Open Geospatial Consortium, Inc.

Date: 2009-07-29

Reference number of this document: OGC 09-033

Version: 0.3.0

Category: Public Engineering Report

Editor: Simon Jirka, Arne Bröring

OGC® OWS-6 SensorML Profile for Discovery Engineering Report

Warning

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements.

Document type:	OpenGIS® Public Engineering Report
Document subtype:	NA
Document stage:	Approved for public release
Document language:	English

Preface

This draft document serves to describe a basic SensorML profile to be used in sensor discovery scenarios. The document defines a minimum set of metadata that has to be provided in order to use a SensorML document as input for populating a sensor registry. Furthermore a structure is defined, which ensures that metadata are described in a consistent way.

This work was developed under the Sensor Web Enablement thread during the OGC Web Services Phase 6 and is based on results of the EU funded projects OSIRIS[1] and GENESIS[2].

Suggested additions, changes, and comments on this draft report are welcome and encouraged. Such suggestions may be submitted by email message or by making suggested changes in an edited copy of this document.

The changes made in this document version, relative to the previous version, are tracked by Microsoft Word, and can be viewed if desired. If you choose to submit suggested changes by editing this document, please first accept all the current changes, and then make your suggested changes with change tracking on.

Forward

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium Inc. shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

[1] http://www.osiris-fp6.eu/

[2] http://genesis-fp7.eu/

OWS-6 Testbed

- OWS testbeds are part of OGC's Interoperability Program, a global, hands-on and collaborative prototyping program designed to rapidly develop, test and deliver Engineering Reports into OGC's Specification Program, where they are formalized for public release. In OGC's Interoperability Initiatives, international teams of technology providers work together to solve specific geoprocessing interoperability problems posed by the Initiative's sponsoring organizations. OGC Interoperability Initiatives include test beds, pilot projects, interoperability experiments and interoperability support services - all designed to encourage rapid development, testing, validation and adoption of OGC standards.

- In April 2008, the OGC issued a call for sponsors for an OGC Web Services, Phase 6 (OWS-6) Testbed activity. The activity completed in June 2009. There is a series of on-line demonstrations available here:

- http://www.opengeospatial.org/pub/www/ows6/index.html

- The OWS-6 sponsors are organizations seeking open standards to address their urgent interoperability requirements. After analyzing their requirements, the OGC Interoperability Team recommended to the sponsors that the content of the OWS-6 initiative be organized around the following threads:

1. Sensor Web Enablement (SWE)
2. Geo Processing Workflow (GPW)
3. Aeronautical Information Management (AIM)
4. Decision Support Services (DSS)
5. Compliance Testing (CITE)

Additional background on these threads and the Request for Quotation / Call For Participation (RFQ/CFP) issued by OGC can be found at:

http://www.opengeospatial.org/projects/initiatives/ows-6.

The OWS-6 sponsoring organizations were:

- U.S. National Geospatial-Intelligence Agency (NGA)
- Joint Program Executive Office for Chemical and Biological Defense (JPEO-CBD)
- GeoConnections - Natural Resources Canada
- U.S. Federal Aviation Agency (FAA)
- EUROCONTROL
- EADS Defence and Communications Systems
- US Geological Survey
- Lockheed Martin
- BAE Systems
- ERDAS, Inc.

The OWS-6 participating organizations were:

52North, AM Consult, Carbon Project, Charles Roswell, Compusult, con terra, CubeWerx, ESRI, FedEx, Galdos, Geomatys, GIS.FCU, Taiwan, GMU CSISS, Hitachi Ltd., Hitachi Advanced Systems Corp, Hitachi Software Engineering Co., Ltd., iGSI, GmbH, interactive instruments, lat/lon, GmbH, LISAsoft, Luciad, Lufthansa, NOAA MDL, Northrop Grumman TASC, OSS Nokalva, PCAvionics, Snowflake, Spot Image/ESA/Spacebel, STFC, UK, UAB CREAF, Univ Bonn Karto, Univ Bonn IGG, Univ Bunderswehr, Univ Muenster IfGI, Vightel, and Yumetech.

Contents

Page

1 Introduction 1
1.1 Scope 1
1.2 Document contributor contact points 1
1.3 Revision history 1
1.4 Future work 2
2 References 2
3 Terms and definitions 2
3.1 SensorML Profile 2
3.2 Sensor Discovery 2
4 Conventions 3
4.1 Symbols (and abbreviated terms) 3
5 SensorML Profiles Overview 4
6 SensorML Profile Description 4
6.1.1 System Description 5
6.1.2 Component Descriptions 10
7 SensorML Profile for Discovery 12
7.1.1 Common Profile Rules 13
7.1.2 System Specific Profile Rules 15
7.1.3 Component Specific Profile Rules 17
8 Summary and outlook 17
8.1 Summary 17
8.2 Issues 18

Listings

Page

Listing 6-1: Example of a keywords section 5
Listing 6-2: Example of an identification section 5
Listing 6-3: Example of a classification section 6
Listing 6-4: Example of a validTime section 6
Listing 6-5: Example of a capabilities section 7
Listing 6-6: Example of a contact section 8
Listing 6-7: Example of a position section 8
Listing 6-8: Example of a inputs section 9
Listing 6-9: Example of an outputs section 9

