GES 2007 German e-Science

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



# On the Applicability of OGSA-BES to D-Grid Community Scheduling Systems

S. Freitag, C. Grimme, A. Papaspyrou, and L. Schley

Robotics Research Institute - Section Information Technology, University Dortmund,
44227 Dortmund, Germany
*email:* {stefan.freitag, christian.grimme, alexander.papaspyrou,
lars.schley}@udo.edu

**Abstract**

In this paper, we exemplary review the requirements of two Grid communities in the D-Grid project and identify similarities in the addressed scientific applications respectively. To facilitate Grid scheduler interoperability on the underlying heterogeneous middleware systems we extend the standardized OGSA-BES interface and propose a basic concept for the exploitation of collaboration potential in the D-Grid community in general. Compared with existing meta-scheduling architectures there will be no need for a central scheduler instance.

## 1 Introduction

Due to vastly divergent applications, the requirements to Grid meta-scheduling systems in different scientific communities are highly heterogeneous and therefore result in very specific realizations for each participant. Nevertheless, it is desirable to enable collaboration on Grid middleware level not only within each implementation, but also across community borders.

To this end, we review the feasibility of an interconnection between two D-Grid[1] community projects, namely the Collaborative Climate Community Data and Processing Grid (C3Grid) and the High Energy Physics Community Grid (HEP-CG). These communities use different and mutually incompatible Grid middleware solutions, in terms of scheduling services as well as data management. To overcome this deficiency, we propose the implementation of Open Grid Services Architecture Basic Execution Standard (OGSA-BES) [6], enabling the exchange of jobs between the two middlewares without requiring structural modifications on the already available schedulers.

Moreover, this approach can be generalized to other D-Grid communities, resulting in a comprehensive D-Grid ecosystem.

The remainder of this paper is organized as follows: in Section 2, we give an overview on the community requirements and identify synergy potentials and limitations. Section 3 reviews the applicability of OGSA-BES to the investigated communities. Finally, we conclude this work with an outlook to future developments in Section 4.

[1] http://www.d-grid.de

## 2 D-Grid Community Application Requirements

The German D-Grid initiative aims to provide a nation-wide eScience infrastructure with the main goal to design, build, and operate a network of distributed resources and services which allow the processing of large amounts of scientific data. Following, we review the diverging requirements for two of the six community projects, namely high-energy physics and earth sciences.

### 2.1 Requirements in HEP-CG Applications

High energy physics investigates among others the quark-gluon plasma physics and tries to discover how the charge and parity symmetry violation influences the imbalance of matter and antimatter during the universe birth [14]. To this end, the Large Hadron Collider (LHC) will start in 2007 to provide logging data from four large experiments, which will result in vast amounts of data available to thousands of scientists around the world.

#### 2.1.1 User Requirements

Today, the data which is produced by the emerging particle physics experiments (Alice, CMS) at CERN is distributed and replicated to a hierarchical storage structure [7] manually to make it accessible for analysis. The HEP Community Grid focuses on the orchestration of data management, data distribution, and computational analysis to improve the data analysis of experiments significantly. In this context, the following requirements, which concern three types of data, are to be fulfilled:

(a) Simulation data has to be generated, reliably stored, and indexed. Although simulations need only small input data sets for generating output, the procedure of data generation is computationally highly demanding. Due to this large effort, the produced data is to be stored safely to prevent damage or loss. Additionally, the distributed simulation data have to be indexed to ensure correct versioning and worldwide access.

(b) Experimental output has to be reprocessed or reconstructed. However, due to the vast amount of data it can typically not be done at its origin. To this end, in a decentralized approach, the data is forwarded to participating computing sites where the processing is done.

(c) Users want to analyze already acquired experimental and simulation data on their local workspace. Thus, necessary data has to be transported to the location at which the analysis takes place without having the user to know about where the data is located, how many replicas of a given data set exist, and what storage management system is used. Altogether, the data has to be staged prior to analysis and the results have to be published to the user afterwards.

Recapitulating, the aforementioned three systems, computing elements, storage elements, and indexing system have to collaborate to ensure the co-allocation of resources for data processing.

#### 2.1.2 System Requirements

Due to the strong commitment to the LHC Computing Grid the HEP community is bound to the use of the EGEE/gLite [10] middleware components. This in particular includes the use of the gLite Workload Management System (WMS) as a resource brokerage system and the application of dCache [5] as a storage management system. Although new components for co-allocation of computational jobs and data have to be developed, integration to the current system is mandatory to ensure the continuous usage of existing middleware mechanisms.

### 2.2 Requirements in C3Grid Applications

Earth system science investigates dynamic climate processes, their chemical formation as well as human-induced changes to climate. In this context, large amounts of highly structured data are acquired, selected, preprocessed, transported, and analyzed by highly demanding applications. As currently no coherent working environment for solving such problems is available to the scientific community, researchers stepwise apply specialized routines to input data acquired and preprocessed manually before.

