

Eldas (Enterprise Level Data Access Services)

Aileen Campbell, Alan Gray, Brian Hills, Stephen Rutherford, Davy Virdee, Rob Baxter
Edikt, National e-Science Centre,
University of Edinburgh, Edinburgh, UK
www.edikt.org/eldas

Abstract

In this paper we present our work on Eldas - Enterprise Level Data Access Services - a middleware solution designed to provide Grid and Web data access services. This paper will describe our current release of the software, both in terms of architecture and design; the service interfaces provided and the messages used by these interfaces. This paper will also cover the web service specifications used and the tools available to create clients. Finally, the paper will present some case studies where Eldas is being applied to specific e-Science application areas.

Data Access Services for Service Grids

Eldas is an interpretation of the Grid Database Service Specification (GDSS). The GDSS is an attempt by the Data Access and Integration Working Group (DAIS-WG) to codify access to heterogeneous and distant data sources attached to a grid. Central to the GDSS is the Grid Data Service (GDS), an Open Grid Service Architecture (OGSA) based grid service for data access. Once the GDSS is finalised and accepted as a GGF recommendation, Eldas will fully adhere to the specification. Eldas provides both a grid service interface via the Open Grid Service Infrastructure (OGSI), and a Web Service interface.

Enterprise Level software

Core to the ethics of the team that produced Eldas is the belief in producing commercial quality software using industrial software engineering practices. This involves employing design techniques that facilitate robustness, scalability and performance; leveraging proven, world-class platforms such as Sun's Java 2 Enterprise Edition (J2EE) framework and the JBoss Application Server; and following best practices in software engineering.

From OGSI to WSRF

Recently OGSI has been superseded by the Web Services Resource Framework (WSRF). WSRF is intended to assimilate some of the advances made by OGSI into the wider Web Services community. The current implementation of OGSI used by Eldas is the Globus Toolkit (GT3), an open source initiative produced by the Globus Alliance. GT3 is planned to be replaced by GT4 later in 2004, and will support WSRF. Eldas will transition to this new platform once it is released. The Eldas architecture separates the functionality into two layers: the *presentation layer* which contains the interfaces, and the *business layer* which contains the data access components. This can be seen in Figure 1 where the web container is the presentation layer which supports grid and web services, and the EJB container is the business layer. This decoupling allows new standards like WSRF to be plugged-in, in a modular fashion.

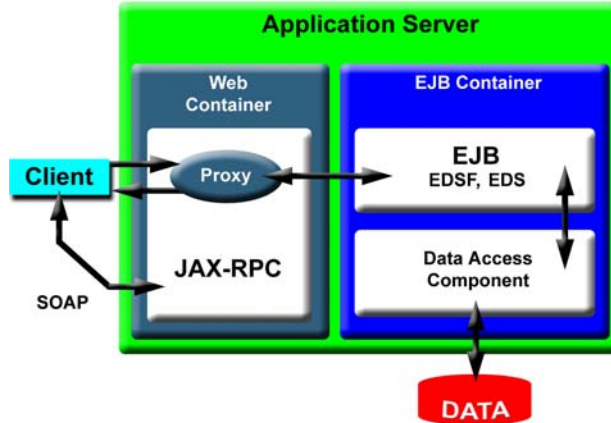


Figure 1: The Eldas Architecture.
The Proxy provides either web service or grid service interfaces to the EJB.

Service Interfaces

There are three services provided by Eldas each with their own interface:

1. A data service,
2. A data service factory,
3. A web service.

Eldas data services are transient services, intended to have finite life times and be created and destroyed as necessary by the client. The Eldas data service provides two methods: `perform`, and `performStreamResults`. Both of these methods take as input a request document specifying the data access operation to be performed, for example, a SQL query. The document also provides the name of the data source, the user name and the password supplied by the user. The `perform` method returns the results of the data access operation in one single block which for large results can overload the memory capacity of the Java Virtual Machine (JVM). The `performStreamResults` method returns smaller blocks upon each invocation with the same request document until the entire result set has been returned. This streaming approach allows much larger results sets to be accessed without overloading the JVM memory capacity. The result of such an approach is an increase in the time required to complete such end-to-end processing.

The Eldas data service also inherits the following methods from the OGSi grid service:

- `findServiceData`
- `setServiceData`
- `requestTerminationBefore`
- `requestTerminationAfter`
- `destroy`

Eldas data service factory creates instances of the Eldas data service. The factory is itself a static grid service, created when the container starts and existing until the container is shutdown. The factory inherits the same methods from the OGSi grid service as the Eldas data service but, in addition defines its own method `createService` which clients call to create an instance of an Eldas data service. In the case of the factory the `findServiceData` method is overridden to return information to the client about the data sources that can be accessed via the Eldas data service. One factory can create Eldas data services capable of talking to a multitude of different data sources depending on how the factory is configured.