Listing 6-10: Example of a components section 10

Listing 6-11: Example of a metadata description of a component 10

Listing 6-12: Example of a position section of a component 11

Listing 6-13: Example of the inputs and outputs sections of a component 12

Listing 7-1: Schematron rule for ensuring that a KeywordList is provided 13

Listing 7-2: Schematron rule for ensuring that every identifier contains a definition 13

Listing 7-3: Schematron rule for ensuring that every classification element contains a definition 13

Listing 7-4: Schematron rule for ensuring that a uniqueID is provided 14

Listing 7-5: Schematron rule for ensuring that a longName and a shortName are provided 14

Listing 7-6: Schematron rule for restricting the structure of the capabilities section 14

Listing 7-7: Schematron rule for ensuring that the unit of measurement is provided if in the capabilities section a swe:Quantity is used 14

Listing 7-8: Schematron rule for ensuring that an inputs and an outputs section are provided 15

Listing 7-9: Schematron rule for ensuring that the inputs and outputs sections contain a definition element 15

Listing 7-10: Schematron rule fpr ensuring that a member element contains exactly one system 15

Listing 7-11: Schematron rule for ensuring that for every System a validTime element is provided 15

Listing 7-12: Schematron rule for ensuring that an observedBBOX is provided 16

Listing 7-13: Schematron rule for ensuring that for every System a contact element is provided 16

Listing 7-14: Schematron rule for ensuring that a Position (including a referenceFrame attribute) is provided 16

Listing 7-15: Schematron rule for defining the description of the position 17

Listing 7-16: Schematron rule for defining the format of coordinates 17

Listing 7-17: Schematron rule for ensuring that descriptions for the Components of a System are provided 17

Listing 7-18: Schematron rule for ensuring that for every Component the sensor type is described 17

OGC® OWS-6 SensorML Profile for Discovery Engineering Report

1 Introduction

1.1 Scope

This document defines a basic SensorML profile for discovery purposes. Besides a minimum set of metadata also the structure of according SensorML documents is defined in order to ensure a consistent metadata description. This goal is achieved by a set of Schematron rules that can be used to validate if a given SensorML document complies with the profile described in this engineering report.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The OGC shall not be held responsible for identifying any or all such patent rights.

1.2 Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organization
Simon Jirka	University of Münster - Institute for Geoinformatics
Arne Böring	University of Münster - Institute for Geoinformatics

1.3 Revision history

Date	Release	Editors	Primary clauses modified	Description
2009/02/26	0.0.1	Arne Broering	all	initial document
2009/03/11	0.0.1	Simon Jirka	6.x	SensorML example edited
2009/03/12	0.0.1	Arne Broering	all	
2009/03/14	0.0.1	Simon Jirka	all	
2009/03/17	0.0.1	Arne Broering	7.x	Update of the schematron rules
2009/03/18	0.0.1	Simon Jirka	all	Minor revisions
2009/04/06	1.0.0	Simon Jirka	all	Minor revisions
2009/07/10	0.3.0	Carl Reed	Various	Prepare for posting as public ER

1.4 Future work

It is anticipated that this document will be subject of further additional work. Especially the alignment with existing metadata models for the OGC Catalogue will be a topic of future work.

Furthermore, several elements of this SensorML profile rely on URIs in order to point to definitions of their values (e.g. referencing the definition of the phenomenon that is observed by the sensors). It will be important to provide dictionaries for these URIs so that a consistent interpretation and use of such identifiers is ensured.

2 References

The following documents are referenced in this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

OGC 06-121r3, *OpenGIS® Web Services Common Standard*

OGC 07-000, *OpenGIS Sensor Model Language (SensorML) Standard*, Version 1.0

OGC 07-122r2, *OpenGIS SensorML Encoding Standard v 1.0 Schema Corregendum 1 (1.01)*, Version 1.0.1

OGC 03-105r1, *OpenGIS Geography Markup Language (GML) Encoding Standard*, Version 3.1.1

OGC 05-099r2, *GML 3.1.1 simple dictionary profile*

OGC 07-006r1, *OpenGIS Catalog Service Implementation Standard*

In addition to this document, this report includes an XML Schema Document file as specified in Annex A.

3 Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Specification [OGC 06-121r3] shall apply. In addition, the following terms and definitions apply.

3.1 SensorML Profile

A restriction of the SensorML schema.

3.2 Sensor Discovery

The process of searching for sensors or SWE services that encapsulate them.

4 Conventions

4.1 Symbols (and abbreviated terms)

CRS	Coordinate Reference System
EU	European Union
GENESIS	GENeric European Single Information Space for environment
GML	Geographic Markup Language
gml:*	Namespace prefix for the GML schema
IP	Interoperability Program
ISO	International Organization for Standardization
OGC	Open Geospatial Consortium
om:*	Namespace prefix for the Observations and Measurement schema
OSIRIS	Open architecture for Smart and Interoperable networks in Risk management based on In-situ Sensors
OWS	OGC Open Web Services
O&M	Observations and Measurements
SAS	Sensor Alert Service
SensorML	Sensor Model Language
sml:*	Namespace prefix for the SensorML schema
SOS	Sensor Observation Service
SPS	Sensor Planning Service
SWE	Sensor Web Enablement
swe:*	Namespace prefix for the SWE Common schema
TML	Transducer Markup Language
UCUM	Unified Code for Units of Measure
URI	Uniform Resource Identifier

