Distributed Gridded Data Delivery for Marine Research

E.A. King[1], P.P.Y. Mak[2,3], P.J. Turner[1], G.P. Smith[1], K.D. Suber[1], M.J. Paget[1], C. J. Jackett[1], P. Fearns[4], A.L. Rohl[5], F. Goessmann[3,5], L. Majewski[6], S. Reddy[7], C. Steinberg[8].

1. CSIRO Marine & Atmospheric Research, GPO Box 1538, Hobart, Tasmania, 7001;
2. Tasmanian Partnership for Advanced Computing (http://www.tpac.org.au);
3. Australian Research Collaboration Service (http://www.arcs.org.au);
4. Imaging & Applied Physics, Curtin University of Technology, GPO Box U1987, Perth, WA, 6845;
5. iVEC, 'The hub of advanced computing in Western Australia', 26 Dick Perry Avenue, Technology Park, Kensington WA 6151;
6. Bureau of Meteorology, GPO Box 1289, Melbourne, Victoria, 3001;
7. Geoscience Australia, GPO Box 378, Canberra ACT 2601;
8. Australian Institute of Marine Science, PMB 3, Townsville, Queensland, 4810

Abstract

A combination of off-the-shelf open-source software and custom-built middleware is used to unite remotely sensed marine data archives operated across Australia by five different agencies and make them accessible via a common interface to a user located anywhere on the internet. The utilisation of existing storage and some state of the art fileservers with a distributed data model, makes the system low-cost, scalable and robust. The creation of a virtual national data set with automatic cataloguing enables the development of advanced data services including aggregation and spatio-temporal time series and sub-setting. Data sets are served through OPeNDAP and automatically harvested for meta-data including temporal and spatial bounds. Spatio-temporal queries made on the catalogues provide information to allow retrieval of subset data, or if many data sets are returned, the aggregation of data matching the query. The software developed to implement the system is built as a set of layers with well defined interfaces, allowing system modularisation and a range of levels of access.

1. Introduction

Gridded data sets, particularly those generated by satellite-borne sensors, have always presented analysis challenges, primarily because their sheer volume exceeded typical storage capacities and network speeds made remote access impractical. Secondary obstacles arise from format inconsistency, choice of geographical projections and sampling, processing algorithms and metadata completeness; in short, a lack of standardisation. The recent rapid and widespread deployment of high bandwidth networks together with the commoditisation and resultant reduction in cost of online storage systems, has reduced the constraint imposed by data volume and the analysis challenge has been transformed into one of access standards and metadata management (e.g. cataloguing, searching, sub-setting).

There are several regional archives of marine remote sensing data distributed around Australia, and an equally distributed community of users. These archives are components of a virtual national data set. If it were possible to make them compatible, as well as widely and easily accessible, a national data set would be realised and the marine research community would collectively have created a common base data set as a foundation for collaborative research on a national scale. This paper describes our attempt to do just that.

The National Collaborative Research Infrastructure Strategy (NCRIS) is a federally funded program to provide major research facilities, and to support research infrastructure and networks. One of the components of NCRIS is the Integrated Marine Observing System (IMOS), which includes a facility to support Satellite Remote Sensing (SRS). Part of the SRS facility is the development of an Australian Oceans Distributed Active Archive Centre (AO-DAAC). Participants in the AO-DAAC project include both Universities and government agencies. The main institutions and agencies holding publicly available marine satellite data are the Australian Institute for Marine Science (AIMS) in Townsville, the Bureau of Meteorology (BoM) in Melbourne, Geoscience Australia (GA) and CSIRO in Canberra, Curtin University and the Western Australian Satellite Technology and Applications Consortium (WASTAC) in Perth, and the Tasmanian Partnership for Advanced Computing (TPAC) based at the University of Tasmania, and CSIRO, both in Hobart. Additional hardware and software support for the AO-DAAC is being provided by the participation of the iVEC consortium in Perth, and

through staff support by the Australian Research Collaboration Service (ARCS) that is funded, in part, by another NCRIS program component, Platforms for Collaboration.

Figure 1 shows the data acquisition network that will be operating in Australia by the end of 2008. It includes both L-band and X-band reception stations that provide continental coverage extending well into the surrounding region in all directions. These permit ongoing daily acquisition of all regional NOAA/AVHRR and MODIS imagery. Additionally individual institutes host historical data sets, of both raw data and derived products (such as Sea Surface Temperature and ocean colour), that in some cases, encompass over 25 years of observations. A further source of data is overseas organisations, such as NASA and the European Space Agency (ESA), that disseminate sensor data globally. For example, Curtin University and WASTAC are obtaining the entire historical Level 1 MODIS (http://modis.gsfc.nasa.gov) data for Australia, and CSIRO are doing likewise for the ATSR (http://www.atsr.rl.ac.uk) and AATSR series. The goal of the AO-DAAC project is to improve the sharing of as many of these products and data sets as possible amongst agencies by encouraging the adoption of consistent processing standards and formats, supporting storage infrastructure and developing data discovery and access tools.