Eldas web service has one method in its interface: `perform`. The web service is a standard static instance; existing for the lifetime of the container. As web services are essentially stateless

services (at least until WSRF is implemented) the Eldas web service does not support streaming of results which requires the services to remember previous invocations of the method. The Eldas web service does not accept a document for input in the same manner as the Eldas data service, rather it accepts the data source identifier, user name, password and data access operation as an array of strings. There is no semantic difference between the xml documents passed in the grid type interface and the array of strings in the web service interface. The methods exposed by the web services interface will be reviewed for future Eldas releases.

Eldas Provides two types of interface: stateful and stateless. The functional difference between the two is that the former supports streaming of query results to handle larger data-sets, and the latter does not. Conversely, the stateful interface bloats the data due to XML mark-up, whereas the stateless interface, which only uses XML for the soap messaging, keeps the data to a more manageable size.

Service Messaging

All three Eldas services have lightweight interfaces due to an adoption of a document based approach. This was dictated to an extent by the GDSS which seeks to move the complexity of a data access service interface into the message documents passed to and from the service. The benefit being a greatly reduced number of methods and port types. The corollary of this approach being that greater processing is required on the client and server side to generate and process the documents to drive the data access transactions and handle the results.

Another benefit of the document approach is the flexibility of processing the documents. In trying to provide data access services to heterogeneous data sources the meta-data pertaining to the data sources is apt to being divergent in structure. Attempting to capture the structure of the meta-data in an interface type would be not only very difficult, but also restrictive in the long term. When a client needs to access the meta-data pertaining to a data source the first point of reference is the `findServiceData` method on the Eldas data service factory. This provides the data source identifier (unique to the installation of Eldas), data type (MySQL database for example), data source URL, the driver type (JDBC for example) and the query language. From this information the client can then create an Eldas data service and query the data source meta-data. In the case of a MySQL database this will involve the use of the SQL query "show tables", to retrieve the names of the tables the user has privileges on, and "desc <<table_name>>", to retrieve the meta-data information for a specific table. The meta-data returned in these cases will be a XML document which the client can view either as is or perform basic XML processing to make more human readable.

Similar considerations regarding data structure versus interface complexity apply when performing queries on the actual data at the data source. In order to produce a flexible and extensible solution to accessing data sources, Eldas service interfaces are orthogonal to the structure of the data.

Web Service specifications

The web service specifications used are:

- W3C SOAP v1.1
- W3C Web Service Description Language (WSDL) v1.1
- Sun SOAP with Attachments API for Java (SAAJ) v1.1
- Sun Java API for XML-Based RPC (JAX-RPC) v1.0
- W3C XML Schema v1.1

These are the specifications implemented by Axis 1.1 which Eldas uses to enable the SOAP messaging for the web services interface.

At present Eldas 1.0 does not adhere to the Web Services Interoperability (WS-I) Basic Profile 1.0 (BP1). This situation will be reviewed for future releases of Eldas. Scope for using the WS-I *testing tools* for assessing conformance will also be reviewed.

Client tooling support

All third-party software used by Eldas is available for free and can be downloaded from the internet. Eldas is currently supported to run in the JBoss Application Server v3.2.2 and can access MySql databases. Other application server and databases, such as WebSphere and DB2 will be supported in the near future.

The Eldas software is split into two downloads; server and client. The server download consists of the files required to deploy Eldas onto a pre-installed JBoss server. The client download contains sample clients for each of the service interfaces (pre-compiled and source) and the relevant third-party libraries. The client download also includes a utility called the *EldasQueryTool* which is an exemplar piece and not shipped with source code. A screen shot is shown in Figure 2.

