processes that augment those provided by the OSG. Such communities support and use their *own community grid, layered over the common platform* provided by the OSG. The OSG has active collaborations with community projects (e.g. the Data Intensive Science University Network and the LIGO Data Grid) to leverage the strengths and activities of the different organizations and to help harmonize the resulting systems.

At the other end of the scale, the researchers in smaller communities have little or no time available even to learn how to use the technologies and processes. For such communities the OSG provides tools, documentation and support of *low overhead use of the common platform*. The OSG thus supports "ready-made" end-user services and software and embedded help for adapting and developing user applications to run on the OSG facility and help for running local distributed infrastructures.

OSG does not own any computing or storage resources. Rather, these are contributed by the members of the OSG Consortium and used both by the owning organization as well as shared for use by others. The resource owners can select which communities they provide access to their clusters and storage, and the priority assigned to jobs scheduled on their site. The resources accessible through the OSG infrastructure span more than 70 university and laboratory institutions in the United States and South America. Today, mid-2010, the OSG provides access to 113 compute clusters, 65 data or tape storage systems, and 9 partner campus and regional infrastructures. Nearly all of the resources are commodity clusters running a variant of the Linux operating system and NFSor otherwise managed data disk-caches appropriate for the OSG's focus on high-throughput and data intensive applications. Typically any application will use more than a dozen of the accessible clusters in any production run. An overview of OSG CPU usage and data transfer is shown in Tables 1 and 2, respectively.





Sample chart showing different VOs running over time

Sample chart showing Sites being used over time.

Table 1: Accounting Summaries of VO and Site Usage

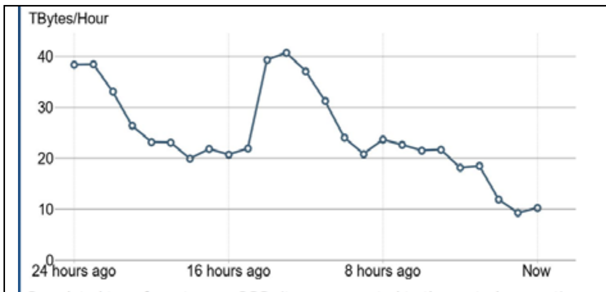
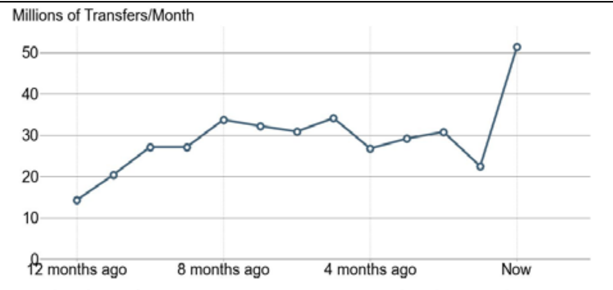
 Completed transfers at many OSG sites are reported to the central accounting system. The above graph shows the number of terabytes moved and accounted for. A total of 1,869,000 transfers totaling 648 TB were recorded.	 Completed transfers at many OSG sites are reported to the central accounting system. The above graph shows the number of terabytes moved and accounted for. A total of 380,494,000 transfers totaling 141,121 TB were recorded.
Chart showing typical variation in data transfer during a day (taken at midnight)	Chart showing recent ramp up in number of file transfers per month due to 2 recent new applications.

Table 2: Accounting Summaries of Data Movement and File Transfer

The OSG architecture is designed to allow contributors to: 1) provide access to their computing and storage resources and/or software developments, 2) benefit from the use of and support of the common software stack and operational services, and 3) store, access and process their data on the ensemble of resources made accessible. Only a thin layer of middleware is shared between them. The architecture allows the resources to be managed and controlled autonomously, especially with regards to operations. Resource providers operate their local distributed resources according to local policies, preferences and expertise. These resources are often integrated with non-remotely accessible resources at the local institution.

The use of the OSG has shown a fifty percent increase over the past two years. About thirty percent of the cycles were executed on resources not owned by the user's community. Non-HEP usage has increased substantially due to the increased ability of LIGO to submit Einstein@Home jobs supporting pulsar analysis. From June 2009 to June 2010, the fraction of non-HEP usage increased 4-fold from about five percent to about twenty percent. The major applications being executed are protein simulations, molecular dynamics in several sub-disciplines, and structural biology.

One measure of the contribution to science is the number of publications where the author's recognize the OSG's services. Three hundred and sixty seven papers have been published over the past 12 months that have benefited from using the OSG (see the OSG NSF Annual Report, June 2010). These publications depended not only on OSG "cycles", but also on OSG–provided software, monitoring and testing infrastructure, security and other services. Of these, 25% are from non-particle physics—a significant increase from the previous year.

3. Principles and Concepts

We describe here the principles, governance and concepts that the OSG has adopted and which provide the underlying foundations for the use and value of the infrastructure and multi-disciplinary collaborations that contribute to it. We can only briefly cover these here due to space constraints, but encourage interested readers to learn more through the main OSG web site[2].

3.1. Principles

The principles that govern the OSG architecture and activities were developed at the start of the project and are described in a "blueprint" which is continuously evolved based on our improved understanding and stakeholder needs. Adherence to and consideration of these principles enables us to work towards a coherent, consistent technical eco-system in an environment of diverse implementations and technologies [7]. While additional principles have been added, we have found that those developed early on have stood the test of time and helped with the improvement and extension of the services offered by the OSG. Examples of some of the principles are:

Foundational Computer Science: Symmetry and recursion in any concepts, architectures, designs and implementations developed for the OSG (for example, no service needs to be unique; if there is one instantiation of the information service there can be many, and the design and implementation of the information software and deployments must allow for this); Services should work toward *minimizing their impact* on the hosting resource, while fulfilling their functions; Services are expected to protect themselves from malicious input and inappropriate use.

