

Bioinformatics Data and the Grid: The GeneGrid Data Manager

Noel Kelly
P.V. Jithesh
David R Simpson
Paul Donachy
Terrence J harmer
Ron H Perrott

Belfast e-Science Centre
www.qub.ac.uk/escience

Jim Johnston
Paul Kerr
Mark McCurley

Fusion Antibodies Ltd
www.fusionantibodies.com

Shane McKee

Amtec Medical Limited
www.amtec-medical.com

Abstract

GeneGrid is an industrial e-Science project under development in the Belfast e-Science Centre in association with commercial partners – “Fusion Antibodies Ltd” and “Amtec Medical Ltd” who provide real industrial engagement and commitment to this pioneering project. The goal of the project is to create a virtual bioinformatics laboratory [23], by developing grid middleware to integrate existing technologies, applications and datasets into a grid environment, allowing all relevant parties to access their collective skills, experience and results in a secure, reliable and scalable manner.

Data access, management and integration are key areas of interest for the project, with the “GeneGrid Data Manager Component (GDM)” addressing GeneGrid requirements in these areas. The GDM is a stand alone component within the GeneGrid architecture which consists of a number of cooperating GT3 grid services. Its role in meeting the above requirements is to integrate disparate and heterogeneous distributed data sets into the GeneGrid environment, and to provide an interface for all relevant services to discover and interact with said data. In this regard, GDM has adopted OGSA-DAI as the basis of its design, extending and enhancing functionality as required.

This paper will present the background to GeneGrid, and provide an overview of the project architecture, focusing in particular on the GeneGrid Data Manager. In addition, it will outline the experiences and issues with the deployment of the GDM within the GeneGrid Test-bed framework.

1 Introduction

The commercial partners in the GeneGrid project, “Fusion Antibodies Ltd” and “Amtec Medical Ltd” have a common goal – to advance bioinformatics genomic research and data mining. Presently the individual companies do not have any dedicated in-house bioinformatics expertise or high performance processing capability. The low speed of data transfer between the parties, the lack of HPC processing power as well as the need for security mechanisms across the disparate administrative & organisation boundaries is an impediment to the rapid advancement of this important area of science and research.

The GeneGrid project proposes to exploit Grid [1] technology, existing micro array & sequencing technology and the large volumes of data generated through screening services to develop specialist tissue-specific datasets relevant to the particular disease being studied. The advantage is that all the genes that relate to a disease can be studied *in silico*, enabled by the creation of a Grid based framework that will integrate the generation and analysis of cancer and infectious disease-specific genetic information from various distributed international sources, public domain data sets and other unique data generated by e-Science projects.

In March 2004, the first GeneGrid test bed was prototyped in the Belfast e-Science Centre, consisting of a number of cooperating Grid Services [1] based on the Open Grid Services Architecture (OGSA) model [2], derived from the Open Grid Service Infrastructure specification (OGSI). All services were developed using the Globus Toolkit 3.0.2 (GT3) [3], and were categorised logically into three components,

- **GeneGrid Data Manager (GDM)** responsible for the access and integration of datasets.
- **GeneGrid Applications Manager (GAM)** responsible for the integration of bioinformatics applications.
- **GeneGrid Workflow & Process Manager (GWPM)** as the central component of GeneGrid with responsibility for scheduling and resource allocation.

The second test bed release followed in August 2004.

This paper aims to provide an overview of the GeneGrid Architecture, as well as discuss in more detail the GDM, drawing on experiences and issues encountered from both the first and second GeneGrid test bed releases.

2 GeneGrid Architecture

The GeneGrid architecture consists of a number of cooperating GT3 services, which may be categorised logically into three components.

The central component of GeneGrid is the GeneGrid Workflow & Process Manager (GWPM), which is responsible for the processing of all submitted experiments, or workflows, including the scheduling and allocation of resources to bring said workflows to completion. The GWPM consists of two persistent Grid Service Factories [1] – the “GeneGrid Workflow Manager Service Factory” (GWMSF) and the “GeneGrid Process Manager Service Factory” (GPMSF), as well as all transient instances of these factories.

