



QCDGrid2 Metadata Catalogue Web Service Specification

Project Title: QCDGrid2

Document Title: QCDGrid2 Metadata Catalogue Web Service Specification

Document Identifier: QCDGRID2-MDC-WS-SPEC_1.0

Document Filename: QCDGrid2-MDCWS-Specification.doc

Distribution Classification: Public

Authorship: Daragh Byrne, George Beckett

Approval List: QCDgrid Project Team

Distribution List: Public

Document History:

<i>Personnel</i>	<i>Date</i>	<i>Summary</i>	<i>Version</i>
DJB	29 th November 2005	First release.	1.0

Contents

1	Introduction	2
1.1	The QCDgrid project.....	2
1.2	Document purpose	2
2	Related information.....	3
2.1	ILDG web site.....	3
2.1.1	ExploreUseCase	3
2.1.2	QueryUseCase.....	4
2.2	Existing Web Service Implementations	4
2.2.1	The University of Tsukuba prototype.....	4
2.2.2	The DESY/LATFOR implementation	5
2.3	Minutes of Middleware WG meetings	5
3	Specification	5
3.1	Purpose of the metadata catalogue web service	5
3.2	Metadata web service use cases	6
3.2.1	Query ensemble metadata use case and variations.....	6
3.2.2	Query configuration metadata use case and variations	7
3.2.3	Error reporting	7
4	Testing and acceptance	7
4.1	Automated testing	7
4.2	Existing client components.....	8
5	Conclusions.....	8
6	References.....	9

1 Introduction

1.1 The QCDgrid project

Quantum Chromodynamics (QCD) is the study of the forces of nature, quantifying the complex behaviour of fundamental particles called quarks and gluons – the constituents of all nuclear matter. The UK is at the forefront of this research, with scientists from a number of academic institutions combining their efforts as a collaboration called UKQCD. The QCDgrid project was initiated in 2001, in order to help UKQCD to manage the vast amounts (Terabytes) of data that they generate on HPC resources such as QCDOC [5]. The first objective of the project is to provide a software application and supporting infrastructure that simplifies the management, storage and manipulation of this data within UKQCD. The second aim of the project is to provide technical input and software to allow UKQCD to participate in the International Lattice Data Grid (ILDG) data sharing consortium.

In the first three years of the project (2001 – 2004), software engineers at EPCC developed a software application called *QCDgrid* – a data management system that combines the distributed resources of the collaborators into a robust facility called the *UKQCD Grid*. The result is a multi-terabyte storage facility over six UK sites at: Edinburgh (including the University of Edinburgh Advanced Computing Facility), Liverpool, RAL, Southampton, and Swansea. Glasgow is also a member of the consortium.

The facility is based on commodity hardware and open-source software. The hardware consists primarily of high specification PC-based servers running the Linux operating system and managing large RAID storage arrays. On top of this infrastructure, the QCDgrid software (built with Globus Toolkit 2.4, EGEE, and an XML Database Server (XDS)) provides *Datagrid* management and user functionality – furnishing a simple and intuitive environment that hides the complexities of the underlying grid and presents a standard file system to the user. It incorporates a robustness metric that automatically disperses datasets across the grid, providing a resilience that ensures data is not affected by the loss of one (or possibly more) storage nodes.

QCDgrid allows the user to query and manipulate associated metadata using a *Metadata Catalogue Browser*. The software also provides a *Job Submission System* that allows a user to schedule computations on remote HPC systems, from the comfort of their desktop computer. Security is leveraged from the Globus Toolkit, based on digital certificates issued by the UK e-Science Certificate Authority. The result is a reliable, secure data management system.

Looking to the future, the collaboration aims to integrate the UKQCD Grid with the other collaborations that constitute the ILDG, allowing like-minded scientists around the world to share their data and benefit from the scientific progress of each others work. In order to meet this objective, the middleware working group of the ILDG has defined a set of middleware components that each collaboration should implement. These components are:

- A metadata catalogue web service, to allow information stored in local metadata catalogues to be accessed;
- A replica catalogue web service, that allows replicas of data to be located;
- (Optionally) a storage resource management (SRM) web service to allow data to be downloaded efficiently.

