Project No. 249024

NETMAR

Open service network for marine environmental data

Instrument: *Please tick*	CA	✓ **STREP**	IP	NOE

ICT - Information and Communication Technologies Theme

D5.3.2 WPS Server – Fully semantically enabled processing services

Reference: D5.3.2_WPS_Server_Semantic_r1_20120430

Due date of deliverable (as in Annex 1): M0 + 27
Actual submission date: 30 April 2012

Start date of project: 1 February 2010 Duration: 3 years

Plymouth Marine Laboratory

Revision 1

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)

Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

NETMAR
Open service network for marine environmental data
Project Reference: 249024
Contract Type: Collaborative Project
Start/End Date: 01/03/2010 - 31/01/2013
Duration: 36 months

Coordinator: Prof. Stein Sandven
Nansen Environmental and Remote Sensing Center
Thormøhlensgate 47, Bergen, Norway
Tel.: +47 55 20 58 00
Fax. +47 55 20 58 01
E-mail: stein.sandven@nersc.no

Acknowledgements

The work described in this report has been partially funded by the European Commission under the Seventh Framework Programme, Theme ICT 2009.6.4 ICT for environmental services and climate change adaptation.

Consortium

The NETMAR Consortium is comprised of:

- Nansen Environmental and Remote Sensing Center (NERSC), Norway (coordinator).
  Project Coordinator: Prof. Stein Sandven (stein.sandven@nersc.no)
  Deputy Coordinator: Dr. Torill Hamre (torill.hamre@nersc.no)
  Quality Control Manager: Mr. Lasse H. Pettersson (lasse.pettersson@nersc.no)
- British Oceanographic Data Centre (BODC), National Environment Research Council, United Kingdom
  Contact: Dr. Roy Lowry (rkl@bodc.ac.uk)
- Centre de documentation de recherche et d'expérimentations sur les pollutions accidentelles des eaux (Cedre), France.
  Contact: Mr. François Parthiot (Francois.Parthiot@cedre.fr)
- Coastal and Marine Resources Centre (CMRC), University College Cork, National University of Ireland, Cork, Ireland.
  Contact: Mr. Declan Dunne (d.dunne@ucc.ie)
- Plymouth Marine Laboratory (PML), United Kingdom.
  Contact: Mr. Steve Groom (sbg@pml.ac.uk)
- Institut français de recherche pour l'exploitation de la mer (Ifremer), France.
  Contact: Mr. Mickael Treguer (mickael.treguer@ifremer.fr)
- Norwegian Meteorological Institute (METNO), Norway.
  Contact: Mr. Øystein Torget (oysteint@met.no)

Author(s)

- Peter Walker, PML, (petwa@pml.ac.uk)
- Michael Grant, PML, (mggr@pml.ac.uk)
- Jorge Mendes de Jesus, PML, (jmdj@pml.ac.uk)

Document approval

- Document status: Revision 1
- WP leader approval: 2012-04-26
- Quality Manager approval: 2012-04-30
- Coordinator approval: 2012-04-30

Revision History

Issue	Date	Change records	Author(s)
Draft 1	2012-03-07	TOC	Mike Grant
Draft 2	2012-04-16	First version	Peter Walker, Michael Grant, Jorge Mendes de Jesus
Draft 3	2012-04-25	Included review comments and discussion of uncertainty	Peter Walker, Michael Grant, Jorge Mendes de Jesus
1	2012-04-30	Final release approved.	T. Hamre

Executive Summary

The NETMAR project aims to develop a pilot European Marine Information System (EUMIS) for searching, downloading and integrating satellite, in situ and model data from ocean and coastal areas. EUMIS will use a semantic framework coupled with ontologies for identifying and accessing distributed data, such as near-real time, forecast and historical data. NETMAR will develop a set of data delivery services using standard web-GIS protocols.

Processing services and adaptive service chaining services will also be developed, to enable users to generate new products suited to their needs. This document summarises the work done in enhancing standards based technologies to make use of semantic metadata attached to the NETMAR data and processing services to add more intelligent checking of parameters to the process building blocks used in these service chains.