Operational: The OSG infrastructure must always include a *phased deployment, with a clear operations model* adequate to the provision of production-quality service; OSG will provide *baseline services and a reference implementation. Use of other services will be allowed*; The OSG infrastructure must be *built incrementally*. The roadmap must *allow for technology shifts and changes*.

Participatory: All services should support the ability to *function and operate in the local environment* when disconnected from the (wide area) OSG environment; Policy should be the main determinant of effective utilization of the resources. Thus, given sufficient requests and open policies for its use, if the OSG is efficient there should be full utilization of the resources. Users are not required to interact directly with resource providers; the requirements for participation should *promote inclusive participation* both horizontally (across a wide variety of scientific disciplines) and vertically (from small organizations like high schools to large ones like National Laboratories).

3.2. Governance

The distributed and autonomous nature of the organizations that make up the OSG collaboration led us to pay particular attention to a governance model that can effectively meet our goals of "openness" (to all research and science) and "partnership" or benefit through mutual contributions. The aim is to provide community specific value as well as develop common artefacts, share knowledge and software, share resources that are otherwise

[2] www.opensciencegrid.org

unused as well as provide agreed upon throughput when needed, foster and sustain a team of experts and experience for support, take decisions on the evolution and extension of the infrastructure and activities, and provide additional value more broadly outside as well as inside the Consortium itself.

The consortium members include those organizations and individuals using the infrastructure and/or contributing cluster, caches, storage as well as projects and organizations developing and providing software and services to the activities and infrastructure. The governing Council adopts policies that cover usage, security and operations, as well as defining the expectations of individual and organization members that contribute to and benefit from the OSG.

Currently the OSG project includes: Production coordination, software, operations, integration and sites, user support and engagement (of new entrants), campus grids, security, training and content management, “scalability, reliability and usability”, work load management, network monitoring, administration and metrics, communication, and some sustaining support for the Condor software. In addition to the core OSG project, several “Satellite Projects” ↓independent projects providing capabilities or services that will be made available to OSG stakeholders using the OSG infrastructure—contribute to the goals and mission of the Consortium. We are finding that the model of contributory partnerships and satellite projects is providing an effective means to extend the scope and influence of the OSG with a minimum of additional effort by the core project staff.

3.3. Virtual Organizations

The OSG methods and processes are based on the *organization, management and use by community groups or Virtual Organizations (VOs).* VOs are typically long-lived stable collaborations, each with well-defined governance. In the OSG context, a VO includes not only the people who are members of a community, but also the resources, services, software and policies of that community. VOs can contain other VOs and/or sub-VOs, interface with each other and share resources, and can have common services, common organizational policies and methods, and common membership. Many communities deploy, manage and use their own community based distributed systems layered over and dependent upon the core OSG platform. The members of the community are able to use additional resources and services outside of their own system by having common software and interfaces with those of the core part of the OSG. University, regional, research and scientific communities with their own infrastructures are able to selectively integrate with and/or rely on OSG services.

3.4. Security

Security management is integrated throughput every activity in the OSG and abides by the overarching principles of maintaining openness and usability for the scientific communities.

The identity management infrastructure of the OSG is based on the X.509 technologies and extended PKI certificates. The authorization and access control infrastructure is based on the “attribute certificates” as profiled in RFC 3281[3]. The multiple sources and implementations of software to create and manage these certificates across the job, data, information and operations services is an exemplar of how collaboration can effectively be brought to bear to work on interoperation and standardization. An example is the recent work on interoperability between the EU and US services to support the XACML protocols [8].

Support is being extended to include Shibboleth-based “identity end-points” in collaboration with InCommon and CILogon projects and use of local security infrastructures within a campus infrastructure with well defined “adaptors” to support bridging to the wide-area OSG environment. The increased integration of web-based and grid-based access tools has led us to develop integrated user tools to improve the usability across the two user identity and authorization domains.

Many challenges to the usability and simplicity of the end-to-end security infrastructure remain. These include: Dependencies on external organizations for key parts of the security infrastructure, both certificate generation and software; Improving the end users experience in use from the laptop, through the local to the remote infrastructures; Engaging the scientific communities in the necessary policies and processes; Monitoring the entire system for unexpected behaviors and coordinating incident response across the globe; And ensuring consistent and complete application of security and authorization models in the software.

4. Operations

Operations for the OSG includes the operation, maintenance and evolution of several different activities: core “service desk” responsibilities; host based services used and relied on by one or more communities; and help for and interactions with the distributed set of VO, site, and partner support organizations. In delivering a highly available production data center, the quality and scope of the operations services and attributes are critical [9].

[3] www.ietf.org/rfc/rfc3281.txt

OSG operations activities are equally driven by our principles, with the specific result that each “OSG-wide” service and practice is developed to enable, and supported for, reuse and replication on other peer infrastructures.

OSG Operations is distributed across multiple institutions. It provides a single contact point for standardized operational services and is responsible not only for triaging and tracking any request or problem, but also for ultimately ensuring the resolution, capturing the lessons learned, and communication of any changes needed to improve the experience from using OSG services for the future.   Operations sets service expectations through Service Level Agreements (SLAs) between the OSG, service providers and the user communities as appropriate.

The day-to-day operational services provided to the OSG community include: Helpdesk response and triaging; Trouble ticket tracking and troubleshooting; Credential distribution sources for both people and resources; Communication hubs including grid-wide event notification and change management announcements; Round the clock emergency security response; Collaborative documentation infrastructure; And administrative, hardware, and hardware environment services.

Host based grid services that Operations is responsible for include: Topology databases; Information services and validation; Monitoring and availability testing; Accounting repositories (current and historical); Software (middleware) caches; Information consolidation and presentation portal; and several “grid-bridging” services described below.

OSG operations develops and provides grid-bridging services — specific software and processes that support use of multiple infrastructures by the user communities. Our goal here, as with other areas of OSG, is to facilitate other organizations and communities to federate their local infrastructures with the OSG to provide a seamless environment. Currently the grid-bridging services include a unique trouble ticket exchange system that allows automated (as well as manual) routing and exchange of service-desk tickets with other ticketing services. This allows institutions contributing to the OSG, as well as partner projects, to maintain their autonomy and local operational practices using their home or local service desk environment. Figure 1 is a diagram of the current ticket exchange architecture between the OSG Footprints, EU GGUS, RT and Remedy service desk implementations.