1.2 Document purpose

It is the intention of the ILDG middleware working group is to have a *first pass* searchable metadata catalogue web service implemented at each site by the middle of 2006. The specification of this metadata catalogue web service is the focus of this document. Specifically, this document:

- describes the understanding of the QCDgrid team with regard to the ILDG Metadata Catalogue web service component, illuminating any inconsistencies and clarifying any ambiguities;

- specifies the ILDG Metadata Catalogue web service component in sufficient detail to permit the design of the UKQCD implementation to begin.

The remainder of this document analyses the existing information relating to the metadata catalogue web service (Section 2), discusses the purpose and use cases¹ of the metadata catalogue web service component (Section 3), and discusses testing and acceptance for an individual web service component (Section 4). The conclusions of the exercise are documented in Section 5.

2 Related information

This section reviews the pre-existing information resources relevant to the specification of the metadata catalogue web service. Specifically, these resources are:

- Material published on the ILDG website [2];
- The minutes of the two middleware working group meetings in Edinburgh (October 2004) and Tsukuba (October 2005), both available from [2];
- Previous metadata catalogue web service implementations, including the prototype metadata catalogue web service implemented by the University of Tsukuba [3] and the production service implemented by the DESY/LATFOR collaboration.

These items are discussed, in turn, below.

2.1 ILDG web site

The ILDG website [2] contains a number of (slightly dated) use cases that cover ILDG middleware as a whole. Of these, two relate specifically to the use of the metadata catalogue web service. These are described below.

2.1.1 ExploreUseCase

The ExploreUseCase describes the sequence of actions undertaken by a scientist when searching for interesting data on ILDG sites.

The main success scenario in this use case is (quoted from the ILDG website):

1. The user views a web page with a list of metadata parameters that may be available for lattice data files.
2. The user selects a metadata parameter, and submits the selection to the client.
3. The client requests a list of MDS services from the ildg web services registry.
4. The client queries each MDS for the selected metadata of all files for which that MDS is the canonical repository.
5. Using the data from each MDS, the client generates a table showing each value these metadata takes, and the number of LFNs whose metadata matches that value. For example, if the metadata parameter selected were "project," a list of projects and the number of files associated with each project would be returned. If a single file matches a combination of metadata values (that is, the number of matches column in the table would be 1), the table lists the LFN as well.
6. The user selects one or more rows of the table and a new metadata parameter, and submits this selection to the client.
7. The client queries each MDS for the selected metadata values of LFN's matching the metadata values in the selected lines of the table, and generates a new web page similar to that described in step 5.
8. The user may repeat step 6, or quit.

¹ A use case is a description of how end-users will use a software application or component. It describes a task or a sequence of tasks that a user will complete using the software, and includes the responses of the software to user actions.

The role of the metadata catalogue web service in this use case provides a query interface to a local metadata catalogue. A flexible search mechanism allows different types of data to be returned – for example: raw metadata or Logical Filenames (LFNs). The client must be aware of the format for the results of the query it is issuing.

2.1.2 QueryUseCase

The QueryUseCase describes the sequence of actions undertaken by a scientist when searching for metadata matching specific query constraints. The main success scenario in this use case is (quoted from the ILDG website):

1. The user specifies what metadata is desired (either a parameter list or the full QCML) and conditions that the metadata must meet to be returned.
2. The client requests a list of MDS services from the ildg web services registry.
3. The client queries each MDS for the LFNs of data whose metadata matches the specified conditions and is originally hosted at that MDS.
4. The client queries that MDS for the metadata associated with each LFN whose metadata matched, and writes the returned QCDML to disk.

The functionality of the metadata catalogue web service, in this scenario, is a subset of that in the previous use case – specifically, only LFNs are returned.

2.2 Existing Web Service Implementations

2.2.1 The University of Tsukuba prototype

The WSDL definition for the original Tsukuba prototype is presented in [3]. The key features of this definition are:

- The definition of the `IldgMDC` portType;
- The definition of the `doEnsembleQuery` operation within the portType;
- The definition of the `doMetadataQuery` operation within the portType;
- The definition of request and response types for each of the two operations;

The operations are defined in such a way that results of queries can be “streamed”. The semantics of the operations have been defined in the middleware working group meetings and are discussed further on in this document. The request types for the operations include:

- The query format (both “SQL” or “XPath” are allowed – though initially UKQCD will only implement XPath);
- The query – as a string;
- Information about the number of results that are returned, including the start and end indices for a particular subset.