Each time a user submits an experiment for processing in GeneGrid, an instance of the Workflow Manager Service is created to manage it. This service will then break the experiment into

logical constituent jobs, or *tasks*, and dynamically create instances of the Process Manager Service to handle individual tasks. An illustration of the GWPM is shown in figure 1 (with instances of the Workflow Manager Service labelled $W1$ to Wn , and instances of the Process Manager Service labelled $P1$ to Pn), while a more detailed description of its functionality and features is available from [4].

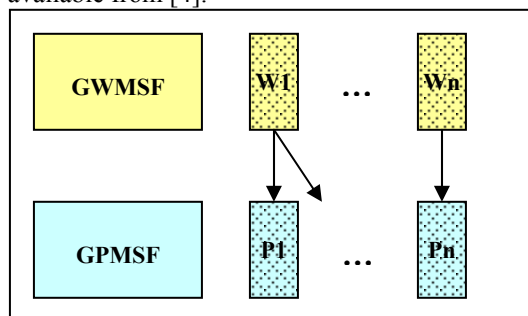


Figure 1 – The GeneGrid Workflow & Process Manager

The GeneGrid Application Manager (GAM) is responsible for the integration of bioinformatics applications in GeneGrid, such as BLAST or SignalP (see [5]). There are four main types of service implemented in GAM. A “GeneGrid Application Manager Factory Service” (GAMFS) appears on each resource available to GeneGrid, integrating bioinformatics applications available on individual resources into GeneGrid. The “GeneGrid Application Manager Service” (GAMS) is an instance of the GAMFS, which is created to interface directly with a specific application residing on the resource. Management of application and resource information is handled by lightweight services called “GeneGrid Node Monitoring Services” (GNMS) which collect usage and performance information for a resource. The final service type in GAM is the “GeneGrid Application and Resource Registry” (GARR), which provides service discovery functionality in the GAM component. All GAMFS and GNMS register with the GARR, providing information on applications supported, and resource status.

A client service may query the GARR for information regarding a given application to identify the appropriate GAMFS with which to communicate based on information sent to the GARR by both the GNMS services and the GAMFS. The client selects a GAMFS to create a GAMS before forwarding on the task for

execution to that GAMS. An illustration of GAM is provided in figure 2, while further information is available in [5].

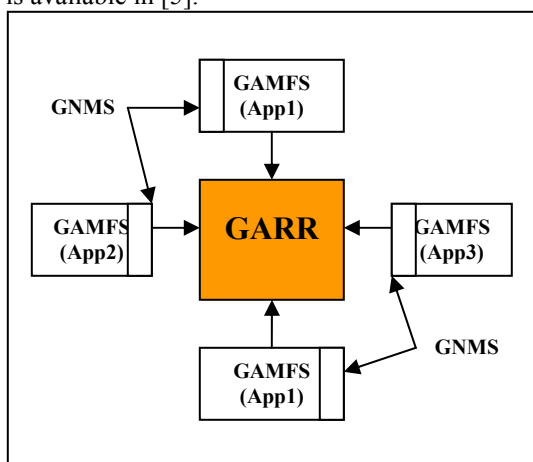


Figure 2 – The GeneGrid Application Manager showing GAM Services registering with GARR. Two GAM services are supporting the same application, but on different resources.

The GeneGrid Data Manager component (GDM) is responsible for the integration and access of a number of databases in GeneGrid. Much of the data which GeneGrid needs to exploit is available in the public domain, and is complemented by the commercial partners’ proprietary data sets. Furthermore, GeneGrid requires a data warehousing capability for the storage of experiment results and operational data e.g. workflow definitions [4], which is provided by the GDM.

