Data Integration With OGSA-DAI

Abhishek Gaurav, Nayden Markatchev, Philip Rizk and Rob Simmonds
Grid Research Centre, University of Calgary.
http://grid.ucalgary.ca

1 Introduction

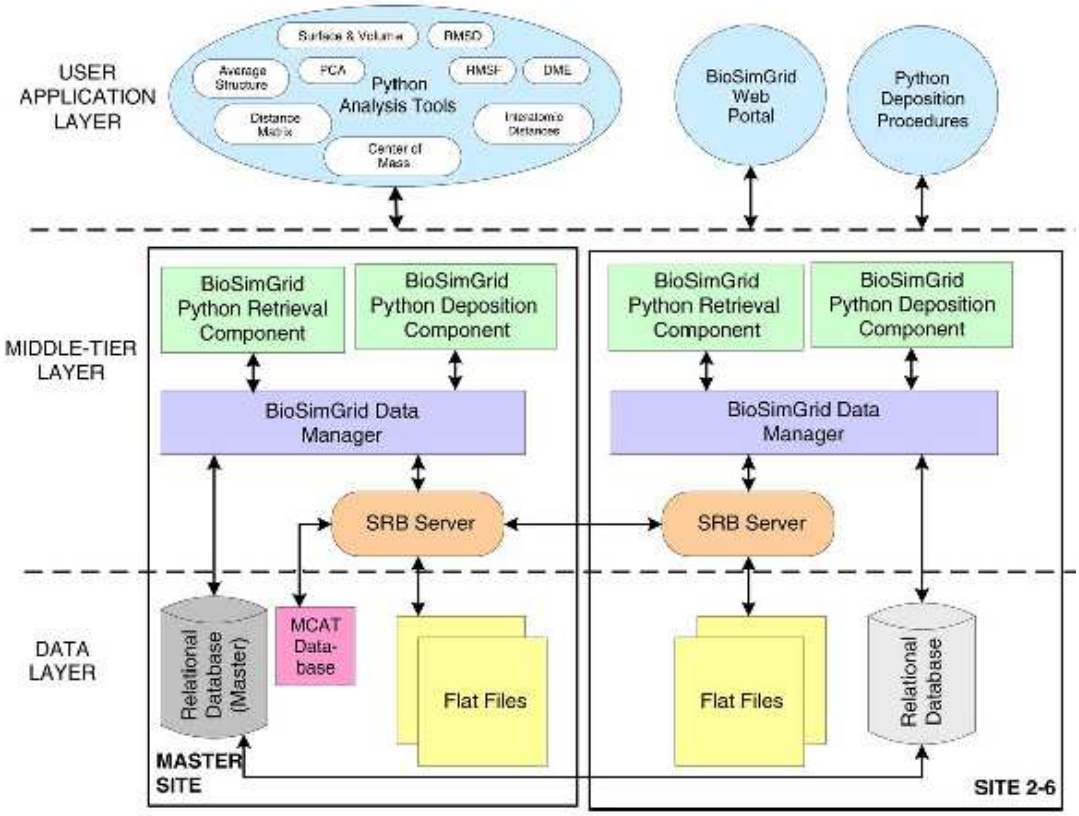
Researchers in the physical sciences continue to generate increasing amounts of data from simulations, sensors, and cataloging efforts such as DNA mappings and climate records. This increase in data has led to the need for need for new data management tools that assist researchers in managing the volume of data entailed as well as distributing the data across disk volumes and administrative domains. Distributing the data facilitates greater collaboration between scientists and maximizes the data's value. These new data management requirements have led to the development of numerous data management systems. Two such systems are the Proactive Data Management System (PDMS) [4] and BioSimGrid [3]. BioSimGrid is a Data Grid project designed to distribute bio-molecular simulation results. PDMS is a data management tool developed by the University of Calgary Grid Research Centre (GRC) that facilitates management and movement of data using metadata rather than physical file locations. The proliferation of different data management systems leads to the need for an extensible framework that facilitates the integration of multiple data sources. OGSA-DAI [2] was designed to meet this goal. This document discusses the integration of services provided by PDMS or BioSimGrid with other data facilities such as databases using OGSA-DAI.

