GALEON OGC IE REPORT – UK NATURAL ENVIRONMENT RESEARCH COUNCIL

20 February 2006
Andrew Woolf (A.Woolf@rl.ac.uk)
CCLRC Rutherford Appleton Lab
Chilton Didcot
Oxon, OX11 0QX
UK

EXECUTIVE SUMMARY

Our experience has demonstrated that the current WCS specification fully supports the provision of netCDF data, with the following caveats:

- by exact analogy with HDF (and HDF-EOS) – the unqualified use of 'netCDF' as an output format is too abstract
- for the thematic community (atmospheric, oceanographic, meteorological, climatological) from which the GALEON IE arose – the 'CF conventions' are the de-facto standard netCDF profile
- a compliant WCS may not provide only netCDF format output – 'CF-netCDF' could be added to the existing formats to support the GALEON community
- existing GML allows the relatively straightforward encoding in binary netCDF files of coverage domainSets (using the xlink mechanism) and rangeSets (using the gml:File sub-element)
- some slight technical modifications on mandatory WCS request parameters are required to support fully-flexible subsetting of four-dimensional data

Finally, we note that an ISO TC211 (and GML) coverage is merely a subclass of an abstract feature having mandatory domain and range attributes – thus, conceptually, any product returned from a WCS must inherit from an ISO (or GML) coverage. This inheritance may be either implicit, through a well-known binary encoding profile (e.g. GeoTIFF, HDF-EOS, DTED, NITF, or CF-netCDF), or explicit, through a GML element inheriting from gml:AbstractDiscreteCoverageType.

Our specific recommendations include:

1. *GML consider the CSML grid schemas for irregular rectified grids.*
2. *ISO 19111 be modified as proposed by the UK standards committee IST/36, and GML be harmonised accordingly.*
3. *GML be rationalised with respect to referencing coverage domain and range sets encoded in binary files (including netCDF). Profiling the existing xlink mechanism seems a useful approach.*
4. *netCDF should be enabled as a supported output format from WCS; to be consistent with the current WCS model (e.g. as for the case of HDF-EOS), the profile of netCDF defined by the CF conventions for climate data should be required – the keyword "CF-netCDF" is suggested for the format specifier.*
5. *The WCS specification be clarified with respect to the ordering of CRS axes in both service metadata (CoverageOfferingBrief, CoverageOffering) and request parameters (BBOX). It would be preferable to require consistency between BBOX (and service metadata) and the data CRS – thus BBOX for EPSG:4979 (for example) should be specified as BBOX=lat1,lon1,height1,lat2,lon2,height2.*
6. *WMS and WCS BBOX and ELEVATION request parameters be harmonised.*
7. *Eliminate the mandatory requirement on WIDTH and HEIGHT (or RESX and RESY) to support arbitrary slicing through four dimensions of data.*
8. *The WCS specification should clarify the difference between subsampling and interpolation – a server may possibly support one but not the other, and the difference is important; in particular, the semantics of the RES{X,Y,Z} (or WIDTH, HEIGHT, DEPTH) request parameters should be clarified.*
9. *Specific conventions on the mapping from netCDF to WCS metadata elements would be useful for consistency of WCS deployment for netCDF data.*
10. *Clarification is needed on the respective roles of 'temporalDomain' and 'spatialDomain/EnvelopeWithTimePeriod' for time-dependant data.*

WCS servers deployed during the GALEON IE are at:

http://glue.badc.rl.ac.uk/cgi-bin/mapserv?map=/var/www/html/jiscInterop/nerc.map

http://glue.badc.rl.ac.uk/cgi-bin/TPAC/WCS

1. WCS ACTIVITY IN NERC

1.1 NERC DataGrid[1]

The current Open Source WCS solutions were evaluated for their suitability as a WCS for netCDF data.

The three main tools evaluated were deegree, GeoServer and Mapserver.

A basic WMS (for visual demonstration) was set up using deegree that read directly from a NetCDF file, however it wasn't simple to create a fully working connector for netCDF within the framework of deegree. We found the codebase difficult to extend, and found this impression consistent with external collaborators. A greater degree of familiarity may have enabled greater success.

Most progress was made with Mapserver. A tool was initially written to convert CF-compliant NetCDF into GMT netcdf, as supported by Mapserver/GDAL. This enabled a two stage process to serve netCDF through Mapserver. Then a rudimentary netCDF driver for GDAL was developed, which allowed us to set up a Mapserver WCS and WMS serving directly from NetCDF data. A very similar netCDF driver is now included in the GDAL distribution. However there are limitations with this approach as GDAL was not written with the purpose of reading and writing n-dimensional data in mind – in fact it is limited fundamentally to two-dimensional data[2].

The URL for the NDG Mapserv WCS is:

http://glue.badc.rl.ac.uk/cgi-bin/mapserv?map=/var/www/html/jiscInterop/nerc.map

At the time of investigation the GeoServer WCS was not fully developed, so we didn't pursue this line of investigation very far. However since this investigation it seems significant progess is being made on nD data delivery with Geoserver[3]. It is likely that future geoserver development will be of significant strategic relevance for netCDF data.

1.2 DEWS[4]

The task of the DEWS (Delivering Environmental Web Services) project is to design and create a data delivery system for different types of UK Met Office data. One aspect of the project is to deliver ocean forecast data (four-dimensional gridded data) via a WCS. This WCS server will be known as GADS-WCS as it is a modification of the University of Reading's GADS system.

GADS-WCS will match the WCS specification as far as possible, but some modifications will need to be made. Data will be available in NetCDF format only. Also, users will be able to download very large amounts of data in a single query. In this case the process of data extraction will take significant time and it will not be practical to return the data as a direct response to the request.

