A Web Service Model for Climate Data Access on the Grid

Andrew Woolf*, Keith Haines and Chunlei Liu
Environmental Systems Science Centre,
University of Reading,
RG6 6AL, UK
{awo,kh,cll}@mail.nerc-essc.ac.uk

Revised March 7, 2003

International Journal of High Performance Computing Applications
special issue on "Grid Computing"

*Now at Rutherford Appleton Laboratory, a.woolf@rl.ac.uk

Running head Climate Data Access on the Grid

Author one (corresponding author):

Name	Dr Andrew Woolf
Address	Rutherford Appleton Laboratory, Chilton, Didcot Oxon OX11 0QX
Phone	(01235) 778027
Fax	(01235) 445945
Email	a.woolf@rl.ac.uk

Author two:

Name	Prof Keith Haines
Address	Environmental Systems Science Centre, University of Reading Reading Berks RG6 6AL
Phone	(0118) 931 8742
Fax	(0118) 931 6413
Email	kh@mail.nerc-essc.ac.uk

Author three:

Name	Dr Chunlei Liu
Address	Environmental Systems Science Centre, University of Reading Reading Berks RG6 6AL
Phone	(0118) 931 8741
Fax	(0118) 931 6413
Email	cll@mail.nerc-essc.ac.uk

Abstract

The problem of sharing environmental and climate data (from measurements or models) across networks does not yet have a standard solution. *Ad-hoc* approaches are common, with complications arising through a number of different file formats and conventions. The state-of-the-art in the climate research community is the Distributed Oceanographic Data System (DODS, also "OPeNDAP"), which maps a file or aggregation of files onto a URL. Data subsets may be retrieved, and limited (and non-standard) metadata queries made. Data is abstracted from storage only to a limited degree. We present an alternate data access mechanism (the Grid Access Data Service, GADS) better suited to Grid applications. Requirements of data abstracted from storage, rich metadata models, flexible delivery, security, and orchestrated workflows all suggest a Web Service solution. The GADS Web Service has two operations: a querying mechanism, and a request mechanism. Three metadata models are required: one for the abstract representation of climate data, one for characterising data usage, and one for describing data storage artifacts. A compatibility interface has been layered on the Web Service to provide DODS/OPeNDAP functionality. A visualisation web portal has also been built to interface with GADS to demonstrate extendable functionality.

1 Climate Data Access on the Grid

Environment and climate scientists have a need to access large geographically and temporally varying data sets, often from a variety of sources, in order to carry out their research. Agencies such as the UK Met Office or Environment Agency, and companies operating in environmental science, often have an additional need to access data in real time, including increasingly real time satellite data. Often archives of historical observations or analyses of spatially gridded fields such as temperature or pressure, are stored in flat files, each representing data at a different time. However it is common for researchers to need to access spatial subsets of these data from many different time periods (files) simultaneously. Add to this the need to make comparisons between observations and results from many model experiments (as for the IPCC report [11]) and it is clear that a great deal of time and effort is expended in data handling. The environment and climate research community are therefore in an excellent position to benefit from a Grid solution to their data handling problems.

An example of an important community dataset is the European Centre for Medium-Range Weather Forecasting (ECMWF) 40-year reanalysis of recent atmospheric conditions, ERA-40[1], currently being prepared based on all available data and the most modern assimilation methods. The resulting dataset will be several tens of terabytes in size [18] and is likely to be accessed by thousands of users. Another important reference dataset is that collected in the most ambitious oceanographic experiment to date, the World Ocean Circulation Experiment[2] (WOCE) [17]. Over 1990–2002, intensive internationally coordinated observations have produced a global ocean dataset containing tens of thousands of full-depth profiles of various water properties, and almost a million upper-ocean only temperature profiles. In addition, satellite measurements of surface temperature, winds and sea-level have been collated over the WOCE period. The data have been distributed on a pair of DVDs but are still being corrected and checked and the most up-to-date version will only be available through the internet [3] using the DODS/OPeNDAP protocol to be discussed below.

Apart from large reference datasets, many climate scientists, such as those at the Hadley Centre, are now generating terabytes of data from new computer models which can only be effectively studied by a team of collaborators. Such coordinated research is much more effective in a distributed Grid environment.

A number of projects are investigating different elements of the distributed data access problem. THREDDS (Thematic Real-time Environmental Distributed Data Services) [5] is developing software tools and federated catalogue systems to enable access to a broad range of distributed environmental data, including local files, and remote data accessible via DODS/OPeNDAP or ADDE[3]. The project is strongly motivated by educational objectives. The Earth

[1] See project documentation at `http://www.ecmwf.int/research/era`.

[2] See `http://www.woce.org`.

[3] The Abstract Data Distribution Environment (ADDE) is a data transfer protocol used by the McIDAS system. This is a bespoke network of datasites and software for sharing and analysing primarily remotely-sensed meteorological data.

System Grid (http://www.earthsystemgrid.org) aims at using Grid technology to facilitate access to petabytes of distributed climate simulation data for cutting-edge research purposes. As part of the project, the DODS/OPeNDAP protocol for distributed data access is being extended to use a GridFTP [1] transport layer. The NERC DataGrid (http://ndg.badc.rl.ac.uk/) is a new (September, 2002) UK project aiming to integrate data holdings of the British Oceanographic Data Centre and the British Atmospheric Data Centre, as well as those registered by individual research groups, into a unified Grid context with tools for data discovery, delivery and use. The infrastructure developed will interoperate with the Earth System Grid. International projects examining specific distributed data access issues in the climate sciences include GODAE [19] and UNIDART [6]. Finally, the European Space Agency recently held a workshop[4] to examine the role of Grid technology in its activities. To the best of our knowledge, none of these projects has proposed a specific alternative to DODS/OPeNDAP as a climate data access solution for the Grid.