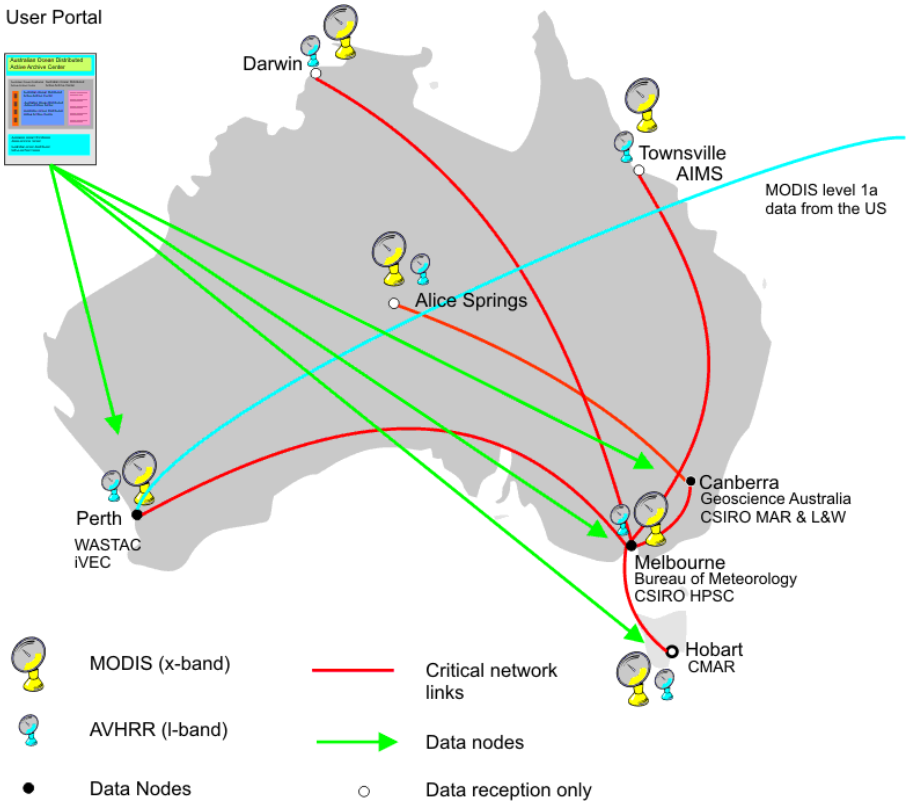

Figure 1. Reception system to be in place for AVHRR and MODIS at the end of 2008.

2. Technologies

Each participating institution has already invested in storage hardware and/or in developing expertise with the software tools. One of the key objectives of the AO-DAAC project is to leverage these existing archives, storage and software technologies wherever possible. This requirement largely dictated the choice of tools that would be used; they are virtually all open-source software packages that implement publicly available standards. The principal advantages of this choice are:

- Reliability - access to existing well-established standards;
- Cost of adoption - a lot of data already conforms;
- Zero software cost – can be deployed widely without costly licensing;
- Existing expertise - there is already a significant user base.

The use of the Java programming language has been partly dictated by the reuse of code already written in Java. Java provides a cross-platform capability and a good set of tools for the development

of applications, web GUIs and web services. We briefly survey the other key software and data management technologies here.

NetCDF and HDF Data Formats

Historically, for most research projects that produce digital data sets, data management has been seen as a topic of secondary interest to the particular measurement experiment at hand and the overriding goal is to get the data into an analysis package as quickly as possible. Long term archival is seldom considered as part of storage design. Consequently file formats have often tended to be data set specific. This situation presents a major obstacle to data reuse because not only is specialised data-specific software required to access the data, there is no guarantee that the files carry any metadata describing the data fields, nor is there any likelihood that the data will be organised in any particular way, or be readable on computer architectures foreign to that on which they were originally written.

A straightforward approach to overcoming this is to define a generic file format standard that makes it both easy and mandatory for data authors to store, name, label and describe data items. By providing software libraries, in a variety of common languages, that operate on multiple platforms and simplify adherence to the file formatting standard, it is possible to create files that are platform independent and self describing. The advantage of this approach is that all data so created becomes long-lived (documented), interchangeable (portable) and accessible to user communities.

