# 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



### Warning

This document is not an OGC Standard. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type: OpenGIS® Implementation Specification
Document subtype: Draft new Implementation Specification
Document stage: Request for Comment
Document language: English

# Contents

 

 

# Figures Page



# Tables Page



 

# i. Preface

This document specifies the interface to a Sensor Observation Service, hereinafter "SOS". The SOS is one of a family of draft specifications that make up the OGC Sensor Web Enablement activity, hereinafter "SWE". Currently, the other draft specifications that pertain to SWE are Sensor Model Language (SensorML), Observations and Measurements (O&M), Sensor Planning Service (SPS), Transducer Markup Language (TML), Sensor Alert Service (SAS), and Web Notification Service (WNS). Work on the predecessor to SOS, Sensor Collection Service (SCS), began during the OGC Web Services 1.1 (OWS 1.1) testbed initiative and focused mainly on the interoperable interface requirements for in-situ sensors and sensor networks. During the OWS 1.2 initiative, the SCS effort focused on the requirements for dynamic remote sensors (and sensor networks) and defined a more explicit interface employing SensorML. The SOS builds on previous SCS efforts and also incorporates an interface employing TML.

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

# ii. Document terms and definitions

This document uses the specification terms defined in Subclause 5.3 of [OGC 05-008], which is based on the ISO/IEC Directives, Part 2. Rules for the structure and drafting of International Standards. In particular, the word "shall" (not "must") is the verb form used to indicate a requirement to be strictly followed to conform to this specification.

 

## iii. Submitting organizations

The following organizations submitted this document to the Open GIS Consortium Inc.

a. IRIS Corporation

b. 3eTI

c. University of Alabama-Huntsville

d. IFGI

e. York University

## iv. Document contributor contact points

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

| Contact | Company | Address |
|---|---|---|
| Arthur Na (editor) | IRIS Corp. | 4220 Varsity Dr, Suite E, Ann Arbor, MI 48108 |
| Mark Priest (editor) | 3eTI | 700 King Farm Blvd., Suite 600, Rockville MD 20850 |
| Simon Cox | CSIRO | CSIRO Exploration and Mining, Perth, W.A., Australia |
| Mike Botts | UAH | GHCC-NSSTC, Huntsville AL 35899 |
| Alexander Robin | UAH | GHCC-NSSTC, Huntsville AL 35899 |
| Alexander C. Walkowski | IFGI | Institute for GeoInformatics, University of Muenster, Germany |
| Ingo Simonis | IFGI | Institute for GeoInformatics, University of Muenster, Germany |
| Steve Liang | York University | York University, 4700 Keele St., Toronto, ON M3J 1P3 |
| John Davidson | Image Matters | 201 Loudoun St, SW, Leesburg, VA 20175 |
| Harry Niedzwiadek | Image Matters | 201 Loudoun St, SW, Leesburg, VA 20175 |

## v. Revision history

| Date | Release | Editor | Primary clauses modified | Description |
|---|---|---|---|---|
| 10/10/05 | 0.1.0 | Arthur Na/ | | Initial draft of SOS IPR, OWS 3, replacing |

| | | | | |
|---|---|---|---|---|
| | | Mark Priest | | Sensor Collection Service, OGC document 03-23r1 |
| 10/11/05 | 0.1.1 | Harry Niedzwiadek | Section 6, Annexes A & C | Moved optional operators to Annex C. Updated Schema in Annex A. Fixed discrepancies in Section 6 or noted inconsistencies. |
| 10/26/05 | 0.1.2 | Arthur Na/ John Davidson | Section 5.3, Sections 8,9,10. | Added XML Spy notation. Changed parameter tables for all operations to four columns. |
| 10/28/05 | 0.1.3 | Mark Priest | Sections 6, 7, 9, Annex A & D | Extensive revisions were made. Added SOS Overview (Section 6), SOS Concept of Operations (Section 7), Transaction Profile (Section 9) and Future Considerations (Annex D). Also added normative schema for transaction operators to Annex A. |
| 10/30/05 | 0.1.4 | Harry Niedzwiadek | Section 10, Annex A | Review and edit of entire document. Moved enhanced operations from Annex C to Section 10, the main body of the document. Added normative schema for these operations in Annex A. |
| 01/24/06 | 0.1.5 | Alexander C. Walkowski | All | Some missing parameters added, as well as exceptions |

The issues in this specification are captured in the following format:

> **Issue Name**: [Issue Name goes here. (Your Initials, Date)]
>
> **Issue Description:** [Issue Description.]
>
> **Resolution:** [Insert Resolution Details and History.] (Your Initials, Date)]

## vi. Changes to the OpenGIS® Abstract Specification

The OpenGIS® Abstract Specification does not require changes to accommodate the technical contents of this document.

## vii. Future work

Further definitions and refinements of the optional enhanced and transactional operations are needed, along with examples of their use once they enter common usage. See Annex C for future work issues.

## Foreword

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. Open Geospatial Consortium Inc. shall not be held responsible for identifying any or all such patent rights. However, to date, no such rights have been claimed or identified.

This version of the specification cancels and replaces all previous versions, including the Sensor Collection Service (SCS) Interoperability Program Report (IPR), Version 0.9.0, 03-023r1, and all previous versions of this document.

OGC 06-009r1 consists of the following parts, under the main body:

- Clause 1: Scope
- Clause 2: Conformance
- Clause 3: Normative references
- Clause 4: Terms and definitions
- Clause 5: Conventions
- Clause 6: Overview
- Clause 7: Concept of Operations
- Clause 8: Core Operations Profile
- Clause 9: Transaction Operations Profile
- Clause 10: Enhanced Operations Profile
- Annex A: XML Schemas (Normative)
- Annex B: SOS Instance Examples
- Annex C: Future Work: Issues to be Resolved
- Bibliography

### Related standards and specifications

This specification was developed under the OWS3 initiative. The SOS Specification is part of the OGC Sensor Web Enablement activity and is related to the following SWE draft specifications.

- Observations and Measurements, OGC Document 05-087
- Sensor Alert Service, OGC Document 05-098
- Sensor Model Language (SensorML) , OGC Document 05-086
- Sensor Planning Service, OGC Document 05-089
- SWE Architecture, OGC 05-090

- Transducer Markup Language (TML), OGC Document 06-010
- Web Notification Service, OGC Document 03-008r1

**Normative Annexes**

Annex A is normative.

**Informative Annexes**

Annexes B and C are informative.

 

## Introduction

The Open Geospatial Consortium's Sensor Web Enablement (SWE) activities, which have been executed principally through the OGC Web Services (OWS) initiatives under the Interoperability Program, is establishing the interfaces and protocols that will enable "Sensor Webs" through which applications and services will be able to access sensors of all types over networks such as the Internet and with the same standard technologies and protocols that enable the Web. These initiatives have defined, prototyped and tested several foundational components needed for a Sensor Web, namely:

1. **Observations & Measurements (O&M)** - The general models and XML encodings for sensor observations and measurements. O&M originated under OWS-1.1 and was significantly enhanced under the OWS-1.2 and OWS-3 testbed initiatives.

2. **Sensor Alert Service (SAS)** – A service by which a client can register for and receive sensor alert messages. The service supports both pre-defined and custom alerts and covers the process of alert publication, subscription, and notification. This service is being defined as part of the Sensor Alert Service Interoperability Experiment begun in 2005.

3. **Sensor Model Language (SensorML)** – The general models and XML schema for describing sensors and sensors as processes. SensorML originated under OWS-1.1, was significantly enhanced under OWS-1.2 and OWS-3 testbed initiatives and is now available as a Best Practices Paper (formerly known as a Recommendation Paper).

4. **Sensor Planning Service (SPS)** – A service by which a client can determine collection feasibility for a desired set of collection requests for one or more sensors/platforms, or a client may submit collection requests directly to these sensors/platforms. This service was defined in the OWS-1.2 testbed and enhanced in the OWS-3 testbed and is now available as a Discussion Paper.

5. **Transducer Markup Language (TML) –** General characterizations of transducers (both receivers and transmitters), their data, how that data is generated, the phenomenon being measured by or produced by transducers, transporting the data, and any and all support data (metadata) necessary for later processing and understanding of the transducer data. This work was brought to OGC in the OWS-3 testbed and is now an OGC Discussion Paper.

6. **Web Notification Service (WNS)** – A service by which a client may conduct asynchronous dialogues (message interchanges) with one or more

other services. This service is useful when many collaborating services are required to satisfy a client request, and/or when significant delays are involved is satisfying the request. This service was defined under OWS 1.2 in support of SPS operations. WNS has broad applicability in many such multi-service applications.

This document specifies Sensor Observation Service. The other SWE components are specified under separate cover.

A Sensor Observation Service provides an API for managing deployed sensors and retrieving sensor data and specifically "observation" data. Whether from in-situ sensors (e.g., water monitoring) or dynamic sensors (e.g., satellite imaging), measurements made from sensor systems contribute most of the geospatial data by volume used in geospatial systems today. The general scenario for in-situ sensors is illustrated in Figure 1, where deployed sensors (S*n*) of various types are grouped into several constellations (C*n*) that are then accessed through some service (e.g., SOS).

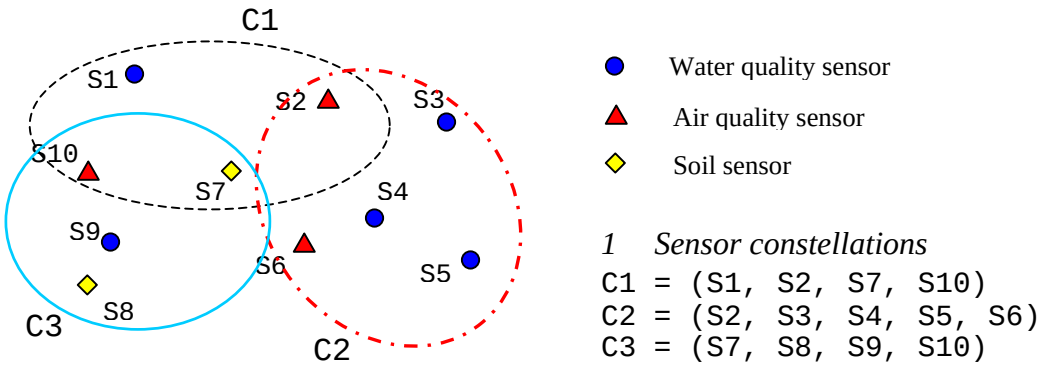


**Figure 1 General Case for In-Situ Sensors**

SOS has three mandatory "core" operations: **GetObservation, DescribeSensor**, and **GetCapabilities**. The **GetObservation** operation provides access to sensor observations and measurement data via a spatio-temporal query that can be filtered by phenomena. The **DescribeSensor** operation retrieves detailed information about the sensors making those measurements and the platforms that carry the sensors. The **GetCapabilities** operation provides the means to access SOS service metadata. Several optional, non-mandatory operations have also been defined. There are two operations to support transactions, RegisterSensor and InsertObservation, and six enhanced operations, including GetResult, GetFeatureOfInterest, GetFeatureOfInterestTime, DescribeFeatureOfInterest, DescribeObservationType, and DescribeResultModel.

Used in conjunction with other OGC specifications, the SOS provides a broad range of interoperable capability for discovering, binding to and interrogating individual sensors,

sensor platforms, or networked constellations of sensors in real-time, archived or simulated environments.

# OpenGIS® Sensor Observation Service

## 1 Scope

This OpenGIS document describes the interface specification for the Sensor Observation Service.

## 2 Conformance

Conformance with this specification shall be checked using all the relevant tests specified in Annex A (normative).

## 3 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of 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.

| |
|---|
| OpenGIS® Abstract Specification, OGC document 99-100r1.<br><br>Topic 0: Overview<br><br>Topic 11 - Metadata (Same as ISO Metadata document 19115)<br><br>Topic 12 - OGC Services Architecture |
| ISO 8601:1988(E), Data elements and interchange formats - Information interchange - Representation of dates and times. |
| Enhanced Filter Encoding, OGC document 05-093. Available to members online under Pending Documents. |
| Geographic Markup Language (GML) Implementation Specification, Version 3.1.1, OGC document 03-105r1. Available to the public online under OpenGIS® Specifications. |

| |
|---|
| Namespaces in XML. W3C Recommendation (14 January 1999). Available [Online]: <http://www.w3.org/TR/1999/REC-xml-names-19990114/> |
| Observations and Measurements, OGC document 05-087. Available to members online under Pending Documents. |
| Sensor Model Language for In-Situ and Remote Sensors, OGC document 05-086. Available to members [Online] under pending documents. |
| Sensor Planning Service (SPS), OGC document 05-089. Available to the public online under Discussion Papers. |
| Transducer Markup Language (TML), OGC document 05-085. Available to the public online under Discussion Papers. |
| XML Schema Part 1: Structures. W3C Recommendation (2 May 2001). Available [Online]: <http://www.w3.org/TR/xmlschema-1/> |
| XML Schema Part 2: Datatypes. W3C Recommendation (2 May 2001). Available [Online]: <http://www.w3.org/TR/xmlschema-2/> |
| OpenGIS® Web Services Common Specification, OGC document 05-008c1. Available to the public online under OpenGIS® Specifications. |

## 4 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

### 4.1 Measurement

An instance of a procedure to estimate the value of a natural phenomenon, typically involving an instrument or sensor. This is implemented as a dynamic feature type, which has a property containing the result of the measurement. The measurement feature also has a location, time, and reference to the method used to determine the value. A measurement feature effectively binds an Observed Value to a location and to a method or instrument.

### 4.2 Observation

An observation is an event with a result which has a value describing some phenomenon. The observation is modelled as a Feature within the context of the ISO/OGC Feature Model. An observation feature binds the result to the feature of interest, upon which it was made. An observation uses a procedure to determine the value, which may involve a sensor or observer, analytical procedure, simulation or other numerical process. The observation pattern and feature is primarily useful for capturing metadata associated with data capture.

### 4.3 Observation Offering

An observation offering is a logical grouping of observations offered by a service that are related in some way. The parameters that constrain the offering should be defined in such as way that the offering is “dense” in the sense that requests for observations that are within the specified parameters should be unlikely to result in an empty set.

### 4.4 Phenomenon

A phenomenon to that can be observed and measured, such as temperature, gravity, chemical concentration, orientation, number-of-individuals. The equivalent term is **measurand** when the values are determined by measurement.

### 4.5 Result

A value describing a natural phenomenon, which may use one of a variety of scales including nominal, ordinal, ratio and interval. The term is used regardless of whether the value is due to an instrumental observation, a subjective assignment or some other method of estimation or assignment.

### 4.6 Sensor

An entity capable of observing a phenomenon and returning an observed value. A sensor can be an instrument or a living organism (e.g. a person), but herein we concern ourselves primarily with modelling instruments, not people.

### 4.7 (Sensor) Platform

An entity to which can be attached sensors or other platforms. A platform has an associated local coordinate frame that can be referenced to an external coordinate frame and to which the frames of attached sensors and platforms can be referenced.

## 5 Conventions

### 5.1 Symbols (and abbreviated terms)

| | |
|---|---|
| API | Application Program Interface |
| DCP | Distributed Computing Platform |
| GML | Geographic Markup Language |
| ISO | International Organization for Standardization |
| OGC | Open Geospatial Consortium |
| OWS | OGC Web Services |
| O&M | Observations and Measurements |
| SensorML | Sensor Model Language |

| | |
|---|---|
| SOS | Sensor Observation Service [formerly Sensor Collection Service, SCS] |
| SPS | Sensor Planning Service |
| SWE | Sensor Web Enablement |
| TML | Transducer Markup Language |
| UML | Unified Modeling Language |
| WNS | Web Notification Service |
| XML | eXtended Markup Language |

### 5.2 UML Notation

Most diagrams that appear in this specification are presented using the Unified Modeling Language (UML) static structure diagram, as described in Subclause 5.2 of [OGC 05-008].

### 5.3 XML schema notation

Most diagrams that appear in this specification are presented using an XML schema notation defined by the XMLSpy[1] product and described in this subclause. XML schema diagrams are for informative use only though they shall reflect the accompanied UML and schema perfectly.

#### 5.3.1 Element

A named rectangle representing the most basic part of the XML Schema notation. Each represents an XML “Element” token. Each Element symbol can be elaborated with extra information as shown in the examples below.


**Figure 2. Simple XML Spy element**

This is a mandatory simple element. Note the upper left corner of the rectangle indicates that data is contained in this element.

[1] XML Spy: http://www.altova.com

### 5.3.2 Optional Element

Optional (non mandatory) elements are specified with dashed lines used to frame the rectangle.


**Figure 3. Optional XML Spy element**

### 5.3.3 Recurring Element

This element (and its child elements if it has any) can occur multiple times.

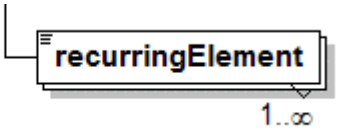


**Figure 4. Recurring XML Spy element**

This example shows a recurring element that must occur at least once but can occur an unlimited number of times. The upper bound here is shown with the infinity symbol.

### 5.3.4 Sequence Connector

The connection box, called a sequence indicator, indicates that the “SequenceElement” data is made up of three elements. In this example, the first two elements are mandatory and the third element is optional.

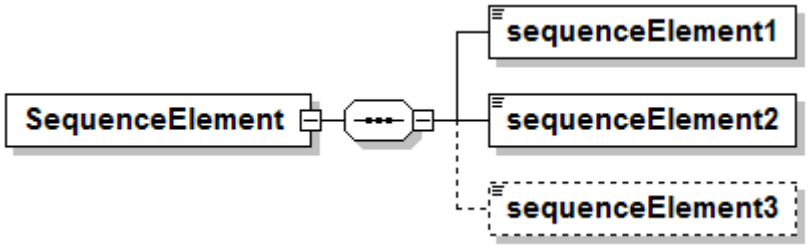


**Figure 5. Sequence Connector in XML Spy**

### 5.3.5 Choice Connector

The connection box here is a “choice” indicator, indicating that there is always going to be exactly one of the child elements listed on the right.

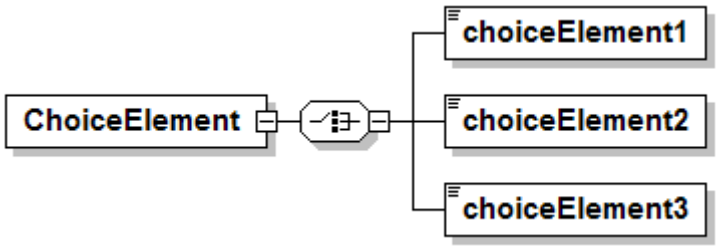


**Figure 6. Choice Connector in XML Spy**

### 5.3.6 Definition with Complex Type

This diagram illustrates the use of a complex type (i.e., “ex:AbstractElementType”) for defining an XML element (e.g., “AbstractElement”).

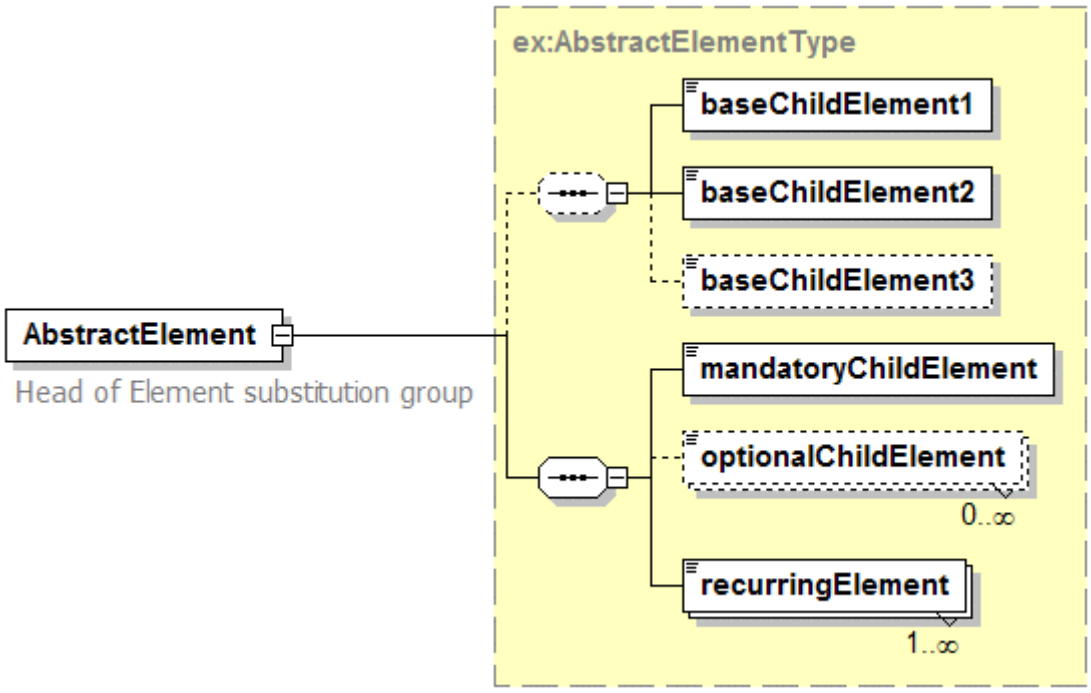


**Figure 7. Definition with Complex Element in XML Spy**

### 5.3.7 Complex Type

This diagram illustrates the definition of a complex type (i.e., “AbstractElementType”), extending another complex type (i.e.,“ex:BaseElementType” ) with three additional elements. Complex types can be reused to specify that different elements are of the same type.