The GDM consists of three types of services – the “GeneGrid Data Manager Registry” (GDMR), “GeneGrid Data Manager Service Factory” (GDMSF) and “GeneGrid Data Manager Service” (GDMS). Each database integrated into GeneGrid has at least one GDMSF bound to it. Each time access is required to any supported database, the relevant GDMSF creates a GDMS to interact with the database management system or file system. All GDMSF services register with the GDMR at start-up to advertise their existence and the database they support. The GDM will be discussed further in a later section.

When integrated together, these three components give rise to the concepts of the *GeneGrid Environment*, and the *GeneGrid Shared Resources*.

The GeneGrid Environment (GE) is the collective name for the (distributed) core elements of the GeneGrid project, consisting of all services from the GWPM component, as well as the GARR from the GAM component and the GDMR from the GDM. Two GDMSF are also included in the GE, configured to support the GeneGrid Workflow Status database, and the GeneGrid Workflow Definition database – both of which are discussed later in this paper. Each GeneGrid Environment will also have a “GeneGrid Environment Interface” (GENI) – a means for end users to interact transparently with the GeneGrid services.

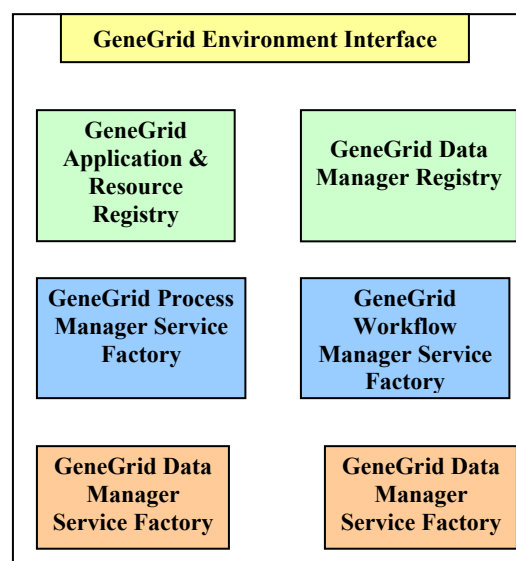


Figure 3 – GeneGrid Environment

The bioinformatics applications, as well as the data sets we wish to access from the GeneGrid Environment are referred to as GeneGrid Shared Resources. Each application has a supporting GAMFS Service and GNMS. Similarly, each data set has a supporting GDMSF. These services make themselves available to the GeneGrid Environment by registering with the GARR or GDMR respectively. Furthermore, these services may be shared across multiple GE by simply registering with each independent GE, as illustrated in figure 4.

Each GE may be considered as a single installation of the GeneGrid project, with all required core elements readily available to the owning organisation(s). While organisations can share a single GE, managing their own independent GE allows organisations to share

resources, while limiting the amount of information they must share.

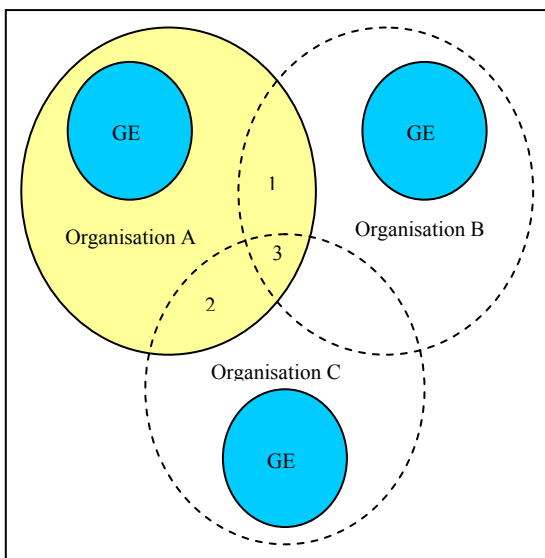


Figure 4 – Shared Resources across multiple GE. Each organisation has resources available to its own GE. Resources falling into area 1 are shared by A and B. Section 2 contains resources shared by A and C, while resources shared by A, B and C are in section 3.