---

1 NERC DataGrid, http://ndg.nerc.ac.uk

2 See, for example, the thread on the GDAL developers list at http://lists.maptools.org/pipermail/gdal-dev/2005-August/006219.html

3 See the Geoserver WCS development wiki at http://docs.codehaus.org/display/GEOS/Experimental

4 Delivering Environmental Web Services, http://www.dews.org.uk

In order to handle these large data extractions, we are incorporating elements of the Web Processing Server (WPS) into the design of GADS-WCS.

The WPS specification describes the interface to a server that performs potentially-lengthy processing on geospatial data. We are effectively treating the data extraction process as a lengthy processing step. Having made a request for data, users will be able to use the WPS mechanisms to monitor the status of the data extraction "job" and retrieve the data when the job is finished. GADS-WCS will be backward-compatible with WCS as far as possible: for small data extractions GADS-WCS will be able to return data as a direct response to the data request.

For more details about GADS-WCS or the DEWS project please contact Jon Blower of the University of Reading at jdb@mail.nerc-essc.ac.uk.

1.3 TPAC-NDG Collaboration[5]

An experimental prototype WCS has been developed collaboratively by TPAC and NDG. It currently provides a WCS wrapper over OPeNDAP/DODS datasets and forms part of TPAC's work developing a Grid infrastructure (data and compute) in Australia for earth systems science. The approach is based around web services, and OGC interfaces are seen as key.

1.3.1 Server architecture

The server architecture is shown in Figure 1 below.

[5] TPAC is the Tasmanian Partnership for Advanced Computing (http://www.tpac.org.au). An ongoing collaboration with NDG has been facilitated through both the AUKEGGS (https://www.seegrid.csiro.au/twiki/bin/view/AUKEGGS/WebHome) collaboration umbrella and a recent TPAC-funded secondment.

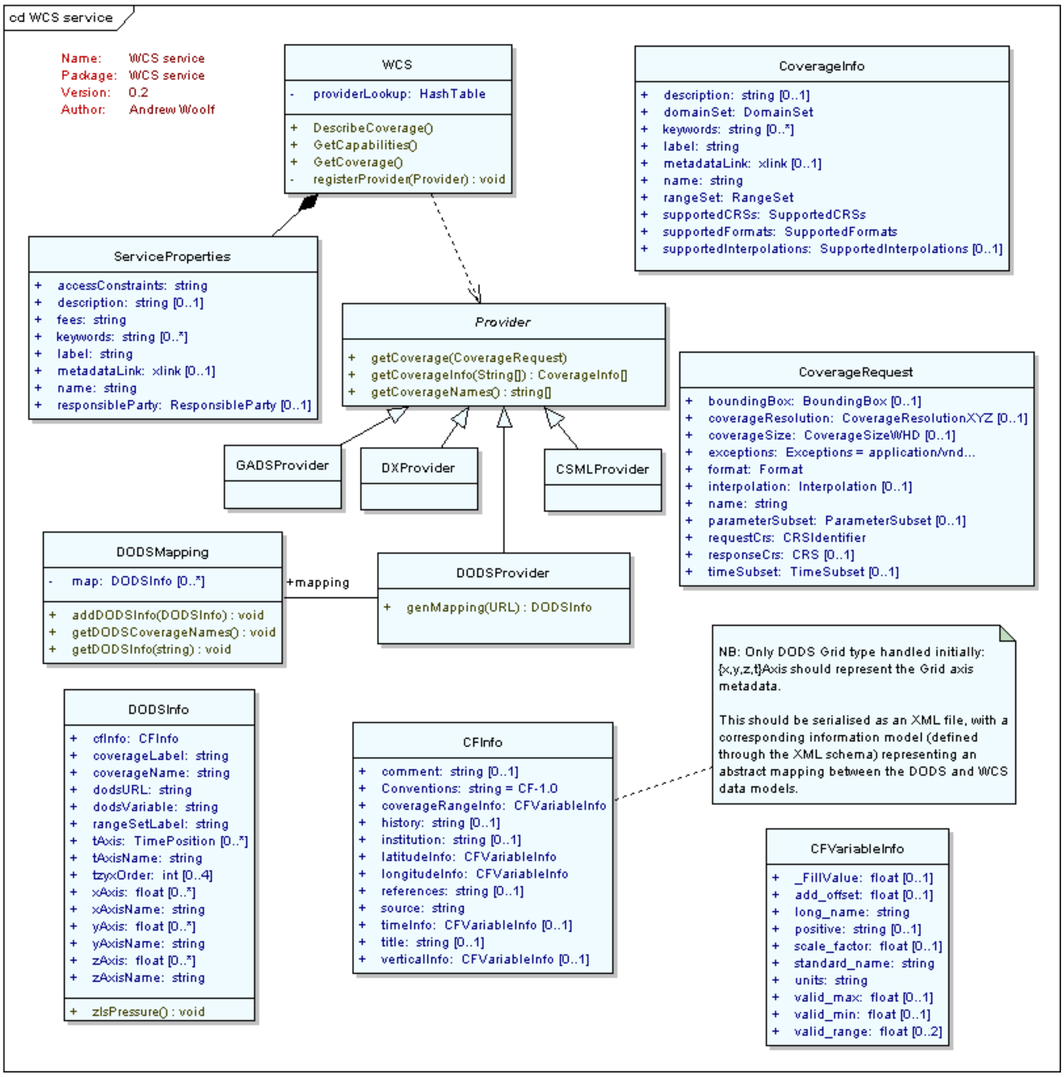


Figure 1: Architecture of WCS server

The currently implemented DODS provider is motivated by recognising the substantial conceptual overlaps between OPeNDAP and WCS functionality. OPeNDAP provides a more general abstract data model than that underlying WCS (i.e. a gridded coverage over some spatiotemporal domain). However, for a large number of cases where OPeNDAP is used for serving netCDF files in the earth sciences, there is a great deal in common.