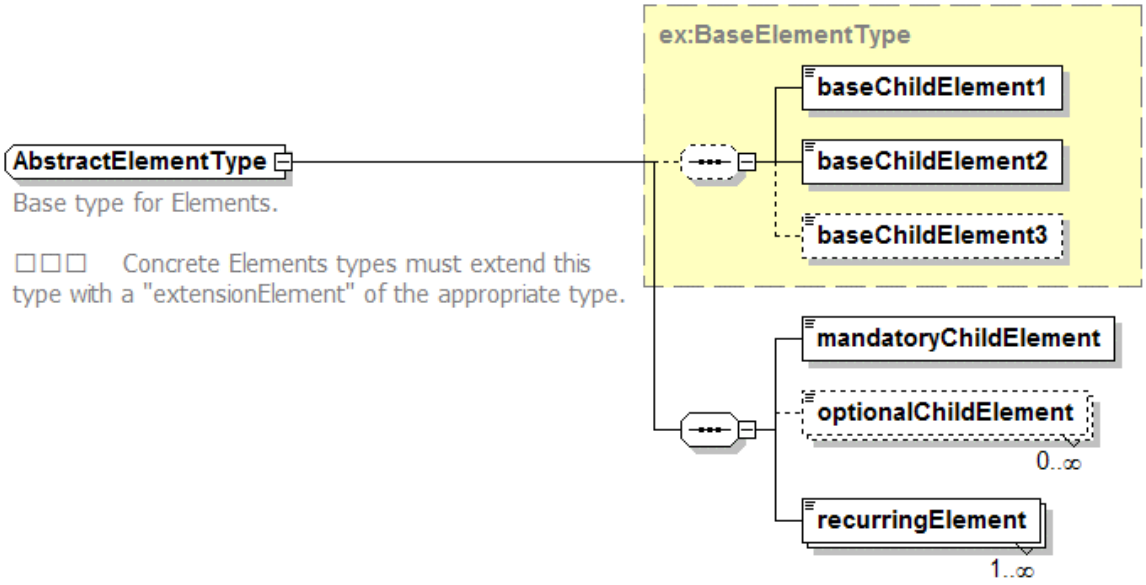


**Figure 8. Complex Type in XML Spy**

## 6 SOS Overview

The goal of SOS is to provide access to observations from sensors and sensor systems in a standard way that is consistent for all sensor systems including remote, in-situ, fixed and mobile sensors. This is a challenging task because the users of sensor data have historically been divided into those who primarily deal with in-situ sensors and those who primarily deal with remote sensors. The terminology, perspective, and expectations of these two broad groups are different. SOS leverages the Observation and Measurements (O&M) specification for modelling sensor observations and the TransducerML and SensorML specifications for modelling sensors and sensor systems.

### 6.1 General Approach

The approach that has been taken in the development of SOS, and the SWE specifications on which it depends, is to carefully model sensors, sensor systems, and observations in such a way that the model covers all varieties of sensors and supports the requirements of all users of sensor data. This is in contrast to the approach that was taken with the Web Feature Service (WFS). WFS provides a generic definition of a geographic feature that is flexible enough to encompass any real-world entity. The WFS uses GML application schemas to define the specific properties of each type of feature. With this approach, interoperability requires organizations to agree on domain-specific GML application

schemas. Clients that access a WFS in a particular domain must have a-priori knowledge of the application schemas used in that domain. The SOS approach defines a common model for all sensors, sensor systems and their observations. This model is not domain-specific and can be used without a-priori knowledge of domain-specific application schemas.

### 6.2 Observation Offerings

An SOS organizes collections of related sensor system observations into Observation Offerings. The concept of an Observation Offering is equivalent to that of a sensor constellation discussed earlier in this document. An Observation Offering is also analogous to a "layer" in Web Map Service because each offering is intended to be a non-overlapping group of related observations. Each Observation Offering is constrained by a number of parameters including the following:

- Specific sensor systems that report the observations,
- Time period(s) for which observations may be requested (supports historical data),
- Phenomena that are being sensed,
- Geographical region that contains the sensors, and
- Geographical region that is the subject of the sensor observations (may differ from the sensor region for remote sensors)

An SOS service instance should factor offerings such that a GetObservation request, with supported query parameters, is very likely to obtain one or more results. The likelihood of getting an empty response for a valid query should be minimized.

For example, suppose an SOS instance advertises two sensors – one that reports wind speed and the other that reports air temperature. If these sensors are attached to the same weather station, they should probably be included in the same offering. That is because a GetObservation request for weather data for a given area that includes the weather station to which the sensors are attached and for time periods that the weather station is reporting will almost always have data for both sensors. If the client asks for wind speed only, air temperature only, or both, the time and location of the results should be the same.

On the other hand, if the two sensors are located on weather stations that are far apart in space or which report during non-overlapping time periods, then they should probably be factored into two distinct offerings. If they were put into the same offering, then the combinatorial space of that single offering would be relatively sparse. In other words, it would frequently be the case that a GetObservation request asking for temperature might return a result where one for wind speed might not. The client might have to make quite a few GetObservation requests on a "sparse" offering before finding data, which is clearly undesirable.

 

### 6.3 Identifying Phenomena and Units of Measure

A critical issue for interoperability is defining a standard way to refer to phenomena that are measured by sensors and the units of measure for those phenomena. This is important for both discovery of SOS service instances in a catalog and for allowing a client to request specific phenomena from a given service instance. Because SOS is intended to be used in a very wide variety of applications in a large number of application domains it is not feasible to construct a single comprehensive and authoritative dictionary for phenomena and units of measure. Specific phenomena and units of measure are generally specific to a given domain and the mechanism used to reference them must support a decentralized approach.

One goal of SOS, and SWE in general, is to specify a standard mechanism for consistently identifying phenomena and units of measure that will scale (up or down) to handle any number of definitions in any application domain. The range of different phenomena and units of measure is quite large, unknown a-priori, and possibly unknowable. Therefore, the mechanism for identifying phenomena and units of measure must be flexible enough to handle this reality.

The solution for identifying phenomena and units of measure is to use GML dictionaries. Services and clients use URIs to refer to specific entries in a particular GML dictionary. A URI can take the form of a URN in cases where the reference is to phenomena or units of measure that are defined by an OGC dictionary or a dictionary hosted by another well-known organization. A URL may be used in the case that a service provides its own dictionary or uses a third-party dictionary that is not well-known. URLs must include a fragment identifier that is used to locate the gml:id of a phenomena within the dictionary that resides at the location indicated by the URL. URN values must follow the format specified in OGC document 05-010 - URNs of Definitions in OGC Namespace.

Entities like units of measure and phenomena are not physical objects in the real world. They are concepts and can only be defined by convention or by their relationship to other intangible concepts. Phenomena or units of measure that are defined in reference to other types can be considered to be derived or constrained entities and can be derived from more basic entities. Within the SWE initiative this is done slightly differently for units of measure and phenomena.

The SWE initiative relies on the existing GML support for defining units of measure as expressed in the GML schema (see http://schemas.opengis.net/gml/3.1.1/base/units.xsd). This schema provides a way to define base units such as the metric units like meter, kilogram, degree Kelvin, etc. These base units are defined using GML metadata tags which consist of a natural language explanation of the units. Derived units can then be constructed from the base units to any level of complexity. The mechanism for deriving units is well-defined and can be done automatically using software as long as the base units are commonly understood.

A GML conformant schema for defining base phenomena and derived phenomena was developed as part of the SWE initiative. This schema allows for the definition of base phenomena in much the same that base units are defined in GML. Derived phenomena

can be developed as a constraint on an existing phenomenon, an aggregation of existing phenomena, or as a composite of existing phenomena.

### 6.4 Filtering and Filter Expressions

The SOS GetObservation operation includes an ad-hoc query capability that allows a client to filter observations by time, space, sensor, and phenomena. It is impossible to know ahead of time all of the possible queries that a client might want to perform, so the SOS design leverages the OGC Filter Encoding specification to support ad-hoc queries. However, the SOS uses strongly-typed filter expressions rather than the more open-ended filter expressions defined in the current Filter Encoding specification in order to reduce complexity. This was done to reduce the burden on those implementing the SOS service. The GetObservation operation factors the spatial, temporal, and property-based query components into separate query elements. Instead of allowing a general filter expression, the filter is built up using a set of specific elements which have very clear semantics and restricted scope. This is consistent with the approach described above where observations are modelled using strongly typed properties.

The capabilities document for SOS includes a FilterCapabilities section that is used to indicate what types of query parameters are supported by the service. The filter capabilities element has been broken up into spatial capabilities, scalar capabilities, ID capabilities, and temporal capabilities. The factored design of the filter capabilities reflects the factored approach to query expressions in SOS.

This concept of defining strongly-typed filter expressions has been proposed as a change request to the Filter Encoding specification (CR 05-093 - Enhanced Filter Encoding) because we believe that it will be useful in other services as well.

## 7 SOS Concept of Operations

### 7.1 Overview

This Concept of Operations section is intended to outline a typical operational context in which an SOS is expected to be used. The intention for including this section is to provide a better understanding of the service and to help explain the operational context that was envisioned when the service was designed. This is only one view of the operational context and the SOS can be used in ways other than those described here. We expect that SOS will be used in ways not anticipated by the contributors to this specification which will help the specification to become more robust and useful than it is now. The Concept of Operations is not meant to constrain how an SOS can be used in an operational setting.

 

The Concept of Operations is divided into two sections. In the first section the perspective is that of a consumer of sensor observations. This is expected to be the most common perspective used with SOS. In the second section the perspective is that of a producer of sensor observations which is using the services of an SOS that supports the transactional SOS profile. The transactional profile allows sensor systems to be dynamically registered and allows sensor observations to be published to the service.

### 7.2 Concept of Operations for Sensor Data Consumer

A sensor data consumer is interested in obtaining sensor observations from one or more sensors. At a high level the consumer might approach this problem from either a sensor-centric or an observation-centric point of view. A sensor-centric point of view would be used if the consumer was already aware of the existence of particular sensors and wanted to find observations for those sensors. An observation-centric point of view would be used if the consumer wanted to see sensor data from a particular geographic area and that capture particular phenomena but is not aware of any particular sensors a-priori.

In either case the consumer would perform service discovery using a catalog service in order to find SOS service instances that can provide the desired sensor observations. After consulting the catalog the consumer could directly obtain observations from services or could perform discovery at the service level or get sensor metadata before obtaining sensor observations. Service-level discovery involves invoking the GetCapabilities operation to return information about the offerings that are available from each service. Detailed sensor metadata can be obtained after extracting the sensor system identifiers out of each observation offering by invoking the DescribeSensor operation.

Figure 9 shows the sensor data consumer in an operational context with OGC CS-W catalogs for services discovery and SOS service instances with observation offerings and observations. A service instance may talk directly to sensors or may be a proxy for other services as indicated in the diagram. Services can be organized into complex topologies using aggregation and other techniques but this is transparent to the data consumer. The consumer only needs to deal with registries and service interfaces.

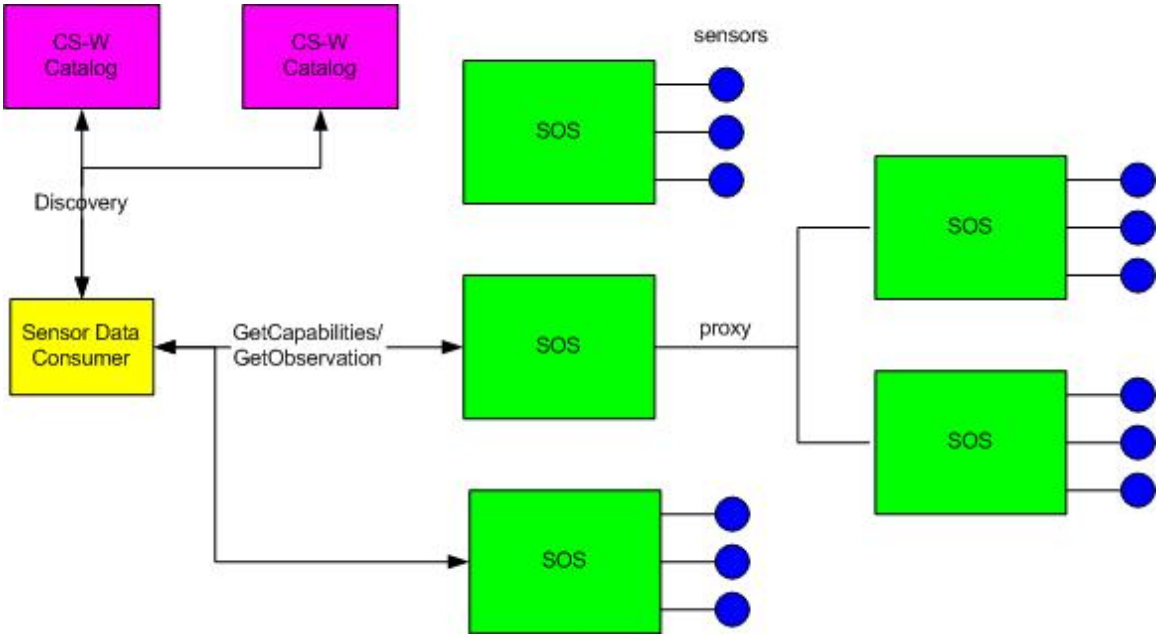


**Figure 9 Sensor Data Consumer in Operational Context**

Figure 10 is a flow chart that shows the mandatory and optional steps that can be followed by the sensor data consumer.

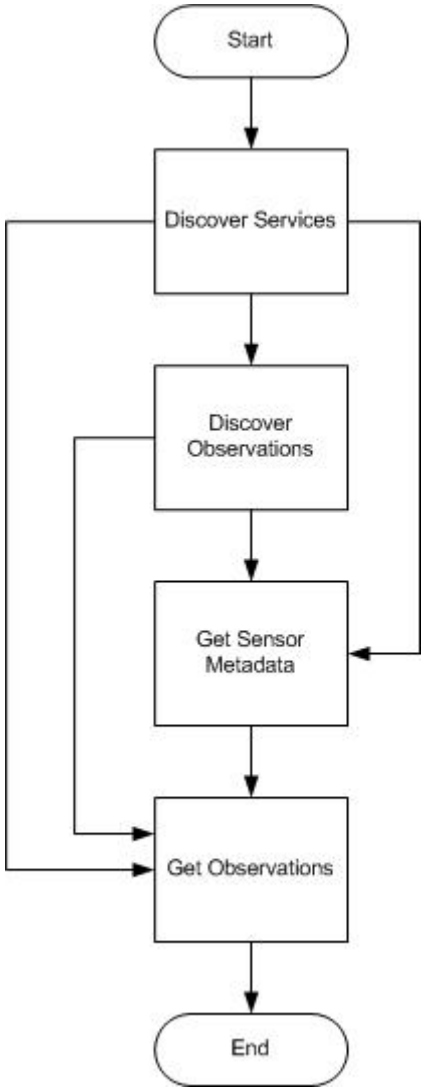


**Figure 10 Sensor Data Consumer Flow Chart**

Figure 11 is a sequence diagram that shows a sensor data consumer discovering two SOS instances from a CS-W catalog by using the GetRecords operation. The consumer then performs service-level discovery on each service instance by requesting the capabilities document and inspecting the observation offerings. The consumer invokes the DescribeSensor operation to retrieve detailed sensor metadata in SensorML or TML for sensors advertised in the observation offerings of the two services. Finally, the consumer calls the GetObservaiotn operation to actually retrieve the observations from both service instances.

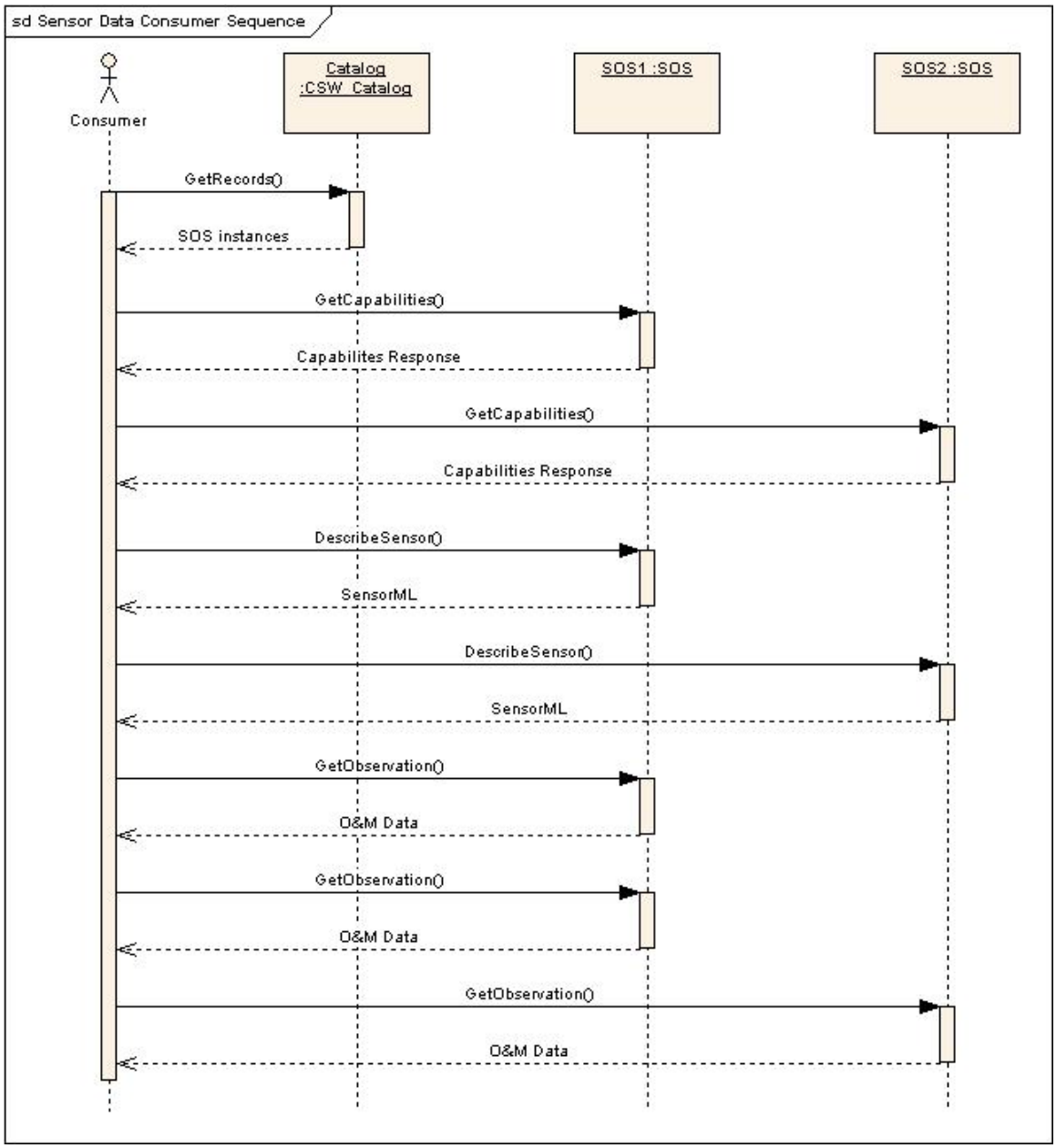


**Figure 11 Sensor Data Consumer Sequence Diagram**

### 7.2.1 Service Discovery

Service discovery is done by using one or more OGC Catalog Service (CS-W) instances. The CS-W specification is out of scope of this document but allows entities to be registered so that they can later be discovered by data consumers. It is also expected that catalogs may periodically harvest information from the capabilities documents of each service they know about in order to keep that service's attributes up to date. The catalogs in the OWS-3 test bed allow the following information to be used in catalog searches:

- Time period of observations
- Phenomena captured by observations
- Spatial extent of observations
- GML names used in observation offerings
- GML descriptions used in observation offerings

In the future, CS-W catalogs might be expanded to allow discovery based on other parameters. As more SWE architectures are deployed over time it will become more evident what service parameters are useful for discovery.

#### 7.2.2 Observation Discovery

Observation discovery happens at the service level. The service capabilities document includes detailed information about all of the offerings that are available from a service. This can be used to further target GetObservation requests. This is an optional step and can be skipped in many cases if the catalog response has sufficient information for the purpose of the consumer.

#### 7.2.3 Get Sensor Metadata

Sensor metadata can be retrieved for any sensor that is advertised in an observation offering using the DescribeSensor operation. This will return a SensorML or TML document with detailed information about the sensor. This could be used to filter out sensors that do not have robust error detection and correction or are not accurate enough for the purposes of the data consumer. This is an optional step that can be skipped in many cases.