The Network Common Data Form (NetCDF) (http://www.unidata.ucar.edu/software/netcdf/) and Hierarchical Data Format (HDF) (http://hdf.ncsa.uiuc.edu) are two implementations of such a solution. Both define abstract data models that support a range of data types, such as scalars, vectors, sequences and grids, and provide software libraries that implement machine independent file formats. NetCDF was developed primarily by the atmospheric research community to manage gridded data sets whereas HDF has a more diverse genesis and supports a complex range of types. Although both have been widely adopted in the physical research communities, in practice the NetCDF Common Data Model (Figure 2) is sufficient for representing most spatial gridded data and the more advanced features of HDF are used less often. Much meteorological and marine data is distributed in NetCDF version 3 and HDF version 4 has been adopted by NASA as a standard for distributing Earth Observation data. The next versions of each (NetCDF4 and HDF5) will bring the two formats closer together by storing NetCDF4 data structures in HDF5 files.

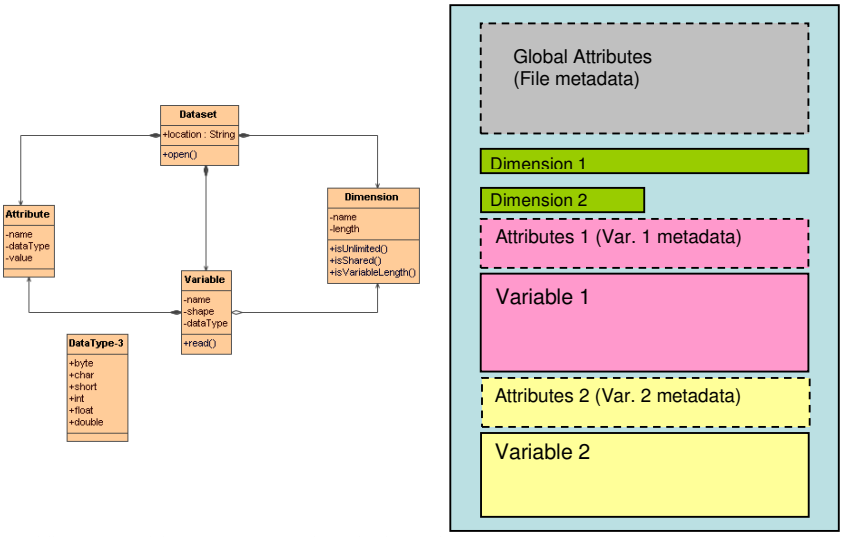

Figure 2. Unified Modelling Language description of the NetCDF Common Data Model (from http://www.unidata.ucar.edu/software/netcdf-java/CDM.html) and a schematic representation of a NetCDF file comprising two two-dimensional gridded data sets with dimension variables and data set-specific metadata (Attributes) and common metadata (Global).

OPeNDAP Data Access Protocol

Traditional data access requires data files to be located locally, or at least on network connected disks, to the software being used to read and analyse them. Since nearly every user needs their own copy, this results in duplication of data sets which is both inefficient in terms of storage, and makes it difficult to

promulgate new versions to the user community when processing methods change. The advent of widespread high speed networks has made it possible to store a single reference copy of the data and provide access to it remotely. The Open-source Project for a Network Data Access Protocol (OPeNDAP) (http://opendap.org) is a framework that simplifies this process.

Like NetCDF and HDF, OPeNDAP is based on an abstract data model, the Data Access Protocol (DAP), and provides software libraries and server software that make it possible to access remote data sets regardless of their native (stored) format. By imposing this data model, DAP places an expectation that the supported file types will provide certain relevant metadata, such as grid dimensions and sizes. The DAP incorporates a facility to return these metadata to the remote user. One other key feature of the DAP protocol, enabled by the metadata retrieval capability, is its ability to support requests for only subsets of data sets, thereby dramatically improving the efficiency of access to small parts of large data sets.

Application software can include calls to the OPeNDAP libraries that directly return data from remote servers with no need for any intermediate file storage. By separating software development and operation from data management, it is possible to greatly speed the data analysis process and to develop new applications that utilise diverse data sets from a variety of locations and providers. OPeNDAP uses HTTP as its transport protocol and so it works as seamlessly as normal web pages across the internet and is easily configured across institutional firewalls. Data can be completely selected and subset using constraint expressions incorporated within HTTP URLs.

The OPeNDAP Consortium provides a modular and highly configurable data server that works with NetCDF3, HDF4, HDF5, as well as a number of other formats common in the meteorological data domain. The THREDDS Data Server (Caron *et al 2006)* developed by the Unidata group at NCAR also supports OPeNDAP for a wide variety of files, though not HDF4 at present (Henry *et al*, 2008).

PostGIS Spatial Database

Relational databases are systems for storing data items and their attributes, and managing the relationships between them. Typically data and attributes can be character strings or numbers, or more complex composite types such as dates, booleans, or monetary quantities, which in turn support their own specialised operations such as arithmetic, logic operations or formatting. A spatially-enabled database adds support for geographical data entities and attributes making it possible, for example, to record locations, vectors and polygons and perform spatial operations such as testing for line intersections, points within polygons and computing distances between points.

