Putting WHOI Data on the Map

Janet Fredericks
AOP&E, WHOI
OGC -- Introduction

The Open Geospatial Consortium, Inc (OGC) is an international industry consortium of 341 companies, government agencies and universities participating in a consensus process to develop publicly available interface specifications. OpenGIS® specifications support interoperable solutions that "geo-enable" the Web, wireless and location-based services, and mainstream IT. The specifications empower technology developers to make complex spatial information and services accessible and useful with all kinds of applications.

OpenGIS® is a Registered Trademark of the Open Geospatial Consortium, Inc (OGC) and is the brand name associated with the Specifications and documents produced by the Open Geospatial Consortium, Inc (OGC). OpenGIS specifications are developed in a unique consensus process supported by OGC industry, government and academic members to enable geoprocessing technologies to interoperate, or "plug and play." You will also find the OpenGIS® trademark associated with products that implement or comply to our specifications. Make sure that your geoprocessing and location services procurement and technology development programs demand OpenGIS® specifications.
Sensor Web Enablement WG

Introduction

OGC members are specifying interoperability interfaces and metadata encodings that enable real time integration of heterogeneous sensor webs into the information infrastructure. Developers will use these specifications in creating applications, platforms, and products involving Web-connected devices such as flood gauges, air pollution monitors, stress gauges on bridges, mobile heart monitors, Webcams, and robots as well as space and airborne earth imaging devices.

OGC members have developed and tested the following candidate specifications. Others are planned.

1. Observations & Measurements (O&M) – Standard models and XML Schema for encoding observations and measurements from a sensor, both archived and real-time.
2. Sensor Model Language (SensorML) – Standard models and XML Schema for describing sensors systems and processes provides information needed for discovery of sensors, location of sensor observation, processing of low-level sensor observations, and listing of taskable properties.
3. Transducer Model Language (TransducerML or TML) – The conceptual model and XML Schema for recording, transmitting, and supporting real-time streaming of data output from sensor systems.
4. Sensor Observations Service (SOS) – Standard web service interface for requesting, filtering, and retrieving observations and sensor system information. This is the intermediary between a client and an observation repository or near real-time sensor channel.
5. Sensor Planning Service (SPS) – Standard web service interface for requesting user-driven acquisitions and observations. This is the intermediary between a client and a sensor collection management environment.
7. Web Notification Services (WNS) – Standard web service interface for asynchronous delivery of messages or alerts from SAS and SPS web services and other elements of service workflows.

Please visit our OpenGIS® Specification page to view and comment on publicly available OGC Sensor Web Enablement candidate OpenGIS® Specifications offered as Discussion Papers and Recommendation Papers. OGC invites organizations to join OGC and participate in OGC Sensor Web initiatives and other initiatives related to interoperable geoprocessing.
OGC 06-009r1

Open Geospatial Consortium Inc.

Date: 2006-02-13

Reference number of this OGC® document: OGC 06-009r1

Version: 0.1.5

Category: OpenGIS® Implementation Specification

Editors: Arthur Na (IRIS Corp.), Mark Priest (3eTI)

OpenGIS® Sensor Observation Service Implementation Specification
Sensor Observation Service

- “Provides an API for managing deployed sensors and retrieving sensor data”
- Key here is access through a service
- Three mandatory “core” operations
  - GetObservation
  - Describe Sensor
  - GetCapabilities
Core Operations

1. **GetObservation** “provides access to sensor observations & measurement data via a spatio-temporal query that can be filtered by phenomena”

2. **DescribeSensor** “retrieves detailed info about the sensors making those measurements and the platforms that carry the sensors”

3. **GetCapabilities** “provides the means to access SOS service metadata”
Other non-mandatory Operations

In support of transactions:
  RegisterSensor and InsertObservation

Enhanced Operations:
  • GetResult
  • GetFeatureOfInterest
  • GetFeatureOfInterestTime
  • DescribeFeatureOfInterest
  • DescribeObservationType
  • DescribeResultModel
SensorML

What is it?
SensorML provides standard models and an XML encoding for describing any process, including the process of measurement by sensors and instructions for deriving higher-level information from observations.