#### 7.2.4 Get Sensor Observations

Sensor observations are obtained using the GetObservation operation. This operation supports a query mechanism described below that supports subsetting the observations that will be returned from a call to GetObservation. This should not be used for discovery since discovery is already handled through the concept of the observation offering and the catalog. GetObservation allows the client to filter a large dataset to get only the specific observations that are of interest.

### 7.3 Concept of Operations for Sensor Data Publisher

The SOS includes an optional transactional profile with operations for inserting new sensors and sensor observations into an SOS. SOS instances that use the transactional profile are expected to be dependent on intermediary external services that talk with sensors rather than talking directly to sensors themselves. The intent is to support the development of commercial SOS products that can be dropped into any sensor network that includes sensor data producers that have a rudimentary SOS client capability. The assumption is that sensor data producers are software entities running on resource constrained platforms that are attached to sensors – either physically or through a wireless connection of some kind. The producers have just enough processing power to construct simple XML insert documents that can be published to the transactional SOS. The transactional SOS is assumed to be a software entity running on a capable server in a data center which can easily handle a large number of simultaneous connections from data producers. Data consumers would request data from the transactional SOS using the core profile and would not need to communicate with the actual data producers. Figure 12 shows sensor data producers in the operational context described above. The catalog still appears in the diagram but there is only one and it is assumed to be co-located with the SOS service. The catalog would be used for discovery of the SOS instance if the data producers were capable enough to support a catalog client application. In some cases the sensor data publishers will need to be configured with the location of the SOS when they are deployed. The diagram is meant to represent a facility that is under the control of a single organization. Scalability would be achieved through aggregating a number of sensor data producers on the sensor side and by aggregating a number of SOS instances on the data center side for providing data to consumers.

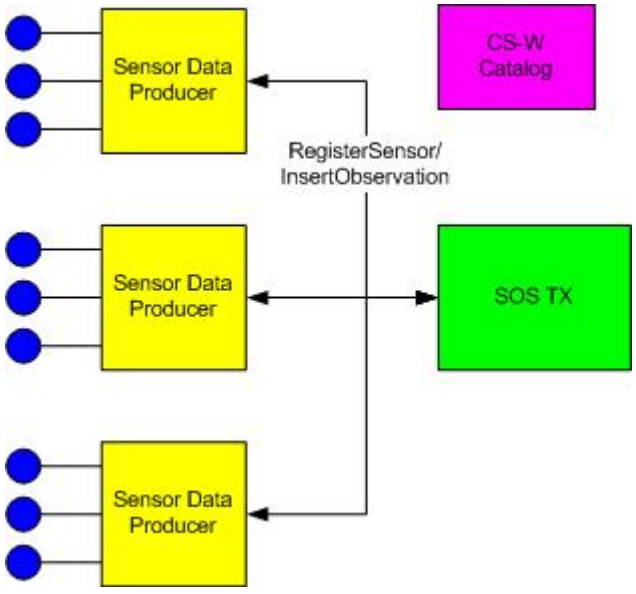


**Figure 12 Operational Context for Sensor Data Producers**

Figure 13 is a flow chart that shows the steps that can be followed by the sensor data producer. All of the steps in the flow are optional.

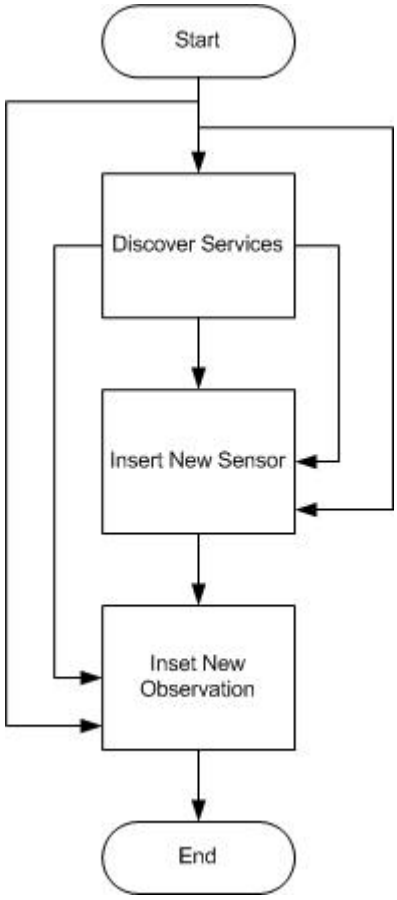


**Figure 13 Sensor Data Producer Flow Chart**

Figure 14 is a sequence diagram that shows how two sensor data producers interact with an SOS instance that supports the transactional profile. The first producer discovers the SOS using a CS-W catalog. The second producer has been configured to know the location of the SOS. When the producers determine that they are communicating with new sensors they register those sensors with the SOS. The producers publish sensor observations from all sensors in their control to the SOS. The diagram also shows a sensor data consumer. Consumers can request observations from the SOS at any time using the GetObservation operation. The consumer does not need to be aware of the existence of the producers and vice versa.

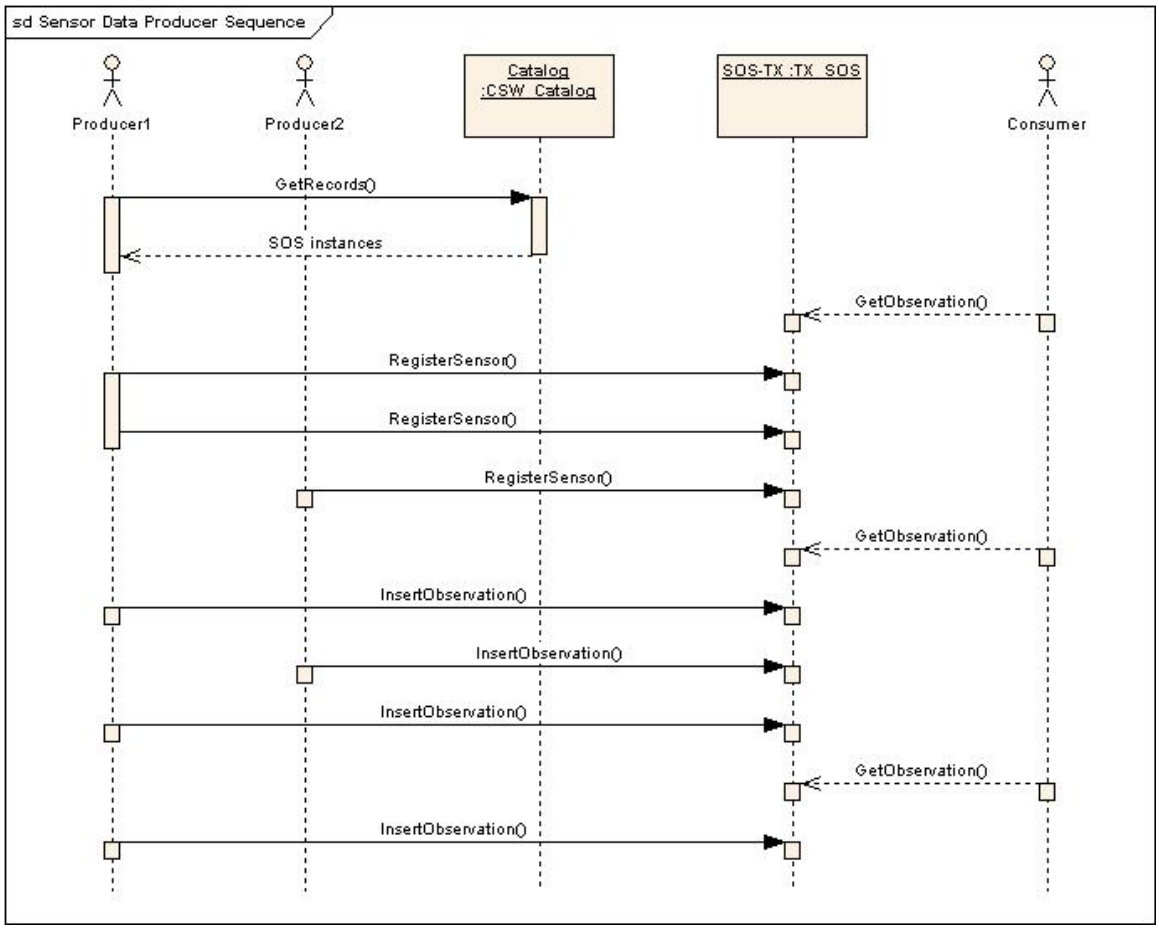


**Figure 14 Sensor Data Producer Sequence Diagram**

### 7.3.1 Service Discovery

Service discovery is done by using one or more OGC Catalog Service (CS-W) instances as described in section 7.2.1 above. We anticipate that sensor data publishers are likely to be more tightly bound to SOS instances than consumers and are less likely to incorporate a catalog client. Therefore, this step will probably be skipped for most producers and the producers will instead be manually configured with the location of the SOS instance when they are deployed.

### 7.3.2 Sensor Registration

A producer can only publish observations to an SOS if that SOS already knows about the sensor that generated the observations. The producer can look at the capabilities document of the SOS to determine whether the sensor is already known to the SOS. If not then the producer must register the sensor using the RegisterSensor operation.

### 7.3.3 Publishing of Observations

Once the producer has registered a sensor with an SOS instance it can begin publishing observations for that sensor. The SOS has the responsibility of packaging the observations into offerings and providing them to sensor data consumers.

## 8 Core Operations Profile

Operations defined for a Sensor Observation Service fall into four profiles: *core*, *enhanced*, *transactional* and *entire*. The mandatory *core* operations are **GetCapabilities**, **DescribeSensor** and **GetObservation**. There are also a number of *transactional* and *enhanced* operations which are described in Sections 9 and 10 respectively. The *entire* profile of SOS implements all of these operations (*core*, *enhanced* and *transactional*). The implementation and use of SOS operations is specified in accordance with the HyperText Transfer Protocol (HTTP) Distributed Computing Platform (DCP). Future versions may apply to other DCPs.

### 8.1 GetCapabilities (mandatory)

#### 8.1.1 Introduction

This operation allows clients to retrieve service metadata about a specific service instance. In this XML Schema encoding, no "request" parameter is included, since the element name specifies the specific operation.

#### 8.1.2 Request

The GetCapabilities operation request shall be as specified in Subclauses 7.2 and 7.3 of [OGC 05-008]. The value of the “service” parameter shall be “SOS”. The allowed set of service metadata (or Capabilities) XML document section names and meanings shall be as specified in Tables 3 and 7 of [OGC 05-008], with the additions listed in Table 1 below.

**Table 1 — Additional Section name values and meanings**

| Section name | Meaning |
|---|---|
| Contents | Return the contents section in service metadata document that contains information about the observation offerings. |
| Filter_Capabilities | Return the filter capabilties section in service metadata document that contains information about the supported filters. |

The “Multiplicity and use” column in Table 1 of [OGC 05-008] specifies the optionality of each listed parameter in the GetCapabilities operation request. Annex B provides an example of a SOS GetCapabilities document.

### 8.1.3 Response

The response to a GetCapabilities operation is an XML encoded document. This document provides clients with service metadata about a specific service instance, including metadata about the tightly-coupled data served. If the server does not implement the updateSequence parameter, the server shall always return the complete Capabilities document, without the updateSequence parameter. When the server implements the updateSequence parameter and the GetCapabilities operation request included the updateSequence parameter with the current value, the server shall return this element with only the "version" and "updateSequence" attributes. Otherwise, all optional elements shall be included or not depending on the actual value of the Sections parameter in the GetCapabilities operation request.

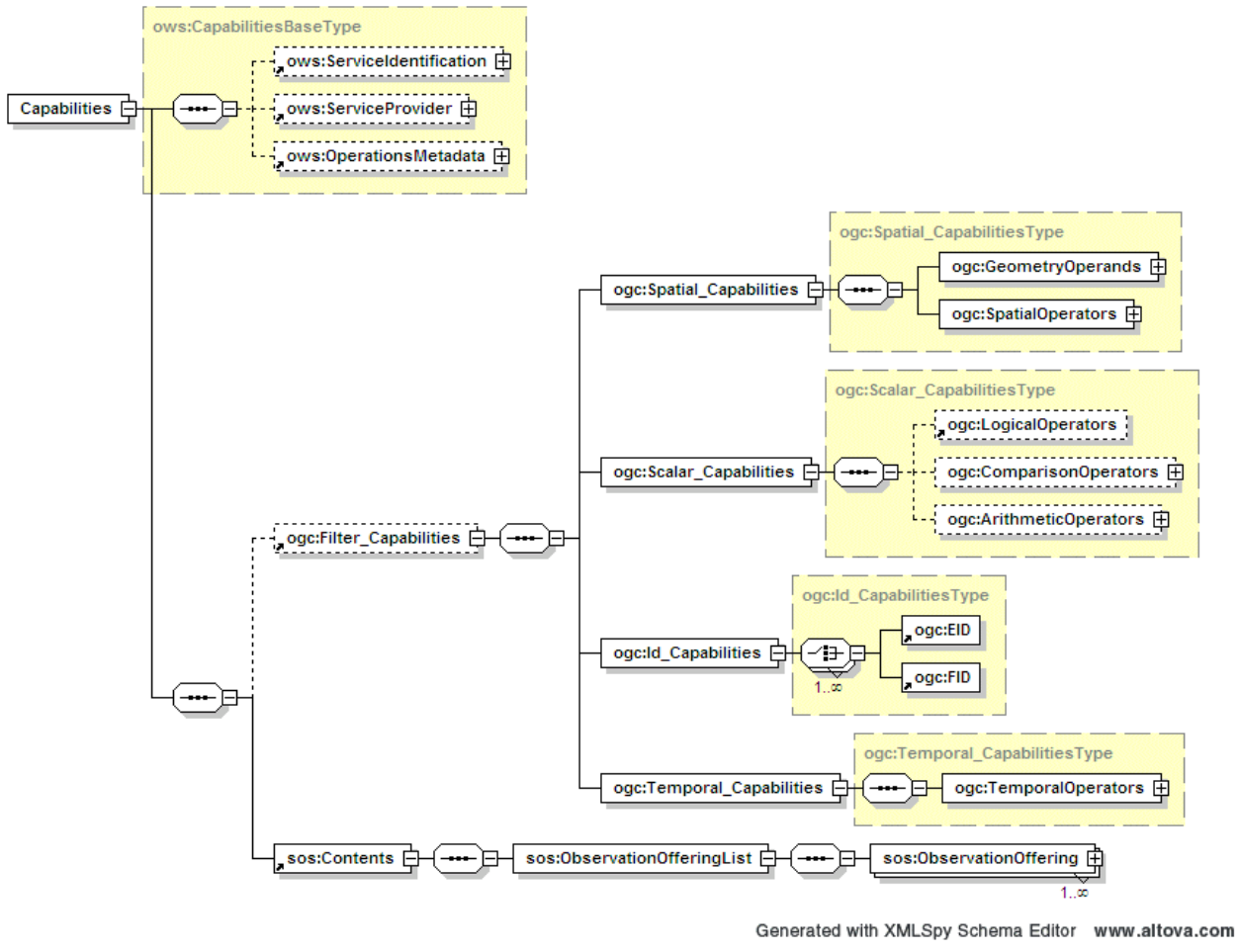


**Figure 15. GetCapabilities Response**

The portions of the GetCapabilities response document that are defined in the OWS Common specification have not been modified for SOS. The subsections that follow contain only the sections of the response that are in the service-specific SOS contents section.

#### 8.1.3.1 FilterCapabilities Section

The FilterCapabilities section is used to indicate what types of query parameters are supported by the service. These capabilities refer to the parameters of the GetObservation operation which is the only operation that includes OGC filter-like expressions.

#### 8.1.3.2 Observation Offering Section

The concept of an Observation Offering is discussed in section 6.2 above. It is a way to organize sets of sensor observation groupings that can be queried using the GetObservation operation. Each Observation Offering is constrained by a number of parameters described in the table below.

The Observation Offering element derives from the GML abstract feature type so it includes a GML Envelope that describes the geographic footprint of the offering. It also includes a gml:id and standard GML object metadata elements.

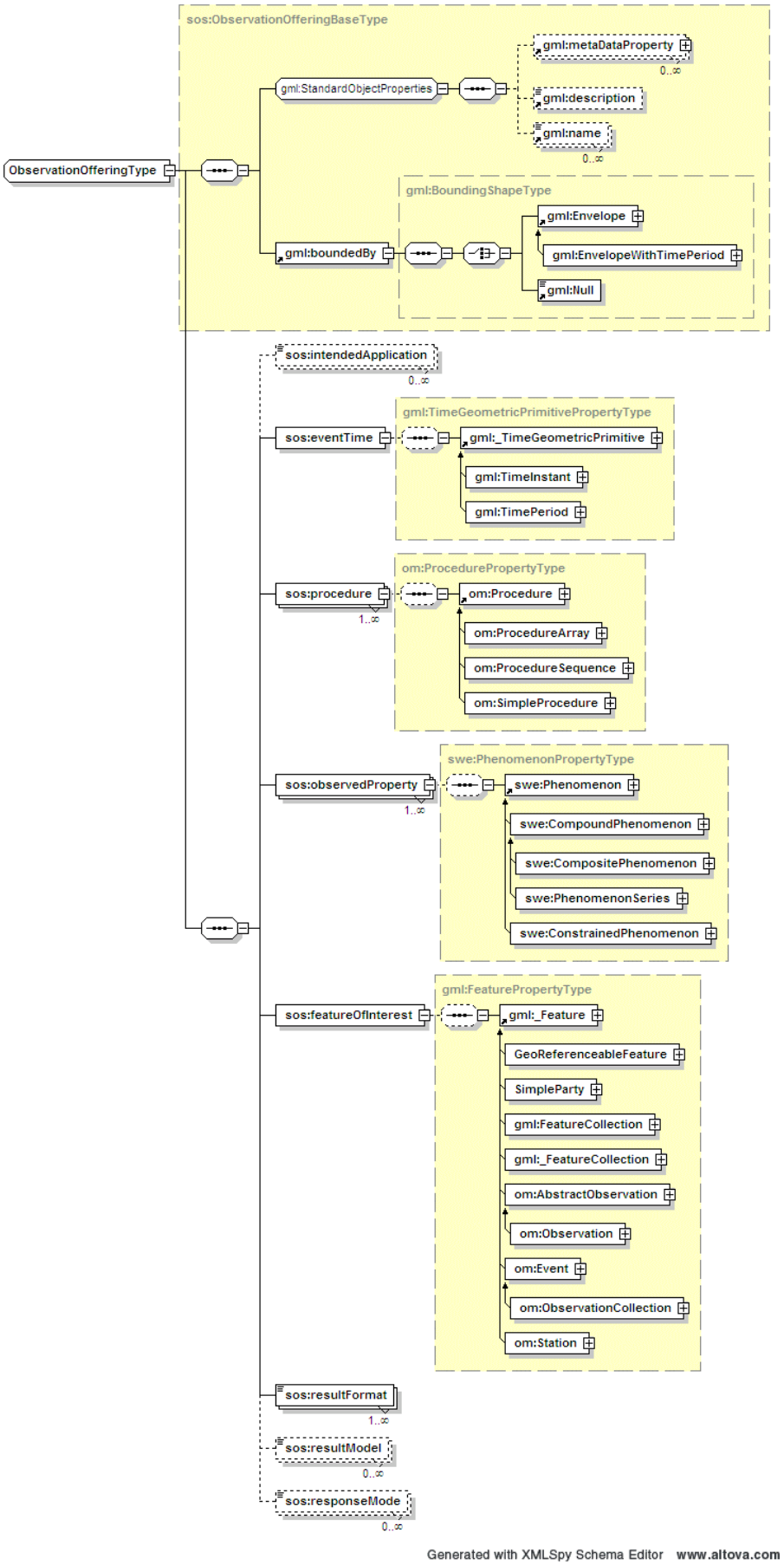

**Figure 16 Observation Offering**

**Table 2 Observation Offering Constraints**

| Name [a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| intendedApplication | The intended category of use for this offering such as homeland security or natural resource planning | Token | Optional |
| eventTime | A series of time periods for which observations can be obtained. This supports the advertisement of historical as well as real-time observations. | gml:TimeGeometricPrimitivePropertyType | One (mandatory) |
| procedure | This is a reference to one or more sensor systems that supply observations in this offering. The DescribeSensor operation can be called to provide a SensorML or TML description for each system. | ProcedurePropertyType | One or more (mandatory) |
| observedProperty | These are the observables/phenomena that can be requested in this offering. | swe:PhenomenonPropertyType | One or more (mandatory) |
| featureOfInterest | This is a single feature or a collection of features that represents the object on which the sensor systems are making observations. In the case of an in-situ sensor this may be a station to which the sensor is attached or the atmosphere directly surrounding the sensor. For remote sensors this may be the area or volume that is being sensed, which is not co-located with the sensor. In general, different GML feature types will be used here depending on the organization that is hosting the SOS. However, all such features are expected to include the optional GML boundedBy element with a GML envelope if the location of the feature is known. This GML Envelope is periodically harvested by OGC service registries. | gml:FeaturePropertyType | One (mandatory) |
| resultFormat | MIME type of the data that will be returned as the result of a GetObservation request. This is usually text/xml; subtype="om/1.0.30". In the case that data is delivered out of band it might be text/xml;subtype="tml/2.0" for TML or some other MIME type. | Ows:MimeType | One or more (mandatory) |
| resultModel | This indicates the schema type of the result element that will be returned from a call to GetObservation for this offering. The base O&M Observation type is defined as anyType so this element is necessary to show the actual types that can be returned for this offering. | Character String type | Optional |
| responseMode | Indicates what modes of response are supported for this offering. The value of "resultTemplate" is used to retrieve an observation template that will later be used in calls to GetResult. The other options allow results to appear inline in a resultTag (inline), external to the observation element (out-of-band)or as a MIME attachment (attached). | Character String type | Optional |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

#### 8.1.4 Exceptions

When a SOS server encounters an error while performing a GetCapabilities operation, it shall return an exception report message as specified in Clause 8 of [OGC 05-008]. The allowed exception codes shall include those listed in Table 5 of Subclause 7.4.1 of [OGC 05-008], if the updateSequence parameter is implemented by the server.