Features of the server include:

- the ability to plug in multiple backend providers – currently an OPeNDAP provider has been implemented, but the architecture is intended to be able to work with NDG's DataExtractor[6] tool, the University of Reading's GADS[7], the Climate Science Modelling Language[8], etc.

---

[6] The Data Extractor (dx), http://home.badc.rl.ac.uk/astephens/software/dx/

[7] GADS (Grid Access Data Service), http://www.resc.rdg.ac.uk/twiki/bin/view/Resc/GridAccessDataService. Note that WCS refactoring of

- Exploiting the full generality of the WCS spec to allow arbitrary subsetting in time and space; i.e. a four-dimensional (x-y-z-t) dataset may be sliced in one, two, three or four dimensions). This is crucial for the earth sciences where, for example, a four-dimensional data volume may need to be sliced to extract a longitude-time array at fixed latitude and height.

- Returned data is always netCDF compliant to the community standard CF conventions[9], regardless of the underlying data source. Thus, standard names from a controlled vocabulary are used to define the physical parameter of the coverage rangeSet, and axes are always ordered t-z-y-x on the returned netCDF coverage.

- A majority of the information required for internal server configuration is available by querying the OPeNDAP server for DAS and DDS metadata[10].

1.3.2 Prototype deployment

The prototype is currently deployed at the British Atmospheric Data Centre (UK) at the following URL:

http://glue.badc.rl.ac.uk/cgi-bin/TPAC/WCS

A static page providing example WCS URLs is available at that address.

2. GML ISSUES

2.1 Irregularly-spaced Grids

Numerical circulation models of atmosphere and ocean typically are formulated on finite-difference grids (Figure 2).

GADS is occurring as part of the DEWS project (see Section 1.2), independent of the TPAC-NDG prototype.

8 Climate Science Modelling Language, http://ndg.nerc.ac.uk/csml/

9 NetCDF Climate and Forecast (CF) Metadata Convention, http://www.cgd.ucar.edu/cms/eaton/cf-metadata/CF-1.0.html

10 See OPeNDAP documentation at http://www.opendap.org for details.

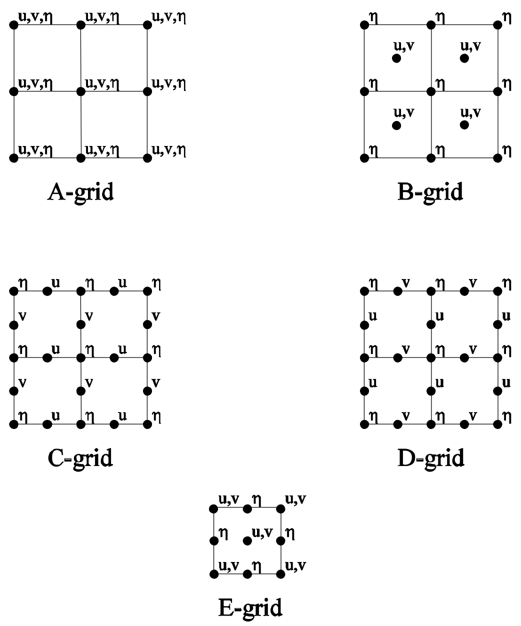


Figure 2: Finite-difference 'Arakawa' grids typically used for metocean models

The following issues result:

- While georeferenced ('rectified'), the grids are rarely regularly spaced. For instance, circulation models may have increased resolution over some arbitrary region of interest; or a conformal mapping may be applied, for instance to relocate the pole and avoid a meridional convergence singularity.

- Grids may be 'staggered' – consider, for instance, the 'η' points in the E-grid of Figure 2.

- Numerical models may be formulated on spectral coordinates, with a transformation required for geographic coordinates. To ensure alias-free transformations between spectral and spatial domains, the resulting spatial grids have fewer longitudinal points towards the poles.

GML provides only for regularly-spaced rectified grids, and this flows through to the WCS specification:

- the DescribeCoverage response CoverageDescription/CoverageOffering/domainSet/spatialDomain may provide only a (regularly-spaced) gml:RectifiedGrid for specifying geolocated grids.

CSML extends gml:GridType for arbitrarily-spaced geo-referenced grids (see section 1)

Recommendation 1: GML consider the CSML grid schemas for irregular rectified grids.

2.2 Vertical coordinate reference systems

Atmospheric and oceanographic data extend over the three spatial dimensions. A rich spectrum of vertical coordinate reference systems are routinely used for metocean data (see Table 1).

Table 1: **Selection of vertical coordinate reference systems**

Coordinate reference system	Vertical coordinate axis

Pressure	Static pressure
Isentropic	Entropy or potential temperature
Sigma	Normalised terrain-following
z	Lineal distance

Work is needed to classify and characterise the full range of vertical CRS used for metocean data. The widely used EPSG codes, for instance, omit the complexity of metocean vertical CRS.

The issue is reflected in recent revisions of GML and ISO 19111. GML v3.0.0 (OGC 02-023r4) provided for a vertical datum (gml:VerticalDatumTypeType) to be one of 'geoidal', 'depth', 'barometric' or 'othersurface', while GML v3.1.0 reverted to any referenceable external datum definition (gml:CodeType with required gml:codeSpace).