The deliverable provides the recommended WPS server implementation and framework that will support semantically aware processing services in the NETMAR project, using semantic metadata provided by data services, for instance, for validating data inputs (e.g. a temperature-oriented processing service that is intelligent enough to ensure it has relevant, compatible and correct input data representing temperatures).

The enhancements and metadata recommendations are compatible with existing OGC standards and services designed to use them would still be able to interact with pre-existing services, which did not support this metadata.

Contents

1. Introduction 4
  1.1. Background and project linkage 4
  1.2. Objective of this report 4
  1.3. Terminology 4
  1.4. Organisation of this report 5
2. PyWPS overview...6
3. Requirements for semantically enabled services 7
  3.1. Data service metadata requirements...7
    3.1.1. Web Map Service (WMS) 7
    3.1.2. Web Coverage Service (WCS) 7
    3.1.3. Web Feature Service (WFS)..7
  3.2. Processing service metadata requirements.......................................8
4. Uncertainty metadata 9
5. Implementation..10
  5.1. Metadata markup 10
  5.2. Unit and parameter compatibility verification 11
  5.3. Unit compatibility and transformation...12
6. Example services 13
  6.1. Web Map Service (GeoServer)...13
  6.2. Web Processing Service (PyWPS)...14
7. Recommendations for integration into NETMAR 15
  7.1. Processing and data services...15
  7.2. Discovery 15
  7.3. GIS viewer 15
  7.4. Recommendations for semantic editor integration......................15
  7.5. Feedback of standards..15
8. References...16

1. Introduction

This deliverable provides the recommended WPS server implementation and framework that will support semantically aware processing services in the NETMAR project, using semantic metadata provided by data services, for instance, for validating data inputs (e.g. a temperature-oriented processing service that is intelligent enough to ensure it has relevant, compatible and correct input data representing temperatures).

Service providers will use this package to implement semantically enabled processing services to meet the requirements of the use cases and to provide demonstration services in the EUMIS portal. A specific goal is to support parameter validation both within service chains and within a service itself (run-time checking). The semantic enhancements developed for NETMAR are compatible with existing standards so services may be made available via international registries, such as GEOSS, and can be used with other standards-based projects even if other clients have not been altered to take advantage of the semantic metadata.

1.1. Background and project linkage

In September 2010, Plymouth Marine Laboratory began working on a new WPS (Web Processing Service) implementation based on PyWPS (Python API supporting WPS) [GRASS06]. The results allow easier web service integration and functionalities inside the EUMIS platform, and have been fed back to the PyWPS community by the NETMAR project.

Newly developed functionalities like WSDL (Web Service Description Language) [WSDL01] and SOAP (Simple Object Access Protocol) [SOAP00] allow WPS services to be integrated into orchestration platforms such as Taverna [TAV06] or to be used in web service execution languages like BPEL (Business Procedure Execution Language). This work resulted in a peer-review paper [JES11].

The current development builds upon the semantic framework developed in NETMAR to provide a consistent approach to binding semantic information such as units and parameter types to data and processing services.

1.2. Objective of this report

The aim of this report is to briefly document the development work carried out by PML and to provide sufficient information for NETMAR partners to set up their own semantically enabled processes.

1.3. Terminology

AJAX	Asynchronous JavaScript and XML
GEOSS	Global Earth Observation System of Systems
GIS	Geographic Information System
OGC	Open Geospatial Consortium
REST	Representational state transfer
RDF	Resource Description Framework
SOAP	Object Access Protocol
WCS	Web Coverage Service
WFS	Web Feature Service

WMS	Web Map Service
WPS	Web Processing Service
WSDL	Web Service Description Language

1.4. Organisation of this report

Section 2 gives a shorts overview of the PyWPS implementation of the WPS standard, while section 3 defines the requirements for semantically enabled services (both delivery and processing services). Section 4 illustrates how uncertainty metadata can be included. Section 5 explains how handling of the semantic metadata are implemented, and Section 6 shows examples of semantic metadata for data delivery (WMS) and processing (WPS) services. Finally, section 7 gives some recommendations for integrating semantic metadata into the EUMIS platform of NETMAR.