#### 8.1.5 Examples

##### 8.1.5.1 Example 1

Annex B has an example of a GetCapabilities response document.

### 8.2 DescribeSensor (mandatory)

#### 8.2.1 Introduction

One method of obtaining metadata that describes the characteristics of a sensor or sensor constellation is to retrieve it from a catalog. However, detail provided in a catalog might only contain high-level information about the types of observables, locations, contact information, etc. Due to the possibly voluminous amounts of sensor related metadata, the SOS specifies some describe operations that support the highest level of detail about the platforms and sensors associated with an SOS. This operation is designed to request detailed sensor metadata.

Likewise, the response to a GetCapabilities request could also provide a list of sensors associated with an SOS. Sensors are devices for the measurement of physical quantities. Both Sensor Model Language (SensorML) and Transducer Markup Language (TML) provide a detailed definition of a sensor for In-situ and Remote Sensors specification. For the purpose of this discussion, there are two main kinds, in-situ and remote-dynamic. The DescribeSensor operation is useful for obtaining detailed information of sensor characteristics encoded in either SensorML or TML. The sensor characteristics can include lists and definitions of observables supported by the sensor. The SensorML and TML specifications detail the response model for the DescribeSensor operation.

#### 8.2.2 Request

Figure 17 illustrates the SOS DescribeSensor operation.

 

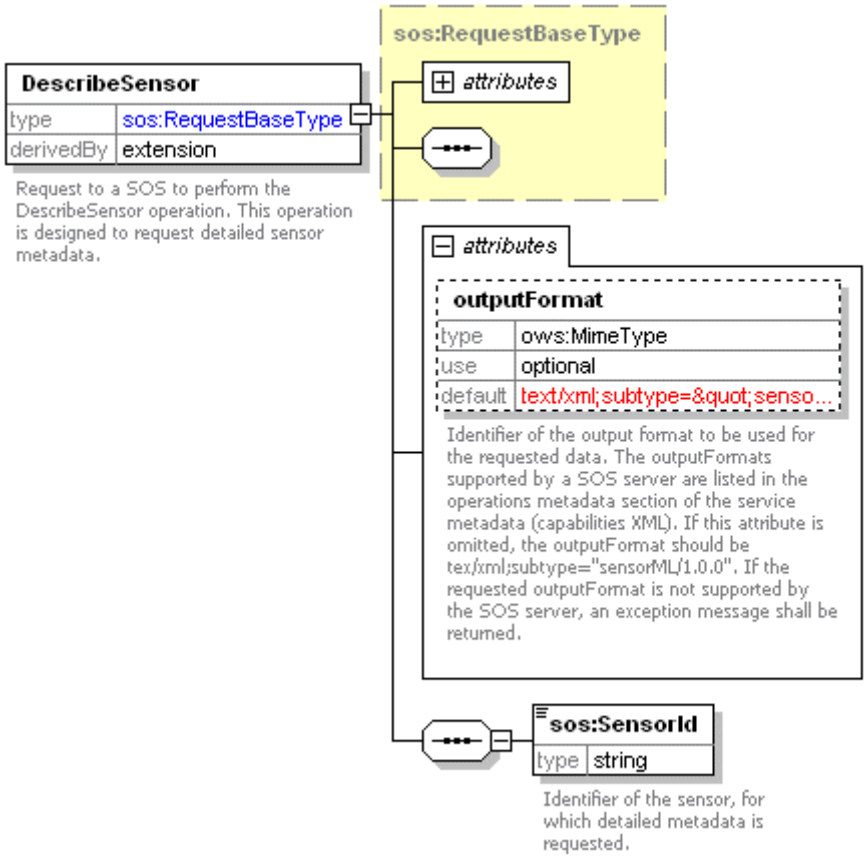


**Figure 17 DescribeSensor Request**

**Table 3 Attributes of DescribeSensor Request**

| Name [a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| outputFormat | The outputFormat attribute specifies the desired output format of the DescribeSensor operation. This can be a MimeType or QName for example. This is text/xml;subtype="sensorML/1.0.0" for SensorML or text/xml;subtype="TML/2.0" for TML. | Character String type | Optional |
| SensorId | The sensorId parameter specifies the sensor for which the description is to be returned. | Character String type | One (mandatory) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., “WMS”, “SOS”) | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

### 8.2.3 Response

A SensorML or TML document describing the sensor system.

### 8.2.4 Exceptions

When a SOS server encounters an error while performing a DescribeSensor operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 8.2.5 Examples

#### 8.2.5.1 Example1

This example illustrates requesting information about a specific sensor instance. The request:

```
<DescribeSensor version="0.0.31" service="SOS"
outputFormat="text/xml;subtype=&amp;quot;sensorML/1.0.0&amp;quot;"
xmlns="http://www.opengeospatial.net/sos">
<SensorId>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-1</SensorId>
</DescribeSensor>
```

The response:

```
<sml:Sensor xmlns:swe="http://www.opengis.net/swe"
xmlns:xlink="http://www.w3.org/1999/xlink" version="1.0"
xmlns:sml="http://www.opengis.net/sensorML"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" id="system.acada-
sensor-1">
    <sml:identification>
     <sml:IdentifierList>
        <sml:identifier name="manufacturer">
          <sml:Term qualifier="urn:ogc:identifier:manufacturer">ACME
Corporation</sml:Term>
        </sml:identifier>
      </sml:IdentifierList>
    </sml:identification>
    <sml:inputs>
      <sml:InputList>
        <sml:input name="electricalsignal">
          <swe:Quantity definition="urn:ogc:phenomenon:frequency"
uom="urn:ogc:unit:hertz"/>
        </sml:input>
      </sml:InputList>
    </sml:inputs>
    <sml:outputs>
      <sml:OutputList>
        <sml:output name="Concentration">
          <swe:Quantity id="HD-conc"
definition="urn:acada:definition:phenomenon:chemicalpresence:HD"
uom="urn:ogc:unit:part"/>
        </sml:output>
      </sml:OutputList>
    </sml:outputs>
```

```
    <sml:referenceFrame>
      <sml:LocalTimeCRS id="SENSOR_TIME">
        <sml:srsName>GMT</sml:srsName>
        <sml:usesCS xlink:href="urn:ogc:crs:GMT"/>
      </sml:LocalTimeCRS>
    </sml:referenceFrame>
</sml:Sensor>
```

## 8.3 GetObservation (mandatory)

### 8.3.1 Introduction

The GetObservation operation is designed to query sensor systems to retrieve observation data in the form defined in of the Observation and Measurement specification. Upon receiving a GetObservation request, a SOS shall either satisfy the request or return an exception report.

### 8.3.2 Request

A <GetObservation> message contains one or more elements that constrain the observations to be retrieved from a Sensor Observation Service. Each GetObservation query element has mandatory attributes of <service> and <version>. The mandatory <version> element attribute must correspond to the specific service interface version negotiated between the service and client during the service binding process. The required <service> attribute explicitly specifies the service type. A fixed value of SOS should be used for the SOS interface.

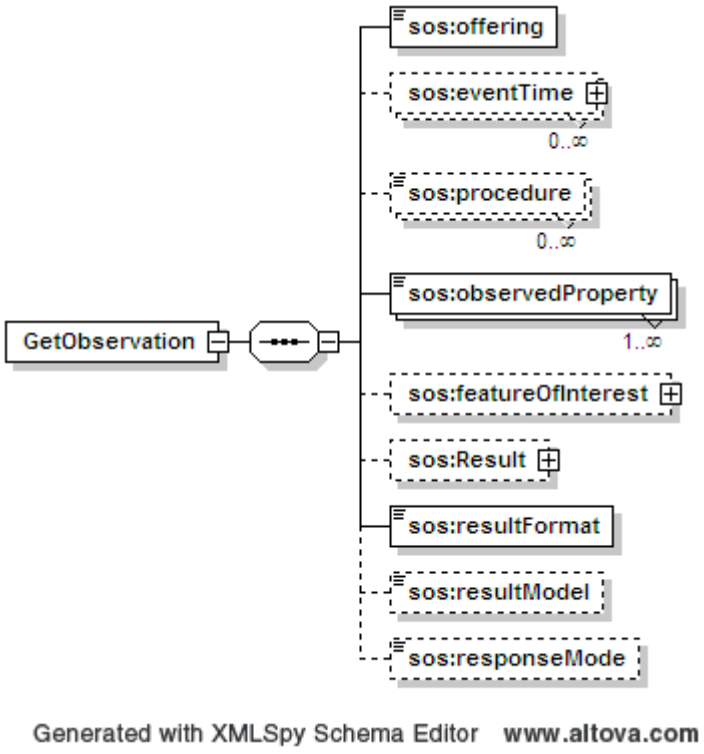


**Figure 18 GetObservation Request**

**Table 4 Parameters of GetObservation Request**

| Name[a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| Offering | The Offering parameter specifies the offering identification advertised in the GetCapabilities document. All of the following parameters are dependent on the selected offering. | Character String type, not empty | One (mandatory) |
| ObservedProperty | The ObservedProperty parameter specifies the observable(s) for which observations are requested. This is a further filter on what was advertised in the GetCapabilities document. This specifies the identifier of a phenomenon advertised in the GetCapabilities document with all possible phenomena listed in the selected offering capabilities. | Character String type, not empty | One or many (mandatory) |
| eventTime | The eventTime parameter specifies the time period(s) for which observations are requested. This allows a client to request observations from a specific instant, multiple instances or periods of time in the past, present and future. The supported range is listed in the selected offering capabilities. | Time type, not empty<br>Value is specified in the selected offering capabilities. | Zero or many (Optional) |
| procedure | The procedure parameter specifies the sensor system(s) for which observations are requested. This is a further filter on what was advertised in the GetCapabilities document. | Character String type | Optional |
| featureOfInterest | The featureOfInterest parameter specifies the feature for which observations are requested. This can either be represented by a reference to a feature ID advertised in the capabilities document or can be a generic GML spatial constraint. This is a further filter on what was advertised in the GetCapabilities document. The featureOfInterest represents the object on which the sensor systems are making observations. In the case of an in-site sensor this may be a station to which the sensor is attached or the atmosphere directly surrounding the sensor. For remote sensors this may be the area or volume that is being sensed. | Character String type | Zero or one (Optional) |
| Result | The Result parameter provides a place to put in OGC filter expressions based on property values. This instructs the SOS to only return observations where the result matches this expression. | Character String type | Zero or one (Optional) |
| resultFormat | The resultFormat parameter specifies the desired resultFormat MIME type for results (e.g. TML, O&M native format, or MPEG stream out-of-band). This specifies the identifier of the output format to be used for the requested data. The supported output formats are listed in the selected offering capabilities. | Character String type | One (mandatory) |
| resultModel | The resultModel parameter specifies the desired format for results (e.g. generic Observation or Common Observation format). The parameter gives the identifier of the result model to be used for the requested data. The resultModels supported by a SOS server are listed in the contents section of the service metadata, identified as QName values. If the requested resultModel is not supported by the SOS server, an exception message shall be returned. | Character String type | Zero or one (Optional) |

| | | | |
|---|---|---|---|
| responseMode | The responseMode parameter specifies whether results are requested in-line, out-of-band, as an attachment, or if this is a request for an observation template that will be used for subsequent calls to GetResult. This is provided to allow the client to request the form of the response. The value of resultTemplate is used to retrieve an observation template that will later be used in calls to GetResult. The other options allow results to appear inline in a resultTag (inline), external to the observation element (out-of-band) or as a MIME attachment (attached). | Character String type | Zero or one (Optional) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., "WMS", "SOS") | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |
| a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008]. | | | |

### 8.3.3 Response

The response to a GetObservation request will be encapsulated within Observation Features, Observation Collections and Observation Arrays, and within Discrete Observation Coverages. The SOS interface is optimized for access to observations and associated information. A detailed description of the response model and examples can be found in the Observations and Measurement encoding document[5]. Annex B contains an example of a full GetObservation response in O&M format.

### 8.3.4 Exceptions

When a SOS server encounters an error while performing a GetObservation operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 8.3.5 Examples

#### 8.3.5.1 Example1

Here is an example of a GetObservation request:

```
<?xml version="1.0" encoding="UTF-8"?>
<GetObservation service="SOS" version="0.0.31"
xmlns="http://www.opengeospatial.net/sos"
xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:swe="http://www.opengis.net/swe">
```

```
  <offering>offering-1</offering>
    <eventTime>
      <ogc:During>
        <gml:TimePeriod>
          <gml:beginPosition>2005-09-01T11:54:32</gml:beginPosition>
          <gml:endPosition>2005-09-02T14:54:32</gml:endPosition>
        </gml:TimePeriod>
      </ogc:During>
    </eventTime>
  <procedure>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-
1</procedure>

<observedProperty>urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed</observed
Property>
<observedProperty>urn:ogc:def:phenomenon:OGC:1.0.30:AverageAirTemperatu
re15Minute</observedProperty>
  <featureOfInterest>
    <ogc:BBOX>
      <gml:Envelope srsName="EPSG:4326">
        <gml:lowerCorner>38.6 -77.9</gml:lowerCorner>
        <gml:upperCorner>41.0 -77.85</gml:upperCorner>
      </gml:Envelope>
    </ogc:BBOX>
  </featureOfInterest>
  <resultFormat>text/xml;subtype="OM/1.0.30"</resultFormat>
</GetObservation>
```

Annex B has an example of a response to this request.

# 9 Transaction Operations Profile (optional)

## 9.1 RegisterSensor (optional)

### 9.1.1 Introduction

The **RegisterSensor** operation allows the client to register a new sensor system with the SOS as part of the transactional profile. Sensor observations can only be inserted for sensors that have first been registered with the SOS. **RegisterSensor** is mandatory for the transactional profile.

### 9.1.2 Request

A RegisterSensor request includes a SensorML or TML system description and an O&M Observation instance that is a template for the observations that will be published for this sensor using the InsertObservaton operation described below.

Each RegisterSensor operation has mandatory attributes of <service> and <version>. The <version> element must correspond to the specific service interface version negotiated between the service and client during the service binding process. The required <service> attribute explicitly specifies the service type. A fixed value of SOS must be used for the SOS interface.

 

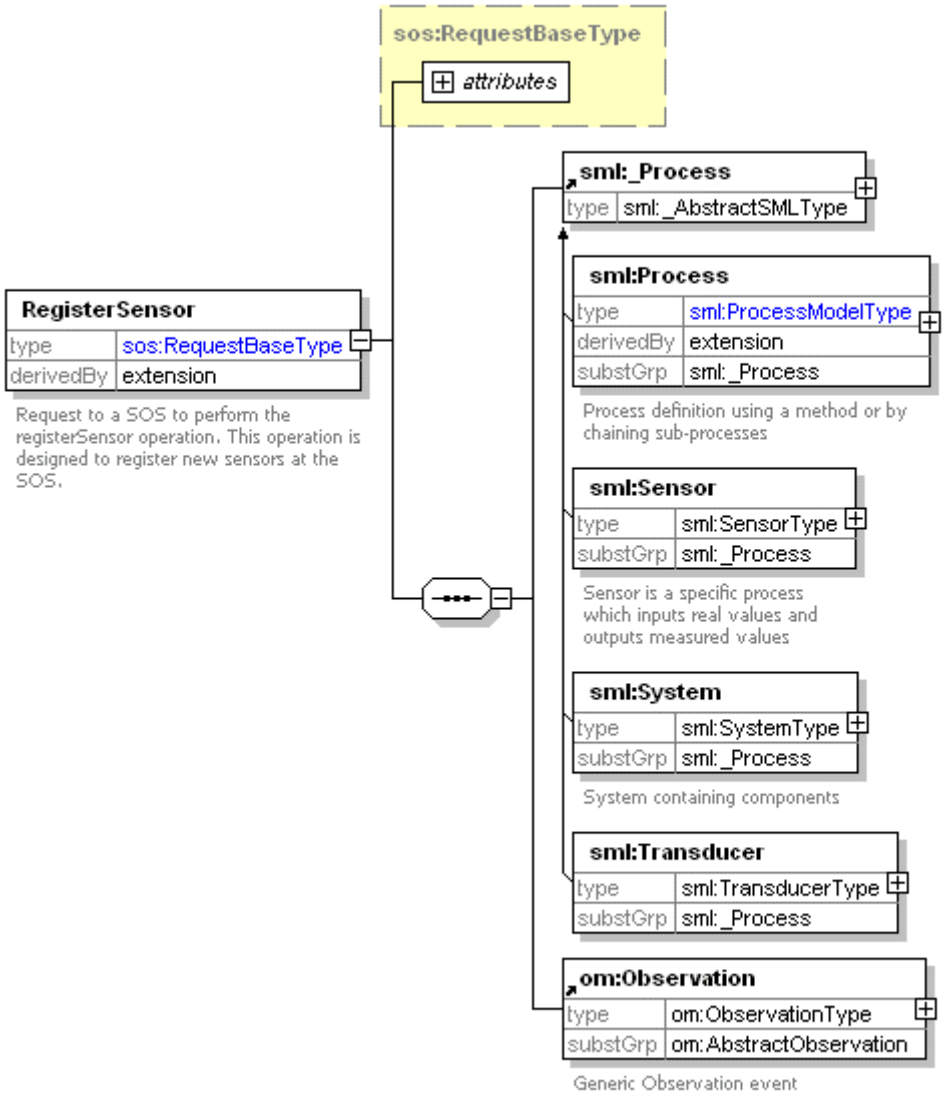


**Figure 19 Parameters of a RegisterSensor Request**

**Table 5 Parameters of RegisterSensor Request**

| Name[a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| SensorML_Process or TML_Process | This is the head of a substitution group that can include a Sensor or System element from either SensorML or TML. This is the detailed description of this sensor or system. | XML document | One (mandatory) |
| O&M:Observation | The O&M:Observation parameter is a template for observation instances that will be inserted for this sensor or system. | XML document | One (mandatory) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., “WMS”, “SOS”) | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

### 9.1.3 Response

The response to a RegisterSensor request is an InsertId which is the identifier used by the SOS to reference the new sensor.

### 9.1.4 Exceptions

When a SOS server encounters an error while performing a RegisterSensor operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 9.1.5 Examples

#### 9.1.5.1 Example1

Here is an example of a RegisterSensor request:

```
<RegisterSensor service="SOS" version="0.0.31"
xmlns="http://www.opengeospatial.net/sos">
  <sml:Sensor xmlns:swe="http://www.opengis.net/swe"
xmlns:xlink="http://www.w3.org/1999/xlink" version="1.0"
xmlns:sml="http://www.opengis.net/sensorML"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" id="system.acada-
sensor-1">
    <sml:identification>
     <sml:IdentifierList>
        <sml:identifier name="manufacturer">
          <sml:Term qualifier="urn:ogc:identifier:manufacturer">ACME
Corporation</sml:Term>
```

```
        </sml:identifier>
      </sml:IdentifierList>
    </sml:identification>
    <sml:inputs>
      <sml:InputList>
        <sml:input name="electricalsignal">
          <swe:Quantity definition="urn:ogc:phenomenon:frequency"
uom="urn:ogc:unit:hertz"/>
        </sml:input>
      </sml:InputList>
    </sml:inputs>
    <sml:outputs>
      <sml:OutputList>
        <sml:output name="Concentration">
          <swe:Quantity id="HD-conc"
definition="urn:acada:definition:phenomenon:chemicalpresence:HD"
uom="urn:ogc:unit:part"/>
        </sml:output>
      </sml:OutputList>
    </sml:outputs>
    <sml:referenceFrame>
      <sml:LocalTimeCRS id="SENSOR_TIME">
        <sml:srsName>GMT</sml:srsName>
        <sml:usesCS xlink:href="urn:ogc:crs:GMT"/>
      </sml:LocalTimeCRS>
    </sml:referenceFrame>
  </sml:Sensor>
  <om:Observation xmlns:om="http://www.opengis.net/om" >
    <om:time xsi:nil="true"/>
    <om:location xsi:nil="true"/>
    <om:procedure xlink:href="system.acada-sensor-1"/>
    <om:observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGA"/
>
    <om:featureOfInterest>
      <om:Station>
        <om:position/>
        <om:procedureHosted/>
      </om:Station>
    </om:featureOfInterest>
    <om:result xsi:nil="true"/>
  </om:Observation>
</RegisterSensor>
```

Here is an example of a RegisterSensor response:

```
<RegisterSensorResponse xmlns="http://www.opengeospatial.net/sos">
<InsertId>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-1</InsertId>
</RegisterSensorResponse>
```

## 9.2 InsertObservation (optional)

### 9.2.1 Introduction

The **InsertObservation** operation allows the client to insert new observations for a sensor system. This is a request to an SOS to perform the Insert operation. The request is constrained by the following parameters: ID obtained by the RegisterSensor operation (identifying the sensor and the observationType), and the observation encoded as O&M. InsertObservation is mandatory for the transactional profile.