3 The GeneGrid Data Manager

As mentioned in section 2, the GDM is responsible for the integration and access of all required data in GeneGrid, which fall into two categories.

Biological Data includes data presently available in the public domain e.g. SwissProt [6], as well as data sets proprietary to the commercial partners. Some of this data is stored in Relational Database Management Systems, such as Oracle [7] in the case of one proprietary data set, or MySQL [8] in the case of the public data set ENSEMBL [9]. However, the vast majority of biological data in the public domain is stored in flat file(s), structured specific to the database in question.

GeneGrid Data is data either generated by GeneGrid, such as Workflow Results, or operational data required by GeneGrid such as Workflow Definitions [4]. Since much of this data is passed around between GeneGrid services, it

was decided to store all GeneGrid Data in XML databases, such as Xindice [10].

The integration of such disparate data sets does present a major challenge, exacerbated by the lack of standardisation across public biological data sets. Therefore, GeneGrid has adopted the OGSA-DAI [11] approach to data access and integration, extending the solution as required. The use of OGSA-DAI provides the GDM with an “off-the-shelf” functional solution to a lot of the GDM requirements, including a facility for service discovery, as well as access and integration of a number of database management systems including Xindice [10], Oracle [7] and MySQL [8].

The OGSA-DAI architecture of GT3 services fitted in conveniently with the GeneGrid architecture, and provided a means of interaction with the other GeneGrid components.

The GDM consists of three types of cooperating services, correlating with those services offered in OGSA-DAI.

The GDMR is a persistent service whose main role is that of service discovery. By registering with a GDMR, a GDMSF advertises its existence and functionality to all potential clients. The GDMR is the GeneGrid implementation of the Data Access & Integration Service Group Registry [11].

The GDMSF is the GeneGrid implementation of the Grid Data Service Factory [11]. It is a persistent service configured to support access and interaction with a single data set. Its role is to provide a facility for creating GDMS for interacting with the database.

The GDMS is the GeneGrid implementation of the Grid Data Service [11]. It is the primary service of the GDM, created and configured by a GDMSF to allow clients to interact with a specific data set. Unlike the other types of GDM service, the GDMS is a transient service, created and destroyed by its owning client, or, once created expires after a set period.

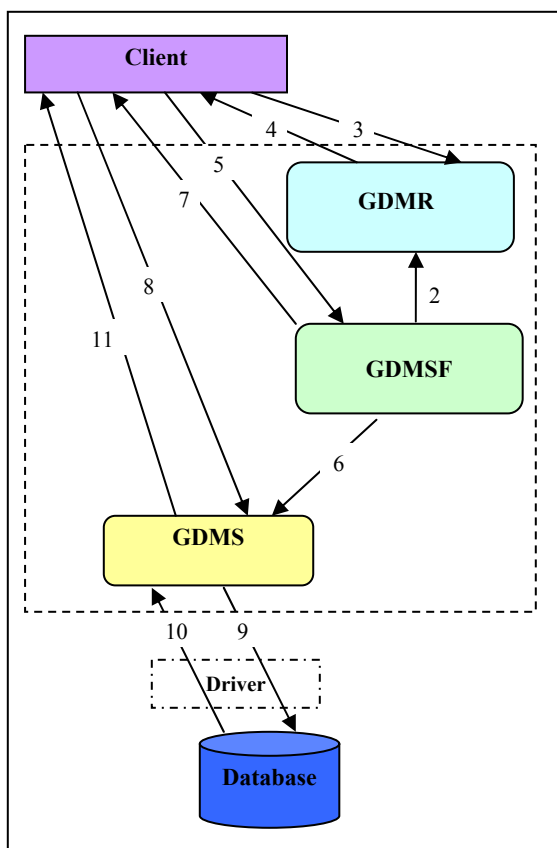


Figure 5 – Service interaction in the GDM.