2. PyWPS overview

PyWPS[1] is a WPS implementation written in the Python language. The current stable version, 3.2.0, offers WPS 1.0.0 support. PyWPS provides a framework where programmers can deploy their geospatial algorithms and projects such as wps-grass-bridge[2] can be used to provide many processes “out of the box”.

PML has continued to use PyWPS as the base software for NETMAR semantic enhancements, extending the code to provide metadata for all inputs and outputs to a process. This allows code within the process to verify data inputs, for example, for correct dimensionality or to apply unit conversion if required. These extensions are all 100% OGC WPS compliant and could be incorporated in any WPS implementation.

PML has also contributed to the general development of PyWPS with enhancements to improve workflows (allowing the use of URL references by default within processes in a workflow to prevent any unnecessary transfers of data). PML has also worked extensively on unit testing and enhancements to the PyWPS MapScript interface which allows a process to return results as a service reference (such as a WMS layer).

PyWPS has also been extended to provide support for mimetypes in inputs and outputs which will allow developers to take advantage of the best practice discussion paper[3] proposed by 52North at the OGC TC meeting.

[1] http://pywps.wald.intevation.org
[2] http://code.google.com/p/wps-grass-bridge
[3] https://portal.opengeospatial.org/files/?artifact_id=47860

3. Requirements for semantically enabled services

A semantically enabled processing service will make use of semantic metadata provided for its data inputs and produce outputs that also contain semantic metadata. A number of requirements have been identified which must be fulfilled before semantic processing can take place. It should be emphasised that if these requirements are not fulfilled the processing service should fall back to a non-semantic processing approach, for instance not checking parameter types.

3.1. Data service metadata requirements

Data inputs to semantic processing services must provide semantic metadata such that the processing service can verify the type of data input parameters. This is achieved by creating an RDF fragment containing this information (using standard vocabularies) and providing a link within the existing metadata to the fragment which could be stored on a separate server.

Methods for including this metadata which remain compatible with the existing OGC WxS standards are listed below.

3.1.1. Web Map Service (WMS)

In WMS the *GetCapabilities* return includes a *MetadataURL* element which can include a *OnlineResource* element pointing to the RDF fragment. In the simplest case all elements of this type would be checked to see if they linked to semantic metadata. Optionally the tag could be marked with the attribute ‘xlink:title="Semantic Metadata"’ and, if found, only elements with this attribute would be followed resulting in fewer network accesses.

```
<MetadataURL type="other">
  <Format>text/plain</Format>
  <OnlineResource xlink:type="simple" xlink:title="Semantic Metadata"
    xlink:href=" http://vocab.nerc.ac.uk/collection/P01/current/PSSTRT01"/>
</MetadataURL>
```

3.1.2. Web Coverage Service (WCS)

In WCS the *CoverageDescription* includes a *metadataLink* element which can be used for this purpose. The only difference to the WMS case is that the URL is provided as text within the element rather than as a sub-element.

```
<wcs:metadataLink metadataType="other" about="Semantic Metadata" xlink:title="Semantic Metadata">
  http://vocab.nerc.ac.uk/collection/P01/current/PSSTRT01
</wcs:metadataLink>
```

3.1.3. Web Feature Service (WFS)

For WFS the *FeatureType* return functions in the same way as for WCS, using *MetadataURL*.

```
<wfs:MetadataURL type="other" format="text/plain" xlink:title="Semantic Metadata">
  http://vocab.nerc.ac.uk/collection/P01/current/PSSTRT01
</wfs:MetadataURL>
```