### 9.2.2 Request

An InsertObservation request includes the InsertId that was returned from the RegisterSensor operation when the sensor that made this observation was registered. It also includes an O&M Observation that follows the template that was provided when the sensor was registered.

Each InsertObservation operation has mandatory attributes of <service> and <version>. The <version> element must correspond to the specific service interface version negotiated between the service and client during the service binding process. The required <service> attribute explicitly specifies the service type. A fixed value of SOS must be used for the SOS interface.

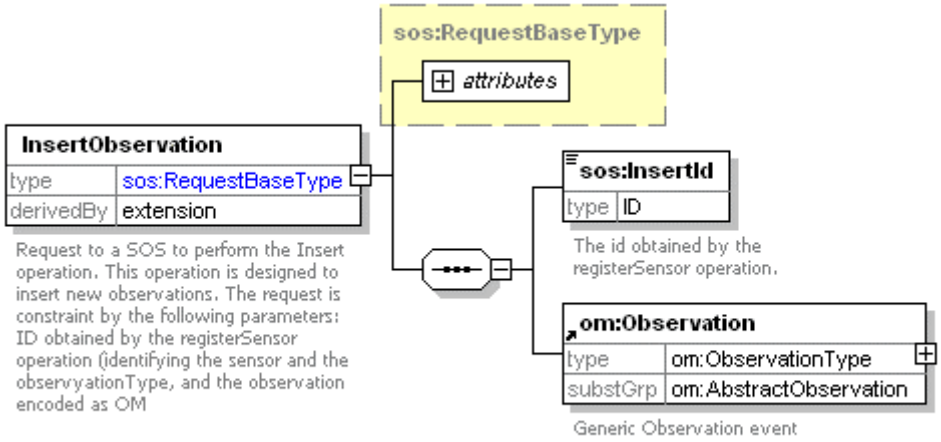


**Figure 20 Parameters of an InsertObservation Request**

**Table 6 Parameters of insertObservation Request**

| Name[a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| InsertId | The InsertId parameter specifies the identifier for the sensor that made this observation. This identifier is obtained by the RegisterSensor operation. | Character String type | One (mandatory) |
| O&M:Observation | The Observation parameter provides the new observation for insertion. | Character String type | One (mandatory) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., “WMS”, “SOS”) | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

### 9.2.3 Response

The response to an InsertObservation request is an empty InsertObservationResponse tag.

### 9.2.4 Exceptions

When a SOS server encounters an error while performing a InsertObservation operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 9.2.5 Examples

#### 9.2.5.1 Example1

Here is an example of an InsertObservation request:

```
<InsertObservation service="SOS" version="0.0.31"
xmlns="http://www.opengeospatial.net/sos"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:om="http://www.opengis.net/om"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<InsertId>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-1</InsertId>
  <om:Observation xmlns:om="http://www.opengis.net/om"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink">
    <om:time>
      <gml:TimeInstant>
        <gml:timePosition>2005-10-28T04:03:39.000Z</gml:timePosition>
      </gml:TimeInstant>
    </om:time>
```

```
    <om:location xlink:href="#location0"/>
    <om:procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:acada-
sensor-1"/>
    <om:observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGA"/
>
    <om:featureOfInterest>
      <om:Station>
        <om:position>
          <gml:Point gml:id="location0">
            <gml:pos srsName="EPSG:4326">33.062018 -
116.616526</gml:pos>
          </gml:Point>
        </om:position>
        <om:procedureHosted/>
      </om:Station>
    </om:featureOfInterest>
    <om:result xsi:type="xsi:boolean">true</om:result>
  </om:Observation>
</InsertObservation>
```

# 10 Enhanced Operations Profile (optional)

## 10.1 GetResult (optional)

### 10.1.1 Introduction

The purpose of the GetResult operation is to allow a client to repeatedly obtain sensor data from the same set of sensors without having to send and receive requests and responses that largely contain the same data except for a new timestamp. A common use case is for a client to repeatedly request current sensor data from one or more sensors on a recurring basis. This operation supports that use case and allows it to occur using much less bandwidth than would be necessary for a full GetObservation call. The motivation for including the operation is to support a data center requesting data from a node that talks directly to sensors over a low bandwidth connection such as a 3G wireless link.

The GetResult operation relies on the creation of an O&M template from a previous call to GetObservation. The ID of the template is used for subsequent GetResult calls instead of sending a duplicate GetObservation XML document. The response contains only the result portion of the O&M Observation because the other components are included by reference in the template.

Here is a typical sequence of even that might occur with the GetResult operation:

1. Client sends a GetObservation request with responseMode="resultTemplate" indicating that he wants an observation template for later calls to GetResult

2. SOS responds with a collection of Observation templates. There is one observation template per sensor that matches the query criteria. Each template has a GML id that is unique to the SOS server. Each template has a valid time period which acts like a lease.

The template is no longer valid after the expiration and subsequent GetResult requests fail after the expiration.

3. Client sends a GetResult request with no TimeInstant parameter.

4. SOS Server responds with result(s) including all observations since the begin time of the lease.

5. Client sends a GetResult request with an "after" time representing the greatest time of an observation from the previous request. This allows the client to filter out duplicates and means the server does not need to maintain state for each GetResult request.

6. SOS Server responds with result(s) including all observations after the time specified.

7. Client sends request...

8. SOS server responds...

etc. etc

### 10.1.2 Request

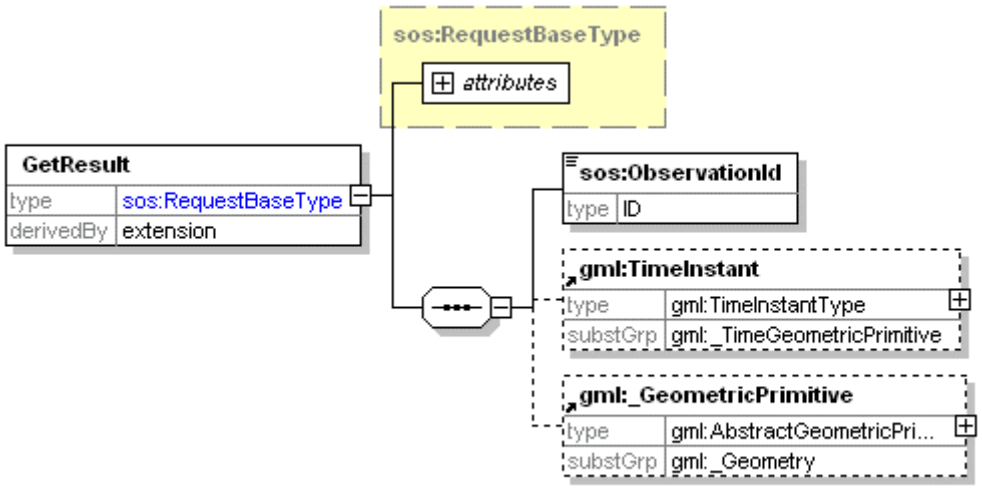


**Figure 21 Parameters of GetResult Request**

**Table 7. Parameters of GetResult Request**

| Name[a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| ObservationId | The ObservationId parameter specifies the Observation template whose result is desired. This must match the gml:id of an Observation from a template that resulted from a previous call to GetObservation. | Character String type, not empty | One (mandatory) |
| TimeInstant | The TimeInstant parameter supports the common use case that data is desired for the period of time since the last call to GetResult was made. The indeterminatePosition attribute of TimeInstant can be set to "after" and the time set to the last time for which data was received during the last request to get all data since the last request. | GML TimeInstant | Optional |
| gml:Geometric Primitive | This element is the abstract head of the substitution group for all (pre- and user-defined) geometric primitives. This is used to subset the response by geographic area, if desired. | GML Geometric type | Optional |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., "WMS", "SOS") | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

### 10.1.3 Response

The response to a GetResult request is a GetResultResponse document. The response contains only a result element with an RS attribute. The RS attribute is a URI that reference the O&M template that provides the context for the in-line result value. The template is the same as what was returned from the call to GetObservation that preceded the call to GetResult (see explanation above). The result is the contents of the result that would be inserted in the referenced template to create a complete O&M Observation document.

### 10.1.4 Exceptions

When a SOS server encounters an error while performing a GetResult operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 10.1.5 Examples

#### 10.1.5.1 Example1

Here is an example of the initial GetObservation request that creates a template for subsequent GetResult requests. The indicator that this is a request for a result template is in the use of the optional responseMode tag with a value of "resultTemplate".

```
<GetObservation service="SOS" version="0.0.31"
xmlns="http://www.opengeospatial.net/sos"
xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:swe="http://www.opengis.net/swe">

  <offering>offering-1</offering>
    <eventTime>
       <ogc:After>
          <gml:TimeInstant>
             <gml:timePosition indeterminatePosition="now"/>
          </gml:TimeInstant>
       </ogc:After>
    </eventTime>
<procedure>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1</procedure>
<observedProperty>urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed</observed
Property>
  <featureOfInterest>
    <ogc:BBOX>
      <gml:Envelope srsName="EPSG:4326">
        <gml:lowerCorner>38.6 -77.9</gml:lowerCorner>
        <gml:upperCorner>41.0 -77.85</gml:upperCorner>
      </gml:Envelope>
    </ogc:BBOX>
  </featureOfInterest>
  <resultFormat>text/xml;subtype="OM/1.0.30"</resultFormat>
  <resultModel>swe:DataValueType</resultModel>

  <responseMode>resultTemplate</responseMode>

</GetObservation>
```

Here is a response from an SOS with a single result template because only one sensor system was requested. The <time> element of the collection indicates that the template is valid for a period of one hour. After this time a new template will need to be requested. The id of the template is the gml:id attribute of the Observation within the collection (a specialized CommonObservation element in this example). If there were more than one sensor there would be more than one template – each with its own id. The CommonObservation resultDefinition element and its sub-elements show how the result that is retrieved with the subsequent GetResult request should be parsed by the client.

```
<om:ObservationCollection xmlns:om="http://www.opengis.net/om"
xmlns:gml="http://www.opengis.net/gml"
xmlns:swe="http://www.opengis.net/swe"
xmlns:meta="http://www.seegrid.csiro.au/xml/metaLite"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <om:time>
      <!-- template is valid for one hour -->
      <gml:TimePeriod gml:id="ValidInterval">
         <gml:beginPosition>2005-08-05T12:01:32</gml:beginPosition>
         <gml:endPosition>2005-08-06T13:01:32</gml:endPosition>
```

```
      </gml:TimePeriod>
   </om:time>
   <om:location xsi:nil="true"/>
   <om:member>
      <om:CommonObservation gml:id="Template-12345">
         <om:time xlink:href="#ValidInterval"/>
         <om:location xsi:nil="true"/>
         <om:procedure
xlink:href="urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1"/>
         <om:observedProperty>
            <swe:CompositePhenomenon gml:id="UserDefinedID"
dimension="4">
               <gml:name>UserDefinedComposite</gml:name>
               <swe:component xlink:href="urn:ogc:data:time:iso860"/>
               <swe:component
xlink:href="urn:ogc:phenomenon:location:EPSG:4326:longitude"/>
               <swe:component
xlink:href="urn:ogc:phenomenon:location:EPSG:4326:latitude"/>
               <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed"/>
            </swe:CompositePhenomenon>
         </om:observedProperty>
         <om:featureOfInterest>
            <om:Station>
               <om:position/>
               <om:procedureHosted/>
            </om:Station>
         </om:featureOfInterest>
         <om:resultDefinition>
            <swe:DataDefinition id="DataDefinition">
               <swe:dataComponents>
                  <swe:DataGroup>
                     <swe:component name="datetime">
                        <swe:IsoDateTime
definition="urn:ogc:data:time:iso8601"/>
                     </swe:component>
                     <swe:component name="longitude">
                        <swe:Quantity
definition="urn:ogc:phenomenon:location:EPSG:4326:longitude"
uom="urn:ogc:unit:degree"/>
                     </swe:component>
                     <swe:component name="latitude">
                        <swe:Quantity
definition="urn:ogc:phenomenon:location:EPSG:4326:latitude"
uom="urn:ogc:unit:degree"/>
                     </swe:component>
                     <swe:component name="windspeed">
                        <swe:Quantity
definition="urn:ogc:def:phenomenon:OGC:1.0.30:AverageWindSpeed15Minute"
uom="urn:ogc:urn:ogc:def:uom:OGC:1.0.30:speed-ms"/>
                     </swe:component>
                  </swe:DataGroup>
               </swe:dataComponents>
               <swe:encoding>
                  <swe:XMLTuple decimalSeparator="."
tokenSeparator="," tupleSeparator="@@"/>
               </swe:encoding>
            </swe:DataDefinition>
```

```
          </om:resultDefinition>
          <om:result xsi:nil="true"/>
       </om:CommonObservation>
   </om:member>
</om:ObservationCollection>
```

Here is the first GetResult request from the client. There is no time specified because all results available are requested. Note that the ObservationId matches the gml:id of the template above.

```
<GetResult service="SOS" version="0.0.31"
xmlns="http://www.opengeospatial.net/sos">
   <ObservationId>Template-12345</ObservationId>
</GetResult>
```

An example of the response might be as follows. The URL in the RS attribute resolves to the contents of the template shown above.

```
<GetResultResponse xmlns="http://www.opengeospatial.net/sos"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <result RS="http://3eti.com/sos/templates/Template-12345">
2005-08-05T12:01:32Z,33.06351,-116.61463,5.3@@
2005-08-05T12:02:13Z,33.06351,-116.61463,5.2
  </result>
</GetResultResponse>
```

If the client made an additional request at 2005-08-05T12:20:32 and included the timestamp of the last time for which it received data the GetResult request would return all of the data since the last request. Here is an example of what that request might look like:

```
<GetResult service="SOS" version="0.0.31"
xmlns="http://www.opengeospatial.net/sos"
xmlns:gml="http://www.opengis.net/gml">
  <ObservationId>Template-12345</ObservationId>
  <gml:TimeInstant>
<!-- This time is the last sensor reading that was received.  The
client doesn't want to get duplicate records -->
    <gml:timePosition indeterminatePosition="after">2005-08-
05T12:02:13Z</gml:timePosition>
  </gml:TimeInstant>
</GetResult>
```

Here is an example of what might get returned by the SOS. Note that no duplicate records are sent to the client because of the TimeInstant argument. If no TimeInstant were specified then the client would receive all records for the full hour of the template

including the ones already received. The client is responsible for keeping track of state, if desired.

```
<GetResultResponse xmlns="http://www.opengeospatial.net/sos"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:gml="http://www.opengis.net/gml">
  <result RS="http://3eti.com/sos/templates/Template-12345">
2005-08-05T12:03:32Z,33.06351,-116.61463,5.3@@
2005-08-05T12:04:32Z,33.06351,-116.61463,5.2@@
2005-08-05T12:05:32Z,33.06351,-116.61463,4.7@@
2005-08-05T12:06:32Z,33.06351,-116.61463,6.4@@
2005-08-05T12:07:32Z,33.06351,-116.61463,4.1@@
2005-08-05T12:08:32Z,33.06351,-116.61463,7.3@@
2005-08-05T12:09:32Z,33.06351,-116.61463,3.8@@
2005-08-05T12:10:32Z,33.06351,-116.61463,2.5@@
2005-08-05T12:11:32Z,33.06351,-116.61463,4.8@@
2005-08-05T12:12:32Z,33.06351,-116.61463,8.3@@
2005-08-05T12:13:32Z,33.06351,-116.61463,7.2@@
2005-08-05T12:14:32Z,33.06351,-116.61463,1.2@@
2005-08-05T12:15:32Z,33.06351,-116.61463,3.3@@
2005-08-05T12:16:32Z,33.06351,-116.61463,5.3@@
2005-08-05T12:17:32Z,33.06351,-116.61463,3.6@@
2005-08-05T12:18:32Z,33.06351,-116.61463,7.2@@
2005-08-05T12:19:32Z,33.06351,-116.61463,7.1
  </result>
</GetResultResponse>
```

## 10.2 GetFeatureOfInterest (optional)

### 10.2.1 Introduction

GetFeatureOfInterest returns a featureOfInterest that was advertised in one of the observation offerings of the SOS capabilities document. This could be a Station for in-situ sensors, for example.

### 10.2.2 Request

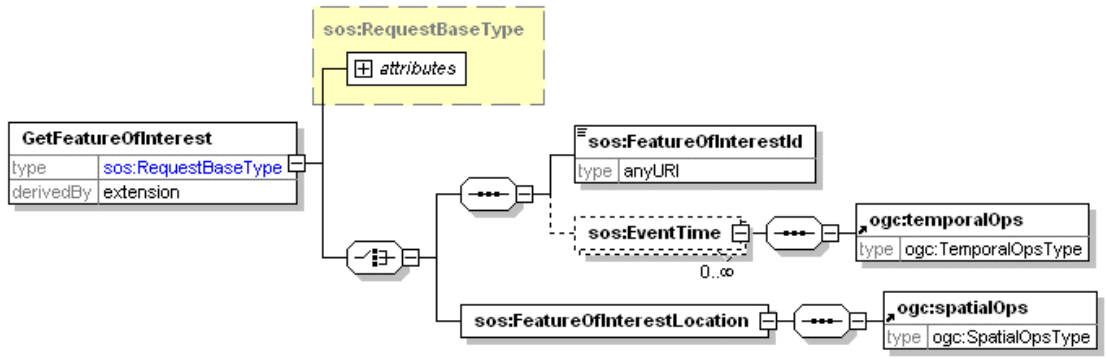


**Figure 22 Parameters for a GetFeatureOfInterest Request**

**Table 8. Parameters of GetFeatureOfInterest Request**

| Name [a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| FeatureOfInterestId [b] | This parameter specifies the identifier of the target feature being returned, for which detailed information is requested. These identifiers must be listed in the Contents section of the service metadata (GetCapabilities) document. | Character String type, not empty | Choice 1 - One (mandatory) |
| EventTime | This parameter specifies the time for which the target feature is of interest. This uses a modified version of filter.xsd and allows a client to request targets from a specific instance, multiple instances or periods of time in the past, present and future. This is useful for dynamic sensors for which the properties of the target are time-dependent. Multiple time parameters may be indicated so that the client may request details of the observation target at multiple times. The supported range is listed in the contents section of the service metadata. | Character String type | Choice 1 - Optional |
| FeatureOfInterestLocation [b] | This argument allows the request to be made based on location such as a BBox rather by specific feature Id. | ogc:spatialOpsType | Choice 2 – One (mandatory) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., “WMS”, “SOS”) | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

b The schema presents a choice between requesting based on Id or location

### 10.2.3 Response

The response to a GetFeatureOfInterest request is a GML feature.

### 10.2.4 Exceptions

When a SOS server encounters an error while performing a GetFeatureOfInterest operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 10.2.5 Examples

#### 10.2.5.1 Example1

Here is an example of a capabilities document with a reference to a featureOfInterest that happens to be a Station. Most of the parameters of the Station are suppressed in capabilities document to save bandwidth.

```
<om:featureOfInterest>
  <om:Station gml:id="station1">
    <om:position/>
    <om:procedureHosted/>
  </om:Station>
</om:featureOfInterest>
```

The client wants to retrieve the feature instance for the station to see where it is located and what sensors are attached to it. Here is what the request might look like. Note that the FeatureOfInterestId matches the gml:id of the featureOfInterest from the capabilities document.

```
<GetFeatureOfInterest xmlns="http://www.opengeospatial.net/sos"
service="SOS" version="0.0.31"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:gml="http://www.opengis.net/gml"
xmlns:ogc="http://www.opengis.net/ogc">
  <FeatureOfInterestId>station1</FeatureOfInterestId>
</GetFeatureOfInterest>
```

The response might look like this indicating the location of the station, some additional metadata, and the identifiers of three sensors that are attached to the station.

```
<Station xmlns="http://www.opengis.net/om"
xmlns:gml="http://www.opengis.net/gml"
xmlns:swe="http://www.opengis.net/swe"
xmlns:meta="http://www.seegrid.csiro.au/xml/metaLite"
xmlns:xlink="http://www.w3.org/1999/xlink" gml:id="station1">
<gml:description>Weather station at ACME headquarters</gml:description>
<gml:name>station1</gml:name>
<position>
  <gml:Point>
    <gml:pos srsName="EPSG:4326">33.06351 -116.61463</gml:pos>
  </gml:Point>
</position>
<procedureHosted xlink:href="urn:ogc:object:feature:Sensor:acme:1234"/>
<procedureHosted xlink:href="urn:ogc:object:feature:Sensor:acme:4567"/>
<procedureHosted xlink:href="urn:ogc:object:feature:Sensor:acme:3234"/>
</Station>
```

## 10.3 GetFeatureOfInterestTime (optional)

### 10.3.1 Introduction

GetFeatureOfInterestTime returns the time periods for which the SOS will return data for a given advertised feature of interest.