PostgreSQL (http://www.postgresql.org) is a mature open-source relational database system that runs on most major operating systems. PostGIS (http://www.postgis.org) is an open-source extension to PostgreSQL that adds support for the spatial features defined by the OpenGIS Consortium Simple Features Specification. These objects include points, linestrings, polygons, multi-points, multi-linestrings, multi-polygons and mixed collections of these. Our basic requirement is to be able to record the polygon boundaries of satellite imagery products and then to search for overlap of these with either points or regions of interest.

3. AO-DAAC Components and Design

System Overview

The design of the AO-DAAC is based on a tiered model, with each layer building upon the functionality of those below it (Figure 3). By modularising the design in this way it is possible to develop and test individual components separately and to control complexity by clearly defining the interfaces between modules.

The foundation tier comprises the OPeNDAP data servers, located at the participating custodian agencies, that provide direct access to both metadata and the data itself. Built on top of this infrastructure is a 3-layer metadata management system. The first layer of this system is a metadata harvester, which is essentially a DAP-literate web robot that automatically collects relevant metadata that is used to populate a spatial database in the layer above. The top layer of the metadata system is a web service that uses the contents of the database to construct OPeNDAP URLs that are returned to clients in response to requests for data. The top layer of the AO-DAAC is the user tier, which accesses

the web service to obtain the OPeNDAP URLs, and then goes directly to the OPeNDAP servers to retrieve the data. Each of these components is described in more detail below.

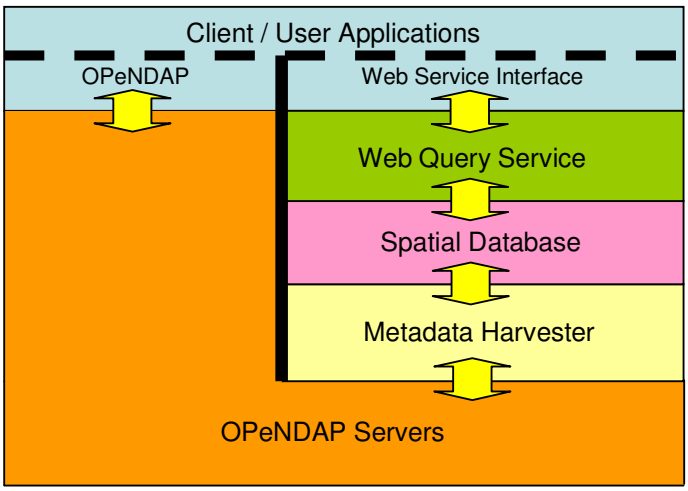

Figure 3. The multi-tiered design of the AO-DAAC. The OPeNDAP servers deployed at data custodian's institutions provide the base layer. They are accessed by the metadata harvester which populates the spatial database, the contents of which are in turn exposed by the web query service. Clients and user applications are able to access the web query service to obtain OPeNDAP URLs that can then be used to access the data servers directly.

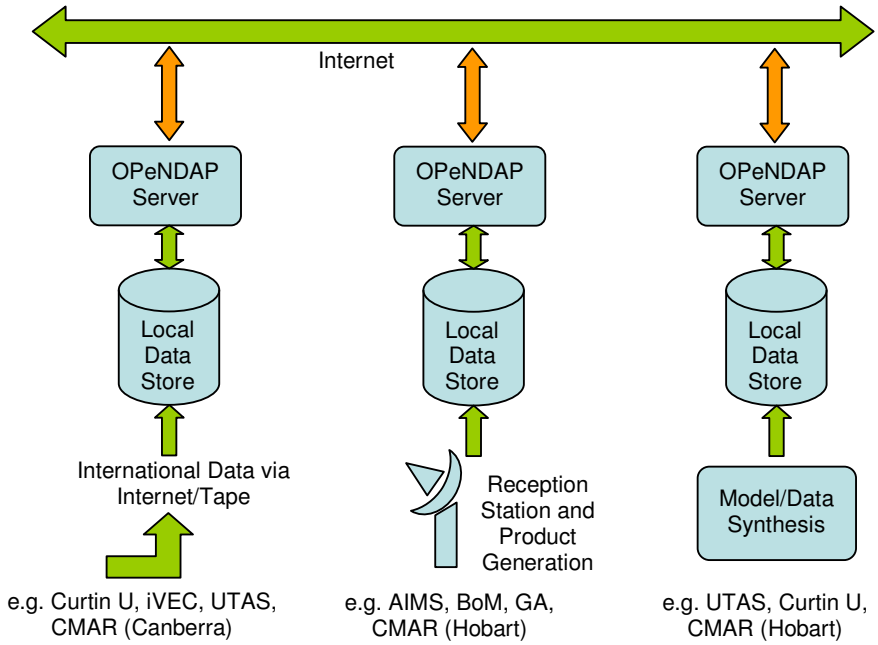

Figure 4. The OPeNDAP server layer underpins the whole of the AO-DAAC system by providing remote access to all data sets, regardless of whether they are locally received, synthesised or imported from a third-party agency overseas.

