

## Enabling Campus Grids with Open Science Grid Technology

This article has been downloaded from IOPscience. Please scroll down to see the full text article.

2011 J. Phys.: Conf. Ser. 331 062025

(<http://iopscience.iop.org/1742-6596/331/6/062025>)

View [the table of contents for this issue](#), or go to the [journal homepage](#) for more

Download details:

IP Address: 131.225.23.169

The article was downloaded on 04/06/2012 at 16:52

Please note that [terms and conditions apply](#).

# Enabling Campus Grids with Open Science Grid Technology

Derek Weitzel<sup>1</sup>, Brian Bockelman<sup>2</sup>, Dan Fraser<sup>3</sup>, Ruth Pordes<sup>4</sup> and David Swanson<sup>5</sup>

<sup>1,2,5</sup> University of Nebraska Holland Computing Center, 118 Schorr Center, Lincoln NE 68588, USA

<sup>3</sup> Argonne National Laboratory, 9700 S. Cass Ave., Bldg 240, Argonne, IL 60439, USA

<sup>4</sup> Fermilab, MS 369, PO Box 500, Batavia IL 60510, USA

E-mail: <sup>1</sup>dweitzel@cse.unl.edu, <sup>2</sup>bbockelm@cse.unl.edu, <sup>3</sup>fraser@anl.gov,

<sup>4</sup>ruth@fnal.gov, <sup>5</sup>dswanson@cse.unl.edu

**Abstract.** The Open Science Grid is a recognized key component of the US national cyber-infrastructure enabling scientific discovery through advanced high throughput computing. The principles and techniques that underlie the Open Science Grid can also be applied to Campus Grids since many of the requirements are the same, even if the implementation technologies differ. We find five requirements for a campus grid: trust relationships, job submission, resource independence, accounting, and data management. The Holland Computing Center's campus grid at the University of Nebraska-Lincoln was designed to fulfill the requirements of a campus grid. A bridging daemon was designed to bring non-Condor clusters into a grid managed by Condor. Condor features which make it possible to bridge Condor sites into a multi-campus grid have been exploited at the Holland Computing Center as well.

## 1. Introduction

A grid is a combination of computer resources from multiple administrative domains utilized to reach a common goal. Grids have been successful in accomplishing both homogeneous interfaces and resource pooling at the national level, although it has taken much longer to accomplish than originally expected. At the US national level, there are two large grid organizations - the Open Science Grid (OSG) [1] and the XSEDE. OSG is working to become a service provider to XSEDE.

The Open Science Grid focuses on High Throughput Computing (HTC), concentrating on tasks that require as much computing power (throughput) as possible over long periods of time [2]. This style is typified by ensembles of independent single processor jobs. HTC can use pooled CPU resources mostly interchangeably and as such is well suited to distributed and grid computing models. The OSG has demonstrated its technologies are successful; in Q4 2010, the OSG averaged over a 400,000 jobs and a million computational hours a day using HTC.

The principles and techniques the OSG follows are just as powerful at the campus level as they are at the national level. This has been recognized independently in several campus grids across the US, with several implementations; these motivations and exemplar implementations are covered in 2. In Section 3, we explore the characteristics we will use as a rubric for examining campus grids.

We have attempted to refine the models used by other campuses to implement a new campus grid at the Holland Computing Center at the University of Nebraska-Lincoln. Section 4 examines this new model, which focuses on using a Condor-based core and integrating non-Condor resources. Finally, Section 5 discusses bridging a campus grid out to the OSG or other campuses.

## 2. Examples of Campus Grid Implementations

Building a campus grid represents a significant commitment for both users and resource providers, which should be evaluated against the benefits. The primary benefits are:

- **Resource sharing:** Resources are typically bought for peak, not average, usage; using the idle time across the entire campus improves the value of the investment.
- **Homogeneous interfaces to multiple resources:** By providing a homogeneous interface across the campus, researchers can quickly utilize new heterogeneous resources.
- **Independence from any single computational resource:** If we do not rely on a specific single cluster, individual cluster downtimes have a smaller impact. This reduces the need for high levels of redundancy and stretches the campus computing budget further.