### 10.3.2 Request

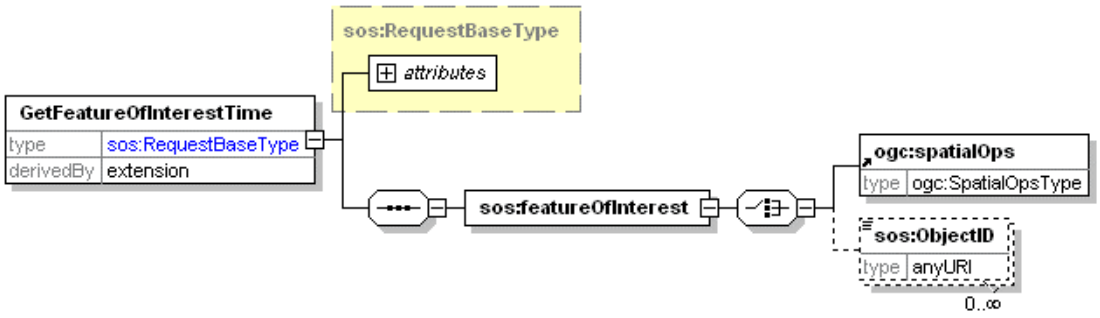


**Figure 23 Parameters for a GetFeatureOfInterestTime Request**

**Table 9. Parameters of GetFeatureOfInterestTime Request**

| Name [a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| Ogc:spatialOps [b] | Request that features be selected by a spatial extent. | ogc:spatialOpsType | Choice 1 - One (mandatory) |
| ObjectID [b] | Request that a feature be selected by ID. | Character String type, not empty | Choice 2 - One (mandatory) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., “WMS”, “SOS”) | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |
| a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008]. | | | |
| b The schema presents a choice between requesting based on Id or location | | | |

### 10.3.3 Response

The response to a GetFeatureOfInterestTime request is a GML time primitive which lists one or more time periods for which observations from that feature of interest are available.

### 10.3.4 Exceptions

When a SOS server encounters an error while performing a GetFeatureOfInterestTime operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 10.3.5 Examples

#### 10.3.5.1 Example1

Here is an example of a capabilities document with a reference to a featureOfInterest that happens to be a Station.

```
<om:featureOfInterest>
  <om:Station gml:id="station1">
    <om:position/>
    <om:procedureHosted/>
  </om:Station>
</om:featureOfInterest>
```

Here is a GetFeatureOfInterestTime request that seeks the times for which this station provides observations.

```
<GetFeatureOfInterestTime xmlns="http://www.opengeospatial.net/sos"
service="SOS" version="0.0.31">
  <featureOfInterest>
    <ObjectID>station1</ObjectID>
  </featureOfInterest>
</GetFeatureOfInterestTime>
```

Here is what the response might look like. In this example the SOS is reporting that data from station 1 is available from 2005-10-25T14:45:00.000Z to 2005-11-01T01:25:00.000Z in 5 minute intervals.

```
<gml:TimePeriod xmlns:gml="http://www.opengis.net/gml">
  <gml:beginPosition>2005-10-25T14:45:00.000Z</gml:beginPosition>
  <gml:endPosition>2005-11-01T01:25:00.000Z</gml:endPosition>
  <gml:timeInterval unit="minute">5</gml:timeInterval>
</gml:TimePeriod>
```

## 10.4 DescribeFeatureOfInterest (optional)

### 10.4.1 Introduction

DescribeFeatureOfInterest returns the XML schema for the specified GML feature of interest advertised in GetCapabilities.

### 10.4.2 Request

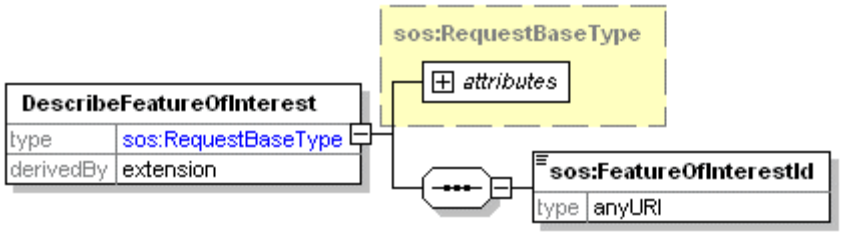


**Figure 24 Parameters for DescribeFeatureOfInterest Request**

**Table 10. Parameters of DescribeFeatureOfInterest Request**

| Name [a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| FeatureOfInterestId | The gml:id of the feature of interest from the capabilities document. | Character String type, not empty | One (mandatory) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., “WMS”, “SOS”) | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

### 10.4.3 Response

The response to a DescribeFeatureOfInterest request is the XML Schema that defines the feature type for the specified feature.

### 10.4.4 Exceptions

When a SOS server encounters an error while performing a DescribeFeatureOfInterest operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 10.4.5 Examples

#### 10.4.5.1 Example1

Here is an example of a DescribeFeatureOfInterest request for the schema of the feature called station 1 used in the previous examples.

```
<DescribeFeatureOfInterest xmlns="http://www.opengeospatial.net/sos"
service="SOS" version="0.0.31">
   <FeatureOfInterestId>station1</FeatureOfInterestId>
</DescribeFeatureOfInterest>
```

The response would be the schema for Station which might look like this:

```
<schema xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:meta="http://www.seegrid.csiro.au/xml/metaLite"
xmlns:om="http://www.opengis.net/om"
xmlns:swe="http://www.opengis.net/swe"
targetNamespace="http://www.opengis.net/om"
elementFormDefault="qualified" attributeFormDefault="unqualified"
version="0.0.31" xml:lang="en">
   <import namespace="http://www.opengis.net/gml"
schemaLocation="./gml4sos.xsd"/>
   <element name="Station" type="om:StationType"
substitutionGroup="gml:_Feature"/>
   <complexType name="StationType">
      <complexContent>
         <extension base="gml:AbstractFeatureType">
            <sequence>
               <element name="position" type="gml:PointPropertyType"/>
               <element name="procedureHosted"
type="gml:ReferenceType" maxOccurs="unbounded"/>
            </sequence>
         </extension>
      </complexContent>
   </complexType>
</schema>
```

## 10.5 DescribeObservationType (optional)

### 10.5.1 Intoduction

The DescribeObservationType operation returns the XML schema that describes the Observation type that is returned for a particular phenomenon. This allows the SOS to list the set of Observation types that it can deliver (by default the value is "om:Observation"). These are specialized observation types, in which one or other of the standard properties are specialised or restricted, or additional properties added. For example om:Measurement is a specialization in which the result is type="gml:MeasureType". It might be an observation type defined in a specific

 

application domain (e.g. GeoChemMeasurement, defined in XMML, WaterQualityMeasurement, defined by NSW-DIPNR, etc.).

### 10.5.2 Request

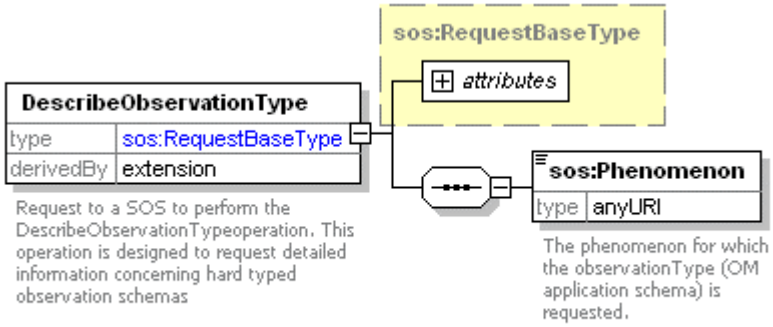


**Figure 25 Parameters for a DescribeObservationType Request**

**Table 11 Parameters of DescribeObservationType Request**

| Name [a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| Phenomenon | The phenomenon parameter specifies the phenomenon for which a DescribeObservationType operation is performed. | URI | One (mandatory) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., “WMS”, “SOS”) | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

### 10.5.3 Response

The response to a DescribeObservationType request is the XML Schema that defines the specialized observation type returned for the given phenomenon.

### 10.5.4 Exceptions

When a SOS server encounters an error while performing a DescribeObservationType operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 10.5.5 Examples

#### 10.5.5.1 Example1

Here is an example of a DescribeObservationType request for the phenomenon of wind speed.

```
<DescribeObservationType xmlns="http://www.opengeospatial.net/sos"
xmlns:ows="http://www.opengeospatial.net/ows" service="SOS"
version="0.0.31">
   <Phenomenon>urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed</Phenomenon>
</DescribeObservationType>
```

The response indicates that the SOS will use the Measurement observation type to report this phenomenon.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:meta="http://www.seegrid.csiro.au/xml/metaLite"
xmlns:om="http://www.opengis.net/om"
xmlns:swe="http://www.opengis.net/swe"
targetNamespace="http://www.opengis.net/om"
elementFormDefault="qualified" attributeFormDefault="unqualified"
version="pre-release">
   <import namespace="http://www.opengis.net/gml"
schemaLocation="./gml4om.xsd"/>
   <import namespace="http://www.seegrid.csiro.au/xml/metaLite"
schemaLocation="../../metaLite/0.1.30/metaLite.xsd"/>
   <import namespace="http://www.opengis.net/swe"
schemaLocation="../../sweCommon/1.0.30/swe.xsd"/>
   <include schemaLocation="./observation.xsd"/>
   <include schemaLocation="./procedure.xsd"/>
   <complexType name="MeasurementType">
      <complexContent>
         <extension base="om:AbstractObservationType">
            <sequence>
               <element name="result" type="om:ResultMeasureType"
nillable="true"/>
            </sequence>
         </extension>
      </complexContent>
   </complexType>
   <element name="Measurement" type="om:MeasurementType"
substitutionGroup="om:AbstractObservation">
      <annotation>
         <documentation>Measurement event</documentation>
      </annotation>
   </element>
</schema>
```

## 10.6 DescribeResultModel (optional)

### 10.6.1 Introduction

**DescribeResultModel** returns the schema for the result element that will be returned when the client asks for the given result model by the given ResultName. The “generic” om:Observation has result type ="xs:anyType", so it is not until an instance is received that the client discovers what the model for the result is, either by inspection of the value of an xsi:type attribute on the <result> element, or merely through the appearance of a sub-element from a namespace declared for the document or local context, with a schemaLocation provided. In order to be able to prepare to use this, the result of a **DescribeResultModel** operation advises the client what the result will contain. A similar argument applies to the other observation types that have soft-typed results, i.e. CommonObservation, ComplexObservation, ComplexMeasurement. The qualified name of the result element can be obtained from the schema that is returned by DescribeObservationType.

### 10.6.2 Request

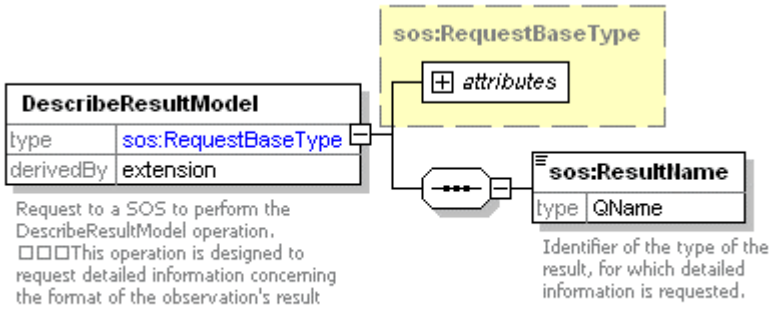


**Figure 26 Parameters for the DescribeResultModel Request**

**Table 12 Parameters of DescribeResultModel Request**

| Name[a] | Definition | Data type and values | Multiplicity and use |
|---|---|---|---|
| ResultName | The ResultName parameter specifies the qualified name of the <result> element in an Observation for which the DescribeResultModel operation is performed. | Qualified name | One (mandatory) |
| service | Service type identifier | Character String type, not empty<br>Value is OWS type abbreviation (e.g., “WMS”, “SOS”) | One (mandatory) |
| version | Specification version for operation | Character String type, not empty<br>Value is specified by each Implementation Specification and Schemas version | One (mandatory) |

a The name capitalization rules being used here are specified in Subclause 11.6.2 of [OGC 05-008].

### 10.6.3 Response

The response to a DescribeResultModel request is the XML Schema that defines the specialized result element type returned for the given qualified name.

### 10.6.4 Exceptions

When a SOS server encounters an error while performing a DescribeResultModel operation, it shall return an exception report message as specified in clause 8 of [OGC 05-008].

### 10.6.5 Examples

#### 10.6.5.1 Example1

Here is an example of a DescribeResultModel request for the qualified name om:ResultMeasureType that was obtained from the call to DescribeObservationType in the previous example.

```
<DescribeResultModel xmlns="http://www.opengeospatial.net/sos"
xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:om="http://www.opengis.net/om" service="SOS" version="0.0.31">
    <ResultName>om:ResultMeasureType</ResultName>
</DescribeResultModel>
```

The response is the XML schema for om:ResultMeasureType.

```
<schema xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:gml="http://www.opengis.net/gml"
xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:meta="http://www.seegrid.csiro.au/xml/metaLite"
```

```
xmlns:om="http://www.opengis.net/om"
xmlns:swe="http://www.opengis.net/swe"
targetNamespace="http://www.opengis.net/om"
elementFormDefault="qualified" attributeFormDefault="unqualified"
version="pre-release">
   <import namespace="http://www.opengis.net/gml"
schemaLocation="./gml4om.xsd"/>
   <import namespace="http://www.seegrid.csiro.au/xml/metaLite"
schemaLocation="../../metaLite/0.1.30/metaLite.xsd"/>
   <import namespace="http://www.opengis.net/swe"
schemaLocation="../../sweCommon/1.0.30/swe.xsd"/>
   <include schemaLocation="./event.xsd"/>
   <include schemaLocation="./procedure.xsd"/>
   <complexType name="ResultMeasureType">
      <simpleContent>
         <extension base="gml:MeasureType">
            <attribute name="relativeMeasure"
type="om:RelativeMeasureCode" default="equals"/>
         </extension>
      </simpleContent>
   </complexType>
   <simpleType name="RelativeMeasureCode">
      <annotation>
         <documentation xml:lang="en">This enumerated data type
specifies values for relative measures.</documentation>
      </annotation>
      <restriction base="string">
         <enumeration value="lessThan"/>
         <enumeration value="lessThanOrEquals"/>
         <enumeration value="equals"/>
         <enumeration value="greaterThanOrEquals"/>
         <enumeration value="greaterThan"/>
      </restriction>
   </simpleType>
</schema>
```

# Annex A
# (normative)

# SOS Schema

These schema are available on the OGC Subversion Repository.

## A.1 *GetCapabilities* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:sos="http://www.opengeospatial.net/sos" elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosGetCapabilities.xsd 2005/05/11</appinfo>
        <documentation>
            <description>This XML Schema encodes the SOS GetCapabilities operation request and
response.</description>
            <copyright>Copyright (c) 2005 Institut for Geoinformatics University of Muenster (Alexander C.
Walkowski)</copyright>
        </documentation>
    </annotation>
    <!-- ============================================================
        includes and imports
    ============================================================ -->
    <import namespace="http://www.opengeospatial.net/ows" schemaLocation="./ows4sos.xsd"/>
    <include schemaLocation="sosContents.xsd"/>
    <import namespace="http://www.opengis.net/ogc" schemaLocation="./ogc4sos.xsd"/>
    <!-- ============================================================
        elements and types
    ============================================================ -->
    <element name="GetCapabilities">
        <annotation>
            <documentation>Request to a SOS to perform the GetCapabilities operation. This operation allows a
client to retrieve service metadata (capabilities XML) providing metadata for the specific SOS server. In this XML
encoding, no "request" parameter is included, since the element name specifies the specific operation.
</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="ows:GetCapabilitiesType">
                    <sequence/>
                    <attribute name="service" type="ows:ServiceType" use="required" fixed="SOS"/>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <!-- ========================================================== -->
    <element name="Capabilities">
        <annotation>
            <documentation>XML encoded SOS GetCapabilities operation response. This document provides
clients with service metadata about a specific service instance, including metadata about the tightly-coupled data
served. If the server does not implement the updateSequence parameter, the server shall always return the
complete Capabilities document, without the updateSequence parameter. When the server implements the
updateSequence parameter and the GetCapabilities operation request included the updateSequence parameter
with the current value, the server shall return this element with only the "version" and "updateSequence" attributes.
Otherwise, all optional elements shall be included or not depending on the actual value of the Sections parameter
in the GetCapabilities operation request. </documentation>
```

```
        </annotation>
        <complexType>
            <complexContent>
                <extension base="ows:CapabilitiesBaseType">
                    <sequence>
                        <element ref="ogc:Filter_Capabilities" minOccurs="0"/>
                        <element ref="sos:Contents"/>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
</schema>
```

## A.2 *DescribeSensor* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns:sos="http://www.opengeospatial.net/sos"
xmlns:ows="http://www.opengeospatial.net/ows" xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosDescribeSensor.xsd 2005/06/16</appinfo>
        <documentation>
            <description>This XML Schema defines the DescribeSensor request and response XML elements and
types.</description>
            <copyright>Copyright (c) 2005 Institut for Geoinformatics University of Muenster (Alexander C.
Walkowski)</copyright>
        </documentation>
    </annotation>
    <!-- ============================================================
        includes and imports
    ============================================================ -->
    <import namespace="http://www.opengeospatial.net/ows"
schemaLocation="../../ows/1.0.30/owsCommon.xsd"/>
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ============================================================
        request
    ============================================================ -->
    <element name="DescribeSensor">
        <annotation>
            <documentation>Request to a SOS to perform the DescribeSensor operation. This operation is
designed to request detailed sensor metadata.  </documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element name="SensorId" type="string">
                            <annotation>
                                <documentation>Identifier of the sensor, for which detailed metadata is
requested.</documentation>
                            </annotation>
                        </element>
                    </sequence>
                    <attribute name="outputFormat" type="ows:MimeType" use="optional"
default="text/xml;subtype=&amp;quot;sensorML/1.0.0&amp;quot;">
                        <annotation>
                            <documentation>Identifier of the output format to be used for the requested data. The
outputFormats supported by a SOS server are listed in the operations metadata section of the service metadata
(capabilities XML). If this attribute is omitted, the outputFormat should be tex/xml;subtype="sensorML/1.0.0". If the
requested outputFormat is not supported by the SOS server, an exception message shall be returned.
                </documentation>
                            </annotation>
                    </attribute>
                </extension>
            </complexContent>
```

 

```
        </complexType>
    </element>
    <!-- ==========================================================
        response
    ========================================================== -->
    <!-- Since the response is an SensorML document no explicite response has to be defined.-->
</schema>
```

## A.3 *GetObservation* Schema

```
 <?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml="http://www.opengis.net/gml"
xmlns:ows="http://www.opengeospatial.net/ows" xmlns:sos="http://www.opengeospatial.net/sos"
xmlns:dc="http://purl.org/dc/elements/1.1/" elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosGetObservation.xsd 2005/05/11</appinfo>
        <documentation>
            <dc:description>This XML Schema defines the GetObservation request XML elements and
types.</dc:description>
            <dc:rights>Copyright (c) 2005 Institut for Geoinformatics University of Muenster (Alexander C.
Walkowski)</dc:rights>
        </documentation>
    </annotation>
    <!-- ==========================================================
        includes and imports
    ========================================================== -->
    <import namespace="http://www.opengeospatial.net/ows"
schemaLocation="../../ows/1.0.30/owsCommon.xsd"/>
    <import namespace="http://www.opengis.net/ogc" schemaLocation="../../filter/1.1.30/filter.xsd"/>
    <import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengis.net/gml/3.1.1/base/temporal.xsd"/>
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ==========================================================
        request
    ========================================================== -->
    <element name="GetObservation">
        <annotation>
            <documentation>Request to a SOS to perform the GetObservation operation. This operation is
designed to request sensor data from live sensors as well as sensor data archives.</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element name="offering" type="anyURI">
                            <annotation>
                                <documentation>ID of an offering advertised in the capabilities.
                                    All following parameters are depending on the selected offering.
                                </documentation>
                            </annotation>
                        </element>
                        <element name="eventTime" minOccurs="0" maxOccurs="unbounded">
                            <annotation>
                                <documentation>Allows a client to request observations from a specific instant,
multiple instances or periods of time in the past, present and future. The supported range is listed in the selected
offering capabilities.
                                </documentation>
                            </annotation>
                            <complexType>
                                <sequence>
                                    <element ref="ogc:temporalOps"/>
                                </sequence>
```