Processes described in SensorML are discoverable and executable.

All processes define their inputs, outputs, parameters, and method, as well as provide relevant metadata. SensorML models detectors and sensors as processes that convert real phenomena to data.
SensorML – what can it do

- Electronic Specification Sheet
- Discovery of sensor, sensor systems, and processes
- Lineage of Modification
- On-demand processing of Observation
- Support for tasking, observation, and alert services
- Plug-N-Play, auto-configuring, and autonomous sensor networks
- Archiving of Sensor Parameters
Ocean Science Interoperability Experiment

Summary

The Ocean Science Interoperability Experiment will contribute to a portion of the ocean observing community by understanding various OGC specifications, pilot demonstrations for Ocean Science application areas, and software implementations, and produce a candidate OGC Best Practices document that can be used to inform the broader ocean observing community. To achieve these goals, the Oceans IE will engage the OGC membership to ensure that any community recommendations coming from the Oceans group will properly leverage the OGC specifications. Potentially, Change Requests on OGC Specification will be provided to the OGC Technical Committee to influence the underlying specifications. It is not anticipated that this IE will develop any new specifications.

Initiator Organizations

The OGC members that are acting as initiators of the Interoperability Experiment are:

- Southeastern Universities Research Association (SURA)
- Texas A&M University, Academy for Advanced Telecommunications (TAMU)
- National Center for Atmospheric Research (NCAR)
- The Monterey Bay Aquarium Research Institute (MBARI)
- Gulf of Maine Ocean Observing System (GoMOOS)

Description

The Ocean IE will advance several areas of understanding and application of OGC specifications in to web services for interoperable ocean science. The IE will apply existing specifications in the context of an Ocean Science scientific domain. The IE will refine and inform spec, rather than develop new spec. The GAOLEN IE is a good example in applying WCS access to Atmospheric data.

The Ocean IE will focus on these areas:

- Web Services for Interoperable Ocean Science.
- OGC WMS and WPS access to ocean data, focusing on SOAP bindings.
- Application of the OGC Service Oriented Architecture (SOA) for Web Services to Ocean-observing applications.
- Sensor Web Enablement (SWE) in particular Observations & Measurements and SensorML.
- Sensor Observation Service (SOS) for raw observations.
- OML application schema for Ocean data semantic interoperability using RDF-based ontologies.
- Develop an end-to-end demonstration of web services increasing the interoperability of various regional-time, ocean-observing programs.

A desired outcome from an interoperability experiment is some kind of "Best Practices" document for the use of OGC adopted spec by a community of interest. The report will be posted as an OGC pending document for consideration by the OGC Specification Program, i.e., the OGC consensus process. This kind of "Best Practice" document would show how to use an OGC spec in specific applications.

Background

The Southeastern Universities Research Association (SURA) hosted a workshop in Baltimore October 2005 called OOS Tech 2005 (note: OOS = Ocean Observing System). The workshop included approximately 100 ocean scientists, data managers, and computer science experts from around the country. They learned and talked about "Web Services for Interoperable Ocean Science." After the workshop, a subset of the group agreed to work together on a follow-on activity to implement some of what they had learned. The group agreed to build from their previous experiences using OGC WMS and WPS specifications. In previous years, they had built some basic elements of a Service Oriented Architecture (SOA) demo at www.openocean.org. The OOS Tech 2005 follow-on activity began with 5 loosely defined goals.
OOSTethys Partners

OOSTethys Partners (OOST) is a partnership among several organizations to make the "system of systems" for ocean observation work better. The goal is to make it easier for scientists and engineers to share data and observations.

Getting Started

Two grass-roots community initiatives have aligned to make concrete choices that will advance the "system of systems" concept: the OpenOOS Interoperability Test Bed (OOTB) and the Marine Metadata Interoperability (MMI) project.

OOTB - OOTB is a joint project of the SURA and OpenOOS communities with the goal of building a test bed for interoperability. The test bed is designed to be flexible and open, with the ability to test different scenarios and configurations.