This paper focuses on the problem of accessing climate data from distributed data files. The implementation discussed is based on gridded data produced by climate models but the issues raised and approach proposed is applicable also to observational data. Section 2 reviews the currently prevalent software tools which are being used by the climate community for interactive and distributed data access and looks at their strengths and weaknesses. Section 3 discusses a new approach to climate data access on the Grid and presents a Web Service model. Section 4 describes the implementation, GADS (Grid Access Data Service), within the restricted example of ocean model data. Section 5 presents an interactive visualisation web client, based on the Live Access Server [15], which uses GADS. Section 6 outlines directions for further development in this field.

2 Distributed data access: state of the art in the climate research community

Although the climate research community is beginning to take advantage of internet enabled data exchange, networked data access remains constrained by storage artifacts. Data are stored in discrete files and therefore tend to be indexed and transferred in the same way, and consequently require a plethora of software APIs to read different file formats[5]. Data must be interpreted according to any of a number of conventions[6] that may have been used (or, worse, none at all). Apart from a number of national and international data centres, data providers usually have poor, and non-standard, metadata accompanying the data. In addition, the use of traditional transport protocols (FTP and HTTP) limit security models for access control and thus discourage agency providers, who often sell some of their real time products, from interacting with the wider

[4] See the Workshop webpage at http://esagrid.esa.int/activities/ESAGrid2002.htm.

[5] The most commonly used file formats include netCDF [27], HDF [13] and GRIB [29].

[6] Data conventions include, for example, COARDS and CF [26] for gridded data, and the WOCE conventions [28] for profile measurements.

research community. All of these factors combine to make the use of data often more difficult than its discovery.

A networked data access protocol has been developed by the climate science community — the Distributed Oceanographic Data System (DODS) [25]. This system was developed seven years ago now [9], and was well ahead of its time, but typically it is only now becoming widely used as interactive software interfaces are starting to be developed on top of it. However, because it was developed early on it does not take advantage fully of the possibilities which can now be seen for data access on the Grid. It is this DODS technology, recently renamed OPeNDAP (the Open-source Project for a Network Data Access Protocol), for which an alternative is proposed here, and so we review below some of its salient features relevant to data access on the Grid.

DODS/OPeNDAP provides a URL-based mechanism for accessing the contents of data files via HTTP. While the original DODS servers provided a one-to-one mapping from individual files to URLs, a recent release allows aggregation of files into logical datasets. A URL for the HTTP protocol may take the general form [2]:

`http://<host>:<port>/<path>?<searchpart>`

DODS/OPeNDAP uses the *<path>* to specify the data file (or aggregation), and the *<searchpart>* to specify a variable name within the file (which, of course, will not necessarily use any consistent naming scheme). Subsetting indices may be specified for a variable, and one of three objects requested: the Data Descriptor Structure (DDS), the Data Attribute Structure (DAS), or the DODS object. The first two are forms of metadata relating to the "shape" and attributes respectively of the variable [22, 23], while the third provides the data themselves. Software libraries are available which use the netCDF API[27] to read DODS-enabled datasets.

DODS/OPeNDAP has been an ambitious and successful project. However, it retains a number of features which limit its usefulness for data access on the Grid.

A major problem is simply that any metadata query or data request must be cast, with very limited semantics, onto the syntax for URLs and is thus unable to exploit the generality available through use of XML and SOAP messaging. The metadata returned in a DDS or DAS is in a bespoke ASCII format that is not easily parsed. It implements a limited data model and has no provision for extensibility. Similarly, the binary DODS data object is in a bespoke format, adding yet another data format to the pantheon already in use in the climate research community. There is no flexibility in the mode or format of delivery. As a result of these bespoke formats, access to DODS-enabled data in practice relies heavily on the "DODS Libraries", which provide a netCDF-equivalent API for accessing DODS-enabled files. Thus, DODS/OPeNDAP is most useful for accessing datasets which map well onto netCDF data structures. It is unlikely, for instance, to prove an ideal method to access tracked satellite data, which may contain a single variable indexed by data-model-specific parameters such as "repeat cycle" and "pass number". The API also introduces redundancy since both the URL label and the netCDF API separately allow subsetting operations.

Limitations were recognised in early design documents, with the following five deficiencies explicitly acknowledged [24]:

1. Poor support for version information
2. No formal support for quality rating of data sets
3. No support for location-independent naming
4. No support for binding extent information within the name-space (e.g., for how long will this data set exist)
5. No support for security information

Perhaps the most serious problem with DODS/OPeNDAP for data access on a Grid is simply that it is incompatible with the Web Service and Grid Service standards upon which Grid applications will be based. By contrast, we demonstrate in section 4.5 backwards compatibility of the Web Service developed here, by layering on top of it a DODS/OPeNDAP server.

The DODS/OPeNDAP data service has become much more widely used only recently with the development of the Live Access Server (LAS [15]) data browse tool. This freeware, developed by the Pacific Marine Environmental Laboratory of NOAA, provides web-based interactive visualisation of gridded climate data, including DODS-accessible remote datasets. It uses a conventional three-tier architecture, with a web browser user-interface formulating product requests that are translated by middleware into operations for a backend visualisation package. Figure 1 shows the appearance of LAS in a web browser. The user may choose a particular variable from any of a number of datasets, and then select a time and region of interest. A graphic is then generated for the user's choice using the backend visualisation application (Ferret, [14]) which is able to read DODS-enabled datasets. This interactive visualisation service has proved so successful that many important climate data sets can now be accessed on the internet in this way from a number of LAS server sites, nearly all still in the US. We will not discuss this aspect of data serving in detail here but in Section 5 we describe a similar interactive visualisation client which uses Web Service data access since this is a clear need for the climate community.

3 A Web Service Model for Data Access

3.1 Data Access Requirements

A number of problems were identified earlier with the DODS/OPeNDAP protocol for accessing climate data on a Grid. We list below desirable features for an alternative mechanism before presenting a Web Service solution.

1. **Data abstraction from storage.** Users need to access data, not files. Logical data has an existence independent of file formats, locations, and internal structure.