```
                    </complexType>
                </element>
                <element name="procedure" type="anyURI" minOccurs="0" maxOccurs="unbounded">
                    <annotation>
                        <documentation>Index of a particular sensor if offering procedure is a Sensor
Array. Allows client to request data from one or more sensors in the array. The size of the array should be
specified in the selected offering capabilities. This is to support scenarios with sensor grids (we don't want to have
one offering for each sensor in that case). Note that sensorML can describe Sensor Arrays too.
                </documentation>
                    </annotation>
                </element>
                <element name="observedProperty" type="anyURI" maxOccurs="unbounded">
                    <annotation>
                        <documentation>ID of a phenomenon advertised in capabilities document.
                            All possible phenomena are listed in the selected offering capabilities.
                        </documentation>
                    </annotation>
                </element>
                <element name="featureOfInterest" minOccurs="0">
                    <annotation>
                        <documentation>Specifies target feature for which observations are requested.
Mostly a hepler for in-situ sensors, since geo-location has to be done on the server side. The supported area
should be listed in the selected offering capabilities.
                        </documentation>
                    </annotation>
                    <complexType>
                        <choice>
                            <element ref="ogc:spatialOps"/>
                            <element name="ObjectID" type="anyURI" maxOccurs="unbounded">
                                <annotation>
                                    <documentation>Unordered list of zero or more object identifiers.
These identifiers are usually listed in the Contents section of the service metadata (Capabilities) document. If no
ObjectID value is included, and if only one category of objects is allowed for this operation, the server shall return
all objects of that category. NOTE: Although retention of this ability is allowed by a specific OWS that uses this
operation, such retention is discouraged due to possible problems. Making this ability optional implementation by
servers reduces interoperability. Requiring implementation of this ability can require a server to return a huge
response, when there are a large number of items in that category. </documentation>
                                </annotation>
                            </element>
                        </choice>
                    </complexType>
                </element>
                <element name="Result" minOccurs="0">
                    <annotation>
                        <documentation>Only report observations where the result matches this
expression.
                        </documentation>
                    </annotation>
                    <complexType>
                        <sequence>
                            <element ref="ogc:comparisonOps"/>
                        </sequence>
                    </complexType>
                </element>
                <element name="resultFormat" type="ows:MimeType">
                    <annotation>
                        <documentation>ID of the output format to be used for the requested data. The
supported output formats are listed in the selected offering capabilities.
                        </documentation>
                    </annotation>
                </element>
                <element name="resultModel" type="sos:resultModelType" minOccurs="0">
                    <annotation>
                        <documentation>Identifier of the result model to be used for the requested data.
The resultModel's supported by a SOS server are listed in the contents section of the service metadata, identified
as QName values. The value "external" may also be selected, indicating that the value of the result is not specified
```

by a schema referenced in this instance document and may not even be XML. If the requested resultModel is not supported by the SOS server, an exception message shall be returned.

```
                    </documentation>
                    </annotation>
                </element>
                <element name="responseMode" type="sos:responseModeType" minOccurs="0">
                    <annotation>
                        <documentation>This element allows the client to request the form of the
response. The value of resultTemplate is used to retrieve an observation template
                    that will later be used in calls to GetResult. The other options allow results to appear
inline in a resultTag (inline), external to the observation element (out-of-band)
                    or as a MIME attachment (attached)</documentation>
                    </annotation>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
</element>
</schema>
```

## A.4 *RegisterSensor* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:sml="http://www.opengis.net/sensorML"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:om="http://www.opengis.net/om"
xmlns:ows="http://www.opengeospatial.net/ows" xmlns:sos="http://www.opengeospatial.net/sos"
targetNamespace="http://www.opengeospatial.net/sos" elementFormDefault="qualified" version="0.0.31"
xml:lang="en">
    <annotation>
        <appinfo>sosRegisterSensor.xsd 2005/08/04</appinfo>
        <documentation>
            <description>This XML Schema defines the registerSenor request and response XML elements
and types.</description>
            <rights>Copyright (c) 2005 Institut for Geoinformatics University of Muenster (Alexander C.
Walkowski)</rights>
        </documentation>
    </annotation>
    <!-- ============================================================
    and imports
    ============================================================ -->
    <import namespace="http://www.opengeospatial.net/ows"
schemaLocation="../../ows/1.0.30/owsCommon.xsd"/>
    <import namespace="http://www.opengis.net/om" schemaLocation="../../om/1.0.30/observation.xsd"/>
    <import namespace="http://www.opengis.net/sensorML"
schemaLocation="../../sensorML/1.0.30/base/sensorML.xsd"/>
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ============================================================

    ============================================================ -->
    <element name="RegisterSensor">
        <annotation>
            <documentation>Request to a SOS to perform the registerSensor operation. This operation is
designed to register new sensors at the SOS.</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element ref="sml:_Process">
                            <annotation>
                                <documentation>The SensorML description</documentation>
                            </annotation>
                        </element>
                        <element ref="om:Observation">
```

```
                                        <annotation>
                                            <documentation>The observation to be inserted to the
SOS.</documentation>
                                        </annotation>
                                    </element>
                                </sequence>
                            </extension>
                        </complexContent>
                    </complexType>
                </element>
                <!-- ==================================================

                ========================================================-->
                <element name="RegisterSensorResponse">
                    <annotation>
                        <documentation>returns the Id that is used in the insert operation to link the sensor to a
bservationType</documentation>
                    </annotation>
                    <complexType>
                        <sequence>
                            <element name="InsertId" type="ID"/>
                        </sequence>
                    </complexType>
                </element>
</schema>
```

## A.5 *InsertObservation* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:om="http://www.opengis.net/om" xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:sos="http://www.opengeospatial.net/sos" targetNamespace="http://www.opengeospatial.net/sos"
elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosInsert.xsd 2005/08/04</appinfo>
        <documentation>
            <description>This XML Schema defines the insert request and response XML elements and
types.</description>
            <rights>Copyright (c) 2005 Institut for Geoinformatics University of Muenster (Alexander C.
Walkowski)</rights>
        </documentation>
    </annotation>
    <!-- ============================================================
    and imports
    ============================================================ -->
    <import namespace="http://www.opengeospatial.net/ows"
schemaLocation="../../ows/1.0.30/owsCommon.xsd"/>
    <import namespace="http://www.opengis.net/om" schemaLocation="../../om/1.0.30/observation.xsd"/>
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ============================================================

    ============================================================ -->
    <element name="InsertObservation">
        <annotation>
            <documentation>Request to a SOS to perform the Insert operation. This operation is designed to
insert new observations. The request is constraint by the following parameters: ID obtained by the registerSensor
operation (identifying the sensor and the observyationType, and the observation encoded as OM</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element name="InsertId" type="ID">
                            <annotation>
```

 

```
                                        <documentation>The id obtained by the registerSensor
operation.</documentation>
                                    </annotation>
                                </element>
                                <element ref="om:Observation">
                                    <annotation>
                                        <documentation>The observation to be inserted to the
SOS.</documentation>
                                    </annotation>
                                </element>
                            </sequence>
                        </extension>
                    </complexContent>
                </complexType>
            </element>
            <!-- ======================================================

            ======================================================-->
            <!--Do we realy need a InsertResponse? Either the server returns an ExceptionReport or the server will
return nothing indicating successfull execution.-->
            <element name="InsertObservationResponse" type="sos:ResponseType">
                <annotation>
                    <documentation>The response to a insert request.</documentation>
                </annotation>
            </element>
            <simpleType name="ResponseType">
                <restriction base="string">
                    <enumeration value="successful"/>
                    <enumeration value="failed"/>
                </restriction>
            </simpleType>
</schema>
```

## A.6 *GetResult* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns:gml="http://www.opengis.net/gml"
xmlns:sos="http://www.opengeospatial.net/sos" xmlns:ows="http://www.opengeospatial.net/ows"
xmlns="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosGetResult.xsd 2005/05/11</appinfo>
        <documentation>
            <description>This XML Schema defines the GetResult request and response XML elements and
types.</description>
            <copyright>Copyright (c) 2005 Institut for Geoinformatics University of Muenster (Ingo
Simonis)</copyright>
        </documentation>
    </annotation>
    <!-- ============================================================
        includes and imports
    ============================================================ -->
    <import namespace="http://www.opengis.net/gml"
schemaLocation="http://schemas.opengeospatial.net/gml/3.1.1/base/gml.xsd"/>
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ============================================================
        request
    ============================================================ -->
    <element name="GetResult">
        <annotation>
            <documentation>request to a SOS to perform a GetResult operation. this operation is designed to
request sensor data from live sensors. Instead of retriveing the observations as a full OM document, you will get an
simple value and a link to the reference system</documentation>
        </annotation>
        <complexType>
            <complexContent>
```

```
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element name="ObservationId" type="ID">
                            <annotation>
                                <documentation>The gml:id of an previous GetObservation request response
indicating observations from a certain sensor for a certain target.</documentation>
                            </annotation>
                        </element>
                        <element ref="gml:TimeInstant" minOccurs="0">
                            <annotation>
                                <documentation>optionaly the point in time could be defined for wich the
observation result is requested</documentation>
                            </annotation>
                        </element>
                        <element ref="gml:_GeometricPrimitive" minOccurs="0">
                            <annotation>
                                <documentation>

                                    is this parameter needed??

                                </documentation>
                            </annotation>
                        </element>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <!-- ============================================================
        response
    ============================================================ -->
    <element name="GetResultResponse">
        <annotation>
            <documentation>the response of a GetResult operation</documentation>
        </annotation>
        <complexType>
            <sequence>
                <element name="result">
                    <annotation>
                        <documentation>RS attribute points to the description of the reference system of the
result. The description will contain all information necessary to understand what is provided within the result
response. The most simple case would be a single value.</documentation>
                    </annotation>
                    <complexType>
                        <simpleContent>
                            <extension base="string">
                                <attribute name="RS" type="anyURI" use="required"/>
                            </extension>
                        </simpleContent>
                    </complexType>
                </element>
            </sequence>
        </complexType>
    </element>
</schema>
```

## A.7 *GetFeatureOfInterest* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:sos="http://www.opengeospatial.net/sos" xmlns:gml="http://www.opengis.net/gml"
elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosGetTargetFeature.xsd 2005/05/11</appinfo>
        <documentation>
```

 

```
            <description>This XML Schema defines the GetFeatureOfInterest request and response XML
elements and types.</description>
            <copyright>Copyright (c) 2005 Institut for Geoinformatics University of Muenster (Alexander C.
Walkowski)</copyright>
        </documentation>
    </annotation>
    <!-- ============================================================
        includes and imports
    ============================================================ -->
    <import namespace="http://www.opengis.net/ogc" schemaLocation="../../filter/1.1.30/filter.xsd"/>
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ============================================================
        request
    ============================================================ -->
    <element name="GetFeatureOfInterest">
        <annotation>
            <documentation>Request to a SOS to perform the GetFeatureOfInterest operation. This operation is
designed to request target feaure instances</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <choice>
                        <sequence>
                            <element name="FeatureOfInterestId" type="anyURI">
                                <annotation>
                                    <documentation>Identifier of the feature of interest, for which detailed
information is requested. These identifiers are usually listed in the Contents section of the service metadata
(Capabilities) document. </documentation>
                                </annotation>
                            </element>
                            <element name="EventTime" minOccurs="0" maxOccurs="unbounded">
                                <annotation>
                                    <documentation>Uses modified version of filter.xsd

                                    Allows a client to request targets from a specific instant, multiple instances or
periods of time in the past, present and future.
                                    This is useful for dynamic sensors for which the properties of the target are time-
dependent.
                                    Multiple time paramters may be indicated so that the client may request details of
the observation target at multiple times.
                                    The supported range is listed in the contents section of the service
metadata.</documentation>
                                </annotation>
                                <complexType>
                                    <sequence>
                                        <element ref="ogc:temporalOps"/>
                                    </sequence>
                                </complexType>
                            </element>
                        </sequence>
                        <element name="FeatureOfInterestLocation">
                            <annotation>
                                <documentation>Uses modified version of filter.xsd</documentation>
                            </annotation>
                            <complexType>
                                <sequence>
                                    <element ref="ogc:spatialOps"/>
                                </sequence>
                            </complexType>
                        </element>
                    </choice>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <!-- ============================================================
```

```
    response
    ======================================================-->
    <!-- The response is an GML feature, the schema of the feature instance can be retrieved using the
DescribeTargetFeature request.-->
</schema>
```

## A.8 *GetFeatureOfInterestTime* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns:gml="http://www.opengis.net/gml"
xmlns:sos="http://www.opengeospatial.net/sos" xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:ogc="http://www.opengis.net/ogc" xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosGetTargetTime.xsd 2005/07/31</appinfo>
        <documentation>
            <description>This XML Schema defines the GetFeatureOfInterestTime request and response XML
elements and types.</description>
            <copyright>Copyright (c) 2005 Institut for Geoinformatics University of Muenster (Alexander C.
Walkowski)
            and Commonwealth Scientific and Industrial Research Organisation (Simon J D Cox)</copyright>
        </documentation>
    </annotation>
    <!-- ============================================================
        includes and imports
    ============================================================ -->
    <import namespace="http://www.opengis.net/ogc" schemaLocation="../../filter/1.1.30/filter.xsd"/>
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ============================================================
        request
    ============================================================ -->
    <element name="GetFeatureOfInterestTime">
        <annotation>
            <documentation>Request to a SOS to perform the GetTargetTime operation.
            This operation is designed to request the time that specified target feature instances or target locations
are available</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element name="featureOfInterest">
                            <annotation>
                                <documentation>Specifies the feature of interest for which the time is requested.
                                </documentation>
                            </annotation>
                            <complexType>
                                <choice>
                                    <element ref="ogc:spatialOps"/>
                                    <element name="ObjectID" type="anyURI" minOccurs="0"
maxOccurs="unbounded">
                                        <annotation>
                                            <documentation>Unordered list of zero or more object identifiers.
These identifiers are usually listed in the Contents section of the service metadata (Capabilities) document. If no
ObjectID value is included, and if only one category of objects is allowed for this operation, the server shall return
all objects of that category. NOTE: Although retention of this ability is allowed by a specific OWS that uses this
operation, such retention is discouraged due to possible problems. Making this ability optional implementation by
servers reduces interoperability. Requiring implementation of this ability can require a server to return a huge
response, when there are a large number of items in that category. </documentation>
                                        </annotation>
                                    </element>
                                </choice>
                            </complexType>
                        </element>
                        <!-- changed to corespond to the new observation model-->
                        <!--<element name="TargetFeature" type="anyURI">
```

```
                        <annotation>
                            <documentation>value should identify a feature instance</documentation>
                        </annotation>
                    </element>
                    <element name="TargetLocation">
                        <annotation>
                            <documentation>Uses modified version of filter.xsd</documentation>
                        </annotation>
                        <complexType>
                            <sequence>
                                <element ref="ogc:spatialOps"/>
                            </sequence>
                        </complexType>
                    </element>-->
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</element>
<!-- ========================================================
response
========================================================-->
<!-- The response is an GML time primitive.-->
</schema>
```

## A.9 *DescribeFeatureOfInterest* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.4 U (http://www.xmlspy.com) by Institut für Geoinformatik (Institut für Geoinformatik) --
>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns:sos="http://www.opengeospatial.net/sos"
xmlns:ows="http://www.opengeospatial.net/ows" xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosDescribeTargetFeature.xsd 2005/06/16</appinfo>
        <documentation>
            <description>This XML Schema defines the DescribeTargetFeature request and response XML
elements and types.</description>
            <copyright>Copyright (c) 2005 Institut for Geoinformatics University of Muenster </copyright>
        </documentation>
    </annotation>
    <!-- ============================================================
        includes and imports
    ============================================================ -->
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ============================================================
    request
    ============================================================ -->
    <element name="DescribeFeatureOfInterest">
        <annotation>
            <documentation>Request to a SOS to perform the DescribeTargetFeature operation. This operation is
designed to request detailed information concerning the observation's target</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element name="FeatureOfInterestId" type="anyURI">
                            <annotation>
                                <documentation>Identifier of the feature of interest, for which detailed information
is requested. These identifiers are usually listed in the Contents section of the service metadata (Capabilities)
document. </documentation>
                            </annotation>
                        </element>
                    </sequence>
                </extension>
```

```
            </complexContent>
        </complexType>
    </element>
    <!-- ==========================================================
        response
    ========================================================== -->
    <!--the response is a XML schema-->
</schema>
```

## A.10 *DescribeObservationType* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.4 U (http://www.xmlspy.com) by Institut für Geoinformatik (Institut für Geoinformatik) --
>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns:sos="http://www.opengeospatial.net/sos"
xmlns:ows="http://www.opengeospatial.net/ows" xmlns="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
        <appinfo>sosDescribeObservationType.xsd 2005/06/16</appinfo>
        <documentation>
            <description>This XML Schema defines the DescribeObservationType request and response XML
elements and types.</description>
            <copyright>Copyright (c) 2005 Institut for Geoinformatics University of Muenster </copyright>
        </documentation>
    </annotation>
    <!-- ==========================================================
        includes and imports
    ========================================================== -->
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ==========================================================
    request
    ========================================================== -->
    <element name="DescribeObservationType">
        <annotation>
            <documentation>Request to a SOS to perform the DescribeObservationTypeoperation. This operation
is designed to request detailed information concerning hard typed observation schemas</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element name="Phenomenon" type="anyURI">
                            <annotation>
                                <documentation>The phenomenon for which the observationType (OM application
schema) is requested.</documentation>
                            </annotation>
                        </element>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <!-- ==========================================================
        response
    ========================================================== -->
    <!--The response will be an XML schema defining the OM applicatoin schemas-->
</schema>
```

## A.11 *DescribeResultModel* Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengeospatial.net/sos" xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengeospatial.net/ows" xmlns:sos="http://www.opengeospatial.net/sos"
elementFormDefault="qualified" version="0.0.31" xml:lang="en">
    <annotation>
```

 