3.2. Processing service metadata requirements

The processing service case (as implemented by an OGC WPS) is slightly more complex in that a WPS process can both provide outputs (similar to a data service) and also expect inputs. Luckily the *ProcessDescription* element allows *Metadata* elements within both *DataInput* and *ProcessOutputs* elements which can be used for this.

```
<ProcessDescription wps:processVersion="0.1" storeSupported="true" statusSupported="true">
...
  <ows:Metadata xlink:title="Temperature" xlink:href="http://vocab.nerc.ac.uk/collection/P24/current/KELVIN"/>
  <DataInputs>
    <Input minOccurs="1" maxOccurs="1">
...
     <ows:Metadata xlink:title="Degrees Celsius"
xlink:href="http://vocab.nerc.ac.uk/collection/P06/current/UPAA/"/>
...
    </Input>
  </DataInputs>
  <ProcessOutputs>
    <Output>
...
      <ows:Metadata xlink:title="Degrees Kelvin" xlink:href="http://vocab.nerc.ac.uk/collection/P06/current/UPKA"/>
...
    </Output>
  </ProcessOutputs>
</ProcessDescription>
```

4. Uncertainty metadata

Uncertainty metadata is handled using a similar approach to the other semantic metadata. However, whereas the semantic metadata for a layer in a WMS applies to all instances of that layer (for instance data for different days) the uncertainty information will vary depending on the actual data. To allow for this the uncertainty metadata provides a rule by which the uncertainty information for a dataset may be retrieved rather than the uncertainty itself. The actual uncertainty information would be held elsewhere (probably with the data).

In general the metadata link will use the same format as described in Section 3.1 but the link could point to an XML snippet containing the uncertainty metadata rather than the uncertainty information itself. For instance, a WCS entry might appear as below:

```
<wcs:metadataLink metadataType="other" about="Uncertainty Metadata" xlink:title="Uncertainty Metadata">
     http://rsg.pml.ac.uk/projects/netmar/uncertainty/MODIS_SST_9km.xml
</wcs:metadataLink>
```

The XML snippet will contain the information necessary to retrieve the uncertainty information for a specific layer (uncertainty layer identifier and metadata).

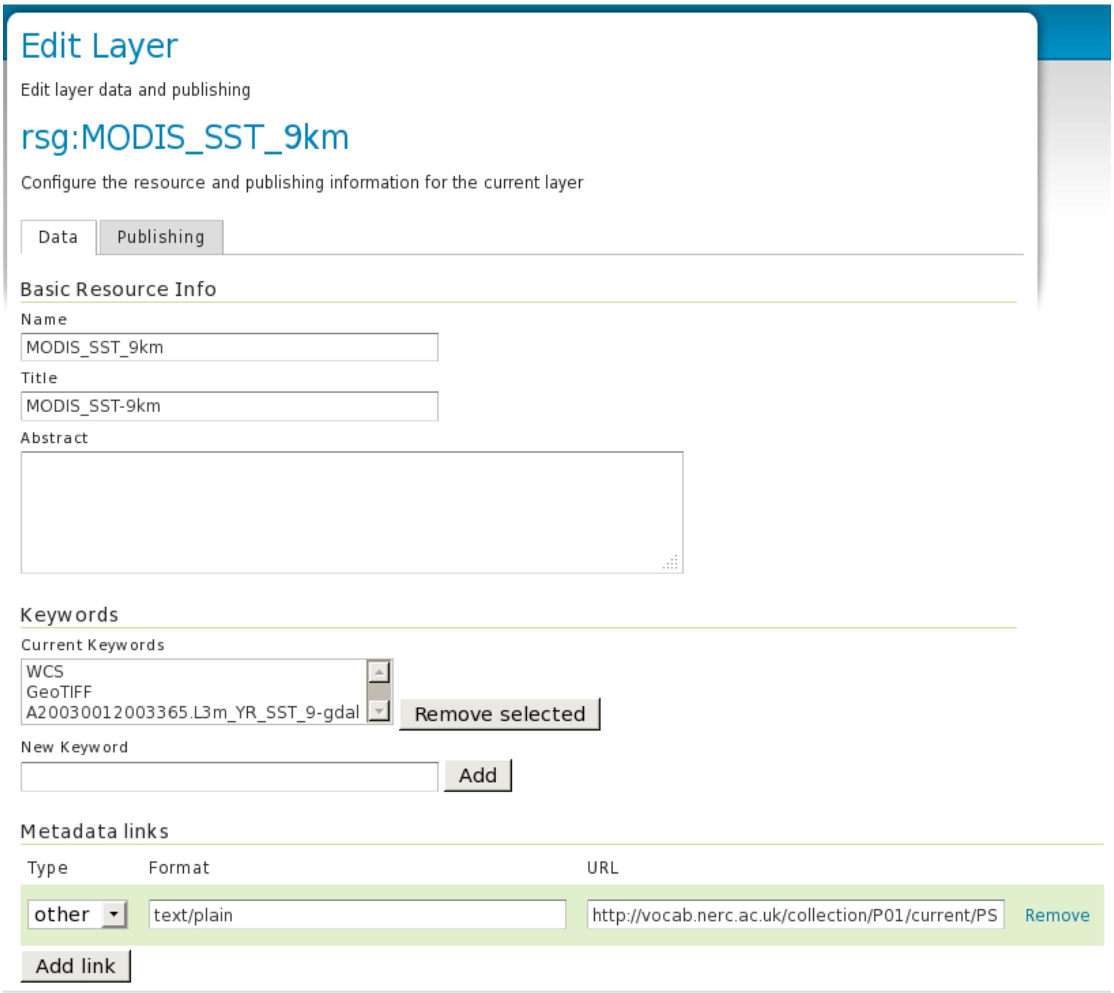
In simple cases the uncertainty layer information may be encoded directly in the metadata link as is done for semantic metadata. For instance, a WMS entry in this form would appear as below:

```
<wcs:metadataLink metadataType="other" about="Uncertainty Metadata" xlink:title="Uncertainty Metadata">
   http://rsg.pml.ac.uk/wfs?service=WMS&version=1.1.0&request=GetMap&layers=rsg:MODIS_SST_9km_variance
</wcs:metadataLink>
```