Data Servers

These form the base layer of the whole system. By using OPeNDAP to permit remote access, they expose the metadata for harvesting and enable access to the data itself. The only real requirement for

data sites is that they be able to place data online. The origin of the data is unimportant, in that it can be locally received, imported from overseas institutions, or even synthesised from a model (Figure 4).

The latest OPeNDAP data server, known as "Hyrax", comprises a front-end that manages the request and supports THREDDS data catalogues that provide a virtualisation capability by enabling data sets to be distributed across multiple hosts. The front-end server passes requests to a back-end server (BES) that contains one or more format-specific file handlers that perform the data extraction. The BES is designed to make it straightforward to create a module to support a new file format. The Hyrax architecture allows one front-end server to use multiple BES installations, on multiple hosts if necessary, so that the file access load on the system can be distributed. In theory, it should be possible to implement the AO-DAAC as a single front-end server that communicates with a BES at each participating site. However, by operating a complete server at each site, data is always available locally regardless of the operational status of the central node.

Metadata Harvester and Aliasing System

The metadata harvester is based on the TPAC Digital Library Crawler which catalogues files and metadata from OPeNDAP sites (Cumming and Hyland, 2005). The crawler begins at some directory root that contains related files (expected to be of the same data set). It finds all files and extracts values published in the Data Attribute Structure (DAS). These values correspond to the attribute values in the HDF and NetCDF files. The crawler was very simple and required no semantic understanding of the metadata. The most recent DAP specification (DAP4) (Gallagher *et al*, 2004) implemented by the Hyrax server, defines a new DDX structure that is an XML rendering of an extended version of the DAS metadata. The harvester has been designed from the outset to use the DDX.

The AO-DAAC harvester is required to retrieve geospatial and temporal bounds of the data stored in each file in each data set. To achieve this, the harvester incorporates a simple spatio-temporal ontology and a more sophisticated understanding of the data file structure and the semantics of some of the information to be retrieved. The initial approach makes an assumption that the spatial and (possibly) temporal information for a given variable will be available through coordinate variables (vectors) associated with the target data arrays in the data files. The geographic bounds of the gridded data variable can be found by retrieving the coordinate variable limits. These coordinate variables normally have units and a name so a check can be made to see if they are in units of degrees and have names like latitude and longitude. The other coordinate required is date (or time) which can potentially be extracted from the corresponding coordinate variable. Although some standards exist (*e.g.* COARDS, http://ferret.wrc.noaa.gov/noaa_coop/coop_cdf_profile.html) temporal information can be stored in a variety of units (*e.g.* days or seconds) and relative to a particular reference time, so there may also be a translation needed to get it into a consistent form for recording in the database. A variant of this problem is where the data corresponds to a single epoch that is identified in the file as an attribute rather than a coordinate dimension, in which case the harvester tries to locate that attribute.

While the original TPAC crawler retrieved everything it found in each data set, the goal of the harvester is to create database entries containing only selected relevant information, so a mechanism has been developed that allows specification of the attributes and data bounds to harvest. This mechanism also allows variable and attribute name translation, or "aliasing", providing the capacity to implement a standard naming system across multiple data sets. Enforcing a consistent naming scheme is crucial to enabling the harvester to correctly identify and retrieve coordinate information

The aliasing system is implemented via an XML file that is placed in the data set's root directory and named "alias.xml". All though this mapping could be done at the harvester, the data set custodian is in a better position to provide the mappings from their local data nomenclature into those adopted by the AO-DAAC and so responsibility for this has been distributed. For the prototype system, only two standard names have been defined: 'Latitude' and 'Longitude'. When the crawler encounters a data set, it first locates the alias.xml file, and then applies the translations to all files in the current and subsequent subdirectories. An example alias.xml file is:

```
<Document>
    <AttributeTable name="NC_GLOBAL">
        <Attribute name="conventions" standard_name="Conventions" />
    </AttributeTable>
```

```
    <AttributeTable name="sst" standard_name="Sea Surface Temperature"/>
        <CoordinateVariable name="lat" standard_name="Latitude"/>
        <CoordinateVariable name="lon" standard_name="Longitude"/>
        <CoordinateVariable name="time" standard_name="Time">
        <Attribute name="unit"/>
     </AttributeTable>
</Document>
```

Briefly, the first entry indicates that in the global attributes, there is an attribute with name 'conventions' that is to be associated with the standard name (in the ontology) 'Conventions'. The file data variable named 'sst' is to be associated with the standard name 'Sea Surface Temperature' and it has three coordinate variables named 'lat', 'lon' and 'time' that are to be associated with standard names 'Latitude', 'Longitude' and 'Time' respectively. There is an attribute of the 'sst' variable named 'unit' that is also to be harvested.

