



CONTRACT NUMBER 031513

eDEISA
**EXTENDED DISTRIBUTED EUROPEAN
INFRASTRUCTURE FOR
SUPERCOMPUTING APPLICATIONS**

European Community Sixth Framework Programme
RESEARCH INFRASTRUCTURES
Integrated Infrastructure Initiative

The DEISA Life Sciences Portal Architecture

Deliverable ID: eDEISA-eSA3-A1

Due date : November, 30, 2006

Actual delivery date: December 21, 2006

Lead contractor for this deliverable: Organisation, Country

Project start date : June 1st, 2006

Duration: 2 years

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Content

Table of Content.....	1
1. Introduction.....	2
1.1 Executive Summary.....	2
1.2 References and Applicable Documents	2
1.3 Document Amendment Procedure	3
1.4 List of Acronyms and Abbreviations	3
2. General Strategy.....	4
2.1 Two Classes of Bioinformatics Applications	4
2.1.1 Alignment and Sequences Comparisons Applications	4
2.1.2 Phylogeny Applications	4
2.2 Deployment Strategy.....	5
2.3 Identification of the Technology.....	6
2.3.1 JMEA.....	6
2.3.2 GridSphere	6
2.3.3 EnginFrame	7
2.3.4 Required Functionalities	8
2.3.5 Choice of the Consortium	9
3. Deployment issues	10
3.1 Description.....	10
3.2 Generic Accounts.....	10
3.3 User Administrative Information	11
3.4 Authentication.....	11
3.5 Authorization.....	11
3.6 Accounting.....	11
3.7 Jobs Management.....	11
4. Enhancements for the Second Year.....	12
4.1 Put UNICORE and DESHL technologies behind the portal.....	12
4.1.1 Presentation of the technologies	12
4.1.2 Job Submission and Monitoring via the DESHL SAGA library	13
4.1.3 Impacts on the User Accounts and AAA issues	14
4.2 Put OGSA-DAI technology behind the portal.	14
4.2.1 Presentation of the Technology	14
4.2.2 Data services Workflow Engine Use Case	15
4.2.3 Proposed Architecture for Audit Trail Functionality	16
4.3 Added Value of UNICORE and OGSA-DAI.....	17
CONCLUSION.....	18

1. Introduction

1.1 Executive Summary

The aim of the DEISA Life Science portal activity is to provide a Web interface to a collection of Life Sciences applications. This activity leverages on the experiences gained in some of the DEISA Joint Research activities to design a general approach to expose DEISA infrastructure services to external users in an Application Service Model where users connect to an external application (the portal) and not to supercomputers that will ultimately perform the job. User's requirements are crucial, because the basic idea is to put supercomputing power at their fingertips without enforcing drastic changes in their working habits. This is why the activity is targeted to one specific discipline: bioinformatics and Life Sciences [1].

The portal will be a testbed for future portal activities in the infrastructure and will act as an application service provider that will deal the security, AAA (Authentication, Authorization and Accounting), job submission and management issues. The technological solution that has been identified by the Executing Committee is the EnginFrame framework which is commercialized and developed by NICE [10].

The purpose of this deliverable is to describe the architecture that has been retained for the DEISA Life Sciences Portal. The first section presents the general strategy that has been adopted for the deployment of the portal. The second section details how EnginFrame will be deployed on the DEISA infrastructure. The third section presents possible architecture enhancements that could be implemented in the second year of the activity: integration of UNICORE and OGSA-DAI technologies.

This document is publicly available.

1.2 References and Applicable Documents

- [1] eDEISA Technical Annex I "Description of Work"
- [2] <http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/information3.html>
- [3] <http://www.phylo.org/news/RAxML.html>
- [4] <http://www.gridisphere.org>
- [5] <http://developers.sun.com/prodtech/portalserver/reference/techart/jsr168/>
- [6] <http://jcp.org/aboutJava/communityprocess/final/jsr154/index.html>
- [7] <http://www.hpc-europa.org/>
- [8] <http://www.collab-ogce.org/ogce2/>
- [9] David F. Snelling, Sven van den Berghe, Vivian Qian Li "Explicit Trust Delegation : Security for Dynamic Grids" Available at <http://www.unigrids.org/papers/explicittrust.pdf>
- [10] <http://www.nice-italy.com>
- [11] <http://apache.tomcat.org>

[12] http://summit.unicore.org/2006/presentations/15_Soddemann_UnicoreSummit06.pdf

[13] <http://www.gridisphere.org/gridsphere/docs/ReferenceGuide/ReferenceGuide.html>

[14] <http://biojava.org>

[15] <http://www.mortbay.org/>

[16] <http://ws.apache.org/axis/>

[17] <http://www.biomoby.org>

1.3 Document Amendment Procedure

The initial document amendment procedure is via communication between members of DEISA eSA3 team. The document is then submitted for review to the eDEISA Executive and an Executive appointed eDEISA reviewer. The document is then amended according to comments received from the Executive and the eDEISA appointed reviewer. It is subsequently re-submitted to the eDEISA Executive for submission to the EU.

1.4 List of Acronyms and Abbreviations

AAA	Authentication, Authorization and Accounting
ACL	Access Control List
BLAST	Basic Local Alignment Search Tool
DAS	Distributed Annotation System
DNA	Deoxyribonucleic Acid
DEISA	Distributed European Infrastructure for Supercomputing Applications
DESHL	DEISA Services for the Heterogeneous management Layer
EAS	Enterprise Application Server
ETD	Explicit Trust Delegation
EJB	Enterprise Java Bean
HTTP	Hypertext Transfer Protocol
JAAS	Java Authentication and Authorization Service
JMEA	Job Management Enterprise Application
JRA	Joint Research Activity
JSR	Java Specification Request
J2EE	Java 2 Enterprise Edition
LDAP	Lightweight Directory Access Protocol
MPI	Message Passing Interface
NIS	Network Information Service
NCBI	National Center for Biotechnology Information
OGCE	Open Grid Computing Environment
UNICORE	UNiform Interface to COmputing REsources
RAXML	Randomized Axelerated Maximum Likelihood
SAGA	Simple API for Grid Applications
SSH	Secure Shell
WAS	Web Application Server
WSRF	WS-Resource Framework
XML	Extensible Markup Language
X509	Standard for digital certificates

2. General Strategy

The global strategy defined by the workgroup is focused on three axes: identification of representative bioinformatics applications, definition of a deployment strategy and choice of a technology provider.