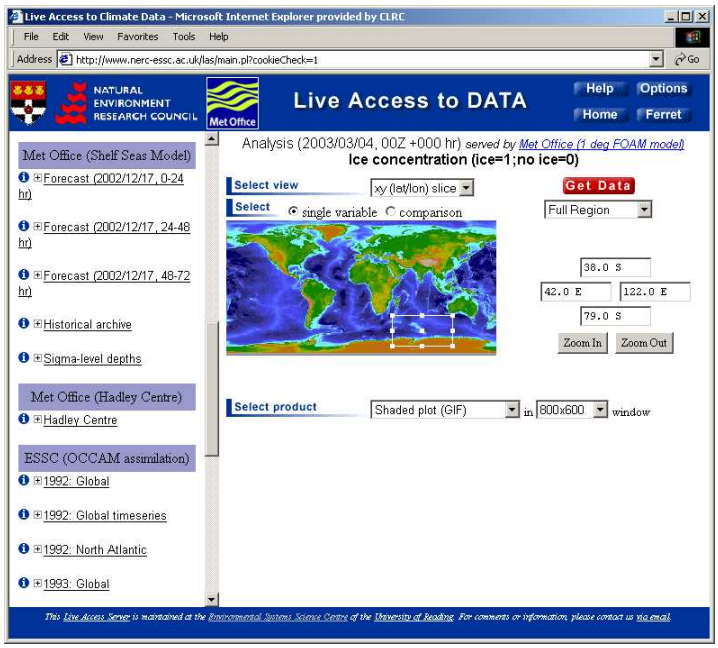

Figure 1: Live Access Server appearance

2. **Standard and extensible metadata.** The data access mechanism should be able to handle a wide variety of different data types, and should be extensible. Queries against data holdings, and data requests, should be possible in standard and useful ways.

3. **Flexible delivery.** Data should be delivered in whatever form is required.

4. **Security.** A security model is needed, able to limit access to metadata or data for individual users or groups as required.

5. **Grid compatibility.** Any new data access mechanism needs to be compatible with the standards upon which Grid technology is being built, if it is to be easily incorporated into the workflow scenarios (including federation) that will be possible.

6. **Backwards compatibility.** Migration to Grid technologies will not be immediate, so alternative access paths should be maintained.

3.2 A Web Service solution

The requirements outlined above indicate a highly-developed and capable data access *service*. By this is meant a mechanism capable of servicing requests which are limited only by the natural characteristics of data, and usage requirements, and not by implementation-imposed constraints of storage or transport

technology. *Web Services* [20] provide just the framework required. These are network-accessible computational services. Remote invocation messages and parameters are encoded with the Simple Object Access Protocol (SOAP), and may be sent over any of a variety of transport protocols (HTTP, SMTP, TLS). Complex input and output objects may be encoded through the use of XML schemas [7]. There are well-defined mechanisms for describing [4] and publishing (`http://www.uddi.org`) Web Services. Mechanisms have also been proposed for orchestrating Web Services into complex workflows[7]. Furthermore, Web Services have been proposed as the foundation for Grid Services under the Open Grid Service Architecture [8, 21]. Rich security models for Web and Grid Services are also under active development [10, 12, 16].

There are two fundamental operations that must be supported by a data access Web Service: querying data holdings, and requesting data products; these functions are provided in varying degrees by current data access mechanisms. Both operations should operate on logical data, with no dependence on physical storage details. Given sufficiently rich implementations, these two operations could support the entire spectrum of data usage scenarios, from search and discovery through complex processing. We therefore propose the two Web Service operations, `dataQuery` and `dataRequest`.

3.2.1 `dataQuery` operation

The scope of the `dataQuery` operation is broad indeed. It must support a range of query types from high-level search and discovery ("Which datasets are available?"), through intermediate levels ("Which physical parameters are contained within the *ERA-40* dataset?", "Which datasets contain deep *temperature* measurements in the *South Pacific* during *1996*?"), to detailed queries about data structures ("What are the dimensions and coordinates of the *salinity* parameter contained in the *ERA-40* dataset?"). The `dataQuery` operation provides a view onto the range of metadata for the data holdings. To fully characterise the domain of the `dataQuery` operation (and thereby to design the input parameter schemas) would require a general metadata model for climate data. It must include everything from data provenance information to object models for different data sources (remote-sensed satellite imagery, atmospheric sounding measurements, output from general circulation models, etc). Such a project is beyond the scope of this paper, but the problem may be solved piecemeal. As mentioned earlier, Web Service input and output parameters are encoded using XML schemas which may model complex objects and are extensible. Different XML namespaces may be used for data models that are inappropriate or too cumbersome to merge into a single schema. The benefits of developing a metadata model for even a small subset of climate data (say output from General Circulation Models (GCMs)) would be substantial.

[7] See, for example, the Business Process Execution Language for Web Services, `http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/`

3.2.2 `dataRequest` operation

The parameters of a data request contain two key pieces of information. The first must identify unambiguously just those data which are required, the second must describe in what form they are required. That is, a data request must identify the logical data required, as well as a specific instantiation of a data object containing them. The query mechanism in the `dataQuery` operation will suffice for the first of these — a schema capable of representing queries at various levels of a data hierarchy must be able to identify specific parts of that hierarchy.

To characterise the data object instantiation, a metadata model is needed for *data use case scenarios*. Controlled vocabularies and schema are required to describe the various ways in which a client may wish to use the data. It should enable a mapping from logical data to a concrete instance. Clearly, this must include at least the object (file format and/or delivery method) in which the data should be instantiated, including semantic characteristics (such as relative axes order for gridded data, or conventions that should be applied). It might, in general, include also limited processing instructions, for instance subsetting or re-gridding, or more advanced server-side calculations. A choice must be made on enabling sophisticated processing at the data source, on the one hand, as against delivering data to further Web Services in a distributed processing workflow on the other.

3.2.3 Data service management

The data access Web Service may be regarded as an 'engine' for mapping stored data onto a data query or request. Two operations are exposed, and two required metadata models have been described. A third metadata model is needed — for characterising the stored data in such a manner that these mappings may be made. It must enable the logical datasets provided by the service to be described by the range of storage artifacts. This 'internal' metadata achieves much of the abstraction of the data away from the storage, hiding details like file locations and formats, internal variable names, etc, casting these instead onto the logical dataset of which they are part.