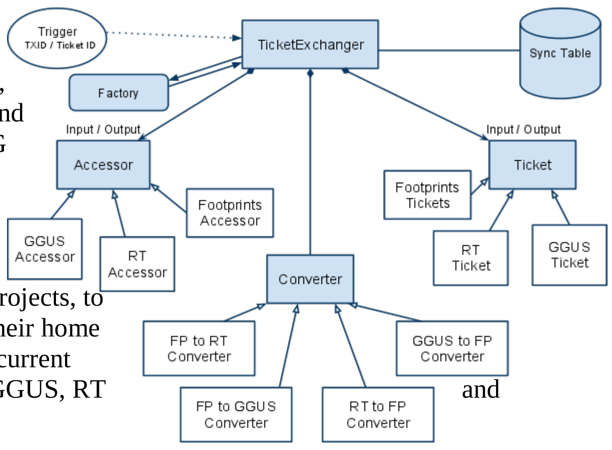


Figure 1: Ticket Exchange Interfaces

Operational security is of paramount importance and requires processes and practices that allow marshalling the whole organization to respond quickly to attacks and threats. The small amount of dedicated project effort is augmented to include, as needed, “all hands to the wheel” to respond to reported concerns, vulnerabilities and incidents. The scale and scope of the diverse, and autonomous sites and use groups and the nature of the partnerships to supply the internationally integrated system for many of the science collaborations means that any banning or removal activities have wide impact. The actions taken to mitigate and respond to threats and incidents benefit from deep attention to the context and risks in the end-to-end eco-system [10].

We are increasingly adopting the ITIL practices of incident, change, configuration, problem and release management. This is bringing us in line with other production data centers in terms of the quality and robustness of our environment. Whether we can apply for ISO20000 certification in the future remains to be seen.

Solid operation is absolutely necessary in a production grid environment. It is the “glue” that holds the whole eco-system together, through providing a small number of processes and host based services needed to run the OSG, and paying attention to measuring effectiveness and cost.

5. Software

To decrease effort for OSG Consortium members and users, OSG provides the OSG Software Stack. Users can install any appropriate subset of the software stack in order to provide or use resources. For the most part, the software stack is built upon existing software (such as Condor, Globus, VOMS to name a few) and OSG does as little software development as possible. In just a couple of circumstances where there are no appropriate software components, OSG has developed it’s own, but our goal is to build upon existing high-quality tools. OSG users are not required to use the software stack, but doing so is the simplest way to

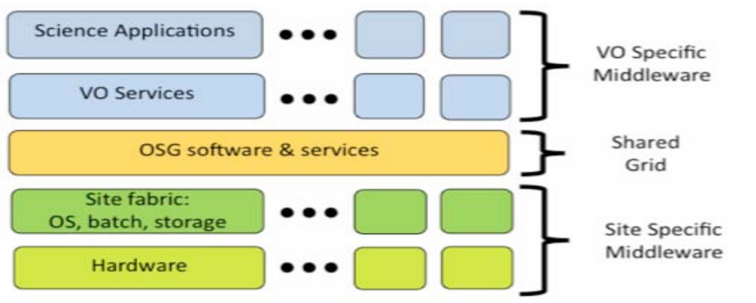


Figure 2: Layers in the End-to-End Software Infrastructure

participate in OSG. The layers of the end-to-end software systems are shown in Figure 2.

The OSG makes integrated tested software releases based on the OSG's Virtual Data Toolkit (VDT [11]) to enable access to and use of the ensemble of processors and storage. The OSG supports these common technologies for both OSG and other projects. We also train new users in their adoption and use. The VDT currently includes more than forty independent components from nearly as many software development groups. The modules span generic open source toolkits to those needed and provided by the user communities themselves. Although we develop little software, it is significant work to build, test, package, distribute, and support these independently developed components into coherent, harmonized sets for end-user computing, data processing and storage services, and operations and management tools. The VDT is released for many variants of Linux and in client mode for Mac OSX. Processes to build, test, and release the software ensure managed evolution and extension of the capabilities offered.

The VDT includes the following software (roughly categorized):

- *Core Distributed Infrastructure Software*: Condor and the Globus Toolkit.
- *Information Services:* including information providers based on the GLUE specification, LDAP repositories, Gratia accounting, monitoring and resource validation scripts, and resource selection and matching services based on Condor ClassAds.
- *Build and testing tools:* the Metronome build and test infrastructure, regression tests etc.
- *Storage Service Implementations:* BeStMan, dCache and XRootd.
- *Security tools and infrastructure:* X509 certificate management, VO management services based on X509 extended attributes, authorization and grid to local account mapping tools.
- *Client Tools:* utilities for accessing OSG services; libraries to read/write data from/to grid-accessible storage sites; workflow tools. These include tools for "submit hosts" which are services separate from the users lap/desk top which provide client services shared by members of one or more VOs.
- *Support Software:* Utilities used by many software packages, including Apache, Tomcat, Berkeley DB, MySQL, OpenLDAP, PHP, Squid web caching and miscellaneous VDT tools to help administrators, support staff and users.

The "software life-cycle" involves much interaction with the users, the site administrators and the OSG operations teams. Requirements for new software versions and components come from the communities and the staff and are prioritized based on the need and effort required. Similarly, a decision to drop support for or remove a software component from the VDT is based on community input.

The software in the VDT is, in the main, built from source and regression tested locally. Then a two phase testing is done, first on a small number of small dedicated OSG sites – where the basic functionality is tested – and then on a larger Integrated Testbed (ITB). OSG operations services, separate from those used in production, are instantiated for the ITB – thus allowing end-to-end testing of the whole infrastructure. The OSG community software and applications are tested as part of the ITB before the final VDT release is authorized. OSG specific configuration scripts and tools are layered over the VDT and supplied by OSG operations. For a major OSG/VDT software release, the whole process spans several weeks.