2.1 *Two Classes of Bioinformatics Applications*

The DEISA Life Science Joint Research Activity (JRA4) is responsible for identifying the applications that will be interfaced via the portal. JRA4 is concerned to find diverse applications widely used by the community, or even better, potentially essential in the future. Those applications should be parallelized already or be excellent candidates for parallelisation in order not to delay the first phase of the project. With JRA4 expertise those applications will be rendered easily accessible, in a transparent manner to the Life Scientists. JRA4 has already identified two classes of applications:

- alignment and sequences comparisons and
- phylogeny.

The portal activity will interface, in the initial pre-production phase, one representative application of each of these two classes. As applications of a given class have similar inputs, outputs and workflows, it will be relatively simple to extend the number of applications in the second year of the project.

2.1.1 *Alignment and Sequences Comparisons Applications*

Alignment and sequences comparisons applications compare biological sequences, such as the amino-acid or the DNA sequences, with well known sequences stored in databases, and identify similarities with a certain threshold. Inputs and outputs of such applications are listed below :

Inputs

- Text (gene sequence)
- Research parameters
- Databases of well known sequences

Outputs

- Simple text file

The first application that has been identified in this class is the BLAST (Basic Local Alignment Search Tool) [2] suite from NCBI. This is an obvious choice since it is the most used application in the similarity search/alignment domain, maybe even in the whole bioinformatics field. It also has an MPI version: mpiBLAST. It is not yet known how the application will perform when deployed onto the DEISA facility but efforts are being made by the developers to also parallelize the I/O.

2.1.2 *Phylogeny Applications*

Phylogeny applications analyze gene sequences of different species and identify ancestral relations between them. Inputs and outputs of such applications are listed below :

Inputs

- Text (gene sequences)

Outputs

- Tree in form of image, PDF file, text file.

The first application that has been identified in this class is RAxML (Randomized Axelerated Maximum Likelihood) [3]. RAxML is a dynamic and promising project that has the merit to be parallelized in an MPI version, and also in an OpenMP version. It may seduce more users than currently, if combined with an adequate user-friendly interface and backed by a powerful supercomputer infrastructure. RAxML is used to build phylogenic trees from biologic sequences multi-alignments, comparing each sequence two by two and building a distance matrix.

2.2 Deployment Strategy

The DEISA supercomputing grid [1] is a composed of two groups of supercomputing resources: the 'AIX super-cluster', which is composed of five IBM supercomputers running LoadLeveler and the 'heterogeneous supercomputing resource' which constitutes a grid of supercomputers and super-clusters from various vendors (IBM, SGI, NEC), running various batch schedulers (LoadLeveler, LFS, NQS II, PBSPro). All DEISA sites will expose computing resources to the portal with one exception - HLRS. This is because no bioinformatics application has been identified by JRA4 for vector platforms. The computing resources of the ten remaining sites can be grouped in three different architectures that are listed in table1.

<i>Vendor</i>	<i>Hardware Architecture</i>	<i>OS</i>	<i>Job Scheduler</i>	<i>Sites</i>
IBM	SP (Power 5)	AIX	LoadLeveler	FZJ, IDRIS, RZG, CINECA, CSC, ECMWF, EPCC
IBM	PowerPC	Linux	LoadLeveler	BSC
SGI	Altix (Itanium)	Linux	LSF and PBSPRO	SARA, LRZ

Table 1: Three different architectures for the portal

JRA4 will compare the performance of each bioinformatics applications on these three architectures in order to identify the most performant architecture for each application. Each bioinformatics application will then be deployed on the most adapted architecture and on all the DEISA platforms of this architecture. A job routing service will be implemented on the portal to allow administrators to specify the submission site. DEISA has also asked NICE to create an automatic submission (*round-robin*) option in the submission service to make sure that all sites of a same architecture are "equally" solicited for a given application.

In the second year of the activity, the number of applications will be increased. JRA4 has currently identified a total of ten bioinformatics applications that can be good

candidates for the portal. These applications belong essentially to the alignment and sequences comparisons classes but also to the linkage analysis and motif search classes.

2.3 Identification of the Technology

Several portal technologies have been investigated for the Life Science portal : JMEA developed by DEISA, GridSphere and EnginFrame. EnginFrame is the technology that has been finally retained.

2.3.1 JMEA

Thomas Soddemann at RZG has developed a portal technology, JMEA (Job Management Enterprise Application) [12] that has been used for the DEISA Material Sciences and Plasma Physics portals. It is a Java Enterprise (J2EE) application that can be accessed via a portal-specific web front-end. JMEA interacts with the UNICORE server-side infrastructure via a refractored UNICORE client library.

Figure 1 shows the flow of information/control for a web browser accessing the portal application via the web application server (WAS) that, in turn, accesses the enterprise application server (EAS), which then connects to the UNICORE.