In summary, there are three fundamental metadata models required for a general data access Web Service: one to model scientific data, one to model the manner in which the data is to be used, and one to describe data storage details. A complete solution would demand comprehensive XML schemas (and controlled vocabularies) for each. Such a complete solution is beyond the scope of this paper, but must remain an aspiration if the *Semantic Grid*[8] is to be realised and sophisticated composition of Grid Services possible.

[8] `http://www.semanticgrid.org`

3.2.4 Grid compatibility and security

It is important that any climate data access service should conform to Grid Service standards as they emerge, since the large data volumes mean that Grid computing methods will be very valuable to the climate community. It is clear that these will leverage strongly off Web Service standards, and here we present initial ideas on Grid compatibility and security for GADS.

Grid Services [21] extend Web Services in a crucial respect by adding statefulness. A transient, stateful Grid Service is instantiated, and referenced through its *Grid Service Handle*. Its state is exposed through public `serviceData` elements that may be queried with the `findServiceData` operation of the *`GridService`* `portType`. GADS could for instance be extended to contain service state information, thus allowing for 'meta-queries' about the service: current server load and available cache size for instance. The Open Grid Services Infrastructure (OGSI) [21] allows operations to be defined with extensible parameters, providing a mechanism for extending the `dataQuery` and `dataRequest` operations to handle further metadata models as they're developed. The Web Service model presented here introduces explicitly the notion of data access as a service, with potential latency involved in a response, eg. if a large number of files or offline storage instruments were involved. As a Grid Service, notification callbacks could be used (through the *`NotificationSource`* `portType`) to alert the client when the requested data became available.

Grid Service security [12, 16] and Web Service security specifications [10] comprise a sophisticated and comprehensive set of solutions to the issues of authentication, authorisation, privacy, trust, and delegation that arise in a loosely-coupled cross-domain Grid infrastructure. The full complexity of this framework is unlikely to be required here. The primary requirement is to guarantee authorised access to metadata (`dataQuery`) and data (`dataRequest`) for identified groups. An immediate solution would be simply to include in the metadata model for data additional attributes specifying access rights. It would fall to the Web Service to enforce these policies based on claims asserted by a client. This should be possible using the *WS-Security* (to attach security tokens to SOAP messages) and *WS-SecureConversation* (to authenticate message exchanges) specifications.

4 The Grid Access Data Service (GADS)

The previous section proposed a general Web Service model for accessing climate data on the Grid. The essential features of the model are:

- The `dataQuery` operation for making enquiries about data holdings
- The `dataRequest` operation for obtaining data
- Metadata models, for:
    1. abstract representation of scientific data

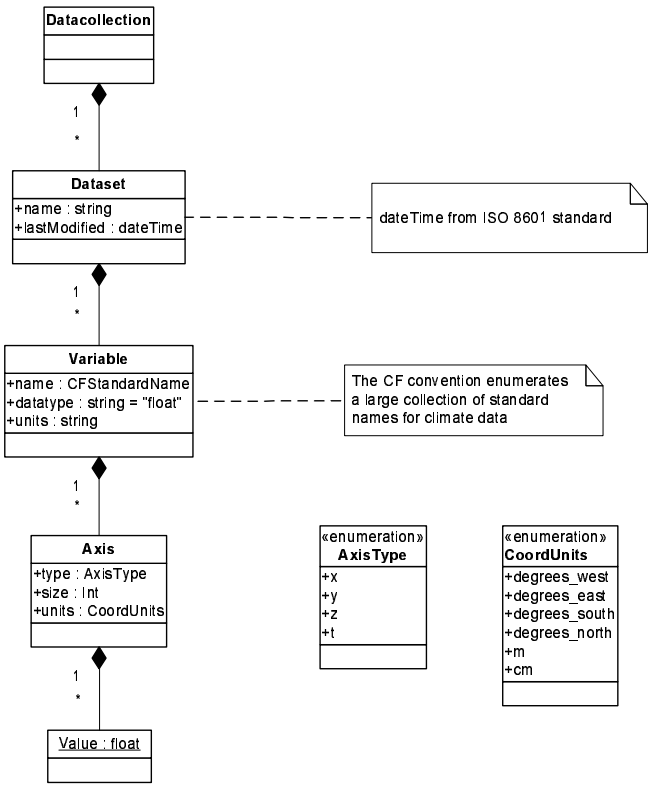

Figure 2: UML class diagram for metadata model of gridded data

2. representing data usage scenarios
3. describing data storage instances

A limited preliminary implementation of such a Web Service has been prototyped and is described in this section. The design is sufficient to encompass a range of gridded model data including those available from the UK Met Office analyses which are of wide interest. The prototype has proven robust and demonstrated the utility of the model. It has been incorporated into a prototype visualisation service described in Section 5.

4.1 Metadata model for scientific data

The Web Service was designed to provide access to time sequences of spatial data on a Cartesian grid, such as is produced by finite-difference general circulation models of the atmosphere or ocean. A metadata model for representing such data is shown (in UML notation) in Figure 2. There is a hierarchy of levels, each

of which is the composition of a number of objects at the next lower level. This model is equivalent to that supported by the Live Access Server visualisation browser, and is also sufficient to represent data stored in COARDS-compliant netCDF files.

4.2 The `dataQuery` operation

The metadata model defined above represents the structure within which queries and data requests may be made. An XML schema corresponding to the model is presented in Appendix C. Any collection of data consistent with this model would be described by a document which would validate against this schema. Meaningful data queries could then be regarded as interrogations against such a document instance and could be made at many levels of complexity. The XML Query Language, XQuery[9], is one attempt to develop a general-purpose XML querying mechanism. However this initial GADS implementation is far simpler and does not use the full generality of the Appendix C schema in its operations.