An obvious requirement for campus grids is having multiple resources on campus. The resources should be interchangeable, albeit not necessarily identical. Complete homogeneity is typically impossible due to individual resource requirements or ownership. Personnel investment must be made by the campuses to engage the user community.

There have been several successful campus grids that have existed alongside or predated the OSG; two are examined here.

Fermigrid [3], made up of resources located at the Fermi National Accelerator Laboratory in Batavia, IL, is the closest example found of a “mini-OSG”. It uses the same CE software, information systems, and storage elements as the OSG. Trust relationships on Fermigrid are based on the Grid Security Infrastructure (GSI), the same authentication method used by OSG. Job submission is managed by Condor-G through a Globus submission layer to the clusters. This same method can be used to submit to OSG, providing one strategy to getting users from campus to the national grid. Some clusters are managed by a central team, while others are done independently. Some of the grid services (authorization and information services, for example) are run centrally. Accounting is done through Gratia, the same software that is used on the OSG. A central cluster file system is available to most clusters, but Globus-based file transfer is also heavily used.

GLOW is an University of Wisconsin grid used at the Madison campus to distribute jobs on their all-Condor grid. Security is based on IP whitelisting of submit hosts. Since all resources are based on Condor, job submission is managed through Condor and distribution of jobs is through Condor flocking. While there is a central team available to assist with management, each resource is free to define its own policies and priorities for local and remote usage. Cluster ownership is distributed, although there’s also a general-purpose cluster available. Software and data is managed by an AFS install and Condor file transfer.

## 3. Considerations for Campus Grid Implementations

We’ve found campus grids can be characterized by how they approach trust relationships, job submission, resource independence, accounting, and data management.

### 3.1. Trust relationships

A successful campus grid must have an acceptable trust model in order to succeed. A trust relationship enables a resource provider to grant campus users controlled access to the resource, and may be established through sociology and/or technology-based security methods.

In the OSG, the trust model used is designed to be homogeneous and to meet the most stringent requirements of all participating sites. The implementation involves using (GSI) with VOMS [4] attributes a PKI extension. While it provides a highly secure, decentralized authorization model proven at the worldwide scale, it is more difficult for end users compared to traditional username/password authentication. Thus, campus grids may be motivated to use alternate trust models.

On-campus resource providers may have a higher degree of trust than at the national level due to sociological reasons. This trust may just be based on locality – it is easier to establish a working relationship with a colleague locally on campus than 1000 miles away. Also, security requirements on some university campuses are simply less stringent than that of federal labs, explaining Wisconsin’s preference for IP security compared to FermiGrid’s GSI.

### *3.2. Job submission*

In order for a HTC-oriented campus grid to function, users need a usable external job submission interface to the cluster. The Globus Toolkit [5] provides the GRAM interface for job submission and batch system abstraction. While GRAM can be used directly, users almost exclusively prefer to interact with it via Condor-G [6], which provides a batch system interface on top of GRAM. Fermigrid relies on Condor-G submission to GRAM for job submission.

An abstraction layer like GRAM introduces a new user experience (even if Condor-G is used), requiring new expertise. An alternate approach is to use batch system software that can interact with multiple instances of itself. By linking resources at the batch system level rather than adding an abstraction layer on top, we improve the user experience - users no longer need to learn additional tools. In the GLOW campus grid, resources are linked through use of a common batch system, Condor, through a mechanism Condor refers to as “flocking”.

In our observations, the closer the “grid user experience” is to the “batch system user experience”, the more likely a user will adopt the campus grid.

### *3.3. Resource Independence*

Compared to a corporate IT environment, one unique aspect of universities is the diversity of management of computing resources. On a campus, several distinct teams may manage distinct clusters due to campus organization or ownership. One characteristic of campus grids is thus the independence of resources - the level of decision-making delegated out to the resource providers.

The simplest campus grids can be formed by requiring all clusters on campus to run the same batch system and linking batch system instances - GLOW’s use of Condor is an example. It may be desirable for a specialized cluster to have a distinct batch system from the rest of the campus; resource independence allows the cluster owners to best optimize their resource to suit their needs.