Arising from NERC concerns, a New Work Item Proposal is likely soon to be proposed by the UK to ISO TC211 for an extension to ISO 19111 "Geographic information — Spatial referencing by coordinates - extension for parametric values". The NWIP scope will be along the following lines:

> To define an extension of ISO 19111:2005 covering coordinate referencing systems where the coordinates are physical or material properties measured by parameters, such as pressure, rather than linear distance measurements. It defines a conceptual schema for the description of spatio-parametric coordinate reference systems. These parametric spatial reference systems will be modelled so that they may be associated with an ISO 19111 [horizontal] spatial coordinate reference system to form a compound spatio-parametric coordinate reference system.

Recommendation 2: ISO 19111 be modified as proposed by the UK standards committee IST/36, and GML be harmonised accordingly.

As part of its commitment to the CF-metadata standard[11], the British Atmospheric Data Centre is considering the issue of formally cataloguing vertical coordinate reference systems used in the climate sciences, along the lines already implemented by EPSG for spatial CRS.

2.3 Binary encodings in GML

GML currently allows resources to be identified by reference, using the W3C 'xlink'[12] recommendation. This 'by-reference' pattern is implemented in many places throughout the GML schemas – providing a flexible means to separate content management.

In addition, GML provides the 'gml:File' element, with 'gml:mimeType' and 'gml:fileStructure' elements to efficiently encode large coverage range sets. However, such encoding efficiency may be needed also for coverage domain sets – or, indeed, any numerical content at all.

These two mechanisms should be harmonised – either explicit support should be provided to describe file-based encoding of coverage domainSets (and other GML numerical content), or else profiles should be defined on the generic 'xlink' pattern to reference coverages (domain and range) within files.

An example of encoding a reference to a netCDF file coverage currently would be:

---

11 Ref email archived at: http://www.cgd.ucar.edu/pipermail/cf-metadata/2006/000812.html.

12 http://www.w3.org/TR/2001/REC-xlink-20010627

```xml
<gml:RectifiedGridCoverage gml:id="netcdf_ID0">
    <gml:RectifiedGridDomain xlink:href="http://myserver/myfile.nc#temperature"
xlink:role="http://netcdf/GMLOps/Grid/spatialDomain"/>
    <gml:rangeSet>
        <gml:File>
            <gml:rangeParameters>
                <gml:Quantity uom="degC"/>
                <fileReference>http://myserver/myfile.nc</fileReference>
                <fileStructure>Record Interleaved</fileStructure>
                <mimeType>application/x-netcdf</mimeType>
            </gml:rangeParameters>
        </gml:File>
    </gml:rangeSet>
</gml:RectifiedGridCoverage>
```

Figure 3: Encoding reference to netCDF Grid variable (domain and range) using current GML capabilities.

In the above example, the xlink:role (http://netcdf/GMLOps/Grid/spatialDomain) is used to indicate an assumed well-known procedure for dereferencing netCDF Grid axes to a GML domainSet.

Alternatively, CSML provides a unified approach to this problem (see section 2 of the appendix) – additionally providing file aggregation capabilities.

Recommendation 3: GML be rationalised with respect to referencing coverage domain and range sets encoded in binary files (including netCDF). Profiling the existing xlink mechanism seems a useful approach.

3. WCS ISSUES

3.1 CF-netCDF output format

While the WCS specification does not prevent netCDF (or any other) format data being served, it may not be provided as the *only* output format.

netCDF as a data encoding format for coverage data is *abstract* in the sense that a given coverage instance may be encoded in netCDF in a variety of ways. The netCDF data model consists only of abstract *variables*, *dimensions* and *attributes*[13]. Conventions are required for applying the abstract netCDF file format to any given coverage data source. The situation is exactly analogous to the specification of HDF-EOS[14] as a profile of the abstract HDF[15] format for earth observation data.

The CF conventions[9] for climate (atmospheric, oceanographic, meteorological) data are the *de-facto* community standard netCDF profile for this thematic domain.

Thus, in order to be consistent with the current approach of WCS, netCDF should be supported as an output format – restricted through the CF conventions for climate-science data.

Recommendation 4: netCDF should be enabled as a supported output format from WCS; to be consistent with the current WCS model (e.g. as for the case of HDF-EOS), the profile of

[13] See netCDF documentation at http://www.unidata.ucar.edu/software/netcdf/.
[14] Hierarchical Data Format - Earth Observing System (HDF-EOS), http://nsidc.org/data/hdfeos/
[15] The NCSA HDF Home Page, http://hdf.ncsa.uiuc.edu/

netCDF defined by the CF conventions for climate data should be required – the keyword "CF-netCDF" is suggested for the format specifier.

3.2 CRS issues

The extension of WCS to netCDF data – particularly as used by the climate science community – challenges existing approaches to coordinate reference systems. WCS is *almost* flexible enough to allow arbitrary slices through a four-dimensional (x,y,z,t) volume. We consider some specific issues below, arising through the implementation of the prototype TPAC-NDG WCS server.

3.2.1 CRS inconsistency with BBOX

The WCS specification states in respect of the GetCoverage parameter BBOX:

> A GetCoverage request may include a 1-D, 2-D, or 3-D spatial constraint expressed as a rectangle (or line, or parallelepiped) aligned with the axes of the spatial reference system given in the CRS parameter. Such a constraint is expressed as a BBOX parameter representing the coordinates of the southwest/lower and northeast/upper corners (in that order) as comma-separated numbers (e.g., minx, miny, maxx, maxy).
>
> NOTE The order (southwest, northeast) often corresponds to (minimum x, minimum y, maximum x, maximum y) – but this is not always the case. For instance, when a Bounding Box expressed in longitude and latitude crosses the antimeridian (the meridian with longitude +/-180 degrees), its northeast corner's longitude is often less than that of its southwest corner.
>
> Each corner's coordinate(s) must be expressed in the order and units given by the CRS.