The rest of this document is structured as follows. Sections 2 and 3 provide an overview of BioSimGrid and PDMS respectively. Section 4 discusses the architecture and limitations of the OGSA-DAI framework. The integration of BioSimGrid and PDMS are discussed in sections 5 and 6. Section 8 summarizes the document.

2 BioSimGrid

The aim of the BioSimGrid project is to make the results of bio molecular simulations widely available. BioSimGrid integrates a set of tools to facilitate the distribution of the bio-molecular simulation results across institutions. These tools constitute a toolkit that provides an abstraction layer that frees researchers from needing to understand the underlying storage structures. It provides transparency of data location and replicates data to minimize data retrieval latency.

Figure 1: BioSimGrid Architecture, as illustrated in [3]



The BioSimGrid system has a three layer architecture as shown in Figure 2. The user application layer consists of Python based tools that implement functionality specific to the field bio-molecular simulations. The middle tier consists of Python based middleware that handles the business logic of retrieving and depositing simulation data. The middle tier also includes a Storage Resource Broker (SRB) server [1]. SRB is a data management system that integrates data storage systems and provides replica management. At the data layer, an Oracle 10g distributed database server in a master/slave configuration stores metadata about the simulations. The simulation data itself is stored as flat files managed by SRB. BioSimGrid constitutes an administrative domain. All additions to BioSimGrid go through a "master site" with all other sites serving as caches for redundancy and improved latency. If the master site becomes unavailable, no additions may be performed but the slave systems can still be used to retrieve data.

BioSimGrid was not designed with integration with other systems in mind and would need to be extended to support any kind of interoperability with any data management system.

3 PDMS

PDMS is designed to facilitate the management of data files based on metadata. Users of PDMS register files with the PDMS system along with metadata describing the files. PDMS manages and keeps track of the replicas of the data files. Users can then request that data matching a specific query on metadata be migrated to a specific location. When PDMS receives such a request, it locates the files that match the metadata, picks replicas to use and reliably transports the data to the new location, using data transfer techniques appropriate for high volume data transfer. These techniques may include using GridFTP with appropriate parameters that improve performance or splitting the data connection into several segments [5]. For a more detailed description of PDMS see [4]. Although PDMS was designed with interoperability with other systems in mind, extensions may still be required in many cases. The functionality of PDMS would also have to be extended to inter-operate with raw data from databases.

4 OGSA-DAI

The OGSA-DAI project addresses many issues related to interoperability of data management systems. OGSA-DAI provides WSRF compliant access to different data types and provides query, transformation and movement services. The OGSA-DAI toolkit also includes data resource accessors (providing functionality to access traditional relational databases, XML databases, or file systems). By having a WSRF compliant framework, transparent security functionality and an easy to use library interface, data flows between data sources can be quickly deployed. Utilizing a consistent interface to the data streams allows transformations to take place without the use of intermediate files. The data is instead piped directly from one transformation to another. The OGSA-DAI toolkit uses XML documents (called a perform document) to describe data flows.

OGSA-DAI is highly extensible through the use of resource accessors or activities. A resource accessor represents a data resource and provides "server" functionality for a resource. It provides data resource specific functionality such as native accessors for activities. One example of these accessor services is the JDBC accessor that provides activities with a standard SQL connection to the database. The simpler approach to extending OGSA-DAI is to develop activities. The creation of an activity simply requires an extension of the base activity class. Activities are the building blocks for data flows and operate on data resource accessors. An activity's interface must take the form of accepting any number of input streams and any number of text attributes while its output can have any number of output streams. For example, in order to develop an activity that transforms a data file into another format, the activity could consist of one input stream in the old data format, options for the transformation and one output stream of the new output format.

Other activities can interact with data sources such as pulling and pushing files to GridFTP servers. Once activities are developed for a resource accessor they can be connected in a manner similar to POSIX style piping. For example a simple data flow that

pulls data from a GridFTP server, transforms it, and pushes the data into another GridFTP server might have a perform document that looks like Figure 2. Note that the variables identify the output from one activity and input to another. A perform document can only act on one data resource. Data transportation between multiple data resources is accomplished with more than one perform document. Typically the output of one document is input to the other.