As of the October 2005 meeting, the Tsukuba WSDL facilitates the following scenarios:

“A client wishes to find an ensemble matching a certain set of physics metadata values. The client constructs an appropriate query and uses the `doEnsembleQuery` operation of the web service to identify one or more Ensemble Metadata Documents (EMDs) that match the query constraints. The raw XML of the EMDs is returned as an array of strings (one per EMD).”

“A client wishes to find configuration metadata. The client constructs an appropriate query — the query may be parameterised on such things as: physics metadata values, ensemble ID (perhaps the result of an ensemble query), user/creation information, etc. The form of the data returned to the client is dependent on the form of the query submitted to the web service. For example, if the client will proceed to download one or more configurations, then the query must be constructed in such a way as to return the LFN for each matching configuration.”

Prior to the October 2005 meeting, the definition of the Tsukuba WSDL specified that a query returned a set of matching LFNs. However, discussion at the October 2005 meeting determined that the result of a query should be generalised to an array of strings containing fragments of XML documents. The current Tsukuba WSDL proposal needs to be updated to reflect this change.

2.2.2 The DESY/LATFOR implementation

The DESY/LATFOR web service that implements the ILDG metadata catalogue web service functionality has an extended portType that offers supplementary DESY/LATFOR-specific operations. In theory, it should be possible to use this service in exactly the same manner as the Tsukuba service. However, it appears that the DESY/LATFOR service as it stands may be incompatible with the Tsukuba service. One incompatibility pertains to the return types specified in the WSDL. Both the Tsukuba and DESY/LATFOR WSDL return: the number of LFNs, a status code, start index, total number of results, and list of LFNs. However, the Tsukuba WSDL also includes: timestamp, query format information, and the query itself in the returned XML. Since the minutes of the Middleware Working Group 2005 meetings state that the WSDL to be used is that specified by the Tsukuba team, UKQCD currently plan to use this version.

2.3 Minutes of Middleware WG meetings

The minutes document the consensus reached (plus any identified obstacles) regarding the interface for the metadata catalogue web service. The main points raised during the first meeting (Edinburgh 2004) were:

- The service consists of a set of operations for querying a metadata catalogue for metadata relating to configurations;
- Queries may be issued against ensemble and configuration metadata separately;
- There was an issue as to what a queries should be permitted to return. One option is to require a query to return a set of LFNs that refer to metadata documents matching the query, which can then be downloaded. An alternative approach allows a query to return entire or partial metadata documents as and array of strings. This discussion was continued at the subsequent meeting in Tsukuba.

The proposed web service interface permits results to be “streamed” in batches, achieved through the specification of a start index and a maximum number of results in the operation signatures. This reduces the likelihood that a service could become overloaded. Optionally, a service may maximise performance by caching the results of a large query, rather than re-running the query for each batch request.

An unresolved issue pertaining to the streaming of query data is the concept of requesting “all results”. As an example, one could specify “start index=0” and “maximum number of results=-1”. Alternatively, it could be left to the particular client to specify a large enough maximum value to accommodate the complete set of results.

3 Specification

In this section we condense the above discussion into a specification for the ILDG Metadata Catalogue web service. The purpose of the component is defined, based on the above discussions. The use cases that the component facilitates/participates in are then formalised. Finally, key features that a component implementation must satisfy, to be accepted by the ILDG system, are discussed.

3.1 Purpose of the metadata catalogue web service

In general, data access layers in a software system can be characterised as “generic” or “domain-specific”:

- **Generic layers** permit access to raw data from the data source (for example, raw XML or database rows): an example of such a technology is JDBC [4]. This approach has the

advantage that any data which is required may be accessed, provided that the client has sufficient knowledge of how to structure an appropriate query.

- In contrast, a **domain-specific layer** separates the data source from the application layer. For example, in an object-oriented application – where there is a concept of a user – raw data (the result of using a generic layer, e.g. query on a database) is mapped into, say, *User* objects by a data access layer. The clients of the mapping layer do not need to know anything about the structure or mechanisms of the data source, and interact with objects that are more appropriate to the application.