The current Web Service `dataQuery` operation allows the data holdings to be queried only at successively deeper levels in the hierarchy. Thus a list of all the datasets may be retrieved, or a given dataset may be queried for its component variables, or a given variable may be queried to discover its structure (component axes); finally, a given axis of some variable may be queried to find the numerical coordinate values. For this simple case, the `dataQuery` operation takes three string arguments representing the names at each level of the hierarchy:
`dataQuery(`*`dataset:String, variable:String, axis:String`*`):String`.
Any subset of the trailing arguments may be empty. The response provides the unique XML nodes selected by the query parameters. An example is provided in Appendix B.

This simplified implementation of the web service uses RPC-style SOAP messaging, and is described in the WSDL document shown in Appendix D. The WSDL description for this prototype has not been published into a public registry at this stage.

Although this implementation avoids the complexity of document-style SOAP messaging, it also limits the generality that might otherwise be possible. An alternative version would allow a more general XML query, based on the schema in Appendix C, and retrieve all matching branches of the metadata document. This would greatly enhance the flexibility of the system, allowing to find all datasets containing a specified variable, for instance, and/or data from a specific geographic location. A query for sea-surface temperature data in the equatorial Pacific would thus take the form:

```
<?xml version="1.0" encoding="UTF-8"?>
<datacollection>
 <dataset>
  <variable name="sea_surface_temperature">
   <axis type="x" units="degrees_east">
```

[9] `http://www.w3.org/TR/xquery`

```
    <value>125</value>
    <value>280</value>
   </axis>
   <axis type="y" units="degrees_north">
    <value>-10</value>
    <value>10</value>
   </axis>
  </variable>
 </dataset>
</datacollection
```

4.3 Data usage metadata, and the `dataRequest` operation

The implemented data usage model was limited to the following three characteristics:

1. File format for data to be delivered in: either raw binary, netCDF, or HDF.
2. Subsetting parameters, using the start index, stride, and count along each axis.
3. Relative order of axes.

Once more, XML schema could have been used to encode such a data usage model, but a simpler solution for this limited implementation was to provide a fixed number of parameters to the `dataRequest` operation in an RPC-style SOAP message:
`dataRequest(`*`dataset:String, variable:String, file_format:String, axis_1_type:String, axis_1_start:Int, axis_1_stride:Int, axis_1_count:Int, ... axis_4_type:String, axis_4_start:Int, axis_4_stride:Int, axis_4_count:Int`*`):String`.
The `file_format` string takes any of the values '`RAW`', '`CDF`', or '`HDF`' to specify raw IEEE binary, netCDF or HDF file formats respectively. As with the `dataQuery` operation, trailing NULL parameters (empty string, or zero) may be used for variables with fewer than four dimensions. The *`axis_type`* parameters take any of the values '`x`', '`y`', '`z`', or '`t`' and jointly specify the relative order of the axes required for the retrieved data.

The `dataRequest` operation applies the request parameters against the internal storage metadata to prepare the data product in accordance with the request. This is placed in a network-accessible cache (with a filename generated from a one-way hash of request parameters). The Web Service returns a choice of URLs at which the requested data is available. Both HTTP and GridFTP [1] transport protocols are supported. A richer data usage model would provide for the delivery method to be specified in the request. More extensive options could be supported including direct streaming to a specified TCP/IP socket, or third-party GridFTP transfers.

An example of the use of the `dataRequest` operation is given in Appendix B.

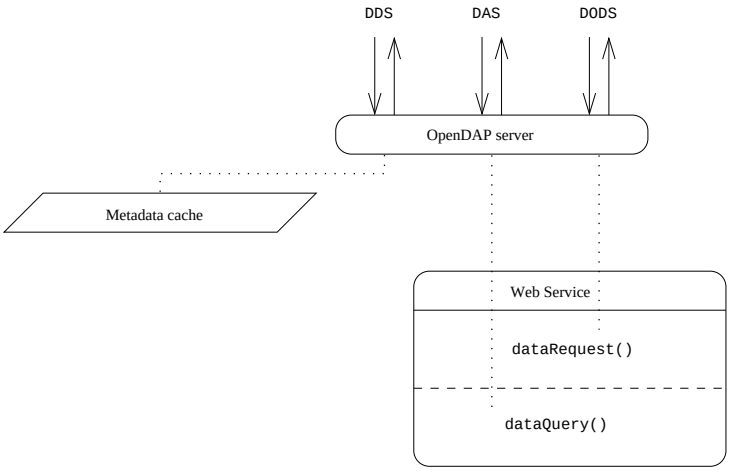

Figure 3: A DODS/OPeNDAP server layered on top of a data access Web Service

4.4 Internal data storage metadata

The final component of the Web Service architecture is the metadata used internally to describe stored files and define logical datasets. Datasets are defined following the structure of Figure 2. A Document Type Definition for the XML document containing this information is given in Appendix A.

This internal metadata is an important component of the complete service. It allows the construction of arbitrary logical datasets and variables from aggregations of files. It affords the opportunity truly to separate data from storage, and so logical variables are defined using the controlled vocabulary of the CF-convention's "Standard Name Table".

4.5 DODS/OPeNDAP server

A DODS/OPeNDAP server was layered on top of the data access Web Service, so that the whole system could be made backwards compatible with the current state-of-the-art. Such a layer provides a "better DODS than DODS", since such DODS-enabled datasets can by default be made CF-compliant, for instance. It is illustrated in Figure 3. This used the standard DODS server template code, with SOAP calls to the Web Service to fulfill requests for metadata (DAS, DDS) or data (DODS) objects.