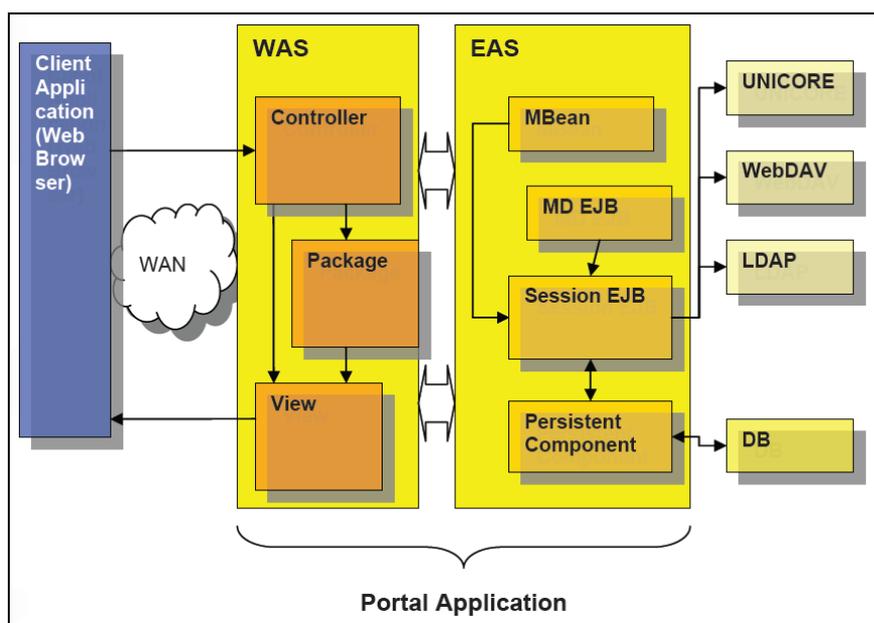


Figure 1 - The JRA1 and JRA3 portal architecture

The Life Sciences portal differs from the other DEISA portals in that it must permit access to the DEISA infrastructure from users external to the DEISA infrastructure. Also, the JMEA uses Explicit Trust Delegation (ETD)¹ for authentication as is required by the JRA1 and JRA3 portals, but in the case of the Life Science portal this feature is not needed (ETD can be however turned off in JMEA).

2.3.2 GridSphere

The GridSphere framework is packaged as a web application that provides a portlet container for managing deployed portlets [13]. It is compliant with the Java portlet

¹ The concept of delegation is defined in [9] : it is *the process whereby one entity on the Grid (usually an end-user) grants rights to another entity on the Grid (usually a process) to perform actions on the entity's behalf.*

specification, JSR-168 [5]. A portlet is a Java component that generates dynamic web content in response to web requests. Portlets run inside a runtime environment provided by a portlet container: these are extension of Java Servlets [6]. GridSphere provides a collection of core portlets as well as a development environment for the creation of new portlet applications. The GridSphere technology has been used by the HPC-Europa project [7] for their *single point of access* portal. Another widely used portal framework that has a similar architecture is OGCE (Open Grid Computing Environment) [8]. Figure 2 details the general architecture of a GridSphere portal.

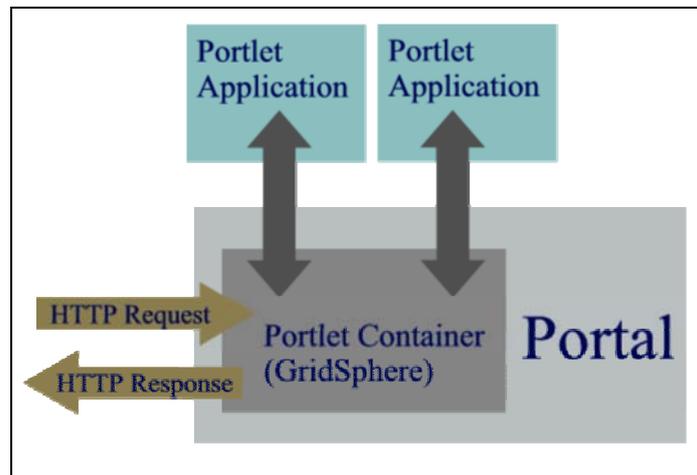


Figure 2 : Architecture of a GridSphere Portal

2.3.3 EnginFrame

The EnginFrame framework is commercialized and developed by the NICE Company [10]. An EnginFrame portal is composed of an EnginFrame server and several EnginFrame agents. The EnginFrame server, which is deployed in a Tomcat Servlet Container [11], is accessed by an end-user's web browser. It can manage the authentication of users, their authorization and access to the agents. EnginFrame supports several authentication mechanisms such as the underlying system methods (LDAP, NIS, GLOBUS) or custom authentication mechanisms developed by NICE. EnginFrame agents interact local applications such as the local batch schedulers. Figure 3 details the general architecture of an EnginFrame portal..