Implementing a domain-specific layer is generally more involved than implementing a generic layer. However, it does have an important advantage of separating the data source from the client application, reducing interdependence between components and simplifying subsequent upgrades to or replacement of these components.

In the ILDG, the preference has been to adopt a hybrid approach. A truly generic approach would involve a single, simple `doQuery` operation. However, the current proposal is to offer two separate query methods that distinguish between the two different types of data – this implies the presence of domain knowledge in the definition of this interface. However, it is not purely a domain level interface, as each operation accepts an arbitrary query. This is appropriate, provided that the implementers recognise the following points:

1. The implementation of an operation need to be aware of the “target” of the operation. For example, the `doEnsembleQuery` operation must ensure that it queries only ensemble metadata documents.
2. A client must incorporate an understanding of the form of the query results that are returned, and of the meaning of the queries they issue.
3. As the relevant domain operations/abstractions become clearer (that is, the most widely used queries and the most suitable way of presenting responses to the client), we can attempt to encapsulate these in a manner that is reusable by all ILDG members;
4. The value of the metadata web service component is in providing a uniform interface to client applications that wish to carry out arbitrary queries on metadata. The current proposal provides a foundation level for this that can be extended in subsequent revisions. This would be less of an issue if the query language allowed such restrictions to be expressed. However, while SQL allows the specification of tables from which results should be selected, XPath does not allow the document types against which the query operates to be specified.

3.2 Metadata web service use cases

There are essentially two major use cases for the metadata web service, corresponding to each of the two query operations discussed above. The successful outcome of each is examined here, alongside the failure cases. In the remainder of this section, “client” is assumed to mean a piece of software – the metadata catalogue web service is assumed to be mainly consumed by other programs.

3.2.1 Query ensemble metadata use case and variations

The most basic form of this use case can be expressed as follows:

“A client wishes to find one or more ensembles matching a certain set of physics metadata values. The client constructs an appropriate XPath query and uses the `doEnsembleQuery` operation of the web service to locate any EMDs that match the query constraints. The raw XML of the EMDs is returned as an array of strings.”

Another variation is possible:

“A client wishes to find the ensemble IDs of ensembles that match a certain set of physics metadata values. The client constructs an appropriate XPath query and uses the `doEnsembleQuery` operation of the web service to search for the names of ensembles that match the physics metadata values. The ensembleIDs are returned as an array of XML fragments.”

For the second case, it is assumed that the XPath query is constructed in such a way as to produce an array of “<markovChainURI>http://some.identifier/</markovChainURI>” elements, which are then parsed by the client. Note that it is responsibility of the client to parse the results.

The next variation illustrates a query for a piece of ensemble metadata when the ensemble ID is known:

“A client wishes to find some metadata (for example, the creator and creating institution) of a particular ensemble whose ID is known. The client constructs an appropriate XPath query, parameterised with the ensemble ID, and uses the doEnsembleQuery operation of the web service to search for the additional metadata. The additional metadata is returned as an array of XML fragments.”

As with the previous usage case, the client is responsible for parsing the results of the query appropriately.

3.2.2 Query configuration metadata use case and variations

Typically, a client will wish to retrieve LFNs from the metadata catalogue. Other possible usage examples include gathering aggregate information about the metadata stored in a catalogue (for example, numbers of documents) and obtaining other arbitrary metadata.

The following use case summarises all potential situations that are currently anticipated.

“A client wishes to find configuration metadata. The client constructs an appropriate query. The query may be parameterised on such things as: physics metadata values, ensemble ID (perhaps the result of an ensemble query), user/creation information, and so on. The data returned to the client is dependent on the form of the query. In order for the client to download configurations, the query must be constructed in such a way as to include LFNs for the configuration, although there is no constraint that this must be done. Upon successful query execution, the results are returned to the client as an array of strings containing XML fragments. These strings are then parsed by the client.”

It is anticipated that components which make it easier to parse client XML will be developed.

3.2.3 Error reporting

At the time of writing, no fault types have been defined in the Tsukuba prototype WSDL. The same is true of the current LATFOR implementation. Suitable error reporting needs to be included at some stage, since there are scenarios where the standard Web Service RemoteException will not be sufficient for reporting errors. At the very least, it should be possible to tell the client that their query is malformed.