URN	Uniform Resource Name
WGS84	World Geodetic System 1984
WNS	Web Notification Service
XML	eXtensible Markup Language
xs:*	Namespace prefix for XMLSchema.2002

5 SensorML Profiles Overview

This document defines a profile of the SensorML OGC standard to be used by sensor and SWE service discovery services and clients. Before the profile definition is presented we outline an example SensorML description of a home weather station. The presented SensorML instance relies strongly on the example of a weather station published by ROBIN (2006). This existing SensorML weather station description is adjusted to the most recent SensorML schema version. This example is supposed to support the reader to understand the specified profile definition. However, the profile is not restricted to weather stations. Instead, it can be used as a generic profile for sensor system descriptions with the purpose of being discoverable.

6 SensorML Profile Description

This section explains the SensorML profile for sensor and SWE service discovery based on a SensorML example that formally describes a home weather station which is able to measure ambient temperature, wind speed, wind direction and rain fall (in order to reduce the complexity of this document only the thermometer, wind sensors and the rain gauge are described). The station is modeled as a `System` which allows a very accurate description of the weather station.

The station can be considered as a platform for several detectors. These detector components are aggregated by the station. In the present example the `System` logically incorporates the following sensors:

- A thermometer
- A wind speed sensor
- A wind direction sensor
- A rain gauge

These logical components do not have to match the real distinct sensors assembled on the weather station. For instance the wind speed and wind direction detectors are in fact realized by one wind measurement recorder. This proceeding is necessary since these two detectors measure different quantities.

6.1.1 System Description

The SensorML `System` may contain multiple sections which allow a detailed description of the station's metadata. We use the sections `keywords`, `identification`, `classification`, `validTime`, `capabilities`, `contact`, `inputs` and `outputs` to depict the station. These metadata sections are especially useful for the discovery of the `System` by the means of specific catalogue services. The data of these elements can be harvested by a catalogue service so that users are able to register and discover sensors or sensor platforms which fit their needs.

The `keywords` section provides a list of keywords which may be used by a user to search for a sensor. This profile requires that at least the observed phenomenon has to be mentioned here. Listing 6-1 shows an excerpt of this section within the SensorML document of the exemplary home weather station. The `keywords` section must be provided by any `System`.

Listing 6-1: Example of a keywords section

```
<keywords>
     <KeywordList>
          <keyword>weather station</keyword>
          <keyword>precipitation</keyword>
          <keyword>wind speed</keyword>
          <keyword>temperature</keyword>
     </KeywordList>
</keywords>
```

The `identification` section informs about general attributes which identify the system among others. This profile requires that at least the following elements are present for any `System`:

- One identifier has to carry the definition "`urn:ogc:def:identifier:OGC:1.0:uniqueID`". The value of its contained "`Term`" element uniquely identifies the instance.
- One identifier has to contain the definition "`urn:ogc:def:identifier:OGC:1.0:longName`". The value of its contained "`Term`" element represents a human understandable name for the instance.
- One identifier has to contain the definition "`urn:ogc:def:identifier:OGC:1.0:shortName`". The value of its contained "`Term`" element represents a short representation of the human understandable name for the instance.

Listing 6-2 shows an excerpt of this section within the SensorML document of the exemplary home weather station.

Listing 6-2: Example of an identification section

```
<Identification>
```

```
    <IdentifierList>
        <identifier name="uniqueID">
            <Term definition="urn:ogc:def:identifier:OGC:1:0:uniqueID">
                <value>urn:ogc:object:feature:Sensor:IFGI:
weatherStation123</value>
            </Term>
        </identifier>
        <identifier name="longName">
            <Term definition="urn:ogc:def:identifier:OGC:1.0:longName">
                <value>OSIRIS weather station 123 on top of the IfGI
building</value>
            </Term>
        </identifier>
        <identifier name="shortName">
            <Term definition="urn:ogc:def:identifier:OGC:1.0:shortName">
                <value>OSIRIS Weather Station 123</value>
            </Term>
        </identifier>
    </IdentifierList>
</identification>
```

The classification section gives further information regarding the type of the `System`. This information may include the intended application of the `System` or the types of the affiliated sensors (see Listing 6-3).

Listing 6-3: Example of a classification section

```
<classification>
    <ClassifierList>
        <classifier name="intendedApplication">
            <Term definition="urn:ogc:def:classifier:OGC:1.0:application">
                <value>weather</value>
            </Term>
        </classifier>
        <classifier name="sensorType">
            <Term definition="urn:ogc:def:classifier:OGC:1.0:sensorType">
                <value>thermometer</value>
            </Term>
        </classifier>
    </ClassifierList>
</classification>
```

The `validTime` section provides information about the time period in which a sensor description is valid. Each `System` must contain this information. As the SensorML schema defines this can be either a `gml:TimePeriod` or a `gml:TimeInstant`. If a `TimeInstant` is provided in the validTime section, the `TimeInstant` must describe the point in time at which the stated metadata about the `System` has become valid (see Listing 6-4).