#### 2.2.1 User Requirements

The C3Grid project is a cooperation of climate research and computer science researchers that aims to provide a Grid technology solution to overcome the aforementioned deficiencies. To this end, the following main requirements – elicited from typical applications – are to be fulfilled:

(a) Standardized access to heterogeneous and distributed data archives, which includes selection and preprocessing to minimize data transfer. According to the user's request, data has to be fetched from a primary data provider or a corresponding replica. This may include for example staging from tertiary storage, preprocessing concerning temporal and spatial constraints. While it is possible that staging and preprocessing are performed by the same provider, these two substeps can also be separated and executed on different sites.

(b) Automatic co-allocation of compute and data resources to ensure data availability during processing time. Depending on the availability of analysis services, prepared data has to be transferred to an appropriate execution site which is selected by the scheduling system. The transferred input data has to be analyzed, typically utilizing an HPC system for processing and calculation. Finally, the results from data analysis have to be made accessible to the user. This again may require data transportation to a user specified location.

(c) Handling of interdependent tasks (workflows) representing complex modular applications, such as humidity flow [8] and stormtrack analysis [3]. These consist of several distinguished steps including preprocessing with respect to temporal and spatial constraints, data conversion, and processing, which

are to be conducted on different sites.

The whole process is orchestrated automatically obeying user-specified interdependencies and minimizing time and effort for application execution.

#### 2.2.2 System Requirements

Arising from the specialized requirements within the C3Grid community, only basic Grid middleware components are used (e.g. the Globus Toolkit, version 4 [4]). Based on this, sophisticated Web Services for workflow scheduling and data management are developed with the goal to integrate them into a single Service Oriented Architecture [11]. The high openness of such systems allows the support of standardized Grid interfaces in a seamless way.

### 2.3 Identification of Differences and Similarities

Obviously, the two projects have significant differences regarding the application context, but also on the technology level. While diversity in the application domain is natural to the different communities on a scientific level and thus cannot be addressed anyway, it is possible to reduce incompatibilities between the middleware systems by defining abstractions which can be covered by both partners.

The current focus in the meta scheduler domain on centralized architectures, i.e. GridWay [9], however, is not suitable for the interaction of highly heterogeneous communities like C3Grid and HEP-CG. Such an approach would require the union of requirements and features from both communities in a single system, producing large overhead in terms of unused or undesired services for each individual partner.

However, it is possible to identify similarities for several use cases regarding subsystems of both communities, enabling the collaborative usage of resources. To achieve this, interfaces for the exposure of common services are required in order to utilize shared features within the Grid.

In the following, we propose a set of basic services common to both community Grids on an abstract level which holds for each community and can be provided independently by the respective partners. These include:

**Data Staging** Selected data has to be fetched from a primary data provider. While HEP-CG utilizes Hierarchical Storage Management (HSM) systems for accessing BLOb[2]-like data, the C3Grid users rely on database or flat file storage with highly structured and randomly accessible data sets. However, both communities consider access to large amounts of input data as well as the storage of computation results as an important part of daily work. As such, community-specific services for data access can be shared by both communities in order to cover a larger range of storage services. Also, since access times for data can be lengthy in both communities, data access can

[2]Binary Large Object.

be considered a schedulable entity managed by the communities' resource management systems.

**Data Analysis** In both communities data is analyzed, typically utilizing a HPC system for processing and calculation. Despite the diversity of scientific applications in both communities there exist common tools which can be applied to all scientific domains such as mathematical analysis and visualization. Although those tools are commonly present on all systems, an information model to ensure exchangeability of corresponding jobs has to be defined such that community schedulers can locate non-community resources which meet the demands of the application. In this way, resources of each community can be offered to the other one for the processing of subsets of jobs.

To bridge the heterogeneities between the two systems, we propose the implementation and extension of OGSA-BES.

## 3 Coupling D-Grid Scheduling Systems using OGSA-BES

The Open Grid Services Architecture Basic Execution Service (OGSA-BES) defines a standardized Web Service interface facading resource managers (RM) for computational entities such that abstract activities can be monitored and controlled in an uniform way without prior knowledge on the concrete RM implementation.