5 Prototype interactive visualisation client

One clear need of the climate community is to be able to visualise sub-components of large data sets in a transparent manner, interactively and in real-time. This

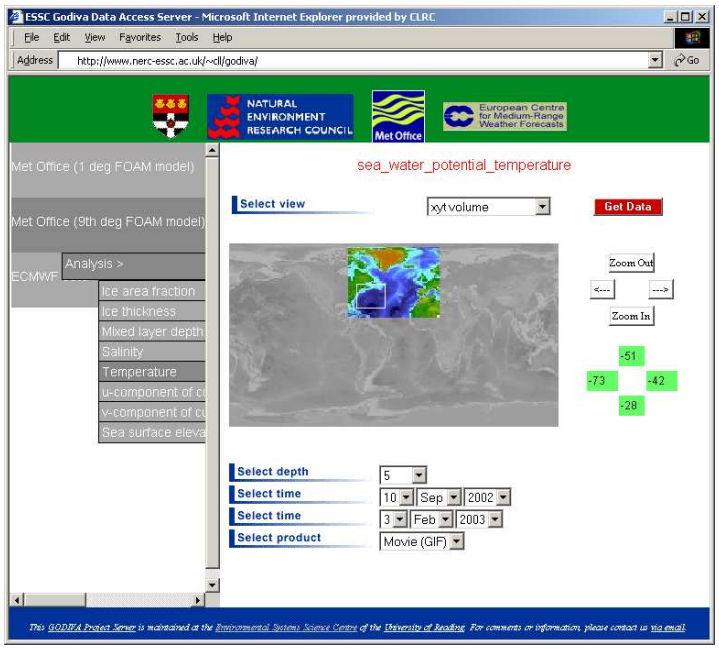

Figure 4: GADS visualisation browser

is demonstrated by the popularity of the Live Access Server. We have therefore designed a first use for GADS in delivering data to an interactive browse tool based on the LAS web portal model as a template. Figure 4 shows the front page of the web portal for this service. This prototype uses both the `dataQuery` and `dataRequest` services to access remote data in real time and bring selected data to a graphics tool (currently Ferret). The portal takes advantage of GADS to dynamically query data holdings. All metadata required to make a `dataRequest` can be obtained via the `dataQuery` operation. This allows the data host to freely manage their data holdings without informing remote sites about small changes that would otherwise disable an LAS-style data portal. There is also an additional functionality over the Live Access Server with an ability to generate animations in real time by distributing the image rendering over several networked machines. Again a web service model is used for the rendering so potentially the rendering could be even more widely dispersed.

Two further areas of active development are underway in order to exploit GADS more fully. One objective will be to allow a computational service layer to lie between the data and the visual rendering. Small computational operations such as forming difference fields, spatial coordinate transformations, or basic operations like field differentiation are easy to precode but more difficult to allow full user interactivity. Such solutions as the Climate Data Analysis Tools

(CDAT)[10] could be built in at this point. More complex services could be developed as stand alone Web or Grid Services and be available as remote operations. The distributed rendering facility of the current portal demonstrates that it should also be possible to pass the data to a computational Grid for more intensive computational analysis.

6 Discussion and conclusions

There is an increasingly urgent need in the climate research community to develop advanced tools to access large distributed datasets. Emerging Grid technology could transform current data handling techniques.

We have presented a general model for a data access Web Service, as well as an implementation (the Grid Access Data Service, GADS) for gridded climate model data. The Web Service abstracts data from storage, includes standard and extensible metadata models, allows flexible data delivery options, and is compatible with emerging Grid security and workflow scenarios. We have also layered a DODS/OPeNDAP server over GADS to demonstrate backwards compatibility with the current state of the art. A web client based on the Live Access Server has been produced to interactively access and visualise data from GADS.

Further development of GADS will continue on two fronts. The first is to develop further the component metadata models (for scientific data, for usage scenarios, and for storage). Developing rich schema for these would expand the usefulness of the Web Service significantly. With the development of such metadata models and formal schemas, we envisage sophisticated (GUI) client tools for drilling through the data object hierarchies and constructing the parameters for the data query and request operations. The NERC DataGrid project will likely undertake this kind of development. Another useful tool could simplify management of the internal (data storage) metadata (Section 4.4) — for instance, it could allow new logical datasets to be defined by graphically selecting files and variables from which the datasets are composed, adding attributes automatically or with user intervention where required. Such client and management tools would provide a high-impact demonstration of the utility of the Web Service model. A second line of development will be to migrate the model to the Open Grid Services Infrastructure [8, 21] as outlined in section 3.2.4.

The true potential for the work presented here will be its incorporation into Grid workflows to undertake novel scientific research and to increase the ease of access for the environmental science user community. This will require a toolkit of other Web Service modules designed to ease the burden of moving large climate and environmental datasets between user designed applications. Examples of other Web Services that would be useful in such a scenario might include regridding/interpolation, generic visualisation services, and computation services. There is great potential for this field of science to become big users of Grid technology in the near future

[10] Available from `http://esg.llnl.gov/cdat/`

7 Acknowledgements

This work has been supported by the UK's Natural Environment Research Council through grants from the Data Assimilation Research Centre and the GODIVA (Grid for Ocean Diagnostics, Interactive Visualisation, and Analysis) e-Science pilot project (NER/TS/2001/00866).

A Document Type Definition for internal storage metadata

Listed below is the DTD for the internal data storage metadata. It allows logical variables and datasets to be assembled as per the model in Figure 2 from netCDF, HDF and GRIB files.

```
<!DOCTYPE datacollection [
  <!ELEMENT datacollection (dataset*)>
  <!ELEMENT dataset (description*, variable+, coord_axis+)>
  <!ELEMENT description (#PCDATA)>
  <!ELEMENT variable (grid, data)>
  <!ELEMENT coord_axis (value_array | value+)>
  <!ELEMENT grid (t | z | y | x)+>
  <!ELEMENT data (file+)>
  <!ELEMENT value_array (start, step, size)>
  <!ELEMENT value (#PCDATA)>
  <!ELEMENT file EMPTY>
  <!ELEMENT start (#PCDATA)>
  <!ELEMENT step (#PCDATA)>
  <!ELEMENT size (#PCDATA)>
  <!ELEMENT t EMPTY>
  <!ELEMENT z EMPTY>
  <!ELEMENT y EMPTY>
  <!ELEMENT x EMPTY>
  <!ATTLIST dataset
    name CDATA #REQUIRED
    last_modified CDATA #REQUIRED
  >
  <!ATTLIST variable
    name CDATA #REQUIRED
    datatype float #FIXED
    units CDATA #REQUIRED
  >
  <!ATTLIST t
    units CDATA #REQUIRED
    axis ID #REQUIRED
  >
  <!ATTLIST z
    units (m | cm) #REQUIRED
    positive (down | up) #REQUIRED
    axis ID #REQUIRED
  >
  <!ATTLIST y
    units (degrees_north | degrees_south) #REQUIRED
    axis ID #REQUIRED
```

```
  >
  <!ATTLIST x
    units (degrees_west | degrees_east) #REQUIRED
    axis ID #REQUIRED
  >
  <!ATTLIST coord_axis
    name IDREF #REQUIRED
  >
  <!ATTLIST file
    location CDATA #REQUIRED
    format (CDF | HDF | GRIB) #REQUIRED
    variable CDATA #REQUIRED
  >
]>
```