While it is noted that (minx,miny,maxx,maxy) is not always an appropriate description for any given bounding box, the use of longitude before latitude is nevertheless implied – even where this is contrary to the CRS definition. In particular, the standard "EPSG:4326" and "EPSG:4979" WGS-84 CRS require latitude before longitude.

A more consistent approach would be for BBOX to specify two bounding corners *in the CRS of the data*. Thus, for EPSG:4979 (three-dimensional lat-lon-height), the BBOX would take the form: BBOX=lat1,lon1,height1,lat2,lon2,height2.

Recommendation 5: The WCS specification be clarified with respect to the ordering of CRS axes in both service metadata (CoverageOfferingBrief, CoverageOffering) and request parameters (BBOX). It would be preferable to require consistency between BBOX (and service metadata) and the data CRS – thus BBOX for EPSG:4979 (for example) should be specified as BBOX=lat1,lon1,height1,lat2,lon2,height2.

While the WMS specification allows for slices in the vertical, it uses a different mechanism to WCS (an 'ELEVATION' parameter and 'BBOX=minx,miny,maxx,maxy' *vs* the WCS 'BBOX=minx,maxx,miny,maxy,minz,maxz'). In addition, there are few, if any, deployed examples of WMS supporting maps of arbitrary slices.

Recommendation 6: WMS and WCS BBOX and ELEVATION request parameters be harmonised.

3.2.2 WIDTH,HEIGHT,DEPTH and RESX,RESY,RESZ for arbitrary slices

In general, a WCS subset through a four-dimensional (x-y-z-t) volume of data may select an arbitrary chunk of data with dimension zero (a single point), one (a line of data along any axis), two (a slice of data through any plane), three, or four. The mandated use of WIDTH and HEIGHT (or RESX and RESY) is inconsistent with this. For example, consider the use of

EPSG:4979 with three-dimensional (lat-lon-height) data. A request for a two-dimensional slice of data in the vertical plane through the Greenwich meridian from the ground to 2km would be specified using[16]: BBOX=0,-90,0,90,0,2000. In this case the WIDTH or RESX parameters for specifying grid width or longitude resolution are irrelevant and should not be mandated.

Recommendation 7: Eliminate the mandatory requirement on WIDTH and HEIGHT (or RESX and RESY) to support arbitrary slicing through four dimensions of data.

3.3 Interpolation *vs* subsampling

The WCS model provides for server-side interpolation, but says nothing about subsampling. This is a common operation for large gridded datasets – the original data may be required (without interpolation or averaging), but at a reduced sampling rate. The netCDF API[13], for instance, provides a standard read method for subsampling by specifying a start, count and stride along each grid axis.

The semantics of RES{X,Y,Z} (or WIDTH, HEIGHT, DEPTH) should be clarified in the WCS specification where interpolation is not supported, but server-side subsampling may be.

Recommendation 8: The WCS specification should clarify the difference between subsampling and interpolation – a server may possibly support one but not the other, and the difference is important; in particular, the semantics of the RES{X,Y,Z} (or WIDTH, HEIGHT, DEPTH) request parameters should be clarified.

3.4 Exposing netCDF attributes in WCS dataset metadata

The TPAC-NDG WCS prototype has made a mapping from the CF-netCDF variable name, 'long_name', and 'standard_name' attributes to the WCS 'name', 'label' and 'rangeSet/label' metadata elements. Further discussion is needed around the most suitable conventions for mapping netCDF metadata to WCS metadata.

Recommendation 9: Specific conventions on the mapping from netCDF to WCS metadata elements would be useful for consistency of WCS deployment for netCDF data.

3.5 The TIME request parameter

A potential inconsistency exists in the WCS 1.0.0 specification – section 9.2.2.8 specifies that a TIME parameter may be specified "(i)f the DescribeCoverage XML reply defines a TemporalDomain on the selected coverage...". However, this is not the only way a server may indicate a coverage on a temporal domain – section 8.3.2 specifies that a DescribeCoverage response may provide a spatialDomain and/or a temporalDomain, but a spatialDomain *itself* may provide a gml:EnvelopeWithTimePeriod.

The specification is unclear about whether the temporalDomain element is mandatory (in addition to spatialDomain/EnvelopeWithTimePeriod) for the case of time-dependant data.

If temporalDomain is not mandatory, clarification is required on the reference 'frame' for specifying temporal limits in a request. Section 9.2.2.8 specifies that the frame in a request should be consistent with that defined (or implied) by the DescribeCoverage 'temporalDomain' element – which may not be mandatory.

---

[16] We use here the (inconsistent) existing WCS conventions of BBOX=minx,miny,maxx,maxy,minz,maxz, and suggest in section 3.2.1 a more consistent approach.

Recommendation 10: Clarification is needed on the respective roles of 'temporalDomain' and 'spatialDomain/EnvelopeWithTimePeriod' for time-dependant data.