```
        <appinfo>sosDescribeResultModel.xsd 2005/07/31</appinfo>
        <documentation>
            <description>This XML Schema defines the DescribeResultModel request and response XML
elements and types.</description>
            <copyright>Copyright (c) 2005 Commonwealth Scientific and Industrial research Organisation
(Australia) </copyright>
        </documentation>
    </annotation>
    <!-- ============================================================
        includes and imports
    ============================================================ -->
    <include schemaLocation="sosCommon.xsd"/>
    <!-- ============================================================
    request
    ============================================================ -->
    <element name="DescribeResultModel">
        <annotation>
            <documentation>Request to a SOS to perform the DescribeResultModel operation.
            This operation is designed to request detailed information concerning the format of the observation's
result</documentation>
        </annotation>
        <complexType>
            <complexContent>
                <extension base="sos:RequestBaseType">
                    <sequence>
                        <element name="ResultName" type="QName">
                            <annotation>
                                <documentation>Identifier of the type of the result, for which detailed information is
requested.</documentation>
                            </annotation>
                        </element>
                    </sequence>
                </extension>
            </complexContent>
        </complexType>
    </element>
    <!-- ============================================================
        response
    ============================================================ -->
    <!--the response is an XML schema-->
</schema>
```

# Annex B
# (informative)

# SOS Schema Examples

## B.1 SOS GetCapabilities Document Example

This section contains a sample SOS GetCapabilities Document conforming to this version of this specification.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Capabilities xmlns="http://www.opengeospatial.net/sos" xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:swe="http://www.opengis.net/swe"
xmlns:om="http://www.opengis.net/om" version="0.0.31" xmlns:ows="http://www.opengeospatial.net/ows"
xmlns:ogc="http://www.opengis.net/ogc" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengeospatial.net/sos:\ogc\svn\trunk\sos\0.0.31\sosGetCapabilities.xsd
http://www.opengis.net/om C:\ogc\svn\trunk\om\1.0.30\observation.xsd">
    <ows:ServiceIdentification>
        <ows:Title>3eTI SOS</ows:Title>
        <ows:Abstract>3e Technologies International SOS Server</ows:Abstract>
        <ows:Keywords>
            <ows:Keyword>Weather, ACADA</ows:Keyword>
        </ows:Keywords>
        <ows:ServiceType codeSpace="http://opengeospatial.net">OGC:SOS</ows:ServiceType>
        <ows:ServiceTypeVersion>0.0.31</ows:ServiceTypeVersion>
        <ows:Fees>NONE</ows:Fees>
        <ows:AccessConstraints>NONE</ows:AccessConstraints>
    </ows:ServiceIdentification>
    <ows:ServiceProvider>
        <ows:ProviderName>3e Technologies International</ows:ProviderName>
        <ows:ProviderSite xlink:href="http://www.3eti.com"/>
        <ows:ServiceContact>
            <ows:IndividualName>Doug Fischer</ows:IndividualName>
            <ows:PositionName>Software Engineer</ows:PositionName>
            <ows:ContactInfo>
                <ows:Phone>
                    <ows:Voice>724-459-7498-1003</ows:Voice>
                    <ows:Facsimile>724-459-5563</ows:Facsimile>
                </ows:Phone>
                <ows:Address>
                    <ows:DeliveryPoint>3e Technologies International</ows:DeliveryPoint>
                    <ows:City>Blairsville</ows:City>
                    <ows:AdministrativeArea>PA</ows:AdministrativeArea>
                    <ows:PostalCode>15701</ows:PostalCode>
                    <ows:Country>USA</ows:Country>
                </ows:Address>
            </ows:ContactInfo>
        </ows:ServiceContact>
    </ows:ServiceProvider>
    <ows:OperationsMetadata>
        <ows:Operation name="GetCapabilities">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get xlink:href="http://ren.3eti.net:8080/ogc2/GetCapabilities.ogc?"/>
                    <ows:Post xlink:href="http://ren.3eti.net:8080/ogc2/GetCapabilities.ogc"/>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="service" use="required">
                <ows:Value>SOS</ows:Value>
```

```
            </ows:Parameter>
            <ows:Parameter name="version" use="required">
                <ows:Value>0.0.31</ows:Value>
            </ows:Parameter>
        </ows:Operation>
        <ows:Operation name="GetObservation">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Post xlink:href="http://ren.3eti.net:8080/ogc2/GetObservation.ogc"/>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="observedProperty" use="optional">
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:AirTemperature</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:AverageAirTemperature15Minute</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:AverageWindSpeed15Minute</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirDPM</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGA</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGB</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGD_GF</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirHD</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirMS</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirVX</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:MaximumRelativeHumidity15Minute</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:MaximumWindSpeed15Minute</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:MinimumWindSpeed15Minute</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:RelativeHumidity</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:WindDirection</ows:Value>
                <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="procedure" use="optional">
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-2</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-1</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-3</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-2</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse1</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse2</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-outside-1</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-outside-2</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-outside-1</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-outside-2</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="eventTime" use="optional">
                <ows:Range>
                    <ows:MinimumValue>2005-10-18T19:54:13.000Z</ows:MinimumValue>
                    <ows:MaximumValue>2005-11-01T03:22:45.000Z</ows:MaximumValue>
                </ows:Range>
            </ows:Parameter>
        </ows:Operation>
        <ows:Operation name="InsertObservation">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Post xlink:href="http://ren.3eti.net:8080/ogc2/InsertObservation.ogc"/>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="InsertId">
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-2</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-1</ows:Value>
                <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-3</ows:Value>
```

```
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-outside-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-outside-2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-outside-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-outside-2</ows:Value>
                </ows:Parameter>
                <ows:Parameter name="procedure">
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-1</ows:Value>
                    <ows:Value>rn:ogc:object:feature:Sensor:3eTI:csi-sensor-3</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-outside-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-outside-2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-outside-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-outside-2</ows:Value>
                </ows:Parameter>
                <ows:Parameter name="observedProperty" use="optional">
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:AirTemperature</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:AverageAirTemperature15Minute</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:AverageWindSpeed15Minute</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirDPM</ows:Value>
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGA</ows:Value>
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGB</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGD_GF</ows:Value>
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirHD</ows:Value>
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirMS</ows:Value>
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirVX</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:MaximumRelativeHumidity15Minute</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:MaximumWindSpeed15Minute</ows:Value>

<ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:MinimumWindSpeed15Minute</ows:Value>
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:RelativeHumidity</ows:Value>
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:WindDirection</ows:Value>
                    <ows:Value>urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed</ows:Value>
                </ows:Parameter>
            </ows:Operation>
            <ows:Operation name="DescribeSensor">
                <ows:DCP>
                    <ows:HTTP>
                        <ows:Post xlink:href="http://ren.3eti.net:8080/ogc2/DescribeSensor.ogc"/>
                    </ows:HTTP>
                </ows:DCP>
                <ows:Parameter name="procedure" use="required">
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-sensor-3</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-outside-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:csi-outside-2</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-outside-1</ows:Value>
                    <ows:Value>urn:ogc:object:feature:Sensor:3eTI:acada-outside-2</ows:Value>
                </ows:Parameter>
            </ows:Operation>
```

```
        </ows:OperationsMetadata>
        <!--============================================================-->
        <!--FILTER CAPABILITIES SECTION-->
        <!--============================================================-->
        <ogc:Filter_Capabilities xsi:nil="true"/>
        <!--the Contents part defines all information about the data that is provided at this SOS. To provideleast
rudimentary support for the sensor driven and the observation driven view, a SensorOfferingListwell as a
ObservationOfferingList are provided. See the class diagram at the Twiki for further information.-->
        <Contents>
            <ObservationOfferingList>
                <ObservationOffering gml:id="offering-1">
                    <gml:description>realtime</gml:description>
                    <gml:boundedBy>
                        <gml:Envelope srsName="EPSG:4326">
                            <gml:lowerCorner>39.11111 -77.17695</gml:lowerCorner>
                            <gml:upperCorner>39.11111 -77.17695</gml:upperCorner>
                        </gml:Envelope>
                    </gml:boundedBy>
                    <eventTime>
                        <gml:TimePeriod>
                            <gml:beginPosition>2005-10-18T15:19:39.000Z</gml:beginPosition>
                            <gml:endPosition>2005-10-18T19:53:53.000Z</gml:endPosition>
                        </gml:TimePeriod>
                    </eventTime>
                    <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1"/>
                    <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:csi-sensor-2"/>
                    <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:csi-sensor-3"/>
                    <observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:AverageAirTemperature15Minute"/>
                    <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:AirTemperature"/>
                    <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:RelativeHumidity"/>
                    <observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:MaximumRelativeHumidity15Minute"/>
                    <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed"/>
                    <observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:AverageWindSpeed15Minute"/>
                    <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:WindDirection"/>
                    <featureOfInterest xlink:href="urn:ogc:def:feature:OGC-SWE:3:transient"/>
                    <resultFormat>application/com-xml</resultFormat>
                    <resultModel>swe:DataValueType</resultModel>
                    <resultModel>om:ObservationType</resultModel>
                    <responseMode>inline</responseMode>
                    <responseMode>resultTemplate</responseMode>
                </ObservationOffering>
                <ObservationOffering gml:id="offering-3">
                    <gml:description>realtime</gml:description>
                    <gml:boundedBy>
                        <gml:Envelope srsName="EPSG:4326">
                            <gml:lowerCorner>33.062018 -116.616526</gml:lowerCorner>
                            <gml:upperCorner>33.06351 -116.61463</gml:upperCorner>
                        </gml:Envelope>
                    </gml:boundedBy>
                    <eventTime>
                        <gml:TimePeriod>
                            <gml:beginPosition>2005-10-25T14:44:10.000Z</gml:beginPosition>
                            <gml:endPosition>2005-11-01T03:22:21.000Z</gml:endPosition>
                        </gml:TimePeriod>
                    </eventTime>

                    <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:csi-outside-1"/>
                    <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:csi-outside-2"/>
                    <observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:AverageAirTemperature15Minute"/>
                    <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:AirTemperature"/>
                    <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:RelativeHumidity"/>
                    <observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:MaximumRelativeHumidity15Minute"/>
```

```
                    <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed"/>
                    <observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:AverageWindSpeed15Minute"/>
                    <observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:WindDirection"/>
                    <featureOfInterest xlink:href="urn:ogc:def:feature:OGC-SWE:3:transient"/>
                    <resultFormat>application/com-xml</resultFormat>
                    <resultModel>swe:DataValueType</resultModel>
                    <resultModel>om:ObservationType</resultModel>
                    <responseMode>inline</responseMode>
                    <responseMode>resultTemplate</responseMode>
                </ObservationOffering>
                <ObservationOffering gml:id="offering-2">
                    <gml:description>realtime</gml:description>
                    <gml:boundedBy>
                        <gml:Envelope srsName="EPSG:4326">
                            <gml:lowerCorner>39.11111 -77.17695</gml:lowerCorner>
                            <gml:upperCorner>39.11111 -77.17695</gml:upperCorner>
                        </gml:Envelope>
                    </gml:boundedBy>
                    <eventTime>
                        <gml:TimePeriod>
                            <gml:beginPosition>2005-10-18T19:54:13.000Z</gml:beginPosition>
                            <gml:endPosition>2005-10-18T19:54:13.000Z</gml:endPosition>
                        </gml:TimePeriod>
                    </eventTime>
                <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:acada-sensor-1"/>
                <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:acada-sensor-2"/>
                    <observedProperty>
                        <swe:CompositePhenomenon gml:id="observable.chemicalpresence">
                            <gml:name codeSpace="urn:3eti">chemicalpresence</gml:name>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirDPM"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirMS"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGA"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGB"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGD_GF"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirVX"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirHD"/>
                        </swe:CompositePhenomenon>
                    </observedProperty>
                    <featureOfInterest xlink:href="urn:ogc:def:feature:OGC-SWE:3:transient"/>
                    <resultFormat>application/com-xml</resultFormat>
                    <resultModel>swe:DataValueType</resultModel>
                    <resultModel>om:ObservationType</resultModel>
                    <responseMode>inline</responseMode>
                    <responseMode>resultTemplate</responseMode>
                </ObservationOffering>
                <ObservationOffering gml:id="offering-4">
                    <gml:description>realtime</gml:description>
                    <gml:boundedBy>
                        <gml:Envelope srsName="EPSG:4326">
                            <gml:lowerCorner>33.062018 -116.616526</gml:lowerCorner>
                            <gml:upperCorner>33.06351 -116.61463</gml:upperCorner>
                        </gml:Envelope>
                    </gml:boundedBy>
                    <eventTime>
                        <gml:TimePeriod>
                            <gml:beginPosition>2005-10-25T14:44:10.000Z</gml:beginPosition>
                            <gml:endPosition>2005-11-01T03:22:45.000Z</gml:endPosition>
                        </gml:TimePeriod>
                    </eventTime>
```

```
                <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse1"/>
                <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:acada-sensor-warehouse2"/>
                <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:acada-outside-1"/>
                <procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:acada-outside-2"/>
                    <observedProperty>
                        <swe:CompositePhenomenon gml:id="observable.chemicalpresence">
                            <gml:name codeSpace="urn:3eti">chemicalpresence</gml:name>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirDPM"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirMS"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGA"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGB"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirGD_GF"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirVX"/>
                            <swe:component
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:ChemicalPresenceInAirHD"/>
                        </swe:CompositePhenomenon>
                    </observedProperty>
                    <featureOfInterest xlink:href="urn:ogc:def:feature:OGC-SWE:3:transient"/>
                    <resultFormat>application/com-xml</resultFormat>
                    <resultModel>swe:DataValueType</resultModel>
                    <resultModel>om:ObservationType</resultModel>
                    <responseMode>inline</responseMode>
                    <responseMode>resultTemplate</responseMode>
                </ObservationOffering>
            </ObservationOfferingList>
        </Contents>
</Capabilities>
```

## B.2 SOS GetObservation Response Example

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<om:ObservationCollection xmlns:om="http://www.opengis.net/om" xmlns:gml="http://www.opengis.net/gml"
xmlns:swe="http://www.opengis.net/swe" xmlns:meta="http://www.seegrid.csiro.au/xml/metaLite"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <om:time>
        <gml:TimeInstant>
            <gml:timePosition>2005-10-29T02:53:38.328Z</gml:timePosition>
        </gml:TimeInstant>
    </om:time>
    <om:location xsi:nil="true"/>
    <om:member>
        <om:Observation>
            <om:time>
                <gml:TimeInstant>
                    <gml:timePosition>2005-09-01T12:00:00.000Z</gml:timePosition>
                </gml:TimeInstant>
            </om:time>
            <om:location xlink:href="#location0"/>
            <om:procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1"/>
            <om:observedProperty xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:WindSpeed"/>
            <om:featureOfInterest>
                <om:Station>
                    <om:position>
                        <gml:Point gml:id="location0">
                            <gml:pos srsName="EPSG:4326">39.1235 -77.8623</gml:pos>
                        </gml:Point>
                    </om:position>
                    <om:procedureHosted/>
                </om:Station>
```

```
                </om:featureOfInterest>
                <om:result uom="urn:ogc:def:uom:OGC:1.0.30:speed-ms"
xsi:type="gml:MeasureType">4.2</om:result>
            </om:Observation>
        </om:member>
        <om:member>
            <om:Observation>
                <om:time>
                    <gml:TimeInstant>
                        <gml:timePosition>2005-09-01T12:00:00.000Z</gml:timePosition>
                    </gml:TimeInstant>
                </om:time>
                <om:location xlink:href="#location1"/>
                <om:procedure xlink:href="urn:ogc:object:feature:Sensor:3eTI:csi-sensor-1"/>
                <om:observedProperty
xlink:href="urn:ogc:def:phenomenon:OGC:1.0.30:AverageAirTemperature15Minute"/>
                <om:featureOfInterest>
                    <om:Station>
                        <om:position>
                            <gml:Point gml:id="location1">
                                <gml:pos srsName="EPSG:4326">39.1235 -77.8623</gml:pos>
                            </gml:Point>
                        </om:position>
                        <om:procedureHosted/>
                    </om:Station>
                </om:featureOfInterest>
                <om:result uom="urn:ogc:def:uom:OGC:1.0.30:degC"
xsi:type="gml:MeasureType">24.6</om:result>
            </om:Observation>
        </om:member>
</om:ObservationCollection>
```

# Annex C
# (informative)

# Future Work: Issues to be Resolved

## C.1 Volume and Density of Data Returned to Client

The concept of Observation Offerings allows the SOS to organize observations into groupings that are meant to be "dense" in the sense that a GetObservation request using parameters within the range identified in the offering will almost always return a non-empty collection of observations. The implementation experienced gained during the OWS-3 test bed demonstrated that this was a very useful construct. However, we also found that there is more work to be done to characterize the density of an observation offering. For example, two SOS implementations might advertise an offering that includes weather data from certain sensors over a two week period. In one instance the weather sensors are streaming real-time instantaneous measurements every three seconds. In the other instance the sensors are only reporting time-averaged data once an hour. The same request would generate widely different record counts on the two service instances. The goal for SOS is to allow clients to discover and request observations from SOS instances with no a-priori knowledge so there is a missing piece that would allow a client to determine the density of an offering. Also, in the case of the SOS that reports real-time weather data every three seconds, the client might want to receive data only every hour. The client might accept hourly data that is just an arbitrary sampling of the raw data or might prefer time-averaged data or statistical summaries of the data that would allow it to see outliers such as a high wind speed value that resulted from a tornado that came near the weather station for 5 minutes sometime during the hour.

It may or may not be appropriate for an SOS to provide time-averaging or statistical computations on raw sensor data. Perhaps a new or existing service could handle that type of functionality. The density of requests and offerings must be addressed, however, so that large data volumes are not transmitted unnecessarily due to a lack of information about service offerings. One idea is to utilize the optional GML timeInterval tag with a GML TimePeriod in the eventTime tag of the observation offering to communicate information about the reporting interval for sensors in that offering.

## C.2 Sensor Identifiers

Sensor identifiers are important within SOS and the SWE initiative because some users want to access data in a sensor-centric fashion. These users are already familiar with particular sensors and want to discover what services can return data from those sensors. Because of the wide variety of scenarios in which SWE technologies will be used and because of the diversity of mechanisms for assigning identifiers to sensors, it is not feasible to define a new globally unique system for assigning identifiers to sensors. The assignment of identifiers to sensors is inherently decentralized just as is the definition of phenomena discussed in a previous section.

Every sensor identifier is associated with some authority that controls the assignment of identifiers. Each of these authorities has its own mechanism for generating and assigning identifiers to sensors. In some cases the identifier may come from the serial number of the sensor that was assigned by the manufacturer. In other cases the identifier may be an arbitrary unique value. In some cases the namespace used for identifiers will be flat and in others it will follow some kind of hierarchy.

One approach for representing sensor identifiers that was considered during the OWS-3 test bed was to define an extension to OGC document 05-010 - URNs of Definitions in OGC Namespace. This extension would add a category label, resource group, resource type, and resource subtype to the existing OGC URN format. The general form would be

```
urn:ogc:category.label:resource.group:resource.type-
resource.subtype:authority:version:code
```

We could add an additional category.label "object" and resource.group "feature" (or maybe collapse these to just "feature") and use the resource.type for the featureType, in this case with the value "Sensor", and then we would need to define the parameters. Following the pattern of the "def" profile, the URN values might look something like this:

urn:ogc:object:feature:Sensor:NASA:version:code

urn:ogc:object:feature:Sensor-DewPointTemperature:3eTI:234567ABCD

urn:ogc:object:feature:Sensor-Elevation:3eTI:987etgh345mnbvx

urn:ogc:object:feature:Sensor:IRIS:WS01:12345

urn:ogc:object:feature:Sensor:IRIS:MOBIL01:12345

urn:ogc:object:feature:Sensor:IRIS:IRIS01_CAMERA:12345

The sensor.subtype would be optional, but if present would be taken from the OGC canonical phenomenon dictionary. The tail of the identifier, after the authority (e.g., IRIS, NASA, 3eTI), is defined by the provider and is usually opaque to the user. There could be additional colon-delimited fields if the authority uses a hierarchical namespace. A list of valid authorities would be maintained in an OGC dictionary.

Another issue regarding sensor identifiers is whether the identifier refers to the sensor or the output of the sensor. Some sensor systems have more than one output. For example the R.M. Young model 05103-5 Wind Monitor reports both wind speed and wind direction. This sensor has two analog outputs: one for each measured phenomena. An identifier could be assigned to the sensor as a whole, to each of the outputs, or to both the sensor and the outputs. In a subsequent version of this specification we should define whether a reference to the identifier for this sensor in a capabilities document or a GetObservation request implies a reference to both sensor outputs.

### C.3 Aggregate and Summary Phenomena

Aggregate and summary phenomena are commonly used in sensor applications. An example of a sensor that measures an aggregate phenomenon is one that measures radiance by measuring several discrete wavelengths that are used to construct a radiance band. Another example is an earthquake location sensor that reports epicentre, depth, and origin time. The observations that capture these aggregate phenomena could be broken into their constituent parts. However, the application domains in which they are used generally require that they be reported as if they were base phenomena. It might not make sense to report each wavelength value separately or to report only two of the earthquake location components, for example.

Summary phenomena are commonly used in meteorology. Weather data often includes things like 15 minute averaged temperature and one hour averaged temperature in addition to instantaneous temperature. In most metrological applications the instantaneous values are not useful because the models require time-averaged values. Data loggers on weather stations frequently calculate the summary data points internally and report them as if they were raw sensor data. In this case the raw values of temperature, wind speed, humidity, etc. are probably not even recoverable since they are only stored temporarily in the data logger for computing the summary data. Theoretically these data could be split into their constituent values. However, just as for the aggregate phenomena discussed above, the application domain requires that the summary data be considered as base phenomena.

We must be careful with how we characterize aggregate and summary observations in the context of the SWE initiative. In general we can characterize these observations as capturing aggregate or summary phenomena directly or we can characterize them as observations that capture base phenomena and which can be transformed through a SensorML process chain or TML transducer characterizations to obtain aggregate or summary observations. In the former case we would explicitly define aggregate or summary phenomena in a dictionary and advertise sensors that generated observations capturing those phenomena. The Observation Offering would include fewer sensor systems and more phenomena (i.e., observedProperty tags). In the latter case we would advertise different sensor systems (i.e., procedure tags) that are actually different outputs from the same sensor which were derived from raw sensor data using a SensorML or TML description. The Observation Offering would include more sensor systems and fewer phenomena.

Flexibility in how observations can be represented is a two-edged sword. Flexibility allows different sensor communities and service providers to advertise capabilities in a way that makes sense for them. However, it also means that essentially the same offering can be encoded in multiple ways. This can lead to interoperability problems because it would be difficult for a client to determine which service offerings are equivalent.

In a future version of the specification we should provide guidance for how to handle aggregate and summary observations.

# Bibliography

[1] ISO 31 (all parts), Quantities and units.

[2] IEC 60027 (all parts), Letter symbols to be used in electrical technology.

[3] ISO 1000, SI units and recommendations for the use of their multiples and of certain other units.

[4] Template for OWS Implementation Specifications, OGC document 05-009r1.

[5] Observations and Measurements, OGC document 05-087.

[6] Sensor Model Language (SensorML) for In-situ and Remote Sensors, OGC document 05-086.

[7] Transducer Markup Language (TML), OGC document 05-085.

[8] Sensor Planning Service (SPS), OGC document 05-089.

[9] Web Notification Service, OGC document 03-008r1.

[10] Sensor Alert Service, OGC document 05-098.

[11] Sensor Web Enablement Architecture, OGC document 05-090.

[12] OpenGIS® Web Services Common Specification, OGC document 05-008

 