A Federated Architecture for Data Access and Integration Services in e-Science

Aileen Campbell[1], Brian Hills[1], Rob Baxter[1], Alan Gray[1], Denise Ecklund[1], Stephen Rutherford[1], Davy Virdee[1], Tom Sugden[2], Bob Mann[1,3], Richard Sinnott[4].

(1) Edikt, (2) EPCC, (3) Institute for Astronomy, University of Edinburgh, (4) NeSC, University of Glasgow.

Abstract

To date, much of the focus of Grid middleware has been to provide services to access data rather than to access *and* integrate data within a Grid environment. Edikt have designed an architecture which enables federated data access and integration using requirements captured from projects in two different scientific disciplines: *BRIDGES* (bioinformatics) and *EdSkyQuery-G* (astronomy). In this paper we will:

- Summarise the aims of both projects;
- Identify the common data access and integration requirements from both projects;
- Discuss our architecture, including key design decisions and current engineering focus.

1. Introduction

Data access and integration within a Grid environment has been the subject of much attention in recent years. The Global Grid Forum is currently seeking to define standards in this area through the Data Access and Integration Services (DAIS) working group with input from both the academic community and key industry players. To date, the software produced by the Grid community has largely focused on enabling data access. The next logical step is the provision of a federated view of data in order to both access *and* integrate data that may exist in large, distributed and heterogeneous data resources. This will encourage a wider adoption of the Grid by both scientists and the business community.

Edikt, based at the National e-Science Centre in Edinburgh, have designed an architecture to enable federated data access and integration which is currently being applied across two projects in very different scientific disciplines: *BRIDGES* in bioinformatics and *EdSkyQuery-G* in astronomy.

2. Application Requirements

Edikt have spent a great deal of effort engaging with specific application scientists to discuss requirements and design issues. As a result of these discussions, Edikt have produced Use Case, Requirements and Design documents. This section provides a summary of both the BRIDGES and EdSkyQuery-G projects and lists some of their common data access and integration requirements.

2.1 BRIDGES

The BRIDGES project employs Grid technologies to develop and explore database integration over six geographically distributed research sites, as shown in Figure 1. The project resides within the framework of a Wellcome Trust biomedical research project [1] to provide a sophisticated infrastructure for bioinfomaticians. Key issues to be investigated in BRIDGES are the integration, federation and security of data. The infrastructure in BRIDGES is being developed to support data sets from numerous heterogeneous sources, with different structures, evolving nature (schemata) and various levels of security from public to shared through to private. All of this is being made as transparent as possible to the end user scientists. To this end, technologies such as Eldas, OGSA-DAI and IBM's Information Integrator are being employed.

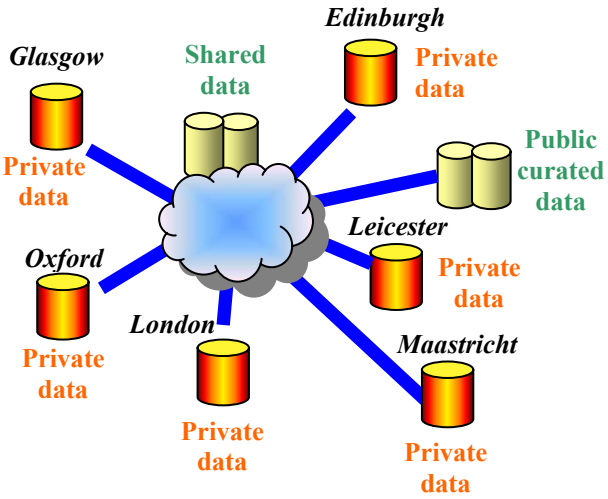

Figure 1: Location of data sources used in the BRIDGES project.

2.2 EdSkyQuery-G

The Virtual Observatory aims to allow astronomers to identify observations of particular celestial objects in multiple, large (often multi terabyte) heterogeneous databases distributed around the world. The EdSkyQuery-G project aims to provide a scalable architecture for doing this through extending the ideas prototyped in the SkyQuery [2] .NET web service developed at Johns Hopkins University. Further aims of the project include integration with existing AstroGrid [3] components and to inform the specification of the International Virtual Observatory Alliance's OpenSkyQuery [4] integration standard.

2.3 Common Requirements

The BRIDGES and EdSkyQuery-G projects address specific problems in different areas of science and, as a result, have different requirements in areas such as raw data formats, security and metadata. These differences can, however, be distilled into basic common requirements for example the support of data mediation tools, flexible security policies and the definition of generic metadata schemas. Working with application scientists Edikt have developed a common set of data access and integration requirements recorded within the project's documentation. These requirements include:

- The need to publish a list of available data and metadata which may be viewed by scientists;
- Joining data between distributed, heterogeneous data resources;
- Application of complex algorithms to data resources;
- Efficient data transport between remote entities;
- Flexible data formats for data transport and presentation;
- Job submission services and scheduling.