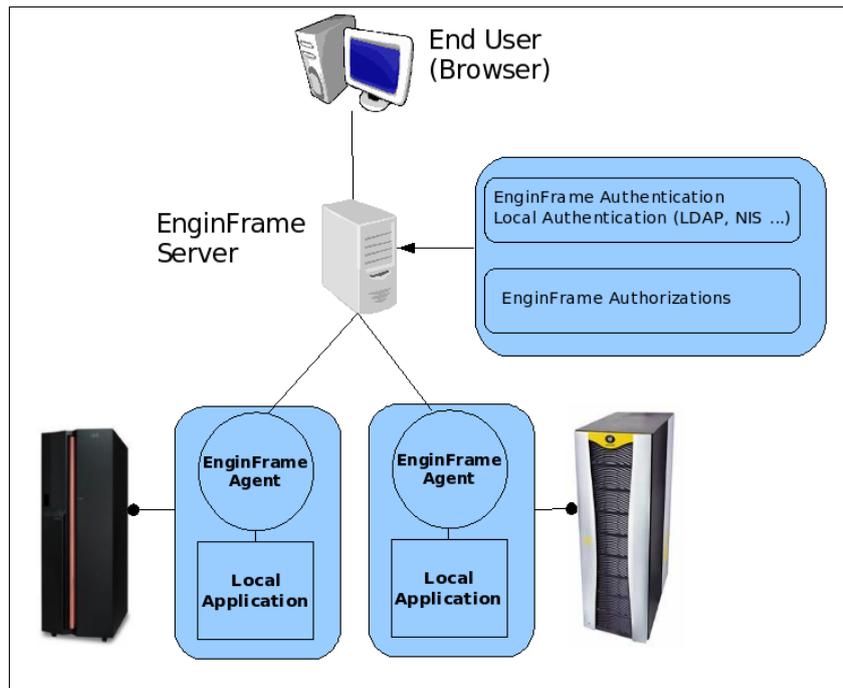


Figure 3: Architecture of EnginFrame

Another alternative to interface the EnginFrame server to a site is to use a plug-in developed by NICE that allows the server to connect to the submission host via SSH. In this case, it is not necessary to deploy an agent on the submission host. There is a performance penalty with the SSH plug-in because the server must re-authenticate on the submission host each time a new action is performed by the user. The direct consequence is that the service exposed to the user will be very unresponsive. This performance penalty may not be significant for DEISA as the aim of the portal is to submit jobs and not to interface “real-time” applications. Actions like jobs submission and monitoring do not require a very strong reactivity: indeed, the core commands that are used to monitor jobs on local systems usually take a few seconds to respond.

2.3.4 Required Functionalities

Several features have been identified as crucial for the activity:

- **Adaptability:** The ability to produce quickly a web service to interface a new application is a very important point for DEISA as the number of services will be significantly increased in the second year of the project.
- **Users Administration :** intuitive graphical tools should be provided to manage big number of projects and user accounts. The possibility to map portal users to generic Unix accounts should also be provided.
- **AAA :** intuitive tools must be provided to manage the AAA aspects.
- **Support for the most common job schedulers:** LoadLeveler, LSF and PBSPRO
- **Maintainability:** The product must be accompanied with a very strong maintenance policy. The maintenance policy can be guaranteed by a contract signed with a technology provider.

Table 1 compares this features.

<i>Functionnality</i>	<i>JMEA</i>	<i>GRIDSPHERE</i>	<i>ENGINFRAME</i>
Type of licence	Not yet defined	Open source	Proprietary
Adaptability	Requires some Java knowledge	Requires some Java knowledge. Includes a portlet for the portal layout customization	Good. A prototype has been developed by the team to validate this point.
Administration of Users	No graphical administration tool.	Includes an administration portlets for creation of users and groups	Shipped with a convenient graphical administration tool : DataGate
Tools for AAA management	No graphical tool	Customisable with JAAS (Java Authentication and Authorization Service)	Login/Password LDAP,NIS, Globus A GUI is provided to manage the ACLs
Job submission	Unicore	Already done through Unicore and Globus	LSF,PBS (open an pro), gLite, grid-MP, LoadLeveler
Maintainability	Very new technology, developed and maintaind by only one person	Serious open source project - frequent releases	Serious company - frequent releases
Overall	Promising technology but maybe not mature enough.	Mature open source framework.	Very mature solution. Appears as the most ready to use solution.

Table 1 : Functionalities of JMEA, GridSphere and EnginFrame

2.3.5 Choice of the Consortium

The technological solution that has been identified by the Executing Committee in July 2006 is EnginFrame. This choice has been motivated by the necessity to deploy an off the shelves solution ready to use technology. DEISA has very few time to demonstrate portal importance in HPC exploitation and can't therefore take the risk to choose a technology that would require some significant developments efforts from the DEISA partners. However, in the second year of the project, the option of investigating how JMEA or GridSphere can be uses to interface UNICORE is still open.

3. Deployment issues

This section describes the deployment choices that have been made for the DEISA Life Science portal. The portal configuration that has been defined for DEISA is an hybrid configuration that combines the use of agents and SSH plug-ins, and where multiple portal users will be mapped to a single DEISA generic account.

3.1 Description

Each site will choose either to run an agent or to use the SSH plug-in. In order to increase the security level of the portal, the EnginFrame server will be split into two parts : the core EnginFrame Server (Tomcat server) that implements all the business logic (authorizations, accounting, services, user data, and so on) will be deployed behind the firewall and an Apache server will be deployed outside to authenticate portal users and to relay the HTTP/HTTPS flow to the EnginFrame server. Figure 4 summarizes this concept.