4. DOES WCS NEED REVISION TO SUPPORT NETCDF?

Our experience has demonstrated that the current WCS specification fully supports the provision of netCDF data, with the following caveats:

- by exact analogy with HDF (and HDF-EOS) – the unqualified use of 'netCDF' as an output format is too abstract
- for the thematic community (atmospheric, oceanographic, meteorological, climatological) from which the GALEON IE arose – the 'CF conventions' are the *de-facto* standard netCDF profile
- a compliant WCS may not provide *only* netCDF format output – 'CF-netCDF' could be added to the existing formats to support the GALEON community
- existing GML allows the relatively straightforward encoding in binary netCDF files of coverage domainSets (using the xlink mechanism) and rangeSets (using the gml:File sub-element)
- some slight technical modifications on mandatory WCS request parameters are required to support fully-flexible subsetting of four-dimensional data

Finally, we note that an ISO TC211[17] (and GML) coverage is merely a subclass of an abstract feature having mandatory domain and range attributes – thus, conceptually, any product returned from a WCS must inherit from an ISO (or GML) coverage. This inheritance may be either *implicit*, through a well-known binary encoding profile (e.g. GeoTIFF, HDF-EOS, DTED, NITF, or CF-netCDF), or *explicit*, through a GML element inheriting from gml:AbstractDiscreteCoverageType.

[17] See ISO 19123 "Geographic information — Schema for coverage geometry and functions"

APPENDIX: CSML

The Climate Science Modelling Language[8] has been developed by the NERC DataGrid as a set of feature types and GML application schema for standards-based integration of climate-related data – both observational and model. It will be used as a wrapper technology enabling the provision of a range of data delivery services in the UK NERC DataGrid[18] project. We provide below some extracts from the "CSML v0.1 User's Manual" relevant to the WCS discussion in this report.

1. CSML GRIDS

GML supports only non georeferenced grids, or grids with equally-spaced gridpoints in some coordinate system. Grids for metocean data may not be equally spaced in any coordinate reference system. For instance, ocean or atmosphere general circulation models often have increased resolution over some arbitrary region of interest. Ocean models often have an

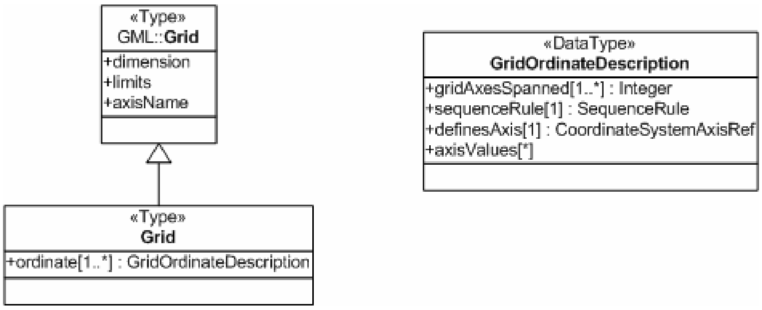


Figure 4: CSML Grid conceptual model

artificially relocated 'north pole' to avoid the meridional convergence singularity. We describe here a mechanism for defining general georeferenced grids in CSML. The conceptual model is shown in Figure 4.

1.1.1 GridOrdinateDescription

A Grid is modelled as being embedded in some space of dimension no smaller, but possibly greater, than the dimension of the Grid itself. For example, a one-d grid may represent a series of points in three-d space. Similarly a two-d slanted grid may extend through three space dimensions. To characterise the gridpoint locations, a GridOrdinateDescription is provided for each space dimension within which the grid is embedded.

- **gridAxesSpanned**: the grid axes along which this space dimension varies, e.g. the latitude ordinate spans only one axis of a lat-lon aligned grid, but spans both axes of a horizontally rotated grid

- **definesAxis**: the particular space dimension being defined, as a reference to a coordinate system definition (e.g. the 'latitude' axis, referenced appropriately)

- **axisValues**: the ordinate values for this space dimension over the gridpoints spanned by it; the 'xlink' attributes of gml:AssociationAttributeType are provided in order to reference numerical values that may be defined indirectly.

- **sequenceRule**: a rule specifying the order of the gridpoints with respect to the supplied axisValues (see GML v3.0.0, section 19.3.14)

[18] NERC DataGrid, http://ndg.nerc.ac.uk.

1.1.2 Grid

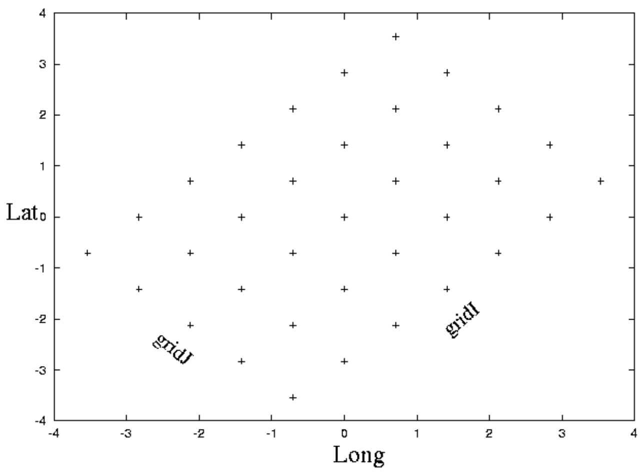
A CSML Grid is subclassed from a GML Grid, and defined by specifying a GridOrdinateDescription for each space dimension within which the grid is embedded. It inherits the following properties from gml:GridType:

- **dimension**: the dimensionality of the grid.
- **limits**: indicates the 'logical' limits of the respective grid indices, e.g. a grid axis of size 5 may have logical index limits of –1 to 3.
- **axisName**: provides a name for each logical grid axis (NB: since the grid doesn't have to be aligned with axes of any CRS, the grid axisName is unlikely to be something like 'latitude', more likely something like 'gridI')

The following useful properties are also inherited (because a gml:GridType derives, in turn, from gml:AbstractGeometryType):