Resource independence comes at a cost to the end-user. Extremely heterogeneous resources can be difficult to integrate at the software level - i.e., a binary compiled for Linux will not be compatible with Windows.

### *3.4. Accounting*

Accounting is critical for the long term health as it provides a quantitative measurement of the grid’s value. Accounting systems do not need to be technically advanced. Most batch systems provide a local accounting system.

Any site-local accounting systems – homegrown, vendor provided, or designed for the national grids – can work at the campus grid level as long as they can answer the following questions for a given time period: How much computing resource was consumed overall? Per user? Per group? How much computing resource did a specific user/group consume on resources they did not own? How much computing resource did a specific cluster provide? To who?

### 3.5. Data Management

Scientific data management presents two challenges for research computing centers: volume of data and archival requirements. The data volume is often larger than a single scientist can keep on his personal systems, and archiving requires expertise outside his field.

Distributed computing can present an additional challenge: managing data location. Data access costs may be variable between different resources on a grid, or required data may simply be unavailable at some locations. A simple solution is to export the same file system to all resources, hiding data locality from the user. Unfortunately, this solution breaks down outside the campus and may break down in highly-distributed campuses.

## 4. The Holland Computing Center Campus Grid

In designing a campus grid for the Holland Computing Center, we attempted to meet the three goals laid out in Section 2: resource sharing, homogeneous submit interfaces, and resource independence.

Like GLOW, we have based the campus grid upon Condor. We chose Condor because of its extensive use in existing grids and it is the user-interface that our stakeholders demand. Further, it can be run as a layer on top of almost any middleware, allowing users to access proprietary or standards-based resources without modifying their workflows. Each resource can be accessed via a Condor-based interface giving an identical experience regardless of what the user considers his or her “local” cluster. Two of the local clusters run Condor as the primary batch system and one is PBS - chosen because of the owner’s requirement for scheduling of large-scale MPI jobs. So, GLOW’s Condor-only approach did not fit our case. For integrating our PBS cluster, we developed the Campus Grid Factory (CGF) to provide a Condor interface for the non-Condor clusters; this is covered in Section 4.1.

To enable decentralized operation, we utilized Condor flocking [7] between clusters. The jobs will continue to be managed by the original user’s submit host, but the execute hosts can be outside what is managed by the local Condor pool. Furthermore, flocking can handle communication errors between remote hosts and can recover, providing disconnected operation when resources are unreachable.

Through Condor flocking and the CGF, we have successfully encompassed all local resources. To provide even more value to HCC, jobs can also bridge to the OSG and other campus grids. The interface to the OSG uses the GlideinWMS [8, 9] frontend software, while we link to other campuses using Condor flocking.

All clusters on the campus grid are managed by the Holland Computing Center, therefore the trust relationship between the hosts are implicitly strong. Since all resources are run by the same team, we have the ability to provide distinct user priorities per resource through Condor. Further, because Condor runs *inside* PBS rather than alongside it, PBS can schedule its jobs without interrupting Condor ones.