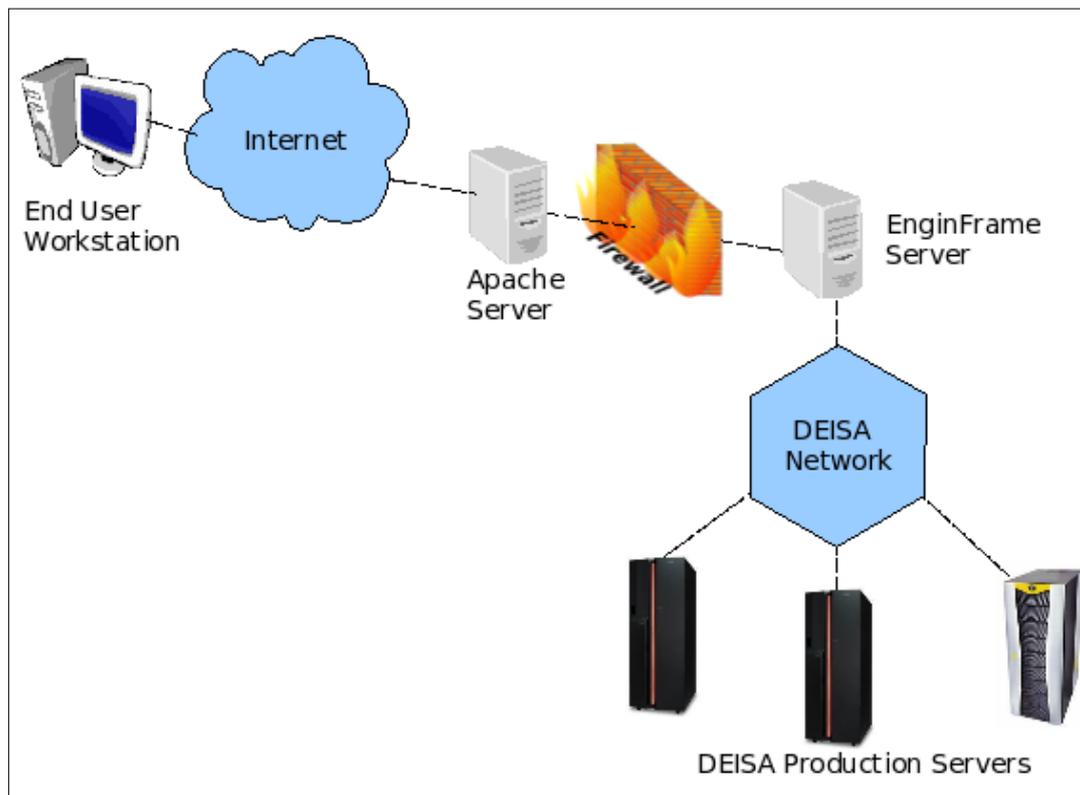


Figure 4 :Multi-sites configuration

3.2 Generic Accounts

The prerequisite of the Life Sciences portal activity is to allow users that are not registered on the DEISA sites to access the Life Sciences portal. Portal users will instead be registered at the portal level. Portal users will belong to projects that will also be registered at the portal level. A project will define a budget (allocation) that can be shared between all users registered to the project. The portal will be in charge of blocking all the accounts of project that have consumed their budget.

On the system point of view, it is clear that standard Unix accounts still have to be used to run the selected applications. The portal will be responsible for mapping multiple portal users to a single DEISA generic Unix account.

3.3 User Administrative Information

The use of a generic portal account does not mean anonymity for portal users. Administrative information about users will be saved in a DEISA internal database that will not be located on the EnginFrame server. This is a more secure approach since no confidential information regarding users is saved on the portal server and hence exposed on the Internet. The portal will include – with each submitted job – the user and project identifiers, so that site administrators can determine the user of the running jobs (via the internal database). These two pieces of information are also needed to maintain the contents of the DEISA accounting databases.

3.4 Authentication

The typical use case for EnginFrame is to rely on the local site authentication services (for example, LDAP or NIS) for user authentication. Unfortunately, this approach will not be suitable for the Life Sciences portal, because no global authentication service is provided in DEISA. The portal will instead use the login/password authentication service included in EnginFrame. NICE will also implement a new authentication service based on X.509 certificates.

3.5 Authorization

EnginFrame provides a technology to configure Access Control Lists on the different services defined on the portal. Portal administrators will therefore have the possibility to explicitly authorize users to access each bioinformatic application. NICE will implement the additional option of defining such access control during the creation of a project on the portal. Therefore, all users within a project will be authorized to access the same applications.

3.6 Accounting

The portal will be aware of the resource consumption of projects allowing portal user accounts that have consumed their budget to be blocked and also permitting users to access their current level of consumption. Each site will be free to block the access to the generic Unix account when the total consumption of this account has reached the allocation that the site has agreed to reserve to DEISA for the Portal Life Science activity. The portal will be responsible for blocking all the portal accounts of projects that have consumed their budget. Some special procedures are currently being defined to interface the portal with the DEISA accounting databases. One option that is still under discussion is to use the OGSA-DAI technology.

3.7 Jobs Management

EnginFrame agents and SSH plugins will interact with the DEISA local schedulers (LoadLeveler, LSF and PBSPRO) to submit jobs. Portal users will have the possibility to monitor the status of their jobs on the DEISA platforms. A cancellation option will also be provided.

4. Enhancements for the Second Year

One important aspect of the Life Sciences portal is its interoperability with middleware technologies. Two architecture improvements have been identified for the second year of the project. The interoperability of the portal with UNICORE could be improved and OGSA-DAI may also be integrated to provide an innovative workflow service to help life scientists keep a record of their data, queries and results.

4.1 Put UNICORE and DESHL technologies behind the portal

UNICORE (Uniform Interface to Computing Resources) is a generic job submission interface to several batch schedulers, which is used within DEISA to insulate the user from these disparate systems. The interfacing of UNICORE through DESHL has already been sketched out.

4.1.1 Presentation of the technologies

The DESHL (DEISA Services for the Heterogeneous management Layer) has been developed by the DEISA Joint Research Activity JRA7. It provides standards-based access for users and their applications to manage jobs and transfer files in the DEISA heterogeneous supercomputing infrastructure.

The DESHL command line tool is implemented as a layered stack with a SAGA (Simple API for Grid Applications) inspired interface at its top and the UNICORE ARCON client at its base. SAGA is a straightforward interface for communicating with Grid systems. The DESHL can be used for data management within the DEISA environment, and to submit, monitor and terminate jobs running on DEISA resources. Figure 5 shows the architecture of the DESHL, how it can be interfaced by the Life Sciences Portal, and the flow of control through the DESHL to the UNICORE server running at a DEISA site. The shaded boxes highlight libraries which form part of the DESHL.