Figure 2: Example perform document from OGSA-DAI

```
<?xml version="1.0" encoding="UTF-8"?>
<perform>
  xmlns="http://ogsadai.org.uk/namespaces/2005/10/types"
  xmlns:pdms="http://telesim.cpsc.ucalgary.ca/wsrf/services/PDMS"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <documentation>
    This example retrieves a file from a GridFTP Sever, transforms it and
    delivers the output to another GridFTP server
  </documentation>

 <deliverFromGFTP name="retrieve">
  <fromGFTP host="dsl101.ucalgary.ca" port="2811" file="/tmp/originaldoc"/>
  <toLocal name="dataSink"/>
 </deliverFromGFTP>

 <ICEValidator name="myTransformer">
    <stringBlocksInput from="dataSink"/>
    <cleanFile name="cleanedFile"/>
 </ICEValidator>

 <deliverToGFTP name="push">
   <fromLocal name="transformedFile"/>
   <toGFTP host="secondhost.ca" port="2811" file="/tmp/transformeddoc"/>
 </deliverToGFTP>

</perform>
```

Through the use of accessors and activities, the OGSI-DAI framework can be extended to integrate just about any data storage system. The OGSA-DAI system does however have significant limitations in its access to simple file storage such as GridFTP or POSIX based file systems.

Currently, file manipulation is not fully matured or supported by OGSA-DAI. For exam-

ple, the file data resource accessor has limited user mapping capability and would need to be extended in many settings. In its current form, the file system accessor can only provide users access to a specific directory, it does not appear to allow different access for different users to sub directories. The GridFTP transfer functionality does not take performance parameters such as buffer sizes or parallelism into account. Pulling and pushing from GridFTP servers are separate, and they go to and from sinks on the OGSA-DAI server. This implies a normal GridFTP third party transfer is not possible without routing the data through OGSA-DAI. This behavior would take a severe performance penalty because the data would have to go through the JVM and not take a direct route. However, it is in fact relatively easy to develop new activities that extend the GridFTP functionality.

Prototyping by the GRC revealed the the OGSA-DAI framework may encounter performance penalties and scalability problems. For example, while using the built in functionally to transfer large files using GridFTP, the JVM crashed due to a lack of memory. This is due to the way GridFTP functionally is implemented and may be correctable.