MMI - MMI is a project that has been demonstrating the exchange, integration, and use of marine data by emphasizing ontologies that employ the Web Ontology Language (OWL) in anticipation of the Semantic Web.

History of the partnership

OpenOOS.org has been demonstrating that "standards enable innovation" for several years, thanks to contributions from Ocean Observing Systems (OOSs) around the country.

Funded by:

NOAA Coastal Services Center
National Science Foundation
Office of Naval Research
This interoperability demonstration represents an effort to develop a Web Services Architecture for Ocean Observing. We are seeking participants who would like to serve their in-situ OOSTethys SOS.

OOSTethys SOS Registry Services

Programming, database and user interface design by gemOOS
SOS Server CGI Cookbook

Steps for installing and setting up an SOS server.

File List
- oosthys - the CGI script
- sos_config.xml - local configuration file
- sosConfHir.xml - template
- sosConfigObservations.xml - template
- observation_text_files
  - e.g. A01_SEA_WATER_SALINITY.TXT

All the files can be downloaded here

System Requirements
- Perl
- Web service capable of running CGI scripts
- CGI perl module
- XML-LibXML perl module
- download and installation instructions Perl Module Installation Tips
- Salinity data to serve.
Required Fields
Not all of the elements in the sos_configuration file are required. Here is a list of the required elements:
- OrganizationName
- OrganizationAcronym
- ContactPersonName
- ContactPersonEmail
- OrganizationURL
- SOSURL
- localPlatformName
- platformURI
- platformlongName
- latitude
- longitude
- startTime
- observedProperty
- SOSDataFile

Creating Data Files
Each sensor and data type has a field called SOSDataFile which should contain the full path of the file containing the data for that sensor and data type.

Note: Typically a Web Server will not allow writing to the cgi-bin directory, otherwise the files could be placed in the same directory as the oostahys_sos.CGI script. In that case you do not need a full path for the observation file name. At COBOSS we run a cron job which queries the COBOSS database every hour and writes salinity data to a temporary data directory. We use the platform name followed by the data type with a .txt extension but any name will work.

Format of the observation file
The observation file is made up of tuples with 5 fields separated by commas. The 5 fields are:
1. Time. This must be in an accepted ISO8601 format, e.g. 2006-09-27T10:00:00Z
2. latitude in decimal degrees. E.g. 43.7905 This in not degrees, minutes, seconds format.
3. longitude in decimal degrees E.g. -70.3453
4. depth in meters E.g. 2
5. Datum value. E.g. 38.78

In the data file each record should be separated by a newline. There would be separate records for different times and depths but each file should represent observations for a single sensor and data type.
Installation Steps

1. Put the sosweb/sos赵QO SCRIPT into your cgi-bin directory and make sure it has execute permission. Typically this in the /cgi-bin/ directory of your Web server. At GoMOOS I installed it in a subdirectory /cgi-bin/sos/.
2. Put the three XML template files in the same directory.
   1. sosGetCapabilities.xml
   2. sosGetObservation.xml
   3. sosDescribeSensor.xml
3. Put the sos_config.xml file in the same directory.
4. Edit the sos_config.xml. See Editing the sos_config.xml file below.
5. Create some data files. See Creating Data Files below.

Editing the sos_config.xml file

The sos_config.xml is a simple XML configuration file which contains the local information necessary to generate the SOS responses to the three SOS requests:
- GetCapabilities
- GetObservation
- DescribeSensor

The DataProvider/Publisher section contains self-explanatory fields for information about your organization. A crucial element is the SOSURL section. This must contain the URL for the cgi-bin SOS script, e.g. http://host/name/cgi-bin/osweb/sos. Unless you have installed the script in some other location or renamed the executable sos script it should always look like this.

The next section is the DataProvider/ObservationList section. The ObservationList is made up of one or more Observation sections. Each Observation section contains information about the platform or sensor and a particular data type. In this version, salinity measurements at PSUS.