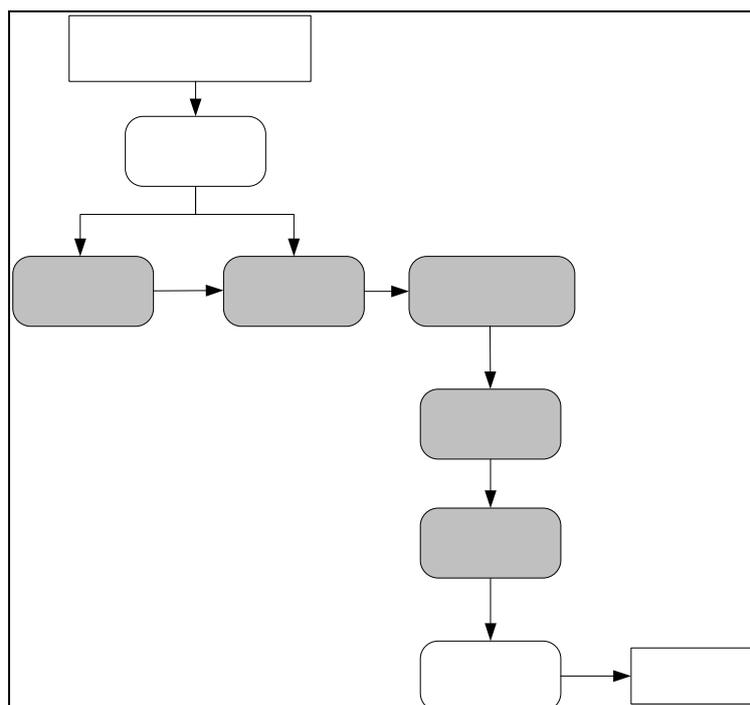


Figure 5: DESHL components within potential portal architecture.

The Life Sciences Portal could access the DESHL either via the command line tool, or via the SAGA library interface. There would be a performance benefit to access via a library interface since this would not require a new process to be spawned for each interaction.

4.1.2 Job Submission and Monitoring via the DESHL SAGA library

When Life Scientists log in to the portal, they are authenticated, and the user and project details are stored in a secure session spooler. This data will later be used to configure the SAGA session that is used to submit the jobs to DEISA. For example, if a life scientist submits a BLAST request, the BLAST service forwards the request to the DESHL service, which in turn calls into the DESHL SAGA interface to submit the BLAST job. A high level architecture describing how the DESHL SAGA interface can be used from an EnginFrame portal to submit and monitor jobs to the DEISA sites is show in the following diagram.

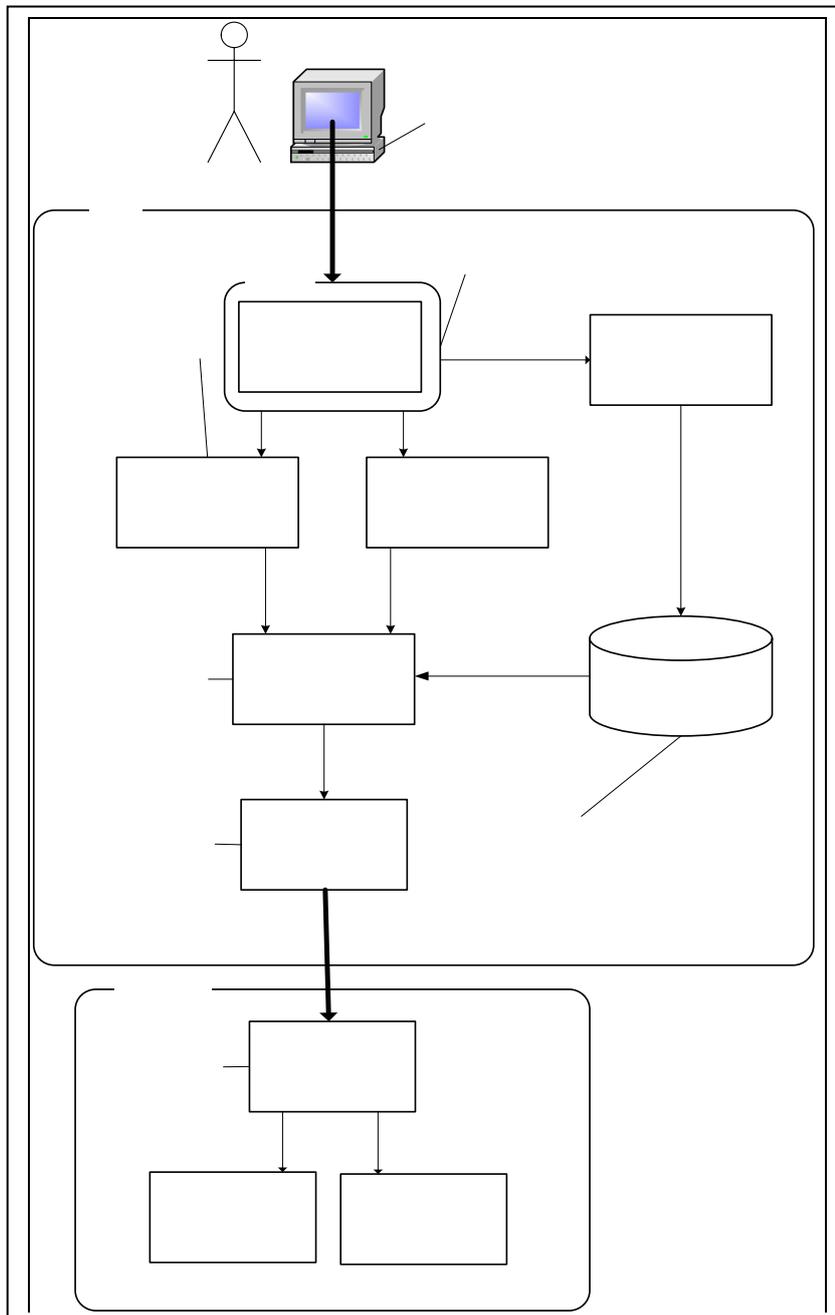


Figure 6: Architecture for interfacing to the DESHL SAGA interface.