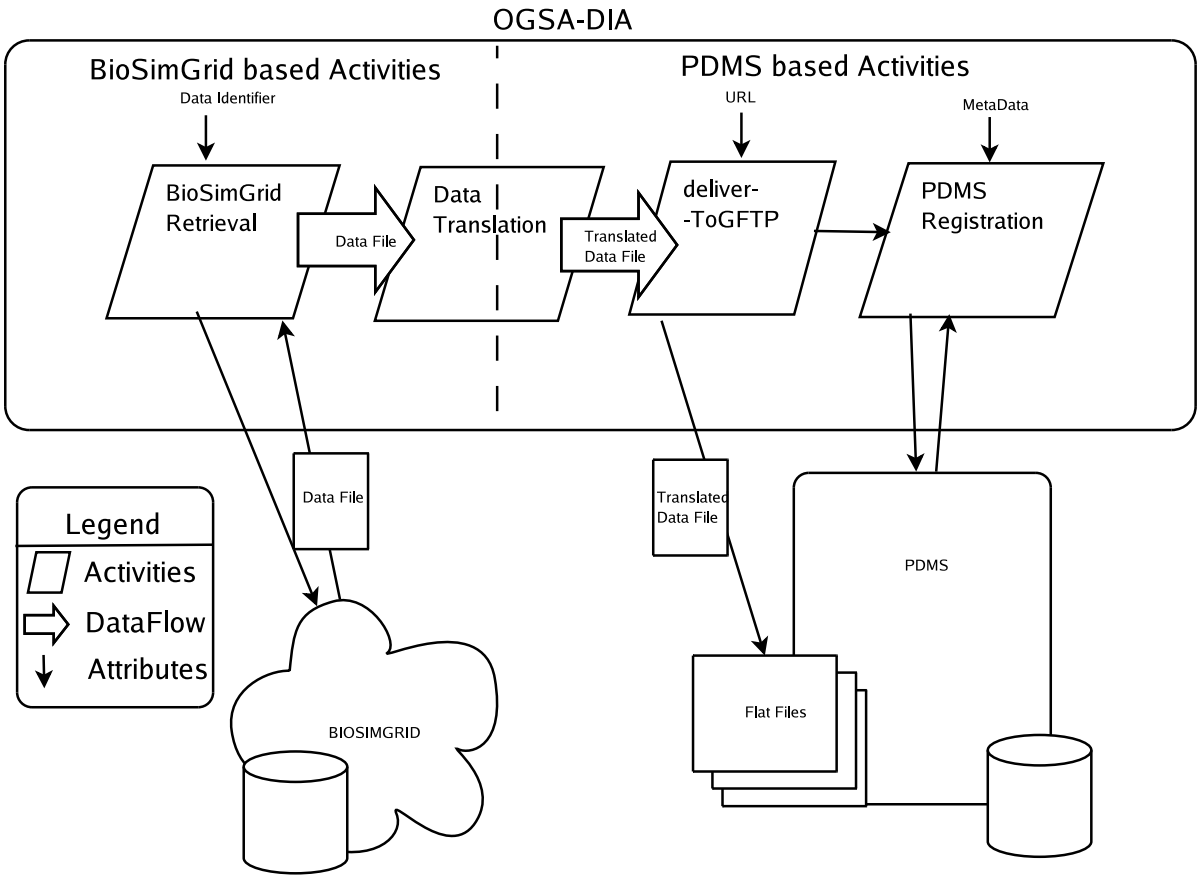
5 BioSimGrid Integration with OGSA-DAI

To integrate BioSimGrid, both data accessors and activities would need to be developed. As mentioned in Section 2, the BioSimGrid system's file storage is based on Storage Resource Broker (SRB) and these files are essentially indexed through a replicated Oracle 10 database server. In order to access the BioSimGrid system an accessor for it would likely have to be developed. If the accessor was developed specifically for the BioSimGrid system it would be able to take queries based on meta-data, and retrieve the data from SRB for a client. Alternatively, a generic SRB accessor could be developed and BioSimGrid specific activities that are aware of BioSimGrid specific implementation details (such as how to query the metadata catalogue for a specific piece of data) could be built around the generic SRB accessors. These activities would, essentially, replicate the Python based middle tier of the BioSimGrid system. Because the system supports multiple data formats, and stores data in a generic form, the data would have to be translated. Activities would need to be developed to translate the generic format into formats that users would want supported. Although BioSimGrid already has mechanisms to translate the document between formats, performance improvements could likely be obtained through the use of activities because the data would be translated in place, rather than having to stage copies specifically for translation.

6 PDMS Integration with OGSA-DAI

A PDMS data resource accessor or activity could be developed to integrate the functionality offered by PDMS. Once a PDMS data accessor or activity is developed, data could be retrieved from a database and pushed into a PDMS tagged file. Conversely, data from a PDMS managed file can be transformed (using XSLT perhaps) and pushed into a database.

Figure 3: Example integration of BioSimGrid and a PDMS system



The simplest way in which PDMS can be integrated into the OGSA-DAI framework is to develop an activity that takes metadata of a file (including its physical location) and registers this information with PDMS. OGSA-DAI already has a functionality that allows data streams to be piped to and from GridFTP servers, the data transfer protocol used by PDMS.

7 BioSimGrid Integration With PDMS

This section describes how OGSA-DAI can be used to realize an integration between an instance of PDMS and the BioSimGrid system. This could occur if a research group was using PDMS to manage its own molecular simulation results but also wanted access to the data available withing the BioSimGrid system. Simply using the BioSimGrid system for all the results may not be an option if the research group considers their own data private. Both the BioSimGrid system and PDMS derive much of their value from having access to the entire relevant data set. This means OGSA-DAI would only realistically be used to import

data from one system to another. OGSA-DAI would not act as a glue that fully integrates the two data sets. Under these conditions, it would be usefully to have a tool that is capable of importing simulation results from BioSimGrid into the group's PDMS system.