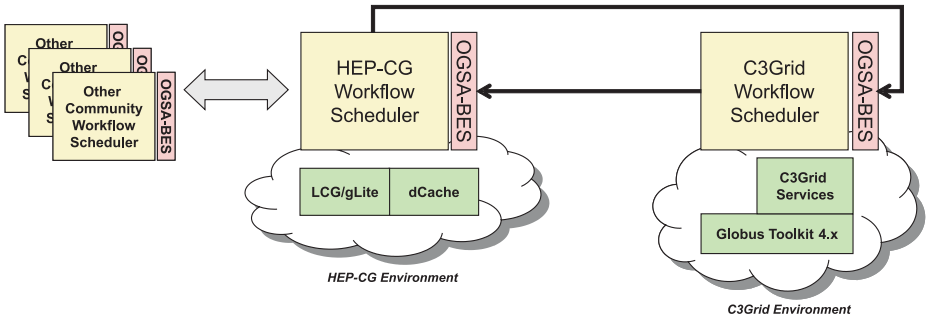


Figure 1: Schematic overview of the collaboration of D-Grid scheduling systems via OGSA-BES Web Services. Every scheduler exposes this interface and acts as a client to others at the same time.

To be able to cope with different RM implementations, OGSA-BES provides a minimal model on activity states, information and resources, which has to be supported by every compatible system.

For the integration of the two afore described communities it is necessary to extend this basic model such that the defined use-cases similar in C3Grid and

HEP-CG are reflected in the state and information model of the BES standard. To this end, we define two extension sets for data staging and data analysis.

### 3.1 Data Staging Extension

To expose its data staging services' capabilities, a community scheduler must publish its individual set of supported storage systems which it is able to manage due to its implementation.

| Name | Context | Multiplicity | Type |
|---|---|---|---|
| SupportedDataSource | BES | 0..* | String |

Table 1: BES-Factory attribute extension for the definition of supported storage backends. The value of this attribute must be the URL prefix of a protocol handler for addressing supported storage systems (e.g. `gsiftp` for GSI-based FTP [1] access, `srm` for Storage Resource Manager [12] access, or `dcap` for dCache SE [5] access).

To this end, the information model of BES is extended by an additional factory attribute, namely `SupportedDataSource` as shown in Table 1. There, a community scheduler can store a set of URL prefixes which denote storage system access protocol handlers it can provide access to. Then, a BES client can query this data and match it with its own data request and if successful, delegate the desired storage access to the queried BES provider. The context of the new parameter is within the `BES` scope of the factory attributes, since it provides information specific to the capabilities of the BES service itself.

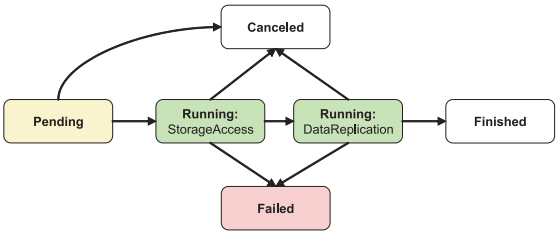


Figure 2: Extension of the OGSA-BES basic state model for the identified similarities in C3Grid and HEP-CG concerning the staging process.

In order to monitor a submitted data staging activity, the community scheduler exposes an extension of the basic BES state model, which differentiates between data access and data replication. The data access part could e.g. denote the process of staging data from tertiary storage such as HSMs and databases

while the data replication part may describe the process of data transport to the users desired target destination.

### 3.2 Data Analysis Extension

In order to execute analysis tasks using standard tools on a non-community system, a community scheduler needs to determine whether all needed software packages and libraries for the requested application run are available on the remote computing element.

| Name | Context | Multiplicity | Type |
|---|---|---|---|
| CPEInformationService | BR | 0..1 | EPR |

Table 2: BES-Factory attribute extension for the definition of a pointer (i.e. the Endpoint Reference to a Web Service) to a service providing information on the available software for a certain system.

Herefore, we extend the information model of BES by another attribute in the factory context to enable the BES provider to publish software packages available on its managed systems. However, the provision of such an information service is beyond the scope of BES and therefore is not handled here, but delegated to external services. To this end, the BES provider only stores an endpoint reference pointing to the Web Service interface of such an information service to enable the BES client to query package information regarding the resources managed by the BES provider.

Regarding the information service, several approaches are possible; a concrete recommendation can however not be made because of the heterogeneity of such systems. To ensure basic compatibility, we propose the use of the BES `<library>` extension to JSDL [2] which allows a minimum definition of package name, version, and description.

The context of the new attribute is within the `BR` scope, as its context depends on the availability of such a service with respect to the managed resource.

To run data analysis jobs on an execution host, we define an additional extension of the BES basic state model which is to be exposed by the community scheduler and can be used for monitoring purposes by BES clients. Here, an extension has been made to the `pending` state in order to distinguish between activities which have been queued (i.e. no decision on the start time of the activity has been made yet) and activities which have been scheduled (i.e. the start time of the activity has been decided on).

Furthermore, additional states have been introduced to denote the process of setting up and tearing down the execution environment for a job. Within these states, software packages may be installed, configured, and removed with respect to the job's JSDL specification mentioned above. The provision of the packages however is beyond the scope of BES and must be handled externally, for example by a CDDLM [13] implementation.