- localPlatformName: the identifier by which you refer to the platform locally.
- platformURI: the URI to the localPlatformName which assures the name is unique across the Internet. Typically this will point to a machine using a supported SensorFarm protocol or name, e.g., at GoMOOS we have a localPlatformName of A01. The platformURI is http://localhost/resource/repository/a01/.
- location: the location of the observation.
- start time, end time: should cover the period for which readings are available and should be in an ISO8601 format in UTC, e.g. 2006-06-28T10:00:00Z.
- observedProperty: in V0.1 this should always be sea_water_salinity. Support does exist in the script for sea_water_temperature.
- SOSDatafile: this should be the full path to a local file which contains data points for this sensor and datatype combination. It can be left blank in which case no sample data within the script will be served. This can be useful for testing purposes. See Creating Data Files below.

If you have more than one platform / sensor the Observation section must be repeated for each one.

See the GoMOOS sample configuration file for an example which handles three platforms serving up salinity data. GoMOOS SOS Configuration XML Example

Required Fields
Creating Data Files

Each sensor and data type has a field called **SOSDataFile** which should contain the full path of the file containing the data for that sensor and data type.

*Note:* Typically a Web Server will not allow writing to the op-bin directory, otherwise the files could be placed in the same directory as the oostechsys_sos.cgi script. In that case you do not need a full path for the observation file name. At GoMOOS we run a cron job which queries the GoMOOS database every hour and writes salinity data to a temporary data directory. We use the platform name followed by the data type with a .txt extension but any name will work.

**Format of the observation file**

The observation file is made up of tuples with 5 fields separated by commas. The 5 fields are:

1. Time: This must be in an accepted ISO8601 format. E.g. 2006-09-27T10:00:00Z
2. Latitude in decimal degrees. E.g. 43.7905. **This is not degrees, minutes, seconds format.**
3. Longitude in decimal degrees. E.g. -70.3459
4. Depth: in meters. E.g. 2
5. Salinity value. E.g. 38.78

In the data file each record should be separated by a newline. There would be separate records for different times and depths but each file should represent observations for a single sensor and data type.

**Sample GoMOOS Data file**

2006-09-27T15:00:00Z,43.5277,-70.5655,1.21,28
2006-09-27T15:00:00Z,43.5277,-70.5655,20,32.07
2006-09-27T15:00:00Z,43.5277,-70.5655,50,32.46

This represents the latest salinity readings for Buoy 401 at three different depths.

At GoMOOS we create a new file each hour with the latest readings, but since each record has a time stamp on it readings for different times could be contained within one file.

*Note:* You can test your service without creating any data text files by leaving the **SOSDataFile** element empty in the sos.conf.xml file. In this case the script will return a sample data record.

Testing the SOS Service

The service is designed to be accessible in a web browser and will default to the GetCapabilities request. E.g. [http://www.gomoos.org/cgi-bin/sos/oostechsys_sos](http://www.gomoos.org/cgi-bin/sos/oostechsys_sos)