Figure 5 provides an illustration of GDM service interaction to allow a client to interact with a given data set. The sequence of steps is as follows.

- 1) The Grid service container creates both a GDMR and a GDMSF.
- 2) The GDMSF registers with the GDMR
- 3) The client queries the GDMR for the location of the GDMSF for the database
- 4) The GDMR returns the location of the GDMSF.
- 5) The client requests the GDMSF to create a GDMS.
- 6) The GDMSF spawns a GDMS
- 7) The GDMSF returns the location of the new GDMS to the client.
- 8) The client submits an operation to the GDMS for execution.
- 9) The GDMS submits the query using a database connectivity driver.
- 10) The database returns a result of the operation to the GDMS.

- 11) The GDMS forwards the result of the operation to the client.

4 GeneGrid Test Bed

GeneGrid is a two year industrial e-Science project which commenced development late 2003. The first prototype was released in March 2004, with the second release following in August 2004. These releases consisted of initial implementations of all three components.

For its part, the GDM in release 1 implemented all three GDM service types, based on the GT3 services from OGSA-DAI release 3 [11]. This allowed integration and access of the following data sets

- **GeneGrid Workflow Definition Database** – a Xindice collection [10] storing template workflow files. This is a core GeneGrid DB, and is supported by a GDMSF in the GE.
- **GeneGrid Workflow Status Database** – a Xindice collection [10] storing copies of submitted workflows for tracking and recovery purposes. This is a core GeneGrid DB, and is supported by a GDMSF in the GE.
- **GeneGrid Results Database** – a Xindice collection [10] storing workflow results.
- **ENSEMBL [9]** – a public MySQL [8] biological database.

An extension was required to the base OGSA-DAI functionality to allow the successful integration and access of the following databases.

- **SwissProt [6]** – a public structured flat file biological (protein) database.
- **EMBL [12]** – a public structured flat file biological (nucleotide) database.

OGSA-DAI employs a JDBC or XML:DB driver to interface with relational database management systems and XML database management systems respectively. These drivers allow OGSA-DAI to submit query or update operations to the database systems using standard query languages such as SQL, XPath and XUpdate. Upon investigation no suitable equivalent driver was found for communicating with structured flat files in this way, so it was necessary to develop a custom means of executing queries against these forms of database. PERL [13] is a popular scripting

language, renowned for its file access efficiency, so we investigated the possibility of developing a PERL based driver for use in the GDM. Furthermore, a set of open source PERL modules called BioPERL [14] provide functionality to interact with a number of biological databases.

For release 1, a set of PERL scripts, using BioPERL modules, were developed to provide query functionality of SwissProt and EMBL. These scripts are invoked by the GDMS as required using a temporary proprietary query language. In turn, the scripts execute the query against an indexed version of the data set (generated in advance using PERL/BioPERL), and create a file containing the query results, which is then read in by the GDMS. Furthermore, these PERL scripts are capable of returning results in a number of different formats other than the native database format, namely in FASTA [15] and XML [16], allowing results to be more portable between GAM supported applications.

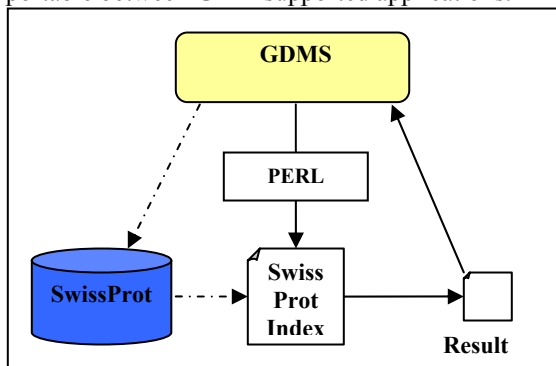


Figure 6 – Querying SwissProt using GDM and PERL. A query is made against the SwissProt database by, in fact, querying an indexed file version of the database. The PERL script used to interface with the database creates a results file, which is subsequently read in by the GDMS

Release 2 saw the refinement of structured flat file database integration, with the PERL scripts used in release 1 being ported to a JPL (a hybrid java and PERL based programming language) driver [19]. This cut the overhead on structured flat file querying, by removing the need for extra configuration, and by allowing the streaming of results directly into the GDMS.