Using the XML file, the harvester can identify the variables that contain the coordinate data. Currently the spatial boundary is created by retrieving the start and end value of each axis to form a bounding box. The prototype has only implemented methods to handle a rectangular projection, however, there are already Java stub classes for handling irregular axes and projections. The spatial database can store arbitrary polygons so it will be possible in future to upgrade the harvester to deal with data in other projections, such as satellite swaths.

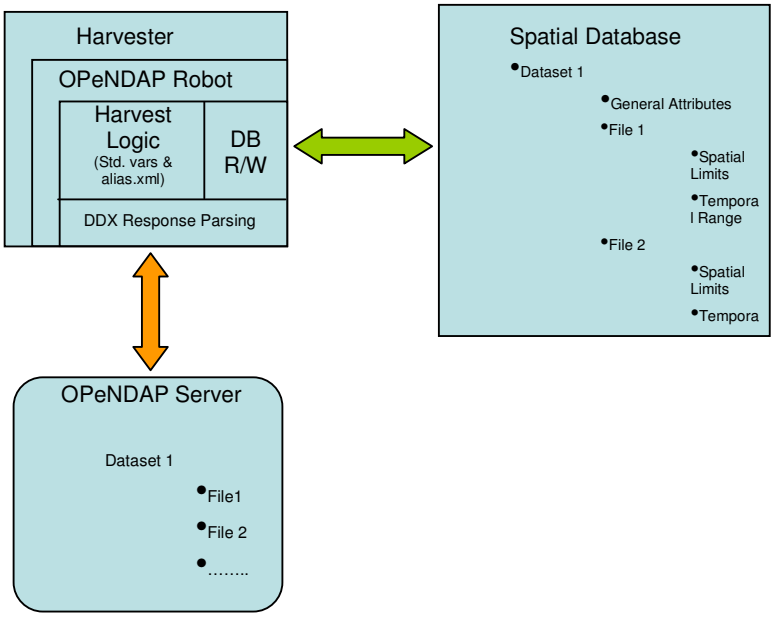

Figure 5. The harvester and database system maintains a spatially-enabled catalogue of data sets that are available from the OPeNDAP servers. The alias.xml file managed by the data provider can be used to map key metadata from local conventions to those used by the AO-DAAC and, for complex data sets, to control which data variables are harvested.

Database

The fundamental concept upon which the database design rests is that of a *product*, which is a variable with consistent characteristics. For example, a Sea Surface Temperature data set computed using a particular algorithm (and possibly stored in multiple locations) would be a single product, whereas two such data sets that represent the same physical property but are computed using two different algorithms would be distinct products. A product name is first registered with the database, essentially defining a unique standard name for a variable, for example "Sea Surface Temperature". It is intended that these names will match those found in the ISO 19115 Marine Profile standard.

When the harvester crawls an OPeNDAP site, it will first search for an alias.xml file. Using this file, any aliases are extracted and stored in the *aliases* table which simply contains pairs of standard and the pre-existing names. If a variable does not have a standard name defined in the alias.xml file, then it is assumed that the standard name is the same as the existing variable name in the file.

The harvester crawls files searching for attributes. Two separate tables are used to record attributes for files at each "site". For these purposes, a site is defined as a separate root directory in the data set catalogue. The *file_attribute_<site>* table records whether an attribute is present (or absent) in a file at <site> and the content of the attribute is stored in the *file_attribute_data_<site>* table.

The Files table can contain a number of products with different spatial and temporal extent. The harvester will store instances of these products as unique Arrays in the database. Each Array entry includes a bounding polygon and a time range. Spatio-temporal searches are carried out by querying this table.

Associated with the Array is set of *coordinate (dimension) variables*. Coordinate variables store metadata relating to the axes of the array, such as the size of each dimension and the minimum and maximum values. Values in this table are used for producing the sub-setting DAP URLs.

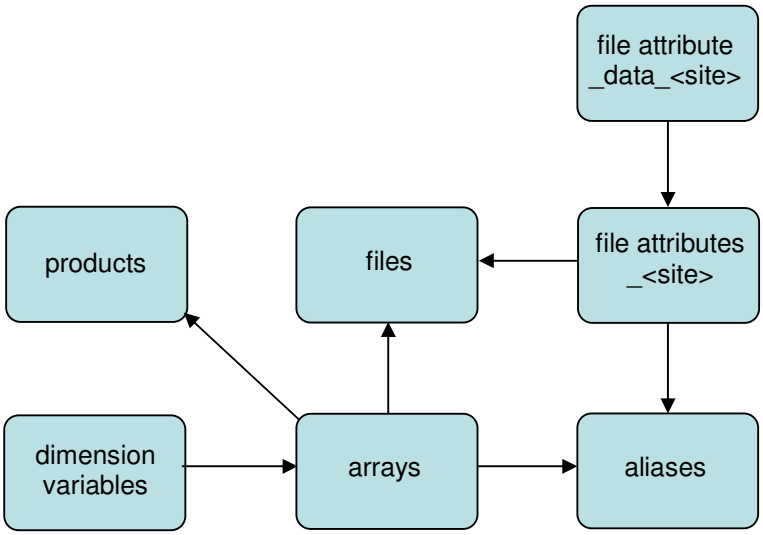

Figure 6. Schematic representation of the key tables and relationships in the database.