Listing 6-4: Example of a validTime section

```
<validTime>
    <gml:TimePeriod>
        <gml:beginPosition>2009-01-15</gml:beginPosition>
        <gml:endPosition>2009-01-20</gml:endPosition>
    </gml:TimePeriod>
</validTime>
```

The `capabilities` element may be used to specify attributes which capture configuration of the present station instance or describe the current status. A `swe:DataRecord` containing a number of `swe:field` elements must be used here to specify the capabilities of the `System` or `Component`. If the child-element of the `swe:Field` is a `swe:Quantity` it has to contain the `swe:uom` element which specifies the `code` attribute.

One `swe:field` has to contain a `swe:Envelope` element with the definition `urn:ogc:def:property:OGC:1.0:observedBBOX`. It describes the bounding box of the area that is observed by the `System`. In case of an in-situ sensor this bounding box only contains the position of the sensor.

An example of this section is presented in Listing 6-5.

Listing 6-5: Example of a capabilities section

```
<capabilities>
     <swe:DataRecord definition="urn:ogc:def:property:capabilities">
          <swe:field name="status">
               <swe:Text definition="urn:ogc:def:property:OGC:1.0:status">
                    <gml:description>The operating status of the system.
                    </gml:description>
                    <swe:value>active</swe:value>
               </swe:Text>
          </swe:field>
          <swe:field name="observedBBOX">
               <swe:Envelope
definition="urn:ogc:def:property:OGC:1.0:observedBBOX">
                    <swe:lowerCorner>
                         <swe:Vector>
                              <swe:coordinate name="easting">
                                   <swe:Quantity axisID="x">
                                        <swe:uom code="m"/>
                                        <swe:value>2592308.332</swe:value>
                                   </swe:Quantity>
                              </swe:coordinate>
                              <swe:coordinate name="northing">
                                   <swe:Quantity axisID="y">
                                        <swe:uom code="m"/>
                                        <swe:value>5659592.542</swe:value>
                                   </swe:Quantity>
                              </swe:coordinate>
                         </swe:Vector>
                    </swe:lowerCorner>
                    <swe:upperCorner>
                         <swe:Vector>
                              <swe:coordinate name="easting">
                                   <swe:Quantity axisID="x">
                                        <swe:uom code="m"/>
                                        <swe:value>2592308.332</swe:value>
                                   </swe:Quantity>
                              </swe:coordinate>
                              <swe:coordinate name="northing">
                                   <swe:Quantity axisID="y">
                                        <swe:uom code="m"/>
                                        <swe:value>5659592.542</swe:value>
                                   </swe:Quantity>
                              </swe:coordinate>
                         </swe:Vector>
```

```
                    </swe:upperCorner>
               </swe:Envelope>
          </swe:field>
     </swe:DataRecord>
</capabilities>
```

The following `contact` section references the organization or person which is responsible for the sensor. This information must be provided for every `System`. In general this is the provider of the sensor (see Listing 6-6).

Listing 6-6: Example of a contact section

```
<contact>
     <ResponsibleParty gml:id="WWU_IfGI_weather_station_contact">
          <organizationName>Westfaelische Wilhelms-Universitaet Muenster,
Institute for Geoinformatics, Sensor Web and Simulation Lab</organizationName>
          <contactInfo>
               <address>
                    <electronicMailAddress>swsl-ifgi@listserv.uni-
muenster.de</electronicMailAddress>
               </address>
          </contactInfo>
     </ResponsibleParty>
</contact>
```

The `position` section specifies the sensor's position in space. It must be given for every `System`. Listing 6-7 presents an exemplary definition of a three dimensional position.

Listing 6-7: Example of a position section

```
<position name="stationPosition">
     <swe:Position referenceFrame="urn:ogc:def:crs:EPSG:6.14:31466">
          <swe:location>
               <swe:Vector gml:id="SYSTEM_LOCATION">
                    <swe:coordinate name="easting">
                         <swe:Quantity axisID="x">
                              <swe:uom code="m"/>
                              <swe:value>2592308.332</swe:value>
                         </swe:Quantity>
                    </swe:coordinate>
                    <swe:coordinate name="northing">
                         <swe:Quantity axisID="y">
                              <swe:uom code="m"/>
                              <swe:value>5659592.542</swe:value>
                         </swe:Quantity>
                    </swe:coordinate>
                    <swe:coordinate name="altitude">
                         <swe:Quantity axisID="z">
                              <swe:uom code="m"/>
                              <swe:value>297.0</swe:value>
                         </swe:Quantity>
                    </swe:coordinate>
               </swe:Vector>
          </swe:location>
     </swe:Position>
</position>
```

The `inputs` section (Listing 6-8) is mandatory for every `System`. It lists up the phenomena observed by the different sensors of the system. Therefore every `input` utilizes the `swe:ObservableProperty` and its `definition` attribute which takes a URN to point to a dictionary entry. This dictionary entry defines the measured phenomenon in detail and declares its semantics.