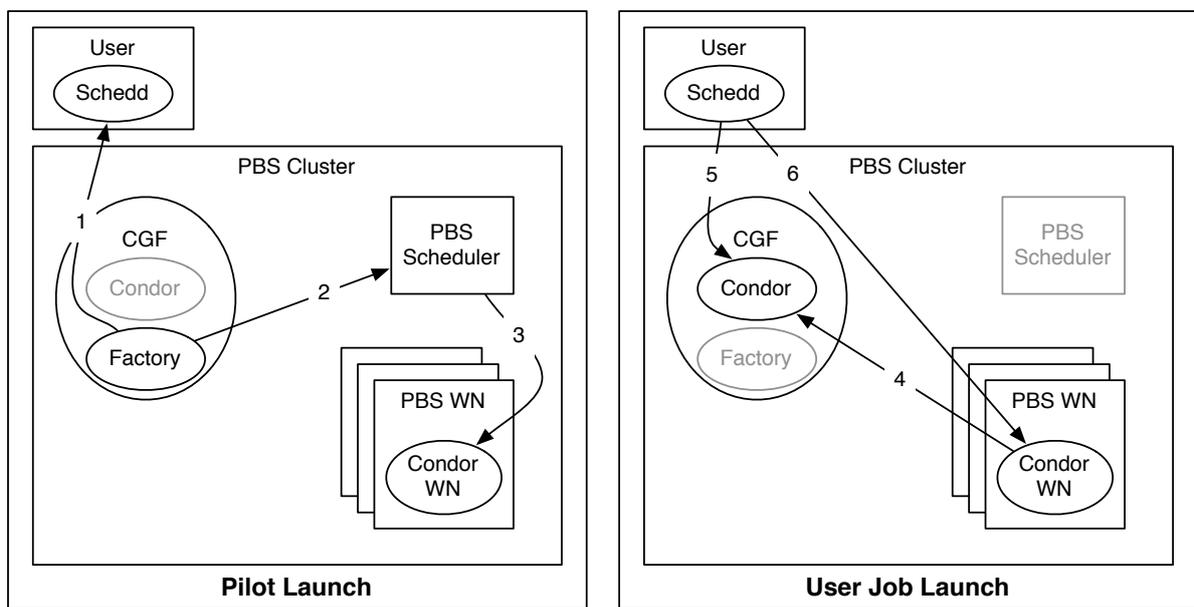
Each submission host runs the Gratia [10] accounting software to provide user accounting. Gratia was chosen because it has a clean separation between remote clusters and the central database (updates are done via HTTP), ability to integrate new resource types easily, and its ability to integrate into the larger OSG accounting. For integration in the OSG, we have extended the software to record both the submission host and the remote OSG cluster utilized.

HCC does not have a shared filesystem across all clusters, so campus grid data management is handled by Condor file transfer. Data management beyond Condor file transfer is left as future work.

### 4.1. The Campus Grid Factory

To solve the issue of providing a Condor interface on a non-Condor batch system, the Campus Grid Factory (CGF) was created. The campus grid factory is a set of software designed to be

operated on a single node with both Condor and the non-Condor batch system (PBS in this case) running. The CGF uses pilot/glidein techniques similar to those used in GlideinWMS or PanDA [11] to submit jobs for one batch system (PBS) that starts up client software for a second batch system (the Condor startd). The Condor startd will then participate in the Condor pool located on the CGF node. This system may also be referred to as a cluster “overlay” system, as it starts a Condor cluster within the PBS cluster. Instead of writing our own PBS submit mechanisms, the CGF utilizes the fact that Condor-G can submit to PBS using the BLAHP from INFN. This has two advantages: the CGF only needs to interact with Condor, and we can reuse the same pilot jobs for other non-Condor batch systems. The CGF is extensible, and would be able to layer over other middleware, such as a OGSA-BES compliant platform.



**Figure 1.** Overview of the Campus Grid Factory

The Campus Grid Factory overview is shown in Figure 1. The factory software starts by querying all the Condor schedd’s on the campus to determine if they have jobs to run (1). If idle jobs are found, the factory will submit (2) a pilot job for execution to the PBS scheduler. When PBS resources are available, PBS will start the pilot job (3) on an execute host, which is a Condor worker node. After starting, the Condor worker node will contact (4) the Condor installation at the CGF and list itself as a node available to run jobs. This is the “pilot launch” sequence.

To launch user jobs, the user schedd will first advertise (5) it has idle jobs to run on the CGF’s Condor collector. The CGF Condor negotiator matches the resources and orchestrates a direct connection (6) between the execute and submit hosts, running the user job.

### 5. Bridging outside the campus

There are two methods for expanding the campus grid described in Section 4: through GlideinWMS to the OSG, or by linking campus grids through flocking.

Unlike the OSG, where the trust relationship is well-defined and common between sites, trust is established between campuses with flocking on a case-by-case basis. The current model for

trust is based on limited trusted hosts. Each site publishes a list of submit and negotiator hosts that are trusted to submit and accept jobs, respectively.