3. Federated Architecture

In order to address the scientist's requirements our architecture builds upon ideas developed by SkyQuery [2] and the DAIS working group [5]. We employ a wrapper-mediator model which includes the three core components shown in Figure 2. These three components are:

- **Query Builder**: A graphical client which enables scientists to enter complex queries.
- **Service Manager**: Receives queries from the query builder, processes queries and interacts with data access services.
- **Data Access Services**: Used to perform queries and cross-database joins. The data access and integration services will be provided by Eldas [6] and OGSA-DAI [7], developed at the University of Edinburgh.

Eldas, developed by Edikt, is a Grid-enabled data access solution designed and built within the J2EE framework. Eldas uses industry adopted Web Service standards, and is deployed within a robust, industry-approved container environment. Eldas is free to download[1] and use. OGSA-DAI, developed by EPCC[2], is an open source database access and integration middleware solution designed to meet the needs of the UK e-Science community.

The incorporation of Eldas and OGSA-DAI within the architecture will serve to provide a valuable evaluation of these technologies which will determine if they can provide the interoperability, performance and scalability required to facilitate e-Science using the Grid.

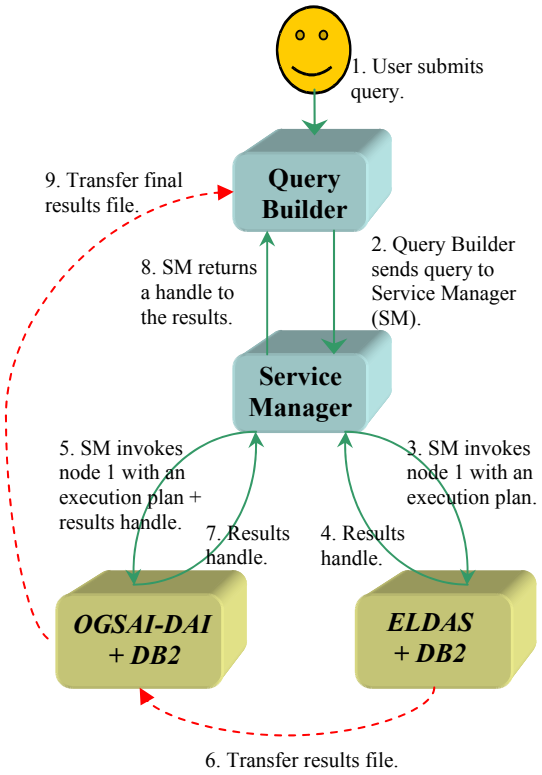

Figure 2: Federated architecture used for the EdSkyQuery-G and BRIDGES projects.

[1] Available from http://www.edikt.org/eldas
[2] Available from http://www.ogsadai.org

3.1 Benefits

Riccardi [5] describes two architectures which may be used to provide data access services for SkyQuery using DAIS. The original SkyQuery employs a recursive approach whereby a node executes its part of the query plan and passes the results and the plan onto the next node, hence the control and data paths are identical. However, we have opted to use an iterative model for control flow as shown in Figure 2. In this model, the Service Manager governs the control path, interacting with each node, whilst data is transferred directly between the servers hosting the databases. The Service Manager may either return the full set of results to the Query Builder or, in the case of a large set of results, a handle to the results file. The handle will enable the Query Builder to retrieve the results when required using an appropriate data transport mechanism. We believe the benefits of this iterative approach include:

- ***Scalability:*** There is no need to return sets of results directly back to the Service Manager and Query Builder. This both minimizes the volume of data being transferred between components and avoids potential Java Virtual Machine (JVM) memory problems.
- ***Robustness:*** The Service Manager can intelligently handle error conditions. For example, if a particular database is unavailable the Service Manager may report a sensible error to the user or even try and use a replica database. Handling error conditions using the recursive model is more complex.
- ***Usability:*** From our initial requirements phase it is clear that scientists wish to perform complex queries which may take many minutes, hours or even days to perform. Therefore it is essential users receive regular status updates on the progress of a particular task. As the Service Manager controls the processing it can send updates back to the Query Builder for example "*invoking SkyNode1 Join service at time Z*". This approach has particular benefits for the BRIDGES project as biologists may wish to view partial/intermediate results and abort or modify the initial query, changing future processing steps.

4. Engineering the Architecture

After the initial requirements capture and design stages we have now moved into a prototyping phase to determine the most suitable data formats and transport mechanisms to meet the scientist's requirements. In order to build the system we are also investigating reusing existing software components. The following sections summarise our work in this area to date.