5. Implementation

5.1. Metadata markup

Methods for marking up data services with semantic metadata will vary depending on the server implementation. For example, using GeoServer[1], the metadata may be entered using the web interface (Figure 5-1).



Edit Layer

Edit layer data and publishing

rsg:MODIS_SST_9km

Configure the resource and publishing information for the current layer

Basic Resource Info

Name: MODIS_SST_9km

Title: MODIS_SST-9km

Abstract:

Keywords

Current Keywords: WCS, GeoTIFF, A20030012003365.L3m_YR_SST_9-gdal

New Keyword:

Metadata links

Type	Format	URL	
other	text/plain	http://vocab.nerc.ac.uk/collection/P01/current/PS	Remove

Figure 5-1 Adding semantic metadata to GeoServer layer.

The metadata created is shown below (WCS 1.0.0):

```
<wcs:CoverageDescription xsi:schemaLocation="http://www.opengis.net/wcs
http://rsg.pml.ac.uk:80/geoserver/schemas/wcs/1.0.0/describeCoverage.xsd" version="1.0.0">
  <wcs:CoverageOffering>
    <wcs:metadataLink metadataType="other">
      http://vocab.nerc.ac.uk/collection/P01/current/PSSTRT01
    </wcs::metadataLink>
    <wcs:description>
      Generated from GeoTIFF
    </wcs:description>
    <wcs:name>
      rsg:MODIS_SST_9km
    </wcs:name>
    <wcs:label>
      MODIS_SST_9km
    </wcs:label>
```

[1] http://geoserver.org

```
…
  </wcs:CoverageOffering>
</wcs:CoverageDescription>
```

To add the semantic metadata to PyWPS processes is simply a matter of adding the attribute to each input and output definition within the initialisation of the process.
For instance, to add semantic metadata to the inputs and outputs of the WPS example shown in Section 6 the following code (shown in bold) would be added.

```
    self.inTemp=self.addLiteralInput("in", "Temperature input value", "Temperature input value that will be
transformed from C into K", type=type(0.0),metadata=[{'title': 'Semantic Metadata', 'href':
'http://vocab.nerc.ac.uk/collection/P06/current/UPAA/'}],default=0.0)
    self.outTemp=self.addLiteralOutput("out","Temperature output value","Returned temperature in
Kelvin",type=type(0.0),metadata=[{'title': 'Semantic Metadata', 'href':
'http://vocab.nerc.ac.uk/collection/P06/current/UPKA'}])
```

5.2. Unit and parameter compatibility verification

The NETMAR project has decided to use the NERC Vocabulary Server (NVS), maintained by BODC, to hold semantic metadata. The metadata is stored as RDF in a number of distinct vocabularies which may be accessed via HTTP GET.