Listing 6-8: Example of a inputs section

```
<inputs>
    <InputList>
        <input name="precipitation">
            <swe:ObservableProperty
definition="urn:ogc:def:property:OGC:1.0:precipitation"/>
        </input>
        <input name="wind">
            <swe:ObservableProperty
definition="urn:ogc:def:property:OGC:1.0:wind"/>
        </input>
        <input name="atmosphericTemperature">
            <swe:ObservableProperty
definition="urn:ogc:def:property:OGC:1.0:temperature"/>
        </input>
    </InputList>
</inputs>
```

Within the `outputs` section (Listing 6-9) which is mandatory for every `System`, the signals recorded by the sensors are defined. The sub-element of each output references the captured phenomenon and specifies the unit of measure in which the measured values are expressed.

Listing 6-9: Example of an outputs section

```
<outputs>
    <OutputList>
        <output name="precipitation">
            <swe:Quantity
definition="urn:ogc:def:property:OGC:1.0:precipitation">
                <swe:uom code="mm"/>
            </swe:Quantity>
        </output>
        <output name="windDirection">
            <swe:Quantity
definition="urn:ogc:def:property:OGC:1.0:windDirection">
                <swe:uom code="deg"/>
            </swe:Quantity>
        </output>
        <output name="windSpeed">
            <swe:Quantity definition="urn:ogc:def:property:OGC:1.0:windSpeed">
                <swe:uom code="m/s"/>
            </swe:Quantity>
        </output>
        <output name="temperature">
            <swe:Quantity
definition="urn:ogc:def:property:OGC:1.0:temperature">
                <swe:uom code="Cel"/>
            </swe:Quantity>
        </output>
    </OutputList>
</outputs>
```

The distinct sensors of the station are separately modeled as `Components` which are associated with the whole `System`. The `components` section of the SensorML description lists up all these sensors. The description of a component can be either defined inline within the `System` document or it can be referenced using the `xlink:href` attribute. In the example below (Listing 6-10) both alternatives are shown.

Listing 6-10: Example of a components section

```
<components>
    <ComponentList>
        <component name="rainGauge"
xlink:href="http://mySensorMLregistry.com?object=98765"/>
        <component name="anemoneter"
xlink:href="http://mySensorMLregistry.com?object=33333"/>
        <component name="thermometer" >
        <component>
            … <!-- inline description of Component -->
        </component>
        </component>
    </ComponentList>
</components>
```

6.1.2 Component Descriptions

To describe the model of the different sensors associated with the system we make use of a SensorML `Component`. A `Component` is similar structured to the `System` element. In the following we describe the encoding of a `Component` by the example of a temperature sensor affiliated with the weather station.

To specify the basic metadata of a `Component` the sections `keywords`, `identification`, `classification`, `capabilities`, `inputs` and `outputs` are used.

Listing 6-11 shows an exemplary instantiation of these sections. The sections `keywords` and `identification` must provide the same information items for any `Component` as for a `System`. In addition in case of a `Component` the `classification` section must provide the information about the sensor type. Within the capabilities section of a `Component` the `observedBBox` is not necessary. If it is not present a registry can assume that the `observedBBox` is identical for the `Component` and the `System` it belongs to.

Listing 6-11: Example of a metadata description of a component

```
<keywords>
     <KeywordList>
          <keyword>weather station</keyword>
          <keyword>temperature</keyword>
     </KeywordList>
</keywords>
<identification>
     <IdentifierList>
          <identifier name="uniqueID">
               <Term definition="urn:ogc:def:identifier:OGC:uniqueID">
                  value>urn:ogc:object:feature:Sensor:IFGI:thermometer123</value>
               </Term>
```

```
          </identifier>
          <identifier name="longName">
               <Term definition="urn:ogc:def:identifier:OGC:1.0:longName">
                    <value>OSIRIS Thermometer at weather station 123</value>
               </Term>
          </identifier>
          <identifier name="shortName">
               <Term definition="urn:ogc:def:identifier:OGC:1.0:shortName">
                    <value>OSIRIS Thermometer 123</value>
               </Term>
          </identifier>
     </IdentifierList>
</identification>
<classification>
     <ClassifierList>
          <classifier name="sensorType">
               <Term definition="urn:ogc:def:classifier:OGC:1.0:sensorType">
                    <value>thermometer</value>
               </Term>
          </classifier>
     </ClassifierList>
</classification>
<capabilities>
     <swe:DataRecord definition="urn:ogc:def:property:capabilities">
          <swe:field name="status">
               <swe:Text definition="urn:ogc:def:property:OGC:1.0:status">
                    <gml:description>The operating status of the
system.</gml:description>
                    <swe:value>active</swe:value>
               </swe:Text>
          </swe:field>
     </swe:DataRecord>
</capabilities>
```

Within the `position` section of a `Component` the position of a sensor can be specified (Listing 6-12). However for a Component this section is optional. If it is not present a registry can assume that the `position` is identical for the `Component` and the `System` it belongs to.