4.1 Data Formats

As suggested earlier, we recognise that different users may wish to process and view different data formats. We have considered using a specific data format to transport results between databases and applying an adapter component to convert the final results into a user defined "presentation" format. The astronomy community have provided us with 20GB of data (10GB in two DB2 databases) which we intend to export into various different formats to determine raw file sizes and the time overheads associated with compression/decompression or marshalling/de-marshalling binary data. We are currently investigating the following approaches:

1. Using the Sun XML WebRowSet metadata and data description [8];
2. Using the XML WebRowSet metadata description with a compressed CSV file..
3. Using an XML description of the data stored as binary, specifically for data transport.

Our initial investigations suggest the second approach may be the most favourable in terms of striking a balance between the size of a file and the need to pre-process the data before loading into a database or presenting to the user. We have found that the first approach results in an excessive "bloat" in the size of the file and the third method introduces a high CPU and time overhead caused by the need to marshal and demarshal the binary data in order to export from and import to a database.

4.2 Data Export, Transport and Import

We have built specific prototypes to examine the feasibility and performance of exporting, transporting and importing data using the following mechanisms:

1. Middleware services such as OGSA-DAI Grid Data Service (GDS) to GDS delivery;
2. Database stored procedures written in Java;

3. Native database routines;
4. GridFTP and FTP.

Our initial investigation of data export and import suggest that using native database functionality such as DB2's export and import utilities to produce DB2-specific IXF binary files provides the most efficient way to export and import data. However this approach is not suited to transferring data between heterogeneous databases as one cannot import a DB2 IXF file into a different vendor's database, for example Oracle or MySQL. Another option we have investigated is invoking a Java program on the database server itself to perform export, transport and import. This program may be run within the database itself as a stored procedure or as a standalone program out with the database. We have found that exporting and importing data using Java code on the machine hosting the database can be around 12 times slower than using the DB2 export and import utilities to produce an IXF file. Furthermore, the use of stored procedures can have a significant impact on the performance of the database itself, particularly if the database is processing concurrent requests.

We investigated the use of GridFTP in order to transport large volumes of data between database servers. Whilst the functionality offered by GridFTP would meet our needs we have decided to use standard FTP as the GridFTP server currently only runs on Unix servers and both the build and installation procedure are extremely complicated and non-trivial.

4.3 Component Reuse

In order to build a prototype and avoid re-inventing many wheels we intend, where appropriate, to reuse existing software components. As previously discussed we are using Eldas and OGSA-DAI to provide the Grid data and integration services. In addition we may use components from the AstroGrid community such as the Query Builder client, Registry and Job Submission tools. Finally, we are tracking specifications in order to provide compatibility with interfaces defined within world wide standards such as the International Virtual Observatory Alliance (IVOA) SkyNode Interface [4].

5. Conclusions

This paper has suggested that, to date, most of the focus of Grid middleware in the data arena has been to enable data *access* with minimal emphasis on data *integration*. Through working with scientists from different disciplines, Edikt have taken the first concrete steps towards understanding the data integration requirements of e-Science in general, and of designing the architecture to meet them. The incorporation of Eldas and OGSA-DAI will serve to evaluate if these technologies can meet the middleware requirements for large scale E-Science projects. In addition, by reusing other existing components, particularly from the astronomy community, we hope to be able to demonstrate a working prototype during autumn 2004.

References

[1] NeSC, University of Glasgow. *Cardiovascular Functional Genomics* [Internet]. http://www.brc.dcs.gla.ac.uk/projects/wellcome_cfg/ [Accessed 2 June, 2004].

[2] T. Malik et al. *SkyQuery: A Web Service Approach to Federate Databases*. In CIDR 2003, Asilomar, CA, USA, January 5th – 8th, 2003.

[3] A. Lawrence. *AstroGrid: The UK's Virtual Observatory*. In Proc UK e-Science programme All Hands Conference, Nottingham, UK, Sept 2003.

[4] IVOA VOQL Working Group. *IVOA SkyNode Interface, Version 0.7.4* [Internet]. Available from: http://www.ivoa.net/internal/IVOA/IvoaVOQL/SkyNodeInterface-0.7.4.pdf [Accessed 2 June, 2004].

[5] G. Riccardi. *DAIS Data Service Interactions and the SkyQuery Portal* [Internet]. Available from: http://www.nesc.ac.uk/~greg/skyquery.pdf [Accessed 2 June, 2004].

[6] A. Campbell et al. *Eldas (Enterprise Level Data Access Services)*. To appear in Proc UK e-Science programme All Hands Conference, Nottingham, UK, 31st Aug-3rd Sept 2003.

[7] Ali Anjomshoaa et al. *The Design and Implementation of Grid Database Services in OGSA-DAI*. In Proc UK e-Science programme All Hands Conference, Nottingham, UK, Sept 2004.

[8] Sun Microsystems Inc. *Sun WebRowset XML Schema Definition* [Internet]. Available from: http://java.sun.com/xml/ns/jdbc/webrowset.xsd [Accessed 2 June 2004].