Patches and bug fixes are treated in a streamlined and expedited process based on the criticality of the change – security related having highest priority – and the likely impact on other software components and services.

6. Using the OSG

Owners of the clusters and disk caches typically provide access to their resources in three "tiers": Guaranteed to the owner's own user community; following agreements between the owner organization and their "friend" user communities; and opportunistic use by other user communities through dynamic resource sharing. By "opportunistic" we mean that owners allow, without specific allocation or agreement, any otherwise unused compute cycles to be used by any community with jobs queued at the local batch or resource management system. As members of the OSG consortium, resource owners are encouraged to provide access of the order of 10% or more of their computing cycles to other communities. As mentioned earlier, in fact, about 30% of the overall usage is opportunistic.

Opportunistic use is a hallmark of the OSG. It provides a low overhead mechanism for users to increase throughput using already provisioned resources. It allows resource owners to automatically enable use of available cycles and storage by other OSG members. It supports the principle of inclusion of members who have no resources of their own but contribute value in other areas.

The *OSG is targeted at high-throughput computing applications,* which means those needing large amounts of computing over long periods of time where the aggregate amount of computing accomplished is more important than the completion of any one specific job. Applications that benefit are large ensembles of loosely coupled parallel applications for which the overhead in placing the application and data on a remote resource is a fraction of the overall processing time. In summary, OSG is particularly effective for: High throughput, pleasantly parallel applications[4]; Job runs of between one hour and several days; Jobs that can be check-pointed; Explicit management of large scale data movement and storage; Ensembles that can effectively run across a large number of resources.

Table 3 below summarizes the types and characteristics of applications running on the OSG, based on a classification from European Grids for EscienceE [12]. Any particular application may have of one or multiple such characteristics.

	Examples on the OSG	**Job and Data Attributes**
Simulation	Physics Monte Carlo event simulation. Protein structure determination. Nuclear physics modeling and simulation. Molecular dynamics.	CPU-intensive. Large number of independent jobs. Large run sequences. Small input data sets; large output data sets.
Production Processing	Processing of physics raw event data. Nanotechnology parameter sweeps. Protein analysis.	Significant amount of database access – locally and remotely. Equivalent (often large) input and output data from remote sources. Reuse of some files by all jobs. Sequence of similar jobs and parameter sweeps.
Complex Workflow	Physics analysis. Text mining. Earthquake simulations. Gravitational wave analysis	Integration of "thick" VO specific higher-level services. Dependencies between tasks and need for good error reporting and response from all layers.
Real Time Response	Testing, validating applications. Operations and monitoring.	Short runs with small amounts of data. Semi-guaranteed response times.
Small-scale Parallelism[5]	Weather forecasting. Chemistry simulations. Molecular dynamics.	Allocation of multiple cores on a single CPU. Need for consistent set of MPI libraries across sites.

Table 3: Application Families on the OSG

6.1. Application (VO) Middleware

Each community or VO develops, integrates and supports its end-to-end (vertically) distributed system. We show two examples, where the details are provided by the reference papers, the ATLAS job submission software [13] and the CMS data management software [14], together with a key to the acronyms used in the diagrams. The main message is to show the complexity in the number of services and the mix of externally and internally provided services needed for these user communities. We would note that the need for multiple community services and integration of a fairly complex system is typical for many different research areas.

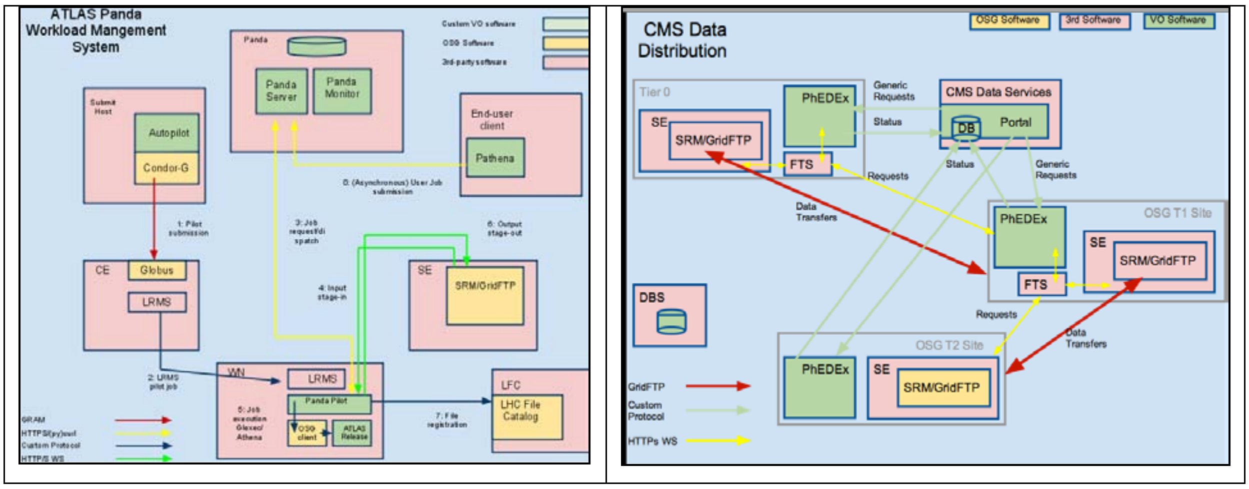


Figure 3: Examples of application service architectures

[4] You may have heard the term "embarrassingly parallel", but we prefer the term "pleasantly parallel". There is nothing to be embarrassed about—we're pleased that there are so many applications that can be parallelized in this way.

[5] Though often these need large-scale parallelism. OSG supports those applications that can be effectively executed with small-scale parallelism.