Listing 6-12: Example of a position section of a component

```
<position name="stationPosition">
     <swe:Position referenceFrame="urn:ogc:def:crs:EPSG:6.14:31466">
          <swe:location>
               <swe:Vector gml:id="SYSTEM_LOCATION">
                    <swe:coordinate name="easting">
                         <swe:Quantity axisID="x">
                              <swe:uom code="m"/>
                              <swe:value>2592308.332</swe:value>
                         </swe:Quantity>
                    </swe:coordinate>
                    <swe:coordinate name="northing">
                         <swe:Quantity axisID="y">
                              <swe:uom code="m"/>
                              <swe:value>5659592.542</swe:value>
                         </swe:Quantity>
                    </swe:coordinate>
                    <swe:coordinate name="altitude">
                         <swe:Quantity axisID="z">
                              <swe:uom code="m"/>
                              <swe:value>297.0</swe:value>
                         </swe:Quantity>
```

```
                    </swe:coordinate>
                </swe:Vector>
            </swe:location>
        </swe:Position>
</position>
```

The `inputs` section of the `Component` specifies the property observed by the sensor. The `outputs` element informs about the unit of measurement which is used when data of this observed property are requested from a Sensor Observation Service (Listing 6-13).

Listing 6-13: Example of the inputs and outputs sections of a component

```
<inputs>
    <InputList>
        <input name="atmosphericTemperature">
            <swe:ObservableProperty
definition="urn:ogc:def:property:OGC:1.0:temperature"/>
        </input>
    </InputList>
</inputs>

<outputs>
    <OutputList>
        <output name="temperature">
            <swe:Quantity
definition="urn:ogc:def:property:OGC:1.0:temperature">
                <swe:uom code="Cel"/>
            </swe:Quantity>
        </output>
    </OutputList>
</outputs>
```

7 SensorML Profile for Discovery

The document outlined above is a SensorML instance which conforms to the profile which we describe next. Nonetheless the profile can be considered as a generic guideline for the design of sensor system descriptions which shall be harvestable by discovery services.

The SensorML data model specifies a majority of its elements as optional. It allows expressing the same information in several, differently structured ways. This open and flexible structure was one of the main aims of the SensorML design in order to make it possible to apply the data model to nearly any type of sensor. For ensuring that SensorML documents which are intended for discovery purposes can be reliably handled by automatic harvesting mechanisms, it is necessary to create a profile for SensorML that defines the information which must be contained in a SensorML document as well as the structure in which the metadata must be encoded.

The profile is intended to be primarily applied to SensorML documents returned by the DescribeSensor operation of a Sensor Observation Service (or that are provided by other types of SWE services). It is assumed that the request of a single sensor description returns the entire model of a system to which it affiliates. Since the entire system model

incorporates the associated sensor descriptions, this approach enables harvesting engines to access the information of the aggregating system as well as its associated components.

To define a formal SensorML profile the Schematron (JELIFFE 2005) language is applied. This language is highly qualified for this task because it allows the formulation of separate rules which restrict an existing schema. The original XML schema does not have to be modified. Instead the profile can be applied to the schema.

The set of Schematron rules which make up the profile are presented in the following. The profile contains rules to restrict a `System` as well as rules that restrict the `Component` element. These are shown in Section 7.1.2 and 7.1.3 respectively. But first, Section 7.1.1 describes rules that have to be executed during the validation process of both element types.

7.1.1 Common Profile Rules

In order to make sure that a list of keyword describing the sensor is provided the Schematron rule shown in Listing 7-1 has been defined.

Listing 7-1: Schematron rule for ensuring that a KeywordList is provided

```
<rule context="//sml:System">
    <assert test="sml:keywords/sml:KeywordList">Error: 'KeywordList' element has
to be present</assert>
</rule>
<rule context="//sml:Component">
    <assert test="sml:keywords/sml:KeywordList">Error: 'KeywordList' element has
to be present</assert>
</rule>
```

The rule defined in Listing Listing 7-2 restricts the possible structure of the `identification` section. Each listed identifier has to contain the `definition` attribute. The value of this attribute serves as a link to the semantic description of the identifier. Listing Listing 7-3 specifies a similar rule for the `classification` section.

Listing 7-2: Schematron rule for ensuring that every identifier contains a definition

```
<rule context="//sml:identification/sml:IdentifierList/sml:identifier/sml:Term">
     <assert test="string-length(@definition) > 0">Error: 'definition' attribute
has to be present and its value has to be > 0.</assert>
</rule>
```

Listing 7-3: Schematron rule for ensuring that every classification element contains a definition

```
<rule context="//sml:classification/sml:ClassifierList/sml:classifier/sml:Term">
     <assert test="string-length(@definition) > 0">Error: 'definition' attribute
has to be present and its value has to be > 0.</assert>
</rule>
```

To be able to identify a `System` or `Component` uniquely the rule in Listing 7-4 is defined. One `identifier` of the `identification` section must declare a `definition`

attribute with the value `urn:ogc:def:identifier:OGC:1.0:uniqueID`. Then the value of the `identifier`'s `Term` element uniquely identifies the instance.

Listing 7-4: Schematron rule for ensuring that a uniqueID is provided

```
<rule context="//sml:identification">
     <assert test="count(sml:IdentifierList/sml:identifier/sml:Term[@definition
= 'urn:ogc:def:identifier:OGC:uniqueID']) = 1" >Error: one identifier has to be
of the type 'urn:ogc:def:identifier:OGC:uniqueID'.</assert>
</rule>
```

Furthermore each sensor shall possess a long and a short name (e.g. weather station 123). This is ensured by the rule shown in Listing 7-5.