B Example `dataQuery` and `dataRequest` usage

The `dataQuery` operation may be used to query the data holdings at successively deeper levels of the model hierarchy. Thus, the following method invocation would retrieve the available datasets:
`result = dataQuery(‘’, ‘’, ‘’).`
The variables contained within the ‘`FOAM_ONE_DEGREE`’ dataset could be found with the following invocation:
`result = dataQuery(‘FOAM_ONE_DEGREE’, ‘’, ‘’);`
while the dimensions of the ‘`sea_water_salinity`’ variable of this dataset could be found with:
`result = dataQuery(‘FOAM_ONE_DEGREE’, ‘sea_water_salinity’, ‘’).`
Finally, the coordinate values of the time axis could be obtained with:
`result = dataQuery(‘FOAM_ONE_DEGREE’, ‘sea_water_salinity’, ‘t’).`
In each case, the result is a string containing an XML snippet. For instance, the following was the result of the variable query in the test implementation:

```
result = dataQuery(‘FOAM_ONE_DEGREE’, ‘’, ‘’)

<dataQueryResponse>
 <variable name="sea_water_potential_temperature" datatype="float" units="degC"/>
 <variable name="sea_water_salinity" datatype="float" units="PSU"/>
 <variable name="sea_surface_elevation" datatype="float" units="cm"/>
 <variable name="eastward_sea_water_velocity" datatype="float" units="m/s"/>
 <variable name="northward_sea_water_velocity" datatype="float" units="m/s"/>
</dataQueryResponse>
```

The `dataRequest` operation specifies the dataset and variable required, axis order and subsetting parameters, and required file format. Thus, the following invocation requests some sea surface elevation data in netCDF format:
`result = dataRequest(‘FOAM_ONE_DEGREE’, ‘sea_surface_elevation’, ‘CDF’,`
`‘t’, 1, 1, 1,`
`‘y’, 10, 1, 100,`

‘x’, 24, 1, 150,
‘’, 0, 0, 0),
with result:

```
 <dataRequestResponse>
  <URL>gsiftp://hydra.nerc-essc.ac.uk/data/ocean/scratch/WService_cache/WServi
ce.2521adea959ac34bcda13bdc6de46036.nc</URL>
  <URL>http://www.nerc-essc.ac.uk/data/ocean/scratch/WService_cache/WService.2
521adea959ac34bcda13bdc6de46036.nc</URL>
 <\dataRequestResponse>
```

C XML Schema for metadata model of scientific data

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:simpleType name="CFStandardName">
    <xs:annotation>
      <xs:documentation>
       Limited to CF convention’s "Standard  Name Table"
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:token"/>
  </xs:simpleType>
  <xs:simpleType name="DataType">
    <xs:annotation>
      <xs:documentation>
       Only "float" supported initially
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:token">
      <xs:enumeration value="float"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="AxisType">
    <xs:restriction base="xs:token">
      <xs:enumeration value="x"/>
      <xs:enumeration value="y"/>
      <xs:enumeration value="z"/>
      <xs:enumeration value="t"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="CoordUnits">
    <xs:restriction base="xs:token">
```

```
      <xs:enumeration value="degrees_west"/>
      <xs:enumeration value="degrees_east"/>
      <xs:enumeration value="degrees_south"/>
      <xs:enumeration value="degrees_north"/>
      <xs:enumeration value="m"/>
      <xs:enumeration value="cm"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="datacollection">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="dataset" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="dataset">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="variable" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="name" type="xs:token"/>
      <xs:attribute name="lastModified" type="xs:dateTime"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="variable">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="axis" maxOccurs="4"/>
      </xs:sequence>
      <xs:attribute name="name" type="CFStandardName"/>
      <xs:attribute name="dataType" type="DataType"/>
      <xs:attribute name="units" type="xs:token"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="axis">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="value"/>
      </xs:sequence>
      <xs:attribute name="type" type="AxisType"/>
      <xs:attribute name="size" type="xs:positiveInteger"/>
      <xs:attribute name="units" type="CoordUnits"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="value" type="xs:float"/>
</xs:schema>
```

D WSDL description of data access Web Service

Presented below is a WSDL description of the Web Service presented here. For the sake of brevity, only the `dataQuery` operation is included.

```
<?xml version="1.0" encoding="UTF-8"?>

<definitions name="GADSDescription" xmlns="http://schemas.xmlsoap.org/wsdl"
 targetNamespace="urn:GADS" xmlns:tns="urn:GADS"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"/>

<message name="dataQueryRequest">
 <part name="dataset" type="xsd:string"/>
 <part name="variable" type="xsd:string"/>
 <part name="axis" type="xsd:string"/>
</message>

<message name="dataQueryResponse">
 <part name="result" type="xsd:string"/>
</message>

<portType name="GADSInterface">
 <operation name="dataQuery">
  <input message="tns:dataQueryRequest"/>
  <output message="tns:dataQueryResponse"/>
 </operation>
</portType>

<binding name="GADSBinding" type="tns:GADSInterface">
 <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
 <operation name="dataQuery">
  <soap:operation soapAction=""/>
  <input>
   <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
  </input>
  <output>
   <soap:body use="encoded"
    encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
  </output>
</binding>