ATLAS or CMS release	A version of the Atlas or CMS software for data analysis.
Autopilot	Utility to manage pilot job submission to site queues via Condor G.
DBS	CMS data set booking service.
CE	Consists of one or more similar computers, managed by a single scheduler/job queue, which is set up to accept and run jobs submitted remotely. The machines do not need to be identical, but must have the same OS and the same processor architecture. In OSG, the CE runs the bulk of the OSG software stack.
Condor-G	Job client interface to diverse grid-job execution environments.
FTS	File transfer service that manages the network pipes across multiple simultaneous data movement actions.
GLEXEC	Permits a pilot job to re-authenticate using the credential of the payload user, as if the user had directly submitted the job.
GlideinWMS	Technology for implementing pilot based job management system over a diverse set of resources.
GRAM	Service that "provides a single interface for requesting and using remote system resources for the execution of 'jobs'.
GridFTP	An implementation of ftp that uses Grid Proxies for authentication and authorization.
LFC	LHC File Catalog System for authenticated registration of mappings between logical file names and one or more physical file names (i.e. replicas).
LRMS	Local Resource Management System, i.e. whatever batch system is in use at the site (Condor, SGE, PBS, etc).
OSG Client	The OSG CLI client suite. Includes all end-user tools to handle credentials, look up information, and submit jobs to sites.
Panda Monitor	The Panda graphic web application interface by which users and administrators can see the state of the system.
Panda Server	The Panda job scheduling/dispatch server. This is where pilot jobs retrieve real user payload
Pathena	The command like utility by which ATLAS users can submit jobs to Panda.
Phedex	CMS data movement and placement service.
SE	The interface through which components communicate with the remote storage unit.
SRM	Middleware that interfaces to diverse implementations storage resources using a standard interface.
VOMS	Manages real-time user authorization information for a VO.

Table 4: Legend for the example application service architectures

6.2. How the Users Submit Jobs

The distributed computing deployment model introduces several problems for the users of the system. Three major problems are the complexity of job scheduling in the distributed environment, the non-uniformity of compute resources, and no easy way to accommodate end-to-end job monitoring.

When providing job submission services to its customers and constituents, users have a variety of technology choices to solve these problems depending on the types and quantities of the jobs managed. To help OSG better understand the tradeoffs in technology; and therefore, be able to better advise its user communities, OSG currently supports three solutions for job routing and distribution: the OSG Matchmaker, Glide-in WMS, and PanDA. (Additional job submission tools that have been used to a lesser degree on the OSG include Gridway, Pegasus and Swift.)

Each of these services relies on the underlying information architecture that includes resource names, availabilities, capabilities, loads, and access mechanisms. This information is used by the application developers to find resources that can effectively execute their jobs.

Every site and resource that is considered part of the OSG is registered in the “OSG Information Management” database that stores basic (mostly static) information about sites and resources such as names, descriptions, and contact information. Dynamic information, such as availabilities, capabilities, and loads, is provided by localized scripts that run at each site and maintain a local information repository. Data from each site is dynamically aggregated and provided to a central collector, from where the data is pushed out simultaneously to several different places controlled by the OSG itself. Users are able to easily access the resources' published information and use it manually or in their automated job management system.

The OSG MatchMaker is a direct extension of the familiar model of submitting jobs to a local batch scheduler. The MatchMaker discovers the list of currently available sites and user submitted jobs that are then *matched* to a specific site. It is not a single batch system, but individual users can instantiate their own matchmaker. In addition to basic job management capabilities, it builds-in sophisticated tests of remote sites and can run "preparation" jobs that set up sites with needed software packages (increasing the probability of success for user jobs).

Many of the larger communities in OSG use an alternative job submission mode: pilot jobs. Before a user submits a job, a pilot job system (or “factory”) submits a workload-management job that can that pull the end-user job once it is running in a site’s batch system. Pilot based systems hide many of the distributed-resource-related failures from the users because the link to the remote system has already been created and because the pilot job can run various tests to ensure the environment is appropriate for user jobs. Handling failures is shifted from the end-user to the operator of the pilot job system. This job submission mechanism has the capability of scaling to a large the number of simultaneous jobs that can be sustained. OSG users have had up to 200,000 jobs in the pilot job system at any given time, with more than 25,000 end-user jobs running simultaneously.

The glideinWMS [15] and PanDA models are quite different in approach although both are pilot based. GlideinWMS was built directly on top of the Condor infrastructure: the pilot is a Condor process that joins a Condor pool. Once a pilot is running and ready to accept a user job, it appears to the user just like any local Condor batch slot. Being Condor-based also means that glideinWMS can readily accept non-trivial workflows created around Condor DAGs (sets of jobs organized as an directed acyclic graph) [16]. Condor provides an inherent ability for managing and prioritizing jobs that are being submitted. This is the most widely used job submission infrastructure on the OSG because of these capabilities.

The PanDA *(Production and Distributed Analysis)* system was designed using standard web service and database components. All job management is handled by a central server and the client installation is minimal. Since PanDA pilot jobs are not coupled to Condor infrastructure (and, in fact, can be submitted in a variety of ways), there is considerable flexibility in functionality of the pilot. This allows the pilot code to be customized for specific conditions and/or perform probing of sites and worker nodes to ensure that adequate configuration and capability are present. Details of pilot jobs execution are reported to the central server and can be used for troubleshooting if necessary. Upon successful completion of the probing stage, the pilot job gets instructions from the central server on how to obtain its payload, and commences its execution. Through a system-wide job database, the PanDA Web portal (Monitor) provides a comprehensive and coherent view of the system and job execution, from high-level summaries to detailed drill-down job diagnostics.

When used in the context of Atlas software stack, PanDA automates data staging, and provides sophisticated ways of matching jobs to the sites where required data has been already staged, optimizing overall system performance. Its satellite system (called Bamboo) provides a fully automated, database-driven submission of Atlas production jobs. The PanDA system has been thoroughly tested on Atlas production workloads and is also running CHARMM VO production jobs [17].