There are numerous ways to design this integration. These choices revolve around the level of functionality to put into each activity. The first step, not shown in the diagram is a query for data within the BioSimGrid system. There are mechanisms within OGSA-DAI that could, potentially, assist in finding simulation data that the BioSimGrid system has, but a particular PDMS deployment does not. This is beyond the scope of this paper.

The data import flow shown in Figure 7 consists of 4 activities. The first activity retrieves particular simulation results the BioSimGrid system. It takes the data identifier as an attribute of the activity. This activity would have to query the meta-catalog to discover how the file is indexed in SRB, and then retrieve the file from SRB. Because the BioSimGrid system stores files in an intermediate format, the file would then have to be translated to whatever format the users of the PDMS system desire. This would be accomplished by a data translation activity which takes the raw data file as input and outputs the translated data file as output. The output from the Data Translation activity is taken as input the the deliverToGFTP activity which stores the file at a location accessible via a GridFTP server. The location is specified as a URL as an attribute for the activity. The deliverToGFTP activity is already part of the OGSA-DAI toolkit. The deliverToGFTP activity does not provide an output stream. The thin arrow connecting it to the PDMS registration denotes that it is entered as input to the registration and ensures the activity is down ofter the delivertoGFTP activity is completed. The MetaData associated with the file comes out of band, and is arrived at by the query process and included as an attribute of the registration activity.

The data flow described is intentionally minimized to demonstrate different ways in which data management systems can be integrated. In this example the BioSimGrid activity queries for the location of the file and retrieves the file in one activity. The PDMS side of the flow separates interaction with the metadata system and the interaction with the GridFTP server. The interaction with BioSimGrid could also be separated into a query for the location of the file, and an activity that retrieve the file from SRB. Similarly, the import into PDMS could have been accomplished by developing an activity that takes the metadata as an attribute and the file as an input. This activity would both transfer the data and register the file with PDMS.

The described importation of data could be accomplished without OGSA-DIA. The security mechanisms, common interfaces, the ability to pipeline the data that OGSA-DAI provides makes it significantly easier. In addition, all the activities used here are re-usable. The objects are not specific to this particular use case.

8 Summary

This document discussed the use of OGSA-DAI to integrate data management systems together. The integration of PDMS and BioSimGrid where both discussed. Rapid prototyping

with OGSA-DAI by this group has revealed OGSA-DAI to be very robust and extensible. The main limitations of OGSA-DAI are likely to be is scalability and performance. The framework is entirely in Java, and the piping mechanisms used can be very memory intensive if not done carefully. Even when extensions to OGSA-DAI are written carefully, scalability may still be an issue for groups with large quantities of data to process. Despite these limitations, OGSA-DAI meets the data integration needs of a large number of users.

References

[1] C. Baru, R. Moore, A. Rajasekar, and M. Wan. The SDSC Storage Resource Broker, 1998. In Procs. of CASCON'98, Toronto, Canada, 1998.

[2] K. Karasavvas, M. Antonioletti, M. Atkinson, N. C. Hong, T. Sugden, A. Hume, M. Jackson, A. Krause, and C. Palansuriy. Introduction to ogsa-dai services. *Lecture Notes in Computer Science*, 3458:1–12, May 2005.

[3] M. H. Ng, S. Johnston, B. Wu, S. E. Murdock, K. Tai, H. Fangohr, S. J. Cox, J. W. Essex, M. S. Sansom, and P. Jeffreys. Biosimgrid: Grid-enabled biomolecular simulation data storage and analysis. *Future Generation Computer Systems*, 22:657–664, May 2006.

[4] U. of Calgary Grid Research Centre. Proactive Data Management System. GRC Internal report.

[5] P. Rizk, C. Kiddle, and R. Simmonds. A GridFTP overlay network service. In *7th IEEE/ACM International Conference on Grid Computing*, 2006.