Access to the full campus grid architecture, with bridging, the user first submits jobs to the local condor cluster. If the local cluster can fulfill the user's needs, then the jobs will remain there. If the local cluster is full or cannot meet the user's demand, Condor flocking will start submitting jobs to other campus clusters, either Condor or CGF-based. If the on-campus resources are unable to meet the user's request, the local Condor schedd will expand its reach again by looking outside the campus. The jobs can also be sent to the OSG via flocking to a GlideinWMS frontend, which creates an overlay pool of grid resources. In this architecture, every effort is given to find resources for the user (local, across campus, or externally), while maintaining the same vanilla Condor interface.

It is important to note that the ever-widening circle of resources expands from the local resource the user knows best (and has the best support for) to the most foreign one. Since all end-user interfaces are Condor vanilla universe, the user never encounters errors translated between systems (a common user frustration) and the user needs to develop expertise in Condor alone.

## 6. Conclusions

The HTC philosophy can be a significant asset for campuses. It improves utilization and gives user jobs better mobility. The Open Science Grid has validated at a nationwide level and campuses including FermiGrid, GLOW, and Purdue have shown different methodologies for scaling grid-based HTC to the campus level. These campus grid solutions can be categorized by how they implement the characteristics discussed in Section 3. HCC has been able to build on these previous ideas for a campus grid and improve on them using the Campus Grid Factory for non-Condor clusters. We have shown these approaches can successfully bridge out to external grids. We acknowledge HTC is a specialization of research computing, but we believe this solution decreases the overall cost of a Condor-based workflow via resource scavenging, thus broadening the appeal.

HCC and OSG plan to continue to build on the CGF as a platform for integrating non-Condor clusters. Over the next year, we plan on deploying it on other sites to establish new grids. As noted, we hope to deploy improved data management strategies. We also hope to more efficiently launch pilot jobs; in Figure 1, the factory currently queries the user schedd to determine if there are waiting jobs, and launch pilots if jobs are waiting. We believe this model will lead to unused pilots in some cases, and can be improved.

## 7. Acknowledgments

This research was done using resources provided by the Open Science Grid, which is supported by the National Science Foundation and the U.S. Department of Energy's Office of Science.

## References

- [1] Pordes R, Petravick D, Kramer B, Olson D, Livny M, Roy A, Avery P, Blackburn K, Wenaus T *et al.* 2007 The Open Science Grid *Journal of Physics: Conference Series* vol 78 (IOP Publishing) p 012057
- [2] Livny M and Raman R 1998 High-throughput resource management *The Grid: Blueprint for a New Computing Infrastructure* ed Foster I and Kesselman C (Morgan Kaufmann)
- [3] Chadwick K, Berman E, Canal P, Hesselroth T, Garzoglio G, Levshina T, Sergeev V, Sfligoi I, Sharma N, Timm S *et al.* 2008 FermiGridexperience and future plans *Journal of Physics: Conference Series* vol 119 (IOP Publishing) p 052010
- [4] Alfieri R, Cecchini R, Ciaschini V, DellAgnello L, Frohner A, Gianoli A, Lorentey K and Spataro F 2004 Voms, an authorization system for virtual organizations *Grid Computing* (Springer) pp 33–40
- [5] Foster I and Kesselman C 1997 *International Journal of High Performance Computing Applications* **11** 115 ISSN 1094-3420

- [6] Frey J, Tannenbaum T, Livny M, Foster I and Tuecke S 2002 Condor-G: A computation management agent for multi-institutional grids vol 5 (Springer) pp 237–246 ISSN 1386-7857
- [7] Epema D, Livny M, van Dantzig R, Evers X and Pruyne J 1996 *Future Generation Computer Systems* **12** 53–65 ISSN 0167-739X
- [8] Sfiligoi I 2008 glideinWMS – a generic pilot-based workload management system *Journal of Physics: Conference Series* vol 119 (IOP Publishing) p 062044
- [9] Sfiligoi I 2008 Making science in the Grid world: using glideins to maximize scientific output *Nuclear Science Symposium Conference Record, 2007. NSS'07. IEEE* vol 2 (IEEE) pp 1107–1109 ISSN 1082-3654
- [10] Canal P, Borra S and Malani M 2006 Gratia, an resource accounting system for osg
- [11] Maeno T 2008 PanDA: distributed production and distributed analysis system for ATLAS *Journal of Physics: Conference Series* vol 119 (IOP Publishing) p 062036