**Sample requests:**

- GetCapabilities
  - [http://www.gomoos.org/cgi-bin/sos/oostechsys_sos?request=GetCapabilities](http://www.gomoos.org/cgi-bin/sos/oostechsys_sos?request=GetCapabilities)
- DescribeSensor
  - [http://www.gomoos.org/cgi-bin/sos/oostechsys_sos?request=DescribeSensor&sensorId=A01](http://www.gomoos.org/cgi-bin/sos/oostechsys_sos?request=DescribeSensor&sensorId=A01)
<SensorML xmlns:sensormL="http://www.opengis.net/sensorML" xmlns:w3c="http://www.w3.org/2001/XMLSchema-instance" version="1.0">
  <member>
    <System>
      <name>12m-Node</name>
      <description>Sensor was pumped or unpumped, depending on deployment</description>
      <keywords>
        <KeywordList codeSpace="http://www.cga.gov/OGC/KeywordList.html">
          <Keyword name="OCEANOGRAPHY"/>
        </KeywordList>
      </keywords>
    </System>
    <Identification>
      <IdentifierList>
        <Identifier>
          < Term>
            <xlink:href="urn:ogc:def:identifier:OGC.longName">12m Node at M500</xlink:href>
          </Term>
        </Identifier>
        <Identifier>
          < Term>
            <xlink:href="urn:ogc:def:identifier:OGC.shortName">12m Node</xlink:href>
          </Term>
        </Identifier>
      </IdentifierList>
    </Identification>
  </member>
</SensorML>
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```xml
<com:Observation xsi:schemaLocation="http://www.opengis.net/om 1.0.0/ows4伊me/schema/ows/brancher/ows-ows4-onom/current/onomObservation.xsd" goml:id="SEA_WATER_SALINITY">
  <gml:description>
    Latest salinity measurement from MVO buoy 12m-Node
  </gml:description>
  <gml:name>MVO Buoy 12m-Node salinity Data</gml:name>
  <gml:location>
    <gml:Point gml:id="OBSEVATION_POINT" srsName="urn:ogc:def:crs:EPSG:6.1.4329">
      <gml:coordinates>41.3366 -76.5564 -10</gml:coordinates>
    </gml:Point>
  </gml:location>
  <gml:observationType>
    <gml:TimePeriod gml:id="DATA_TIME">
      <gml:beginPosition>2007-04-04T12:24:00Z</gml:beginPosition>
      <gml:endPosition>2007-04-04T12:24:00Z</gml:endPosition>
    </gml:TimePeriod>
  </gml:observationType>
  <gml:procedure xlink:href="urn:mivo.org:source:nodes#12m-Node"/>
  <gml:observedProperty xlink:href="http://marinemetdata.org/sea_water_salinity"/>
  <gml:featureOfInterest xlink:href="urn:something:bodyOfWater"/>
  <gml:resultDefinition>
    <swe:DataBlockDefinition>
      <swe:components name="SalinityDataOf12m-Node">
        <swe:DataRecord>
          <swe:field name="time">
            <swe:Time definition="urn:x:phenomenon:time.iso8601"/>
          </swe:field>
          <swe:field name="latitude">
            <swe:Quantity definition="urn:x:phenomenon:latitude:wgs84"/>
          </swe:field>
        </swe:DataRecord>
      </swe:components>
    </swe:DataBlockDefinition>
  </gml:resultDefinition>
</com:Observation>
```
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<sensorML xmlns:schemaLocation="http://www.opengis.net/sensorML/1.0.1/sensorML-current/sensorML.xsd" version="1.0">
  <identifierList>
    <identifier>
      <term>
        <xlink:href="urn:ogc:identifier:12m-Node"/>
        <value>12m-Node at MVCO</value>
      </term>
    </identifier>
    <identifier>
      <term>
        <xlink:href="urn:ogc:identifier:12m-Node"/>
        <value>12m-Node</value>
      </term>
    </identifier>
    <identifier>
      <term>
        <xlink:href="http://www.w3.org/1999/02/22-rdf-syntax-ns#ID"/>
        <value>urn:mvco.org:source#12m-Node</value>
      </term>
    </identifier>
  </identifierList>
</sensorML>
```
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```xml
<ows:Capabilities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.opengis.net/ows/1.0/ows.xsd http://www.opengis.net/ows/ows.xsd" version="0.0.01">
    <ows:ServiceIdentification>
        <ows:Title>OGC: SISO Service</ows:Title>
        <ows:Abstract>
            <ows:Keywords><ows:Keyword>OGC: SISO Service</ows:Keyword></ows:Keywords>
        </ows:Abstract>
        <ows:Provider>
            <ows:ProviderName>Marines Vineyard Coastal Observatory (MVCO)</ows:ProviderName>
            <ows:ProviderSite xlink:href="http://www.whoi.edu/mvco/"/>
            <ows:ServiceContact>
                <ows:IndividualName>James Fredericks</ows:IndividualName>
                <ows:PositionName>Scientist</ows:PositionName>
                <ows:ContactInfo>
                    <ows:Phone>
                        <ows:Voice/>
                    </ows:Phone>
                    <ows:Address>
                        <ows:City>Marines Vineyard Coastal Observatory (MVCO) xlink:href="http://www.whoi.edu/mvco/"/>
                    </ows:Address>
                </ows:ContactInfo>
            </ows:ServiceContact>
        </ows:Provider>
    </ows:ServiceIdentification>
</ows:Capabilities>
```
This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
- <SensorML xmlns:schemaLocation="http://www.opengis.net/sensorML/1.0/sensorML-current.xsd" version="1.0">
  - <member>
    - <System gml:id="12m-Node">
      - <gml:description>
        Sensor was pumped or unpumped, depending on deployment
      </gml:description>
      - <keywords>
        - <KeywordList codeSpace="http://geonames.org/Geonames/keyword-list.html">
          <keyword>CEOLIA</keyword>
        </KeywordList>
      </keywords>
      - <identification>
        - <IdentifierList>
          - <Identifier>
            - <Term>
              <value>12m-Node at MVC0</value>
            </Term>
          </Identifier>
          - <Identifier>
            - <Term>
              <value>12m-Node</value>
            </Term>
          </Identifier>
          - <Identifier>
            - <Term>
              - <swe:codeSpace xlink:href="http://www.w3.org/1999/02/22-rdf-syntax-ns#ID"/>
              <value>urn:uuid:org:source:node#12m-Node</value>
            </Term>
          </Identifier>
          - <Identifier>
            - <Term>
              - <swe:codeSpace xlink:href="urn:xml.org:platform"/>
              <value>/</value>
            </Term>
          </Identifier>
        </IdentifierList>
      </identification>
    </System>
  </member>
</SensorML>
```
MVCO Buoy 12m-Node salinity Data