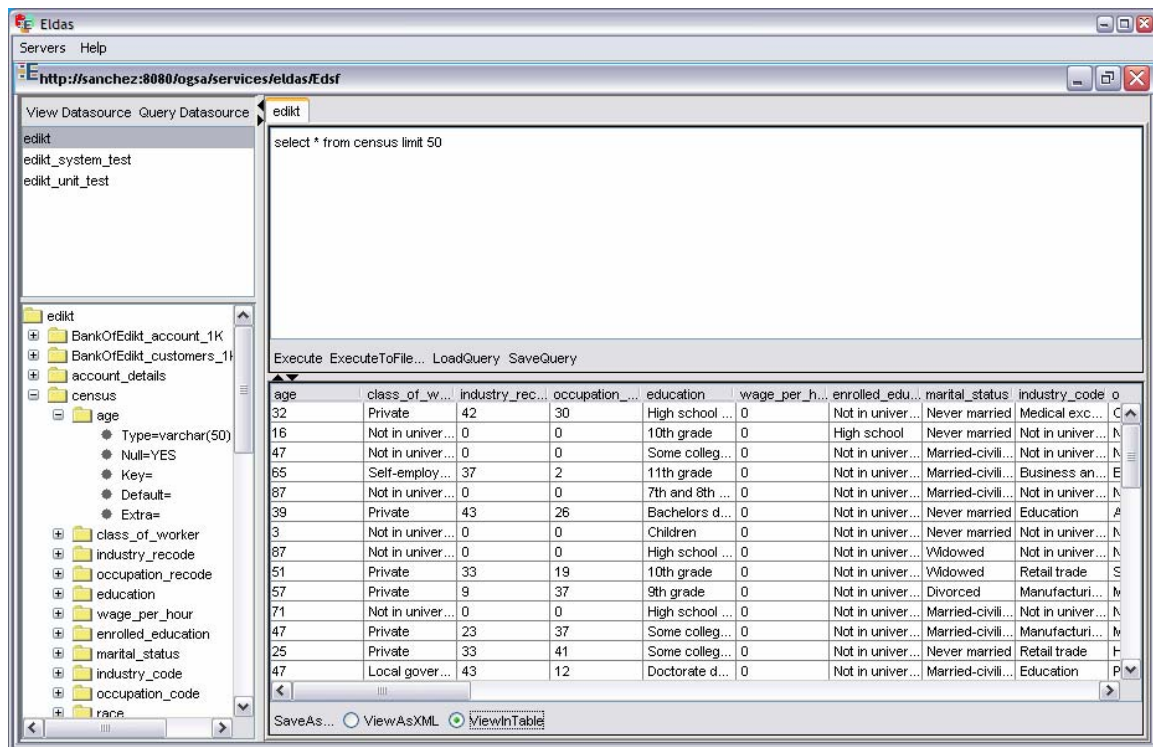


Figure 2: The EldasQueryTool

The grid clients rely on the libraries in the GT3 distribution. The web services client requires the following libraries:

- axis.jar - A SOAP engine from the Apache Software Foundation
- commons-discovery.jar & commons-logging.jar - Tools from the Jakarta Commons project
- jaxrpc.jar - Java API for XML-Based RPC from Sun Microsystems
- log4j-1.2.8.jar - A logging engine from the Apache Software Foundation
- saaj.jar - SOAP with Attachments API for Java from Sun Microsystems
- wsdl4j.jar - Web Services Description Language for Java Toolkit from IBM

Eldas 1.0 does not have a proprietary client development toolkit. Any new clients that users need will have to be built on the basis of existing web or grid service APIs that they have available or can download for free.

Application Area Support Services

Eldas will be used in various application areas including medical research, particle physics and financial services. Example projects, in the areas of bioinformatics and astronomy, are BRIDGES and EdSkyQuery-G.

BRIDGES (University of Glasgow, University of Edinburgh)

The BRIDGES project uses Grid technologies to develop and explore database integration over six geographically distributed research sites in order to provide a sophisticated infrastructure for bioinformaticians. 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 share through to private. All of this is being made as transparent as possible to the end user scientists. To this end data technologies such as Eldas and OGSA-DAI are being employed.

EdSkyQuery-G (University of Edinburgh)

The identification of observations of particular celestial objects in multiple, large (often multi Terabyte), heterogeneous databases distributed around the world, lies at the heart of the Virtual Observatory concept. The EdSkyQuery-G project aims to provide a scalable architecture for doing this through extending the ideas prototyped in the SkyQuery .NET web service developed at Johns Hopkins University. EdSkyQuery-G is being developed with future integration with AstroGrid in mind, and it will inform the specification of the International Virtual Observatory Alliance's OpenSkyQuery data integration standard.

Both of these application areas are providing motivation and requirements for the furtherance of the Eldas core functionality. This includes high demand functionality, such as performing distributed joins over heterogeneous databases. Other functionality that will be included from these applications into Eldas is: database stored procedure invocation and data transport.

Conclusions

We have shown that Eldas provides middleware to facilitate the construction of Service Grids. In particular the adoption of a document-centric approach to service interfaces allows for simpler interfaces for greater flexibility. Further we have shown that Eldas has been adopted across different eScience disciplines.