At the time of writing, the DESHL supports a subset of the SAGA interface, namely SAGA Namespaces and SAGA Jobs. For this approach to be feasible the DESHL SAGA interface would need to be extended to support the SAGA Sessions interface which would provide management of the DEISA authentication data. Support for SAGA Session has been provisionally scheduled by the DESHL development team for inclusion in the April 2007 release of the DESHL.

In addition to the requirements on the DESHL, this approach also necessitates that EnginFrame supports invocation of methods inside a Java library. This functionality is under development, presently an alpha release, and due for production release before the end of 2006.

The Portal can interface DESHL at one of two points, either at the SAGA interface or via the command line tool. The SAGA interface is the favoured approach as the service does not fork an instance of the DESHL command line tool for each request. Each fork will consume time and resources on the portal: something that does not scale well.

4.1.3 Impacts on the User Accounts and AAA issues

The integration of UNICORE through DESHL can also be done with a generic account. This can be made possible by storing the credential (a X509 certificate) of the generic account on the hosts where the EnginFrame agents are running. Therefore the end user should see no difference for the authentication : the authentication with a login or a certificate will be still possible. The authorizations and accounting aspects should also not be impacted.

4.2 Put OGSA-DAI technology behind the portal.

4.2.1 Presentation of the Technology

OGSA-DAI is a grid middleware that supports the exposure of data resources, such as relational and XML databases. The software includes a collection of components for querying, transforming and delivering data in different ways, and a simple toolkit for developing client applications.

The OGSA-DAI Engine is the core component of OGSA-DAI. It interprets and acts upon descriptions of work called Perform Documents.



Figure 7 – The OGSA-DAI engine processing a perform document

Broadly speaking, a Perform Document is used to query data in an OGSA-DAI data resource, transform it, and deliver it to a location (Figure 7). Such documents are not intended for human consumption. They are generated by the OGSA-DAI clients and sent to an OGSA-DAI server for processing. A Perform Document is composed of a sequence of activities, each of which dictates an action to be performed. For example, activities can be created to: query a data resource; transform data in some way; or deliver results in some manner. Within a Perform Document, one can visualize data flowing between activities.

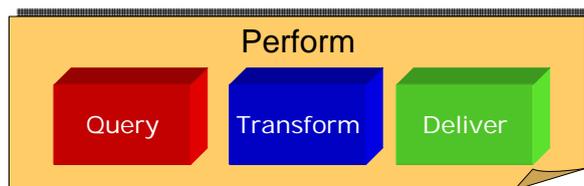


Figure 7 – An OGSA-DAI perform document is composed of a sequence of query, transform and deliver activities

A large number of useful activities are packaged up with OGSA-DAI. For example, there are activities to query an SQL database, or transform an XML document using XSLT. It is also possible for a developer to create new activities and incorporate them into their OGSA-DAI servers and the OGSA-DAI client toolkit.

4.2.2 Data services Workflow Engine Use Case

One of the requirements of the DEISA Life Sciences portal is that it should have 'added-value' over other Life Sciences portals. One way that this requirement could be met is if the portal were to supply an audit trail function to help life scientists keep a record of their data, queries and results.

OGSA-DAI could be used to manage workflow data, for example, to store jobs and their results for the portal users. This can be augmented by a library such as BioJava [14] to translate from one Life Sciences database format to another, which in turn will enable complex workflows to be created within the portal.

This feature should not be confused with the workflow management feature offered by some Life Sciences portals, such as the IBN portal, which uses BioMoby [17] to automatically discover the services which can process data of a certain type and then dynamically invoke them. The DEISA Life Sciences portal will offer a finite set of Life Science applications, which have known input and output data type. Dynamic discovery and invocation of services is not within the scope of this activity.

Typically, when life scientists use a portal to analyse some data, say a sequence, they run many queries on the data. They examine the results of their queries, tweak them slightly and run them again. They also take data from the results of one query and analyse it further using a different application. All of this involves the life scientist keeping a record of their interaction with the portal, the job inputs and results. Hence they are constantly *cutting and pasting* between their records and the portal application.

In order to offer the Audit Trail functionality the portal will need to store the user's data and their associated results. An obvious place to store the data would be in a database. The data stored in the database should be stored in raw form. Therefore, the portal should be able to format this data nicely when they are to be viewed by the user. It should also be able to parse the raw data and convert it from one format to another, so that the data can be moved easily between the different applications.

The challenges outlined above can be addressed within the proposed architecture for eSA3 using two tools: OGSA-DAI and BioJava. BioJava is a framework that provides an API for processing biological data. It can be used to:

- manipulate biological sequences – for example, convert from a protein sequence to a nucleotide sequence;
- parse files – for example, read, write and convert between BLAST and FASTA database files;
- access databases, such as BioSQL and Ensembl;
- interact with DAS (Distributed Annotation System) servers;
- and statistically analyse data.

It is an open-source project which is already used in several real-world bioinformatics applications.

4.2.3 Proposed Architecture for Audit Trail Functionality

A potential architecture for how OGSA-DAI and BioJava could be integrated into EnginFrame to offer the audit-trail functionality is shown in Figure 8. When a Life Scientist submits a job to DEISA through the portal the job input and output data are stored in a database. This database is exposed as an OGSA-DAI data resource. Perform documents are written to input, retrieve and update this data, and BioJava-based activities are written: to transform the data for exchange between jobs or, to format data for presentation to a user. In addition, the OGSA-DAI URL- or SNMP-delivery activities offer the possibility to deliver important results to the user via ftp or e-mail. It would be possible to access the local database directly, without using OGSA-DAI. However, an OGSA-DAI-based solution would be easier to maintain and extend.