An example entry[1], which might be returned for an SST (Sea Surface Temperature) layer, is shown below:

```
<rdf:RDF>
  <skos:Concept rdf:about="http://vocab.nerc.ac.uk/collection/P01/current/PSSTRT01/">
    <skos:prefLabel xml:lang="en">Skin temperature of the water body by infra-red radiometer</skos:prefLabel>
    <skos:altLabel xml:lang="en">RadTmp</skos:altLabel>
    <skos:definition xml:lang="en">Unavailable</skos:definition>
    <dc:identifier>SDN:P01::PSSTRT01</dc:identifier>
    <skos:notation>SDN:P01::PSSTRT01</skos:notation>
    <owlxml:versionInfo>1</owlxml:versionInfo>
    <dc:date>2009-11-03 16:19:38.0</dc:date>
    <skos:note xml:lang="en">accepted</skos:note>
    <owlxml:deprecated>false</owlxml:deprecated>
    <skos:broader rdf:resource="http://vocab.nerc.ac.uk/collection/P02/current/PSST"/>
    <skos:related rdf:resource="http://vocab.nerc.ac.uk/collection/P06/current/UPAA"/>
  </skos:Concept>
</rdf:RDF>
```

A processing service may check its inputs for compatibility by following these steps:

1) Retrieve the parameter RDF from the vocabulary server for each service.
2) Compare the "broader" links to the P02 collection ("parameter groupings" collection)
    a. if they are the same, the parameters are compatible, carry on to unit checking (step 3).
    b. if they are not the same, they are not compatible, return failure.
3) Compare the "related" links to the P06 collection (units)
    a. if they are the same, the units are directly compatible, return success.
    b. if they are not the same, return failure (or implement the unit compatibility checks in Section 5.3).

The check function has been implemented in a Python library, which allows the verification to be carried out by a single function call. This can be used directly (best for embedding in a processing service), or via a REST interface (best for AJAX calls from the service chaining editor), or via a WPS call (for incorporation in a chain).

[1] http://vocab.nerc.ac.uk/collection/P01/current/PSSTRT01

5.3. *Unit compatibility and transformation*

Some parameters may not be identical but it might be possible to map one to the other via a simple unit transformation. The steps for carrying this out are listed below:

1) Retrieve the RDF for the two units from the vocabulary server.
2) Take the units text from each e.g. “kilometres per hour” and “miles per hour”.
3) Canonicalise the units, transforming them to basic SI units, a scale factor and an offset (e.g. using the udunits2 program)
    a. “kilometres per hour” becomes “0.277777777777778 ms^{-1}”.
    b. “miles per hour” becomes “0.44704 ms^{-1}”.
4) if the SI units are not the same, the units are incompatible, return failure.
5) if the SI units are the same, the units are compatible.
6) check if the scale factors are the same and the offset zero; if so, they are directly compatible, return success.
7) if the scale factors and/or offsets are different, the units are compatible but need a transformation; return “yellow light/partial success” indicating this.
8) optionally, return transformation parameters.

6. Example services

6.1. Web Map Service (GeoServer)

An example WMS layer marked up according to these guidelines may be found at http://rsg.pml.ac.uk/wfs?service=WMS&version=1.1.0&request=GetMap&layers=rsg:MODIS_SST_9km&styles=&bbox=-180.0,-90.0,180.0,90.0&width=660&height=330&srs=EPSG:4326&format=image/png (Figure 6-1).