Listing 7-5: Schematron rule for ensuring that a longName and a shortName are provided

```
<rule context="//sml:identification">
     <assert test="count(sml:IdentifierList/sml:identifier/sml:Term[@definition
= 'urn:ogc:def:identifier:OGC:longName']) = 1" >Error: one identifier has to be
of the type 'urn:ogc:def:identifier:OGC:longName'.</assert>
</rule>
<rule context="//sml:identification">
     <assert test="count(sml:IdentifierList/sml:identifier/sml:Term[@definition
= 'urn:ogc:def:identifier:OGC:shortName']) = 1" >Error: one identifier has to be
of the type 'urn:ogc:def:identifier:OGC:shortName'.</assert>
</rule>
```

Listing Listing 7-6 restricts the structure of the `capabilities` section. To ease the usage of the `capabilities` element it is defined that a `swe:DataRecord` containing a number of `swe:field` elements must be used to specify the capabilities of a `System` or `Component`. The child element of each `swe:Field` element has to contain a `definition` attribute so that a client is able to look up the semantics of the capability. The next rule (Listing 7-7) defines that if the child-element of the `swe:Field` is a `swe:Quantity` it has to contain the `swe:uom` element which specifies the `code` attribute.

Listing 7-6: Schematron rule for restricting the structure of the capabilities section

```
<rule context="//sml:capabilities/swe:DataRecord/swe:field">
     <assert test="string-length(child::node()[@definition]) > 0">Error:
'definition' attribute has to be present and its value has to be > 0.</assert>
</rule>
```

Listing 7-7: Schematron rule for ensuring that the unit of measurement is provided if in the capabilities section a swe:Quantity is used

```
<rule
context="//sml:capabilities/swe:DataRecord/swe:field/swe:Quantity/swe:uom">
     <assert test="string-length(@code) > 0">Error: 'code' attribute has to be
present and its value has to be > 0.</assert>
</rule>
```

The rules of Listing 7-8 define that the `inputs` as well as the `outputs` section has to be present for a `System` or a `Component`.

Listing 7-8: Schematron rule for ensuring that an inputs and an outputs section are provided

```
<rule context="//sml:System">
     <assert test="sml:inputs">Error: 'sml:inputs' has to be present.</assert>
</rule>
<rule context="//sml:Component">
     <assert test="sml:inputs">Error: 'sml:inputs' has to be present.</assert>
</rule>

<rule context="//sml:System">
     <assert test="sml:outputs">Error: 'sml:outputs' has to be present.</assert>
</rule>
<rule context="//sml:Component">
     <assert test="sml:outputs">Error: 'sml:outputs' has to be present.</assert>
</rule>
```

The structures of the `inputs` and `outputs` sections are restricted by the rules of Listing 7-9. It is specified that the child elements of each `input` and `output` element must possess a `definition` attribute. The value of this attribute can be considered as a link to the semantics of the `input` and `output`.

Listing 7-9: Schematron rule for ensuring that the inputs and outputs sections contain a definition element

```
<rule context="//sml:inputs/sml:InputList/sml:input">
    <assert test="swe:ObservableProperty/@definition">Error!</assert>
</rule>


<rule context="//sml:outputs/sml:OutputList/sml:output">
    <assert test="child::node()[@definition]">Error!</assert>
</rule>
```

7.1.2 System Specific Profile Rules

A SensorML document has to incorporate one `member` element which contains exactly one `System`. This rule is expressed by the following Schematron snippet:

Listing 7-10: Schematron rule fpr ensuring that a member element contains exactly one system

```
<rule context="/">
    <assert test="count(sml:SensorML/sml:member) = 1">Error!</assert>
    <assert test="count(sml:SensorML/sml:member/sml:System) =
1">Error!</assert>
</rule>
```

Each `System` must contain information about the time it is valid. As the SensorML schema defines this can be either a `gml:TimePeriod` or a `gml:TimeInstant` describing the instant in time at which the stated metadata about the System has become valid.

Listing 7-11: Schematron rule for ensuring that for every System a validTime element is provided

```
<rule context="//sml:System">
     <assert test="sml:validTime">Error: 'validTime' element has to be
present</assert>
```

```
</rule>
```

The dataRecord within the capabilities section of the System has to contain one `swe:field` which specifies a `swe:Envelope` element with the definition `urn:ogc:def:property:OGC:1.0:observedBBOX`. It describes the bounding box of the area that is observed by the System. In case of an in-situ sensor this bounding box only contains the position of the sensor.

Listing 7-12: Schematron rule for ensuring that an observedBBOX is provided

```
<rule context="//sml:capabilities">
     <assert test="count(swe:DataRecord/swe:field/swe:Envelope[@definition =
'urn:ogc:def:property:OGC:1.0:observedBBOX']) = 1" >Error: one "swe:field" of
the "DataRecord" has to contain a "swe:Envelope" element with the definition
"urn:ogc:def:property:OGC:1.0:observedBBOX".</assert>
</rule>
```

A "`contact`" element has to be present for each `System`.