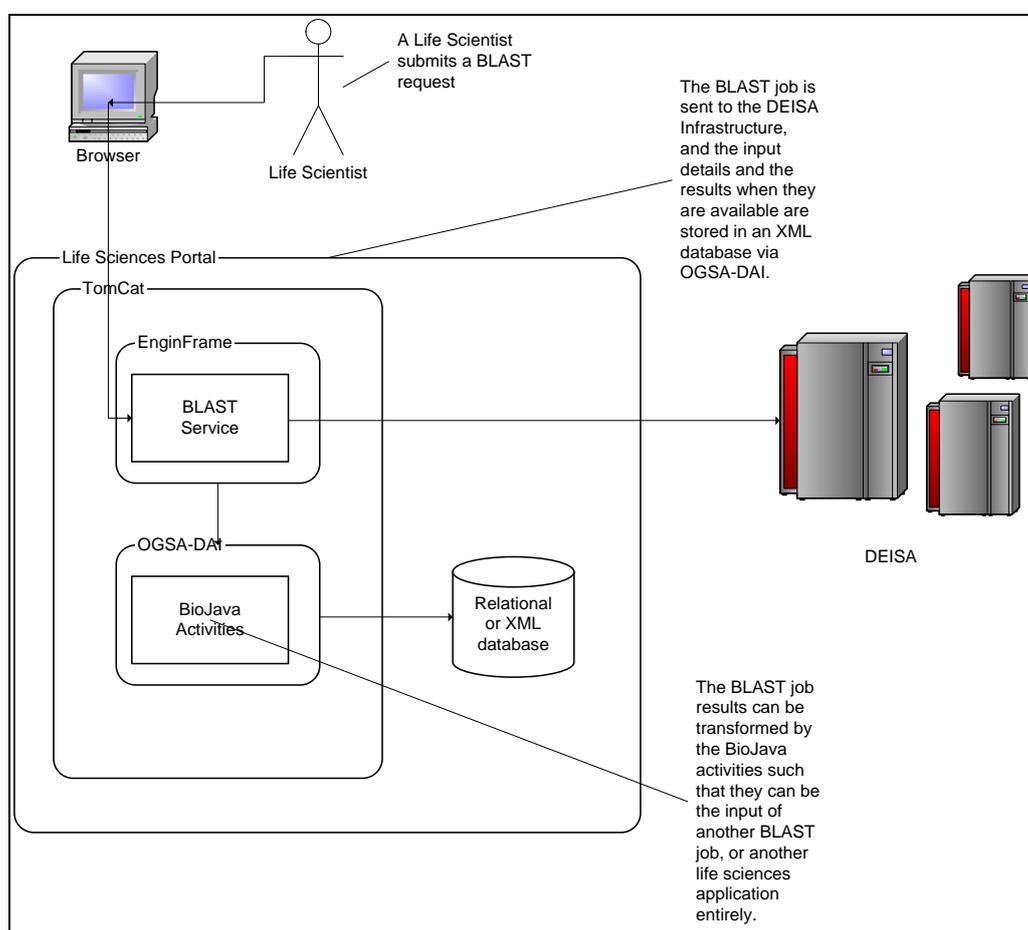


Figure 8 – Architecture of audit-trail functionality using OGSA-DAI and BioJava.

Tomcat is bundled with the EnginFrame distribution; however, the server could be contained within another compliant servlet container such as Jetty [15]. OGSA-DAI can be deployed within Tomcat. There are two versions of OGSA-DAI; the WSRF version of OGSA-DAI is compatible with the Globus Toolkit's implementation of WSRF, and the WSI version which can be deployed in Apache Axis [16]. One of these solutions would need to be deployed in the EnginFrame Tomcat server. A data repository, either a relational or XML database, would also need to be installed on or near to the portal. Special OGSA-DAI perform documents can be

written to add and update data in the database. These can either be passed to the OGSA-DAI server via the command-line data service client or dynamically generated via the OGSA-DAI client toolkit. In the latter case it is necessary for EnginFrame to support Java scripting within the services. As noted previously, this feature is unavailable at the time of writing, though will be included in next version of EnginFrame — due before the end of 2006.

4.3 Added Value of UNICORE and OGSA-DAI

The integration of UNICORE will reinforce the portal architecture by providing a second job submission canal. Moreover, it will increase the interoperability with the HPC resources as UNICORE is currently deployed on many different HPC systems. The integration of UNICORE will enhance the EnginFrame agent access but not the SSH plug-in based access.

OGSA-DAI appears as a very innovative service that can be used to allow users to save the inputs and outputs of their jobs in a database and to reuse them for further operations. OGSA-DAI can be coupled with the BioJava library to create complex workflows involving different bioinformatics applications.

The scope of this activity regarding UNICORE and OGSA-DAI is more a proof of concept than a production issue. The objective is to define two simple scenarios, one for DESHL UNICORE, and the other one for OGSA-DAI. These scenarios may involve only a limited number of sites.

Conclusion

The EnginFrame framework is the core technology that will be used for the DEISA Life Sciences portal. The two initial classes of applications that will be exposed as services through the portal are the alignments and sequences comparisons applications class and the phylogeny applications class. One representative application of each of these two classes will be interfaced during the pre-production period : specifically, BLAST and RAxML. As these applications will be deployed on several DEISA platforms, a submission service will be integrated on the portal to spread the load between them.

The configuration that will be deployed on the DEISA infrastructure in the coming months is a hybrid configuration where a central EnginFrame server will be linked to the submission hosts through EnginFrame agents and SSH plug-ins. The portal will authenticate users, manage the authorization, accounting and all of the job submission and management aspects. The user configuration that has been defined is a many-to-one configuration where multiple portal users are mapped to a single DEISA generic account.

The objectives of the second year of the project will be to increase the number of applications and to enhance the architecture of the portal with the integration of UNICORE and OGSA-DAI technologies.