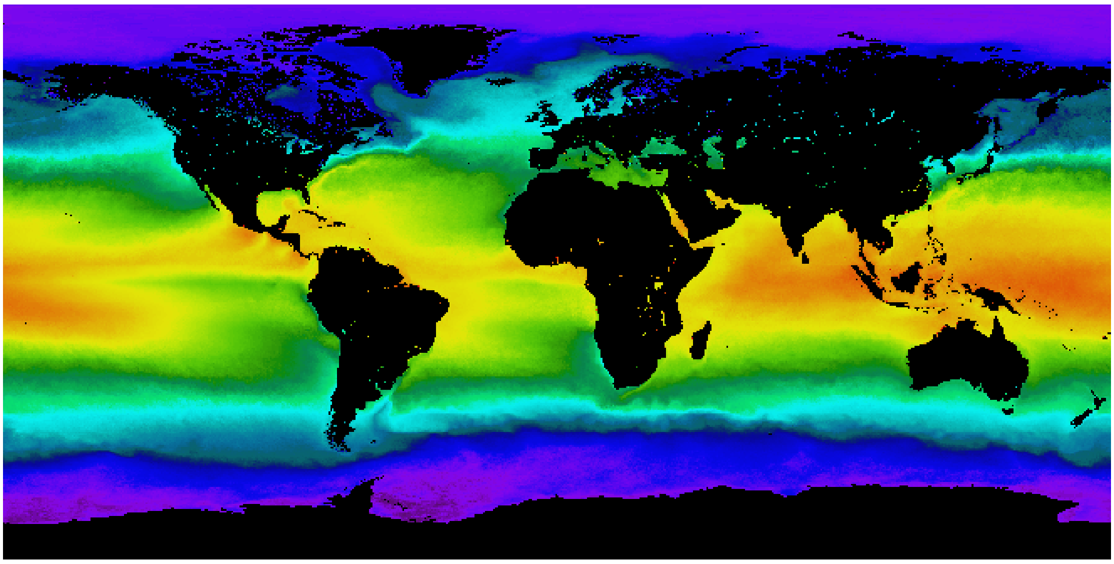


Figure 6-1 MODIS 9KM SST example layer.

A section of the GetCapabilities[1] response for this layer is shown below:

```
   <Layer queryable="1">
    <Name>rsg:MODIS_SST_9km</Name>
    <Title>MODIS_SST-9km</Title>
    <Abstract/>
    <KeywordList>
      <Keyword>WCS</Keyword>
      <Keyword>GeoTIFF</Keyword>
      <Keyword>A20030012003365.L3m_YR_SST_9-gdal</Keyword>
    </KeywordList>
    <SRS>EPSG:4326</SRS>
    <LatLonBoundingBox minx="-180.0" miny="-90.0" maxx="180.0" maxy="90.0"/>
    <BoundingBox SRS="EPSG:4326" minx="-180.0" miny="-90.0" maxx="180.0" maxy="90.0"/>
    <MetadataURL type="other">
      <Format>text/plain</Format>
      <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
xlink:href="http://vocab.nerc.ac.uk/collection/P01/current/PSSTRT01"/>
    </MetadataURL>
    <Style>
      <Name>raster</Name>
      <Title>Raster</Title>
      <Abstract>A sample style for rasters, good for displaying imagery</Abstract>
      <LegendURL width="20" height="20">
        <Format>image/png</Format>
        <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
xlink:href="http://rsg.pml.ac.uk:80/geoserver/wms?request=GetLegendGraphic&amp;format=image%2Fpng&amp;wid
th=20&amp;height=20&amp;layer=MODIS_SST_9km"/>
      </LegendURL>
    </Style>
   </Layer>
```

[1] http://rsg.pml.ac.uk/wfs?service=WMS&version=1.1.0&request=GetCapabilities

6.2. Web Processing Service (PyWPS)

A simple WPS demonstrating how semantic metadata can be processed is available at
http://rsg.pml.ac.uk/wps/generic.cgi?request=describeProcess&service=WPS&identifier=temperatureConverter&version=1.0.0