The table below compares the functionality of the three systems (see https://twiki.grid.iu.edu/bin/view/Documentation/JobSubmissionComparison for full description):

Submission Capabilities and Requirements	**MatchMaker**	**Glide-in WMS**	**PANDA**
Job Distribution Mechanism (User perspective)	Condor-G	Pilot Based	Pilot Based
Is the service centrally hosted?	Each VO must install at least one instance of OSGMM	The Pilot Factory is centrally hosted, VOs must each install their own Job submission infrastructure to submit jobs to the Pilot Factory. VOs can also install their own Pilot Factory if desired	The PanDA server, monitor and their databases are centrally hosted. Pilot submission is a light weight process that is hosted by the VO. VOs must each install their instance of Job submission scripts
Difficulty level for new VOs to install and setup the required job submission infrastructure	Easy -- components are available through the VDT	Moderate/Difficult, new VOs currently require handholding to get started. (There is active work on making it easier.)	Moderate/Difficult, VOs have a minimal number of frontend services to set up and maintain but handholding from developers is still needed.
Difficulty level for new users to get their applications running	Moderate, users must embed their jobs in Condor wrappers. Data movement can be tricky to manage; Condor provides some built-in data handling capabilities.		Easy. Submission is based on a command line infrastructure. Data management can require use of GridFTP or comparable staging mechanism
Is the setup and operation of the system well documented in terms of "thoroughness" and "ease-of-use" for end users (1 to 5, 5 being best)?	Thoroughness (3) Ease-of-use (5)	Thoroughness (4) Ease-of-use (3)	Thoroughness (4) Ease-of-Use (3)
Support for complex workflows	Yes, based on Condor DAGMan	Yes, based on Condor DAGMan	Complex workflows must be implemented manually

Submission Capabilities and Requirements	**MatchMaker**	**Glide-in WMS**	**PANDA**
Support for Job prioritization	Based on Condor. Can only prioritize jobs per submit node↓ not for the whole VO	Full featured, based on Condor infrastructure	Basic prioritization in the brokerage module, gradual throttling of user job priorities depending on submission volume
Error handling of ill configured sites or environments	Regular test/maintenance jobs are sent to make sure sites are up and running. Retries are automatically enabled to minimize end user error messages	User jobs don't start unless a Pilot job is already running at the site, errors are stored in log files for manual inspection	Pilots probe the environment before acquiring a job, and report cases of misconfiguration to the central service
Information systems used to schedule jobs	Resource Selection Service (queried every few minutes)	One time use of registration to install sites; localized infrastructures keep track of dynamic data based on pilot submissions	User jobs are submitted via command line tools, production job submission is automatic, driven by the database and managed by a satellite service
User information systems	Command line tools	Command line tools; limited monitoring capabilities for end users	Portal Interface (Monitor) provides a good interactive view of the entire system for the users and operators
Number of VOs that have used or are using this service in production	3	8	2
Site requirements for job submission (other than a CE and basic correct configuration)	None	Outbound network connections on each worker node if Condor CCB is installed; otherwise requires bi-directional WAN	Outgoing connectivity to WAN on each worker node
Typical number of systems that a VO needs to set up in order to manage 5000 simultaneous jobs	One dual core, 8GB system	1-2 systems, 8GB memory total	One system, very lightweight job submission requirements
Is Root typically required for VOs to set up the submit hosts?	Yes	Yes	No

Table 5: Comparison of the three OSG large-scale job submission systems

6.3. Usability, Reliability and Scalability – Data, Storage and Testing

Many applications running on the OSG sites are data-intensive and require high-throughput. The LHC (Large Hadron Collider) experiments such as ATLAS, CMS and ALICE transfer an unprecedented amount of data. The Tevatron experiments (D0, CDF) as well as non-HEP experiments such as LIGO, SCEC and others also need a significant amount of data transfer. The data scale of transfer varies from petabytes to terabytes depending on demands from Tier-1 and Tier-2centers. Gigabyte scale transfer of data may be created by a single job on a single worker node. The requirements on storage and data management vary widely among the user communities.

The software provided by the OSG can manage the transfer of input and output data files associated with a job, as well as pre-staging of data before launching jobs The sites are responsible for administering and supporting storage available at their sites. The minimal requirement on a site is to provide at least some temporary space for job IO. In order to satisfy this requirement a site usually provides a local storage visible on a gatekeeper and all worker nodes. The persistency of application's input and output data created in this space must exceed the lifetime of the job that created it. A site also guarantees the availability of at least 10 GB of temporary staging space per job on each worker node. The space should be cleaned up after job completion.

An OSG site may also offer access to storage systems managed and interfaced separately from the processing clusters and nodes. On each “storage element” (e.g. physical file system, disk cache or hierarchical mass storage system), data are stored and managed according to the authorization policies provided by the end-to-end security services. To date the majority of accessible storage is configured to be used by the owning community. However, increasingly all applications need some data storage persistent across jobs. To meet this need, sites are asked to provide access to storage for non-owner communities - a chunk of available space (usually up to 5% of the total) with specific, or even uncertain, lifetimes for public usage. A user or user community can use this public space if appropriate authorization settings are in place.

With the ever-expanding number of resources and user communities, the OSG risks reaching limits in its robustness and usability. Software components may become unreliable or stop working altogether when reaching a certain threshold. Tools that were adequate on small scale may become unusable at a large scale. Procedures that worked for a small group of people may become completely infeasible with hundreds or thousands of participants.

To avoid getting into this situation, OSG has dedicated effort to evaluate the current tools and procedures to evaluate their effectiveness at the current scale. Also, effort is being expended to simulate subsystems at the scales expected a few years from now and in evaluating the current tools and procedures in that environment. If any problems are detected in any part of the infrastructure, the owners of that piece are notified of the shortcoming and given all the necessary support to reproduce and fix the issue.