Listing 7-13: Schematron rule for ensuring that for every System a contact element is provided

```
<rule context="//sml:System">
     <assert test="sml:contact">Error: 'sml:contact"' element has to be
present</assert>
</rule>
```

A `System` description must define a position in space. Listing 7-14 - </rule>

Listing 7-16 specify the encoding of this position and restrict the SensorML schema which offers several options to do this. Listing 7-14 specifies the rule that a `position` element has to contain a `swe:Position` and it redefines the "optional" `referenceFrame` attribute of a `swe:Position` as "required". The `swe:Position` element has to incorporate a `swe:location` containing at least two `swe:Vector/swe:coordinate/swe:Quantity` elements to express the position coordinates (</rule>

Listing 7-15). </rule>

Listing 7-16 specifies three conditions. First, it is required that this `swe:Quantity`, which is used to declare the coordinate value, owns an `axisID` attribute to link to the axis to which it refers. Second, a `swe:value` element is utilized to specify the coordinate value. And third the `swe:uom` element uses a `code` attribute to define the unit of the coordinate value.

Listing 7-14: Schematron rule for ensuring that a Position (including a referenceFrame attribute) is provided

```
<rule context="//sml:position/swe:Position">
    <assert test="@referenceFrame">Error!</assert>
</rule>
```

Listing 7-15: Schematron rule for defining the description of the position

```
<rule context="//sml:position/swe:Position/swe:location">
    <assert test="count(swe:Vector/swe:coordinate/swe:Quantity)>
1">Error!</assert>
</rule>
```

Listing 7-16: Schematron rule for defining the format of coordinates

```
<rule
context="//sml:position/swe:Position/swe:location/swe:Vector/swe:coordinate/swe
:Quantity">
    <assert test="string-length(@axisID) > 0">Error!</assert>
    <assert test="swe:value">Error!</assert>
    <assert test="swe:uom[@code]">Error!</assert>
</rule>
```

The `components` section of a system description lists the descriptions of the associated sensors. The `component` element must contain either the attribute `xlink:href` to specify an external reference to the sensor description or a `Component` as a child element which describes the sensor inline. This condition is formulated in Listing 7-17.

Listing 7-17: Schematron rule for ensuring that descriptions for the Components of a System are provided

```
<rule context="//sml:System/sml:components/sml:ComponentList/sml:component">
    <assert test="(@xlink:href and not(sml:Component)) or (not (@xlink:href)
and sml:Component)">Error!</assert>
</rule>
```

7.1.3 Component Specific Profile Rules

Besides the common rules defined in section 7.1.1 a `Component` description has to fulfill the Schematron rule specified in Listing 7-18. It defines that a `Component` element has to contain at least one classifier with the definition "`urn:ogc:def:classifier:OGC:1.0:sensorType`". The value of its contained `Term` element states the type of the sensor.

Listing 7-18: Schematron rule for ensuring that for every Component the sensor type is described

```
<rule context="//sml:Component/sml:classification">
     <assert test="count(sml:ClassifierList/sml:classifier/sml:Term[@definition
= 'urn:ogc:def:classifier:OGC:1.0:sensorType']) >= 1" >Error!</assert>
</rule>
```

8 Summary and outlook

8.1 Summary

This engineering report has shown an approach for defining a SensorML profile that can be used for sensor and SWE service discovery. The profile was developed under the

Sensor Web Enablement thread during the OGC Web Services Phase 6. Furthermore it integrates results and experiences gained during the EU funded projects OSIRIS and GENESIS.

A first version of this profile has already been successfully applied within the OSIRIS project. The experiences gained during the project have shown that the availability of such a SensorML discovery profile brings significant advantages when building harvesting mechanisms in order to populate sensor and SWE service registries.

8.2 Issues

Further issues have to be addressed in the context of sensor discovery that are related to the SensorML profile specified in the engineering report. This includes especially the following two aspects:

- Time dependency of metadata: Sensor networks often possess a highly dynamic structure: sensors may be deployed or removed, sensor parameters can continuously change, mobile sensors may observe different areas depending on the time, etc. The `validTime` element within the presented SensorML profile is a first approach to allow a time dependent metadata description. However, there is a need to investigate mechanisms for quickly updating sensor metadata within SWE services and registries but also to encode efficiently such metadata changes (e.g. not sending a whole new SensorML document if just one element has been changed)

- Dictionaries for identifiers: For example, the phenomena measured by a sensor are identified within a SensorML document by URIs. In order to ensure a consistent use of these phenomenon identifiers and to make the definitions that are assigned to these URIs accessible, it is necessary to provide a phenomenon dictionary or registry. An important functionality would be an operation providing access to the phenomenon definitions (e.g. resolving the URIs) but also for exploiting semantic relationships (e.g. finding equivalent or similar definitions).[3]

- Non-physical processes (Process model and process chain)

[3] An example for such a dictionary (in this case for managing phenomenon definitions) is the Sensor Observable Registry developed by the OSIRIS project.. More information about this approach can be found in the following document: http://www.osiris-fp6.eu/doc/OSIRIS-WP6000-DEL-0037-Archi%20Spec%20and%20Justification.pdf

Bibliography

[1] Robin, A. (2006) - Tutorial I: Using SensorML to describe a Complete Weather Station. Online at: http://vast.nsstc.uah.edu/downloads/documents/SensorML_Tutorial1-Weather_Station_System_preV1.0.pdf

[2] Jeliffe, R. (Ed.) ISO 19757-3 Document Schema Definition Languages (DSDL) - Part 3: Rulebased validation - Schematron. International Organization for Standardization (ISO): Geneva, Switzerland, 2005