The impending release of DFDL [20] may impact this area of the GDM, and so an investigation into its functionality and usage will take place upon its release. Later releases of GeneGrid may well see

the phasing in of DFDL to integrate additional data sets.

Further database integration will also take place with the inclusion of TrEMBL, TrEMBL_new (both SwissProt format) [6] and possibly Pfam [18] – all structured flat file public biological databases. Also, more instances of the GeneGrid Results Databases will be included as well as any identified proprietary databases.

An obstacle encountered implementing OGSA-DAI as the basis of the GDM was the use of CDATA XML tags [16] in query results from XML databases. The CDATA tag has the effect of concealing all data between its opening and closing tags from any XML parser. Use of these tags is necessary in the case of data obtained from a relational database, and now in structured flat file data, due to the content of the results containing illegal XML characters [16]. However results obtained from a Xindice database will not contain any such characters, and therefore the use of CDATA tags in this manner is not strictly necessary. This presented a problem in GeneGrid since operational data is stored in XML databases, and in order for GeneGrid services to utilise the data retrieved by the GDM, further processing is required.

Another OGSA-DAI specific issue was the inability to identify specific GDMSF based on a single call to the GDMR. In order to find a GDMSF supporting a specific database, a client must first obtain a list of registered GDMSF from the GDMR, and follow that with a query of all GDMSF for relevant service data until it identifies the one appropriate. For release 1 of GDM, it was decided to accept this impact on performance since a solution was promised in later releases of OGSA-DAI.

Staying with the OGSA-DAI based registry, with every GDMSF only registering with the GDMR at start up, there will be no registration information available for any GDMSF started before the GDMR. Similarly, if the GDMR is restarted, all registration information is lost, thereby ceasing the ability of the GDMR to advertise services to potential clients. This presents a major issue to an industrial project such as GeneGrid, and a ‘recoverable’ registry must be introduced in later releases of the GDM, or OGSA-DAI on which it is based.

Release 2 saw the upgrade of OGSA-DAI from release 3 to release 4 [17]. One of the new features of this OGSA-DAI release is the sending of registration information by each GDSF during registration with the Registry. This information may now be accessed by querying the Registry, thereby eliminating the need to query each GDSF individually to identify any specific one. Implementation of this feature in the GDM improved the efficiency of service discovery, an improvement necessary as more and more databases were integrated.

This OGSA-DAI upgrade also served to make the interaction with GDM services easier for all GeneGrid services, by inclusion of the new “*Client Toolkit API*” [17] as a replacement to the document based approach used in earlier OGSA-DAI releases.

A final issue which appeared during release 1 development was the size limitation of approximately 5Mb imposed by the Xindice Database Management System. While not a concern initially during release 1, such a limitation may be unworkable in later releases, hence investigation into the possibility of moving the GeneGrid databases from Xindice to an XML flat file format has been undertaken for release 2, impacting future releases.

5 Conclusions

The integration and access of data in a grid environment has proven to be a major challenge to GeneGrid. Evolving standards, such as DAIS [21] are attempting to address this issue of grid computing. Projects like OGSA-DAI, which is evolving towards DAIS compliance, provide an extensible framework for relational and XML database integration.

GeneGrid, by extending OGSA-DAI, has developed the GDM focusing on the industrial requirements presented by the GeneGrid project and its commercial partners’, but not without issues.