The results of the DescribeProcess are shown below:

```
<wps:ProcessDescriptions service="WPS" version="1.0.0" xml:lang="en-CA"
xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
http://schemas.opengis.net/wps/1.0.0/wpsDescribeProcess_response.xsd">
  <ProcessDescription statusSupported="true" storeSupported="true" wps:processVersion="0.1">
    <ows:Identifier>temperatureConverter</ows:Identifier>
    <ows:Title>Simple Temperature Converter, Centigrades to Kelvin</ows:Title>
    <ows:Abstract>Simple Temperature Converter, Centigrades to Kelvin</ows:Abstract>
    <ows:Metadata xlink:href="http://vocab.nerc.ac.uk/collection/P24/current/KELVIN" xlink:title="Temperature"/>
    <DataInputs>
      <Input maxOccurs="1" minOccurs="1">
        <ows:Identifier>in</ows:Identifier>
        <ows:Title>Temperature input value</ows:Title>
        <ows:Abstract>Temperature input value that will be transformed from C into K</ows:Abstract>
        <ows:Metadata xlink:href="http://vocab.nerc.ac.uk/collection/P06/current/UPAA/" xlink:title="Semantic
Metadata"/>
        <LiteralData>
          <ows:DataType ows:reference="http://www.w3.org/TR/xmlschema-2/#float">float</ows:DataType>
          <ows:AnyValue/>
          <DefaultValue>0.0</DefaultValue>
        </LiteralData>
      </Input>
    </DataInputs>
    <ProcessOutputs>
      <Output>
        <ows:Identifier>out</ows:Identifier>
        <ows:Title>Temperature output value</ows:Title>
        <ows:Abstract>Returned temperature in Kelvin</ows:Abstract>
        <ows:Metadata xlink:href="http://vocab.nerc.ac.uk/collection/P06/current/UPKA" xlink:title="Semantic
Metadata"/>
        <LiteralOutput>
          <ows:DataType ows:reference="http://www.w3.org/TR/xmlschema-2/#float">float</ows:DataType>
        </LiteralOutput>
      </Output>
    </ProcessOutputs>
  </ProcessDescription>
</wps:ProcessDescriptions>
```

7. Recommendations for integration into NETMAR

7.1. Processing and data services

All NETMAR data and processing services should have semantic metadata added according to these guidelines. This would allow processing services to be targeted at specific data types, reducing the possibility of incorrect results being produced due to inappropriate data inputs.

7.2. Discovery

Semantic metadata, both in processing and data services, should be used in discovery to identify suitable data inputs for a process (or available processing services for a dataset)

7.3. GIS viewer

Semantic metadata would allow the GIS viewer to perform specific transformations based on types of data.

7.4. Recommendations for semantic editor integration

The service chaining editor should make used of semantic metadata when building workflows to ensure that only compatible services may be chained together. In conjunction with the discovery service it could also suggest suitable processing/data combinations.

7.5. Feedback of standards

In addition to the work carried out within NETMAR, the approach used for semantic metadata enhancement of OGC WxS services should be fed back to the OGC for possible inclusion in future standards and recommendations.

It is understood that the OGC have just put forward a document “Semantic annotations in OGC standards” as a proposed Best Practice and NETMAR will take these proposals into account.

8. References

[GRASS06] Cepicky, J., 2006. GRASS goes web: PyWPS, in: Free and Open Source Software for Geoinformatics, Lausanne, Switzerland.

[JES11] de Jesus J et al, 2011 WPS orchestration using the Taverna workbench: The eScience approach. DOI:http://www.sciencedirect.com/science/article/pii/S0098300411003906]

[SOAP00] Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H., Thatte, S., Winer, D., 2000. Simple object access protocol (SOAP) 1.1, W3C Note 08 May 2000. W3C Note. The World Wide Web Consortium (W3C).

[TAV06] Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M., Li, P., Oinn, T., 2006. 29 Taverna: a tool for building and running workflows of services. Nucleic Acids Research 34, 729–732.

[WSDL01] Christensen, E., Curbera, F., Meredith, G., Weerawarana, S., 2001. Web services description language (WSDL) 1.1, W3C Note 15 March 2001. W3C Note. The World Wide Web Consortium (W3C).