</definitions>
```

References

[1] Allcock, W., Bester, J., Bresnahan, J., Chervenak, A., Liming, L., Meder, S., and Tuecke, S. GridFTP protocol specification. Tech. rep., Global Grid Forum, Sept. 2002. [Online] Available: http://www.globus.org/research/papers/GridftpSpec02.doc [2002, December 13].

[2] Berners-Lee, T., Masinter, L., and McCahill, L. RFC 1738: Uniform resource locators (URL). [Online] Available: http://www.ietf.org/rfc/rfc1738.txt?number=1738 [2002, December 13], Dec. 1994.

[3] Bindoff, N., Woolf, A., Roberts, J., and Sainsbury, F. Online access to WOCE global data V3. In *WOCE and Beyond: Achievements of the World Ocean Circulation Experiment* (2002). 18-22 November 2002, San Antonio, Texas.

[4] Christensen, E., Curbera, F., Meredith, G., and Weerawarana, S. Web Services Description Language (WSDL) 1.1, Mar. 2001. W3C Note. [Online] Available: http://www.w3.org/TR/wsdl [2002, December 13].

[5] Domenico, B., Caron, J., Davis, E., Kambic, R., and Nativi, S. Thematic Real-time Environmental Distributed Data Services (THREDDS): Incorporating interactive analysis tools into NSDL. *Journal of Digital Information 2*, 4 (2002). [Online] Available: http://jodi.ecs.soton.ac.uk/Articles/v02/i04/Domenico/ [2002, November 20].

[6] EUMETNET. UNIDART Homepage. [Online] Available: http://www.dwd.de/UNIDART/ [2002, December 11], 2002.

[7] Fallside, D. C. XML Schema Part 0: Primer. Tech. rep., World Wide Web Consortium, May 2001. [Online] Available: http://www.w3.org/TR/xmlschema-0/ [2002, December 13].

[8] Foster, I., Kesselman, C., Nick, J. M., and Tuecke, S. The physiology of the Grid. An Open Grid Services Architecture for distributed systems integration. Tech. rep., Global Grid Forum, 2002.

[9] Gallagher, J., and Milkowski, G. Data transport within the Distributed Oceanographic Data System. In *The Web Revolution* (1995), Fourth International World Wide Web Conference. [Online] Available: http://www.w3.org/Conferences/WWW4/Papers/67/ [2002, November 20].

[10] IBM Corporation and Microsoft Corporation. *Security in a Web Services World: A Proposed Architecture and Roadmap*, Apr. 2002. [Online] Available: http://www-106.ibm.com/developerworks/library/ws-secmap/ [2002, December 13].

[11] Intergovernmental Panel on Climate Change. *Climate Change 2001: The Scientific Basis*. Geneva, SWITZERLAND, 2001.

[12] Nagaratnam, N., Janson, P., Dayka, J., Nadalin, A., Siebenlist, F., Welch, V., Foster, I., and Tuecke, S. The security architecture for Open Grid Services. GGF5, Edinburgh. [Online] Available: http://www.globus.org/ogsa/Security/ [2002, December 12], July 2002.

[13] National Center for Supercomputing Applications. The NCSA HDF Home Page. [Online] Available: http://hdf.ncsa.uiuc.edu [2002, July 29], 2002.

[14] PMEL/NOAA. Ferret Home Page. [Online] Available: http://ferret.wrc.noaa.gov/Ferret/ [2002, December 11], 2002.

[15] PMEL/NOAA. Live Access Server. [Online] Available: http://ferret.wrc.noaa.gov/Ferret/LAS/ [2002, December 11], 2002.

[16] Siebenlist, F., Welch, V., Tuecke, S., Foster, I., Nagaratnam, N., Janson, P., Dayka, J., and Nadalin, A. OGSA security roadmap. GGF5, Edinburgh. [Online] Available: http://www.globus.org/ogsa/Security/ [2002, December 12], July 2002.

[17] Siedler, G., Church, J., and Gould, J., Eds. *Ocean Circulation & Climate: Observing and Modelling the Global Ocean*. Academic Press, 2001.

[18] Simmons, A., and Gibson, J., Eds. *The ERA-40 Project Plan*. European Centre for Medium-Range Weather Forecasting, Mar. 2000.

[19] Smith, N., Ed. *GODAE Data and Product Server Workshop* (Biarritz, France, 2002). Summary Report [Online] Available: http://www.bom.gov.au/bmrc/ocean/GODAE/Projects/ServerWS/index.html [2002, December 11].

[20] Snell, J., Tidwell, D., and Kulchenko, P. *Programming Web Services with SOAP*. O'Reilly and Associates, Inc., 2002.

[21] Tuecke, S., et al. Open Grid Services Infrastructure (OGSI). Tech. rep., Global Grid Forum, 2003.

[22] University Corporation for Atmospheric Research. *DODS — Data Delivery Architecture*, Aug. 1996. [Online] Available: http://www.unidata.ucar.edu/packages/dods/archive/design/data-delivery-arch/ [2002, December 12].

[23] University Corporation for Atmospheric Research. *DODS — Data Delivery Design*, Aug. 1996. [Online] Available: http://www.unidata.ucar.edu/packages/dods/archive/design/data-delivery-arch/ [2002, December 12].

[24] University Corporation for Atmospheric Research. *DODS - Uniform Resource Locators*, Aug. 1996. [Online] Available: http://www.unidata.ucar.edu/packages/dods/archive/design/urls/urls.html [2002, December 12].

[25] University Corporation for Atmospheric Research. Distributed Oceanographic Data System. [Online] Available: http://www.unidata.ucar.edu/packages/dods [2002, July 29], 2002.

[26] University Corporation for Atmospheric Research. NetCDF Conventions. [Online] Available: http://www.unidata.ucar.edu/packages/netcdf/conventions.html [2002, December 11], 2002.

[27] University Corporation for Atmospheric Research. Unidata NetCDF. [Online] Available: http://www.unidata.ucar.edu/packages/netcdf [2002, July 29], 2002.

[28] WOCE Data Products Committee. *WOCE CD Version 3.0 data and inventory conventions*, 2002. [Online] Available: http://www.cms.udel.edu/woce/utils/netcdf/woce_conventions [2002, December 11].

[29] World Meteorological Organization. *Guide to WMO Binary Code Form GRIB 1.* Geneva, 1994. Technical Report No. 17.

List of Figures