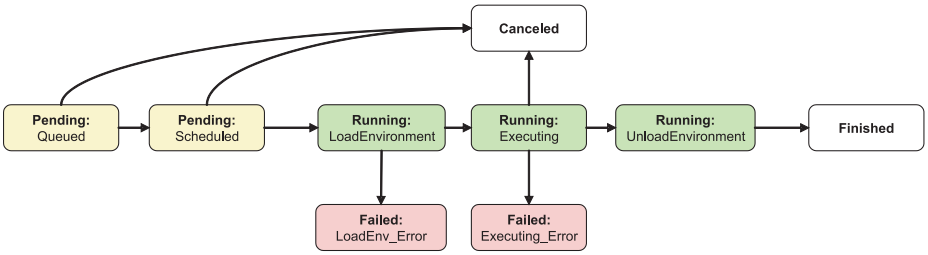


Figure 3: Extension of the OGSA-BES basic state model for the identified similarities in C3Grid and HEP-CG concerning the execution of processes.

Summarizing, the incorporation of the OGSA-BES standard allows the exchange of data and execution computation related tasks between C3Grid and HEP-CG community schedulers with the option to integrate other communities easily as shown in Figure 1.

## 4 Conclusion and Future Work

In this paper, we examined the differences and synergy potentials of two exemplary D-Grid communities. Due to mutually incompatible middleware solutions, we identified similarities on a very abstract level in the general architectures of C3Grid and HEP-CG. For basic interoperability between the two community schedulers we proposed the implementation of OGSA-BES and specified the necessary extensions to this standard regarding the state and information models. In a next step, we plan to realize prototype implementations for both projects, extending and evaluating OGSA-BES properties accordingly to promote our concept to other D-Grid communities.

## Acknowledgements

We would like to thank all partners in the C3Grid and HEP-CG communities for the intense collaboration and support.This work was founded by the German Ministry for Education and Research (BMBF).

## References

1. W. Allcock. GridFTP: Protocol Extensions to FTP for the Grid. online [http://www.globus.org/alliance/publications], April 2003.
2. A. Anjomshoaa, F. Brisard, M. Drescher, D. Fellows, A. Ly, S. McGough, D. Pulsipher, and A. Savva. Job Submission Description Language (JSDL) Specification, Version 1.0. online [http://forge.gridforum.org/projects/jsdl-wg], November 2005.

3. M. L. Blackmon. A Climatological Spectral Study of the 500 mb Geopotential Height of the Northern Hemisphere. *Journal of Atmospheric Sciences*, 33:1607–1623, August 1976.
4. I. Foster and C. Kesselman. Globus: A Toolkit-Based Grid Architecture. In *The Grid: Blueprint for a Future Computing Infrastructure*, pages 259–278. Morgan Kaufman, San Mateo, 1st edition, 1998.
5. P. Fuhrmann and V. Guelzow. dCache – Storage Systems for the future. In *Proceedings of the European Conference on Parallel Computing (Euro-Par)*, pages 1106–1113, Dresden, August 2006.
6. A. Grimshaw, S. Newhouse, et al. OGSA Basic Execution Service Version 1.0. online [http://forge.gridforum.org/projects/ogsa-bes-wg], December 2006.
7. O. Gutsche and K. Bockjoo. Distributed CMS Analysis on the Open Science Grid. In *Proceedings of the Conference on High Energy Physics (CHEP)*, Mumbai, India, February 2006.
8. J. Hansen, G. Russell, D. Rind, P. Stone, A. Lacis, S. Lebedeff, R. Ruedy, and L. Travis. Efficient three-dimensional global models for climate studies: Models I and II. *Monthly Weather Review*, 111(4):609–662, 1983.
9. E. Huedo, R. S. Montero, and I. M. Llorente. A Framework for Adaptive Execution on Grids. *Journal of Software – Practice and Experience*, 34:631–651, 2004.
10. E. Laure, F. Hemmer, et al. Middleware for the next generation Grid infrastructure. In *Proceedings of the Conference on Computing in High Energy Physics and Nuclear Physics (CHEP)*, pages 826–829. Springer, September 2004.
11. B. Portier. SOA terminology overview, Part 1: Service, architecture, governance, and business terms. online, November 2006.
12. A. Shoshani, A. Sim, and J. Gu. Storage Resource Managers: Middleware Components for Grid Storage. In *Proceedings of the 19th IEEE Symposium on Mass Storage Systems (MSS)*, Adelphi, Maryland, USA, April 2002.
13. J. Tatemura. CDDLM Configuration Description Language Specification, Version 1.0. online [http://forge.gridforum.org/projects/cddlm-wg], August 2006.
14. K. Yagi, T. Hatsuda, and Y. Miake. *Quark-Gluon Plasma*. Cambridge University Press, 2005.