The OSG scalability, reliability and usability team has developed tools to automate most of the software testing and, since OSG does not own any hardware, is relying on hardware given by VOs that are most concerned about the scalability and reliability of a certain subsystem. Unfortunately the evaluation of procedures is much harder to automate, so the team mostly evaluates them manually and seeking feedback from the wide OSG user community as much as possible. The OSG also makes significant efforts to insure the scalability and reliability of the support set of storage software and configurations. The certification and performance tests developed in the OSG, executed on the various storage test stands, assure to some extent the scalability and quality of the available software. The results of the tests also provide feedback to the developers to allow them to improve the software. During the last year we have done a lot of work to simplify the installation, configuration and usability of the storage implementations for small sites – particularly in support of the LHC university, Tier-3 sites.

Many challenges remain. Feedback from our users has made it clear that we need to improve the availability and management of storage for use by communities who do not own the resource. The following requirements must be met: Provide managed access to all VOs supported by a particular resource to reserve, use and release requested data storage; Protect sites from bandwidth and other (inadvertent) user misuse; Simplify and automate the users ability to discover and request data transfer and data replication; Provide a common file, and possibly other metadata, cataloguing service; Enable VOs to manage the available and reserved storage allocated to them across multiple sites; Provide an operations service to manage the public storage. The evaluation of iRODS as a space management solution and common metadata catalog is underway to see if it could satisfy the abovementioned requirements.

7. How Sites Participate

Once the owner of a cluster or storage resource decides that they would like to make it accessible to OSG users and communities they follow a few basic steps to: Register the resource to the OSG operations, including identifying the owning community or organization and obtaining a security certificate for the resource; Follow the instructions off the OSG wiki to install the appropriate software collection offered from the OSG software caches; and participate in an operations meeting and join the and site administrators mail lists to meet the support staff and peer technical groups. Once the software is installed the administrator starts the few core services required to participate in the common infrastructure, if so desired. Those services include the interface to the processing nodes and data areas, as well as the security, accounting, information, and validation services.

The site administrator maintains control of the management, priorities and policies of the resource made accessible to the OSG. Installing the OSG software does not obviate use of the cluster or storage areas by local users, through the local batch system and I/O utilities etc. The OSG services are all designed to support this sharing, to be replicated for high-availability and failover, and to operate on infrastructures separate from OSG – either fully locally managed and used, as part of infrastructures bridged or federated with the OSG, or a combination of these with providing direct access to OSG users.

The situation of course is not quite so simple in practice. There is significant diversity across sites in the network and resource hardware implementation and configurations. With regards to networking, the details of site firewalls, NATs, in-bound or out-bound network access to the final execution CPUs and storage systems impact the usability, errors, performance of the applications using the site. Troubleshooting such problems is difficult, as each installation tends to have specific unique features. With regards to the compute and storage hardware, the specific version and variant of Linux, the disk space and memory available to each execution node and the availability of shared file systems, all have impact on the usability and performance of the site for different communities and applications. Luckily, the team nature of the OSG Consortium helps the scaling in support needs. The “many” email lists and engaged contributors mean that invariably new administrator questions and issues are responded to and resolved in a timely fashion by the extant community of experts. This is a valuable commodity and is regarded as a core strength of the collaboration.

In general, sites deal with the communities of users. The security infrastructure supports defining policies and authorization for access by community, group within the community, as well as the ability, in the last instance, to ban or enable an individual per se. VOs are registered in the OSG information databases and have identified contacts for security, troubleshooting and support. Within a VO, the community self-manages priorities between sub-groups, support VO specific software, services and users, thus providing a well layered structure for their end-to-end distributed environments over the set of distributed resources – some of which they own and some of which are "opportunistically" accessed.

8. Federation and Interoperation

As described previously, the scope of OSG's thinking and activities includes support for applications and services that bridge federated distributed computing infrastructures. OSG currently federates with the EGEE/EGI, TeraGrid, the Worldwide LHC Grid, New York State Grid, the Sao Paolo regional Grid (GridUNESP), and the Clemson University, Grid Laboratory of Wisconsin, Fermilab and University of Nebraska campus-wide infrastructures. The model includes "bridges" between local campus grids and the wide-area Grid where job and data are automatically and transparently uploaded from the local to the OSG environment (mapping of the security tokens, management of the input and output data etc.). OSG supports services to publish information from one infrastructure to the next (resource availability, usage etc.).

Fundamental to the thinking is that the different grids can evolve their technologies and services independently. Thus there is a continuing need for testing of interoperation of job submission and execution and data transport and access between and across OSG and the grids with which we federate. We leverage the tools and processes in our software and service release and testing for interoperability and transparency testing.

9. New Job Management Technologies

New applications, new computer hardware, and new software technologies are bringing opportunities for increasing the effectiveness of the computing ensemble for an increased range of applications. The OSG is currently working in several of these areas to provide benefit for additional communities and improve the usability and throughput accessible to the users.

9.1. High Throughput Parallel Computing

With the advent of 4- and 16-core CPUs packaged in commodity CPU systems, OSG stakeholders have shown an increased interest in computing that combines small scale parallel applications with large scale high throughput capabilities, i.e. ensembles of independent jobs, each using 8 to 64 tightly coupled processes. The OSG "HTPC" program (funded through a separate NSF grant) is evolving the technologies, engaging new users, and supporting the deployment and use of these applications. While still in the early stages, there have been some useful applications running to date and submitting publications of their results. Currently the focus of the program is to: Bring the MPI and other specific libraries from the client to the remote executive site as part of the job – thus removing the dependence on the different libraries invariably found on different sites; Adapt applications to only use the number of cores available on a single CPU; And extend the OSG information services to advertise support for HTPC jobs. For example, chemistry applications have been run across 6 sites (Oklahoma, Clemson, Purdue, Wisconsin, Nebraska and UCSD). The work is being watched closely by the HTC communities who are interested in taking advantage of multi-core while not adding a dependency on MPI. Challenges remain in all the above areas as well as adapting the OSG accounting, troubleshooting and monitoring systems to work well with this new job paradigm.