- **srsName**: a reference to the coordinate reference system within which this grid is embedded
- **srsDimension**: the dimension of this coordinate space; may be greater than or equal to the grid 'dimension'

Figure 5 gives an example of a two-dimensional grid sitting in a horizontal plane, and regularly-spaced but rotated through 45°. Thus, the (-3,-2) gridpoint has coordinates (-0.707° E, -3.54° N), and the (1,2) gridpoint has coordinates (-0.707° E, 2.12° N).




```
<Grid srsName="urn:EPSG:geographicCRS:4326" srsDimension="2" dimension="2">
    <gml:limits>
        <gml:GridEnvelope>
            <gml:low>-3 -2</gml:low>
            <gml:high>3 2</gml:high>
        </gml:GridEnvelope>
    </gml:limits>
    <gml:axisName>gridI</gml:axisName>
    <gml:axisName>gridJ</gml:axisName>
    <ordinate>
        <gridAxesSpanned>gridI gridJ</gridAxesSpanned>
        <sequenceRule order="+y+x">Linear</sequenceRule>
        <definesAxis xlink:href="#Latitude"/>
        <axisValues>-3.54 -2.83 -2.12 -1.41 -0.70
        -2.83 -2.12 -1.41 -0.707 0
```

```
        -2.12 -1.41 -0.707 0 0.707
        -1.41 -0.707 0 0.707 1.41
        -0.707 0 0.707 1.41 2.12
        0 0.707 1.41 2.12 2.83
        0.707 1.41 2.12 2.83 3.54</axisValues>
    </ordinate>
    <ordinate>
        <gridAxesSpanned>gridI gridJ</gridAxesSpanned>
        <sequenceRule order="+y+x">Linear</sequenceRule>
        <definesAxis xlink:href="#Longitude"/>
        <axisValues>-0.707 -1.41 -2.12 -2.83 -3.54
        0 -0.707 -1.41 -2.12 -2.83
        0.707 0 -0.707 -1.41 -2.12
        1.41 0.707 0 -0.707 -1.41
        2.12 1.41 0.707 0 -0.707
        2.83 2.12 1.41 0.707 0
        3.54 2.83 2.12 1.41 0.707</axisValues>
    </ordinate>
</Grid>
```

Figure 5: Example encoding of CSML Grid.

2. NUMERICAL ARRAY DESCRIPTORS

While canonical GML utilises inline encoding for numeric values (except coverage rangeSets, which support file-based encoding through the gml:File element) the CSML application schema provides a generic mechanism for defining numeric arrays, as shown in Figure 6.

Numerical array descriptors may be used in two ways in a CSML Dataset – for defining both domain, and range, of coverages. They may be used as the direct content for a GML coverage

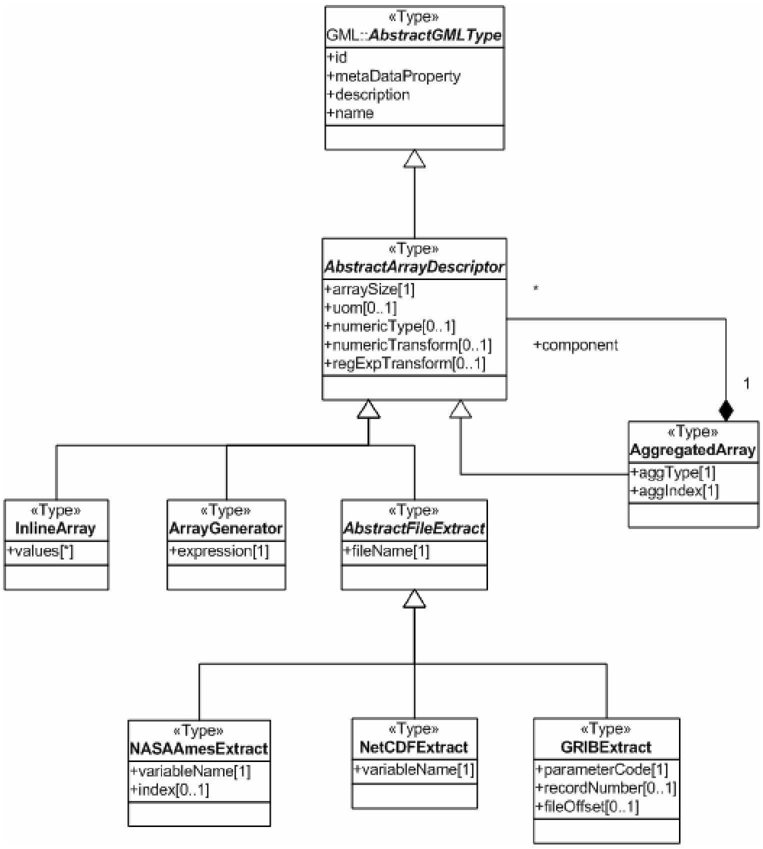


Figure 6: CSML numeric array definition

rangeSet element, and as a standalone descriptor that may be referenced (using xlink attributes) from coverage domainSet elements. The reason for the 'xlinked' content model for the latter is that CSML domains are based ultimately on GML geometry classes with XML lists of numbers – i.e. simple content types that may be supplemented only by attributes (an xlink to the array descriptor in this case) and not child elements. To support this, CSML array descriptors all derive from the GML AbstractGMLType, which allows a unique ID to be specified.

The following four array descriptor subclasses are defined:

- InlineArray: an inline character encoding of array values
- ArrayGenerator: a formulaic expression implicitly generating array values
- AbstractFileExtract: values extracted from file-based storage
- AggregatedArray: an aggregation of two or more arrays along an existing or new dimension

In addition a numeric or regular expression transformation can be applied to any of the above as appropriate.