Web Query Service

The web query service client is a Java servlet that takes a request for a product, including spatio-temporal constraints, and searches the database to return a set of ready-to-use OPeNDAP URLs (including subset constraints), together with some extra information on the coordinate system limits and geographic projection of the data. It is implemented as a web service, which is an increasingly popular framework that allows loosely coupled software pieces to cooperate with each other through an exchange of XML messages over the web (Curbera *et al* 2002). The XML messages follow the Simple Object Access Protocol (SOAP) (Gudgin *et al* 2007) standard that is supported through libraries in a wide range of programming languages, making it relatively easy to build AO-DAAC access into client applications. An additional advantage is that the web service hides all the internal implementation of the data model within the database, exposing only useful functionality so that future evolution need not require changes to client code.

A web service comprises a set of functions that can be 'called' over the web. These functions are advertised in the Web Service Description Language (WSDL) document. This is an XML document that defines what the request message and response will look like. The simplest analogy is a method header that specifies what the input (request) and the output (response) types will be. Most modern Integrated Development Environments will be able to ingest this document and create Java stub classes that will interact with the web service with very little additional work.

The AO-DAAC web service has three methods to support data discovery and retrieval:
1. GetProducts: Returns a list of registered products from the database.
2.GetSubsetInfo: Returns OPeNDAP URLs of a list of products within a specified region and time range. The results are a list of "MatchedFiles", where each element will contain a record for each file

containing the product that overlaps with the requested region. Each file has information on the axes, such as the actual values at the bounds of the subset URL. Note that the web service will always return a slightly larger region than the requested region if there is not an exact match in axes values.

3. DoDirectQuery: This method allows clients to perform direct queries to the database using plain SQL. Commands that modify the database, such as DELETE, TRUNCATE, ALTER and DROP are not permitted. This method is provided to give end users direct access to the database if needed. The user would need to have good understanding of the database schema in order to use this method productively.

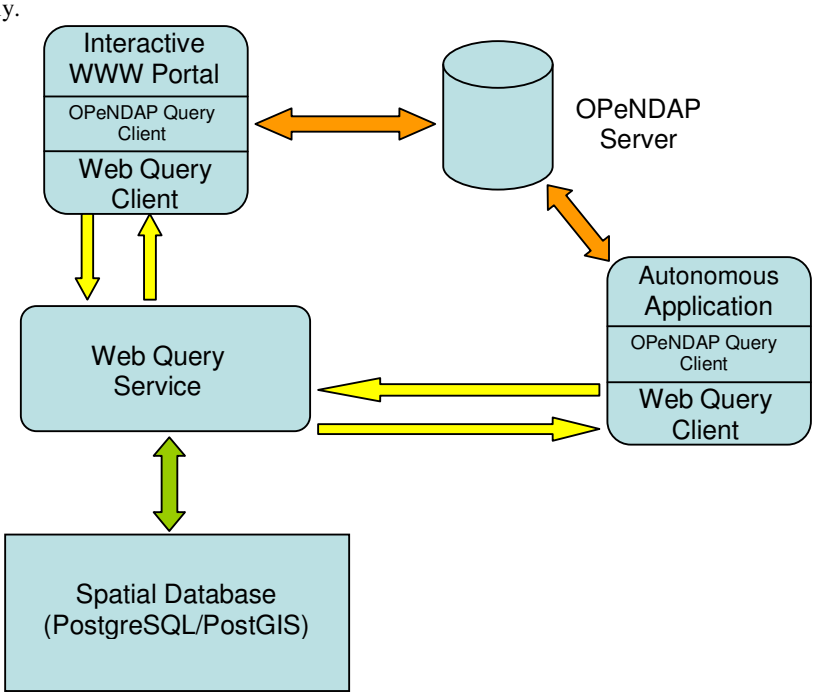

Figure 7. The Web Query Service exposes the searchable contents of the spatial database as OPeNDAP URLs that can then be used directly by clients and user applications to access data from the OPeNDAP servers.