4 Testing and acceptance

4.1 Automated testing

Broadly speaking, for an implementation of the metadata catalogue web service to be accepted into the ILDG, it should behave in the same manner as every other implementation. Since every site will have its own, different, data, it will be difficult to define an automated test that will depend on the data present in a catalogue. Two possible strategies to address this issue are explored.

- **Standard test data** – It should be possible to produce a set of test data with known values. The results of any query on this test data can be computed a priori, making it possible to validate the behaviour of a particular service instance that uses a back-end containing this data. The result of a query can be compared to the predicted value when such a data set is used.
- **Results form testing** – Usually in an automated test, a part of a system is operated (for example, a method is called on an object) and the state of the system after invocation is

compared against some expected value (for example, the return value of the method is compared against the expected value). As noted above, it is impossible to control or know about the data behind a particular metadata catalogue web service implementation. However, it should be possible to build a set of automated tests that examine the form of the results obtained by queries against any of the metadata catalogue web services in the ILDG. The fact that the results are returned as strings means that regular expressions may be used to check that the form of the results is accurate. For example, it could be checked that each query result matches the string “<dataLFN>*</dataLFN>” where * is a wildcard.

It is anticipated that UKQCD will favour standard test data during development (that is, known database content) to test the candidate implementation. If appropriate, suitable results form tests may also be used.

4.2 Existing client components

It should be possible to use a particular metadata catalogue web service instance within an existing client application, without experiencing significant problems. At present, there are two client applications available: the LATFOR data tools and the example Java classes provided by the team at the University of Tsukuba. At the very least, we would expect the UKQCD implementation of the web service to work with the Tsukuba client code (subject to elimination of the anomalies described above). However, it is not clear whether the UKQCD web service can be expected to work with the LATFOR client code, as this code’s functionality is much more extensive than simply querying the metadata catalogue. Even with this in mind, the possibility of using modules of the LATFOR code in testing will be investigated during the design and development phases.

5 Conclusions

The overall purpose of the metadata catalogue web service can be summarised as:

To provide a well-defined mechanism that allows clients to execute queries against the metadata catalogue at one or more ILDG sites.

The form of the metadata catalogue web service interface will facilitate client applications to interact with multiple web service implementations in a well-understood manner. Furthermore, the functionality of the web service will accommodate the use cases that are defined in Section 3.

The following list of open issues remains at this time:

- There are a number of discrepancies between existing prototypes for the WSDL description of the metadata catalogue that need to be resolved. Furthermore, a mechanism for reporting that the user has submitted an invalid query also needs to be defined. These issues will be resolved by discussion on the ILDG mailing list. The results of this discussion will be used to propose a new version of the metadata catalogue web service WSDL, against which the UKQCD collaboration will design and build their implementation of the metadata catalogue component.
- The issue of how to specify that a service call is to return all results from a query will also be resolved by e-mail discussion.
- The QCDgrid will test its implementation using the “known data” approach described in Section 4. Testing techniques that are found useful during implementation will be shared, as they become available, with other ILDG members.

6 References

- [1] QCDGrid: Probing the building blocks of matter with the power of the Grid, QCDgrid Project Homepage available on-line at <http://www.gridpp.ac.uk/qcdgrid/>.
- [2] The ILDG Wiki <http://www.lqcd.org/ildg/tiki-index.php>
- [3] Tsukuba prototype web services <http://www.lqa.ces.tsukuba.ac.jp/WS/>
- [4] JDBC <http://java.sun.com/products/jdbc/>
- [5] [P.A. Boyle](#), [D. Chen](#), [N.H. Christ](#), [M. Clark](#), [S.D. Cohen](#), [C. Cristian](#), [Z. Dong](#), [A. Gara](#), [B. Joo](#), [C. Jung](#), [C. Kim](#), [L. Levkova](#), [X. Liao](#), [G. Liu](#), [R.D. Mawhinney](#), [S. Ohta](#), [K. Petrov](#), [T. Wettig](#), [A. Yamaguchi](#), *Hardware and software status of QCDOC*, Reference hep-lat/0309096.