9.2. Clouds

Cloud computing has emerged as an economic model for computing. The characteristic features of clouds are elasticity, on-demand and a utility principle exhibited via a pay as you go model. Therefore the challenge that many operational grid infrastructures face is to investigate Cloud computing and determine if it is cost-effective to outsource the computer hardware management and if we should adapt the OSG infrastructure and its middleware to support a new cloud model. To that end OSG has encouraged and started grass roots efforts that call on the community to research, test and deploy cloud prototypes and measure the cost and benefits.

Several efforts are currently on-going to evaluate the use of cloud computing with OSG. As an example, the STAR nuclear physics experiment is exploring several implementations or different OSG sites:

- STAR has successfully used *Nimbus* to deploy a fully-fledged grid site on the Amazon EC2 and run batch processing jobs on it. In this experiment, the compute element and the worker nodes were virtualized on EC2 and the standard job submission techniques were used to send jobs to the OSG processing resource running in EC2.

- *Condor*, which is the most widely used batch system on OSG, has recently added a virtual machine universe which offers VMware and KVM support. There the virtual machine is a regular condor job. STAR tested on the Condor VM universe at the University of Wisconsin.
- At *Clemson University*, the OSG processing interface was modified to automatically instantiate VM based on job requests, as well as target the VMs of specific VOs. This technique has the advantage that the user keeps on using his usual workflow while being guaranteed execution in his virtual machine [18].

In addition, OSG sees new testbeds funded by the DOE (Magellan) and NSF (FutureGrid) as partners in the exploration of cloud computing and as a means to inform our future architecture and evolving principles.

10. Conclusion

The OSG provides value and benefit to its collaborations through the successful technical hands-on teamwork (and patience) of the many site administrators and users as well as the efforts of a fantastically dedicated and experienced staff. We have learned that attention to the "open" in OSG fosters this spirit, that the science supported by the OSG provides a meaningful goal to motivate the team, and that the "Grid" is a workable—though challenging—market to be in. We have described just a few aspects of the OSG today. But we don't stand still! Come visit us in person or electronically (via osg-contact@opensciencegrid.org) to find out more.

11. References

[1] Pordes, R. "Challenges facing production grids" in High Performance Computing and Grids in Action. Advances in Parallel Computing, 16th ed. (Amsterdam: IOS Press, 2008), 506-521.

[2] The ATLAS Experiment at the CERN Large Hadron Collider, The ATLAS Collaboration and G Aad et al, Journal of Instrumentation, Volume 3, August 2008, 2008 JINST 3 S0800, doi: 10.1088/1748-0221/3/08/S08003.

[3] The CMS experiment at the CERN LHC The CMS Collaboration and S Chatrchyan et al 2008 JINST 3 S08004 doi: 10.1088/1748-0221/3/08/S08004.

[4] "LIGO and the Detection of Gravitational Waves" Physics Today, October 1999.

[5] Douglas Thain, Todd Tannenbaum, and Miron Livny, "Distributed Computing in Practice: The Condor Experience" Concurrency and Computation: Practice and Experience, Vol. 17, No. 2-4, pages 323-356, February-April, 2005

[6] Globus Toolkit Version 4: Software for Service-Oriented Systems. I. Foster. IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp 2-13, 2006.

[7] Analysis of the current use, benefit, and value of the Open Science Grid, R Pordes (for the Open Science Grid Executive Board), 2010 J. Phys.: Conf. Ser. 219 062024 doi: 10.1088/1742-6596/219/6/062024.

[8] XACML profile and implementation for authorization interoperability between OSG and EGEE, G Garzoglio et al, 2010 J. Phys.: Conf. Ser. 219 062014 doi: 10.1088/1742-6596/219/6/062014.

[9] The Open Science Grid, OSG Executive Board, Journal of Physics: Conference Series 78:012057.

[10] Optimal Response to Attacks on The Open Science Grid, Mine Altunay; Sven Leyffer; Jeffrey T Linderoth ; Zhen Xie Elsevier Editorial System(tm) for Computer Networks COMNET-D-09-3466R1.

[11] A. Roy and the OSG Consortium, "Building and testing a production quality grid software distribution for the Open Science Grid", Journal of Physics: Conference Series, Issue, Volume 180, Number 1, 2009 J. Phys.: Conf. Ser. 180 012052 doi: 10.1088/1742-6596/180/1/012052

[12] Interactive and Real-Time Applications on the EGEE Grid Infrastructure, E. Floros and C. Loomis, Remote Instrumentation and Virtual Laboratories, Springer US DOI 10.1007/978-1-4419-5597-5.

[13] Ganga: User-friendly Grid job submission and management tool for LHC and beyond D C Vanderster ,et al 2010 J. Phys.: Conf. Ser. 219 072022 doi: 10.1088/1742-6596/219/7/072022.

[14] PhEDEx Data Service Ricky Egeland , Tony Wildish and Chih-Hao Huang , 2010 J. Phys.: Conf. Ser. 219 062010 doi: 10.1088/1742-6596/219/6/062010.

[15] glideinWMS—a generic pilot-based workload management system, I Sfiligoi et al 2008 J. Phys.: Conf. Ser. 119 062044 doi: 10.1088/1742-6596/119/6/062044

[16] Peter Couvares, Tevik Kosar, Alain Roy, Jeff Weber and Kent Wenger, "Workflow in Condor", in *In Workflows for e-Science*, Editors: I.Taylor, E.Deelman, D.Gannon, M.Shields, Springer Press, January 2007 (ISBN: 1-84628-519-4)

[17] Damjanovi, A., et al. Open Science Grid study of the coupling between conformation and water content in the interior of a protein. Journal of Physical Chemistry, B, August 2008.

[18] M. A. Murphy, L. Abraham, M. Fenn and S. Goasguen “Autonomic Clouds on the Grid” Journal of Grid Computing Volume 8, Number 1 (March 2010), pages 1-18.