Latest Observations

<table>
<thead>
<tr>
<th>Time</th>
<th>Latitude</th>
<th>Longitude</th>
<th>Depth</th>
<th>Salinity</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007-04-01T2.48.00</td>
<td>41.3365</td>
<td>-70.5564</td>
<td>-12</td>
<td>32.183</td>
</tr>
</tbody>
</table>

Close
**12m-Node**

**Description:** Sensor was pumped or unpumped, depending on deployment

**Martha's Vineyard Coastal Observatory**
http://www.whoi.edu/evco

**Janet Fredericks**
Redlack@whoi.edu

**Keywords from http://gcmd.nasa.gov/Resources/valids/keyword_list.html**

**OCEANOGRAPHY**

**longName**
12m-Node at MVCO

**shortName**
12m-Node

**rdf-syntax-nS#ID**
nmvdco.org/source/nano12m-Node

**Valid begin time:**
2002-04-19T21:25:08Z

**Valid end time:**
now

**SYSTEM_LOCATION Point:**
41.3368 -70.5584 EPSG 81.4329

**Outputs:**
SailInlyDataOff12m-Node

**Additional Information:**
- Sensor was pumped or unpumped, depending on deployment
- Sensor data available for analysis
- Additional keywords and metadata provided for easy integration

**Map Viewer**
- Zoom to specific areas
- Organizations: Marthas Vineyard Coastal Observatory
- Map features: Satellite, Hybrid views

**Terms of Use**
- SOSThethy SOS Registry Services
- Programming, database and user interface design by GeoMOS
Summary of Registry Contents

Identifier: WHOI.salinity.2
Version: 1.0
Modified: 2007-01-02T12:33:10Z
Observation Name: Salinity
Status: Operating
Platform Name: SBP27D_s
Latitude: 41.3366
Longitude: -70.6964
Vertical Position: -1.45
Vertical Datum: MSL
Operator: Martha's Vineyard Coastal Observatory (WHOI)
Organization: MERACOOS
Start Date*: 2002-04-19T21:20:00Z
End Date*: 
Operator URI*: http://www.whoi.edu/mvoc/contact.htm
Platform URI*: 
Data URI*: http://www.whoi.edu/mvoc/data/oceandata.html
Comments*: see additional records for water temp

* optional
Important Future Initiatives?

QA/QC into SensorML: data dictionaries; setting them up in SensorML profiles; providing tutorials for implementation and demonstrating by integration into oostethys_sos