4. Aggregator

The development effort has included the creation of a specialised client that exercises most features of the prototype system to permit creation of aggregate data sets. The aggregator value-adds to the data access process by neatly assembling all the data requested by the user and is a unique capability of the AO-DAAC. The initial version of the aggregator can only deal with gridded data and has very limited semantic and ontological understanding. The input to the aggregator is a list of OPeNDAP URLs fully qualified with sub-setting indices (*i.e.* as returned by the GetSubsetInfo function of the web service) together with an overall set of bounding limits. The aggregator reads the data directly from all the URLs and creates a data 'cube' bounded by latitude longitude and time. The aggregator is not yet able to regrid data and relies on the consistency of grids in each data set. Metadata is included by the very crude means of concatenation in temporal order. This prototype aggregator needs to be expanded to allow:

1. Re-gridding of data.
2. Use of unit semantics when aggregating.
3. Ability to manage data in different projections
4. Ability to deal with non-temporal and non-geographic data bounds (for example, altitude or depth).
5. Ability to merge data (compositing, mosaicing).

The prototype aggregator can be deployed as a user desktop application, allowing the user to tune the aggregation process to their needs and also to distribute the compute load placed on the AO-DAAC. Once it is running robustly in this mode, it will be possible to deploy the aggregator itself as a web-based service.

5. Roadmap

All the components shown in figure 3 are now built and are currently being integrated into the prototype system. These components facilitate the discovery and retrieval of data. The aggregator is a higher level interface to the system and will provide a demanding test of the integration.
Once the prototype aggregator is running and has demonstrated reasonable functionality the system will be placed online. The initial deployment will utilise existing hardware resources at iVEC and two new Sun X4500 file servers that are currently coming online in Canberra and Melbourne.
The initial application of the system will be to serve satellite derived sea surface temperature and ocean colour products to users in the marine research community. This community will encompass users with a wide range of computer skills and data requirements from small text downloads to massive chunks of time series data. The layered design of the AO-DAAC should accommodate the requirements of these users.

Future development of the software will include the generalisation of system components to support more data projections and metadata types, and integration with the authentication systems being developed by Australian Access Federation. In parallel more online marine datasets will be made visible through the system.

6. Conclusions

Integration of existing software and data standards, together with the addition of new functionality, has lead to the creation of a highly capable system that brings the national marine remote sensing community together. The use of standards for interfaces is important for several reasons: it permits interoperability, promotes consistency and enables modularity. All of these contribute to making the system highly scalable so that it should be able to grow with future data acquisitions. The aliasing mechanism enables data custodians to harmonise their existing local naming conventions with the AO-DAAC community to facilitate new data coming online. The design of the system is such that if the data holdings in the metadata database ever grow so large that searching becomes an unacceptable overhead, the database can be split by product (or groups of products) which can then be supported by separate AO-DAAC portals, each with their own database and hardware.

The large gridded data sets in use today demand operational data delivery, even for use in a research context. This project has demonstrated that it is possible to build a system that does this using a distributed architecture. It is enabling the simplification of existing applications and the development of new applications. This has again been assisted by the adoption of software standards that has enabled the incorporation of a great deal of existing code, bringing the whole task within the reach of a small group of developers.

Although the system has been developed with a marine data focus, there is little (if anything) that is marine-specific in the design. It should therefore be very easy to use this system to manage terrestrial or atmospheric gridded data sets. Furthermore, generic applications developed for the retrieval of marine data would be immediately transferable to these other domains, accelerating the path to uptake and utilisation. There is also nothing Australian-centric about this system; it could also be extended to incorporate metadata from OPeNDAP servers located globally, leading to the establishment of a global data system.

6. References

Caron, J., E. R. Davis, Y. Ho, and R. P. Kambic, 2006, Unidata's THREDDS data server, 22nd International Conference on Interactive Information Processing Systems for Meteorology, Oceanography, and Hydrology;

Cumming, I.; Hyland, G. 2005. Oceans and Climate Digital Library Portal, Proceeding of APAC05,Gold Coast, Sep 2005;

Curbera, F.; Duftler, M.; Khalaf, R.; Nagy, W.; Mukhi, N.; Weerawarana, S. 2002. Unraveling the Web services web: an introduction to SOAP, WSDL, and UDDI. Internet Computing, IEEE, 6(2): 86 – 93, March-April 2002;

Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J., Mielson, H., Kamarkar, A., Lafon, Y., 2007, SOAP Version 1.2 Part 1: Messaging Framework (Second Edition), 27 April , viewed 19 May 2008, http://www.w3.org/TR/soap12-part1/;

Gallagher,J., Potter, N., Sgouros, T., Hankin, S., Flierl, G., 2004, The Data Access Protocol – DAP 2.0, 14 September, viewed 19 May 2008, http://opendap.org/archive/design/dap/dap2/html/dap2.0.html

Gallagher, J, Potter, N, and Souros,T. , 2004, DAP Data Model Specification DRAFT – DAP4, 6 November, http:/opendap.org/pdf/dap_objects.pdf

Henry, L., Kearney, M, King, E.A, Paget, M.J., Investigation and Evaluation of the THREDDS Gridded Data Server, 2008, CSIRO Water for A Healthy Country Flagship Client Report.