2.1 AbstractArrayDescriptor

This is the abstract base class for all CSML numeric array descriptors. As well as standard attributes inherited from gml:AbstractGMLType, the following attributes are defined, and inherited by all concrete classes:

- **arraySize**: a sequence of integers specifying the dimensions of the array being defined.
- **uom**: units of measure. This is required whenever the array descriptor is used for a coverage rangeSet (when used for the domainSet, the coordinate axis units are specified in the parent geometry object).
- **numericType**: a string ('float', 'double', 'short', 'int') defining the numeric type to which the values should be cast in a user application.
- **numericTransform**: an expression providing a mathematical transformation to be applied to the numeric values defined by this descriptor.
- **regExpTransform**: a regular expression providing a textual transformation to be applied to the direct character-based data defined by this descriptor (e.g. where inline values are supplied, or the file is ASCII-encoded). This is applied before any cast to a numerical type within the user application

2.2 InlineArray

This class is used for specifying values as direct inline XML content. The only attribute is:

- **values**: a list of numeric values

An example XML encoding is shown in Figure 7. This example utilises all the optional attributes, and demonstrates both the regular expression and numeric transformation. The result of this array descriptor is a five-by-two array of temperatures ranging from 6 to 14 with the final two values equal.

```
<InlineArray>
    <arraySize>5 2</arraySize>
    <uom>udunits.xml#degreeC</uom>
    <numericType>float</numericType>
    <regExpTransform>s/10/9/ge</regExpTransform>
    <numericTransform>+5</numericTransform>
    <values>1 2 3 4 5 6 7 8 9 10</values>
</InlineArray>
```

Figure 7: InlineArray for value array.

2.3 ArrayGenerator

If the numeric values in an array follow a simple pattern, it is more efficient to encode them implicitly through a formulaic expression. The ArrayGenerator class provides this facility. One attribute is defined:

- **expression**: provides a formulaic expression defining values to be generated implicitly

An example XML encoding is shown in Figure 8. This example generates a sequence of 10001 values representing five minute increments in time from zero.

```
<ArrayGenerator>
    <arraySize>10001</arraySize>
    <uom>udunits.xml#minute</uom>
    <numericType>float</numericType>
    <expression>0:5:50000</expression>
</ArrayGenerator>
```

Figure 8: XML encoding example for ArrayGenerator.

2.4 AbstractFileExtract

Numeric values may be defined in file-based storage. The AbstractFileExtract and its subclasses are used for this purpose. The following attribute is defined:

- **fileName**: a pathname to the file.

The class is subclassed for NASA Ames, netCDF and GRIB files.

2.4.1.1 NASAAmesExtract

A NASAAmesExtract is used where data is being extracted from a NASA Ames format file. The following attributes provide the minimal information required to identify the data to be extracted:

- **variableName**: the name of the variable in the NASA Ames file

- **index**: there is no requirement for NASA Ames variable names to be unique. In the event of duplication, the "index" attribute allows a specific instance to be identified

An example is given in Figure 9.

```
<NASAAmesExtract>
    <arraySize>526</arraySize>
    <numericType>double</numericType>
    <fileName>/data/BADC/macehead/mh960607.asv</fileName>
    <variableName>coefficient a1</variableName>
</NASAAmesExtract>
```

Figure 9: Encoding for NASAAmesExtract.

2.4.2 *NetCDFExtract*

Only a variable name attribute is required in addition to the file name to identify an extract of data from a netCDF file:

- **variableName**: the netCDF variable defining the contents of this numerical array

An example is shown in Figure 10.

```xml
<NetCDFExtract gml:id="feat04azimuth">
    <arraySize>10000</arraySize>
    <fileName>radar_data.nc</fileName>
    <variableName>az</variableName>
</NetCDFExtract>
```

Figure 10: Encoding of NetCDFExtract.

2.4.3 *GRIBExtract*

The GRIBExtract class represents data extracted from a GRIB format file. In order to uniquely identify the data required, the following parameters are used:

- **parameterCode**: the GRIB parameter code for the GRIB record to be extracted

- **recordNumber**: since a GRIB file can contain any number of records of the same parameter, the specific record number may be indicated

- **fileOffset**: as an alternative to specifying the record number, an explicit offset within the file to the start of the required GRIB record may be specified (this avoids having to step through the file over earlier records until the required record is found)

An XML example is shown in Figure 11.

```xml
<GRIBExtract>
    <arraySize>320 160</arraySize>
    <numericType>double</numericType>
    <fileName>/e40/ggas1992010100rsn.grb</fileName>
    <parameterCode>203</parameterCode>
    <recordNumber>5</ recordNumber>
    <fileOffset>289412</fileOffset>
</GRIBExtract>
```

Figure 11: XML encoding for GRIBExtract.

2.5 AggregatedArray

The composite pattern is applied so that an array descriptor may be an aggregation of other array descriptors. Arrays may be aggregated along either an existing dimension (thus maintaining the dimensionality of the arrays), or along a new dimension (thus creating an (n+1)-dimensional array from n-dimensional arrays). The following properties are used:

- **aggType**: a character string indicating aggregation along either an 'existing' or 'new' array dimension

- **aggIndex**: the dimension index along which aggregation is performed. For example two 100x200 arrays may be aggregated along the existing index '1' to produce a 200x200 array, or along the existing index '2' to produce a 100x400 array. Aggregation along a new dimension '1' would create a 2x100x200 array; along a new dimension '2' would create a 100x2x200 array, etc.