With much data relevant to scientists stored in structured flat files, we feel this is an area which should be included in the DAIS charter. We also feel that an all inclusive solution to RDBMS, XML databases, and Structured Flat File database

integration would be of great benefit to many grid projects, and essential to the success of GeneGrid. The GDM provides such a solution specific to the requirements of GeneGrid by using PERL / BioPERL based drivers for accessing structured flat file databases in the OGSA-DAI framework.

At the time of writing, DAIS is discussing the inclusion of file support in its charter.

Service Discovery in OGSA-DAI has improved between release 3 and release 4 with the inclusion of registration information by the GDSF at registration time. However, it is still a feature of OGSA-DAI which could be refined. Also, with OGSA-DAI registries losing all registration information if restarted, and their need to be started before any services wishing to register with them, the sharing of data resources across GeneGrid Environments does present quite a challenge.

The inability to use data stored in XML databases for operational purposes without post-processing in the client due to the use of the CDATA tag in query responses has also been an issue for GeneGrid.

With the issues mentioned above, and issues from other e-Science projects in mind, the OGSA-DAI project team have taken the positive step of creating the “OGSA-DAI Users Group”. This group of users and developers of OGSA-DAI meet to provide input and feedback on the OGSA-DAI project, and suggest possible features for future releases. GeneGrid has taken an active role in this initiative.

The first test bed release of GeneGrid, including the GDM, has been successfully deployed across multiple administrative domains, incorporating many geographically disparate resources. Release 2 saw GeneGrid services being deployed on a 32-node cluster in the Belfast e-Science Centre to compliment the resources available in release 1. The first implementation of the GeneGrid Environment Interface has also been included in release 2.

References

- [1] OGSi <http://www.gridforum.org/ogsa-wg/>
- [2] OGSA <http://www.globus.org/ogsa/>
- [3] Globus Toolkit
<http://www.globus.org/toolkit>
- [4] GeneGrid: A practical Workflow Implementation for a Grid Based Virtual Bioinformatics Laboratory, *David R. Simpson, P.V. Jithesh, Noel Kelly et al*
- [5] Bioinformatics Application Integration and Management in GeneGrid: Experiments and Experiences, *P.V. Jithesh, Noel Kelly, David R. Simpson et al*
- [6] SwissProt User Guide
<http://au.expasy.org/sprot/userman.html>
- [7] Oracle RDMBS <http://www.oracle.com>
- [8] MySQL <http://www.mysql.com>
- [9] ENSEMBL <http://www.ensembl.org>
- [10] Xindice <http://xml.apache.org/xindice>
- [11] OGSA-DAI release 3 Overview
<http://www.ogsadai.org.uk/docs/R3/OGSA-DAI-USER-UG-PRODUCT-OVERVIEW.pdf>
- [12] EMBL User Guide
http://www.ebi.ac.uk/embl/Documentation/User_manual/usrman.html
- [13] PERL <http://www.perl.com>
- [14] BioPERL <http://www.bioperl.org>
- [15] FASTA format description
<http://www.ncbi.nlm.nih.gov/BLAST/fasta.shtml>
- [16] XML <http://www.w3c.org/xml>
- [17] OGSA-DAI release 4 Overview
<http://www.ogsadai.org.uk/docs/current/DAIOverview.html>
- [18] Pfam protein database
<http://pfam.wustl.edu/>
- [19] Java PERL Language (JPL) Tutorial
<http://www.sunsite.ualberta.ca/Documentation/Misc/perl-5.6.1/jpl/docs/Tutorial.html>
- [20] Data Format Description Language (DFDL) Working Group
<http://forge.gridforum.org/projects/dfdl-wg/>
- [21] DAIS Working Group
<https://forge.gridforum.org/projects/dais-wg/>
- [22] Global Grid Forum (GGF)
<http://www.ggf.org>
- [23] Grid Based Virtual Laboratory *Paul Donachy, Terrence J. Harmer, Ron H. Perrott et al.*