Open Geospatial Consortium

Approval Date: 2012-04-16

Publication Date: 2012-04-20

Reference number of this document: OGC 12-006

OGC name of this OGC® project document: **http://www.opengis.net/doc/IS/SOS/2.0**

Version: 2.0

Category: OpenGIS® Implementation Standard

Editors: Arne Bröring (52°North)
Christoph Stasch (IfGI)
Johannes Echterhoff (iGSI)

OGC® Sensor Observation Service Interface Standard

Warning

This document is an OGC Member approved international standard. This document is available on a royalty free, non-discriminatory basis. 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: OGC Implementation Standard
Document subtype: Interface
Document stage: Approved
Document language: English

License Agreement

Contents

Page

Figures

Page

Tables

Page

i. Abstract

The SOS standard is applicable to use cases in which sensor data needs to be managed in an interoperable way. This standard defines a Web service interface which allows querying observations, sensor metadata, as well as representations of observed features. Further, this standard defines means to register new sensors and to remove existing ones. Also, it defines operations to insert new sensor observations. This standard defines this functionality in a binding independent way; two bindings are specified in this document: a KVP binding and a SOAP binding.

ii. Preface

This standard defines the interface of the Sensor Observation Service, hereinafter abbreviated as *SOS*. The SOS is one of a family of OGC standards that make up the OGC Sensor Web Enablement framework [OGC 06-021r4], hereinafter know as *SWE*. The functionality of the SOS within SWE is to provide standardized access to measured sensor observations as well as sensor descriptions. To encode observations, the Observations & Measurements (O&M) standard is used. To encode sensor descriptions, the Sensor Model Language (SensorML) is used. The SOS 2.0 builds on the previous SOS 1.0 efforts.

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

If you choose to submit suggested changes or enhancements, please use the OGC online change request application:

http://portal.opengeospatial.org/public_ogc/change_request.php

The previous versions, including version 1.0 [OGC 06-009r6], are deprecated by this standard.

A tutorial describing the utilization of this standard through examples and explanatory text can be found at: http://www.ogcnetwork.net/SOS_2_0/tutorial

iii. Changes to Previous Version

This SOS 2.0 standard improves SOS 1.0 [OGC 06-009r6] by incorporating the following major changes:

- Reduced complexity through clearer structuring in *Core*, *Extensions* and *Profiles*.
- By applying the modular specifications policy [OGC 08-131r3], the implementation of the standard is facilitated. Clear listings of requirements and derived conformance tests also provide better support for CITE testing.
- Increased interoperability through minimum requirements:
    - Certain operators and operands for temporal and spatial filters of *GetObservation*, *GetFeatureOfInterest* and *GetResult* have to be supported as a minimum by each SOS server which supports temporal or spatial filters.
    - The “Spatial Filtering Profile” considerably improves the spatial filter of observations in the *GetObservation* and *GetResult* operations.
- KVP binding: SOS 2.0 defines a simple key-value-pair binding for HTTP GET.
- SOAP binding: SOS 2.0 defines a SOAP binding for all its operations.

- Alignment with other SWE standards by relying on and reusing data types defined in the SWE Service Model [OGC 09-001].

- Improved concept for observation offerings: SOS 2.0 defines that each observation offering is limited to be associated with only a single procedure. This solves the ambiguity of SOS 1.0 of grouping observations from multiple procedures to offerings.

- Improved handling of information on hosted observation offerings through application of *property inheritance mechanism* (defined by OGC 09-001) to decrease the amount of redundant information.

- The Capabilities document of SOS 2.0 lists *related features* instead of all *features of interest*. The related features are selected by the service provider and serve discovery purposes.

- The parameterization of the *GetFeatureOfInterest* operation is extended. Procedure, observed property and spatial operators are now allowed as filter parameters

- The handling of observation results is revised and extended. A new operation *InsertResult* now also allows the efficient upload of plain result values.

- The operations *DescribeObservationType*, *DescribeResultType*, and *DescribeFeatureType* have been removed since the types of features, observations and results are identified by URIs in SOS 2.0 and these URIs are known and are resolvable.

iv. Document terms and definitions

This document uses the standard terms defined in Subclause 5.3 of [OGC 06-121r3], 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 standard.

v. Submitting organizations

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

a) 52° North Initiative for Geospatial Open Source Software GmbH

b) University of Muenster – Institute for Geoinformatics (IfGI)

c) International Geospatial Services Institute GmbH (iGSI)

vi. Document contributor contact points

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

CONTACT	COMPANY	EMAIL
Arne Bröring	52°North	Broering<at>52north.org
Christoph Stasch	IfGI, University of Münster	Staschc<at>uni-muenster.de
Johannes Echterhoff	iGSI	johannes.echterhoff<at>igsi.eu
Peter Taylor	CSIRO	peter.taylor<at>csiro.au
Luis Bermudez	OGC	Lbermudez<at>opengeospatial.org
Alex Robin	Spot Image	alexandre.robin<at>spotimage.fr
Jeff de La Beaujardière	NOAA	jeff.deLaBeaujardiere<at>noaa.gov
Tom Ingold	SAIC	thomas.b.ingold<at>saic.com
Carsten Hollmann	52°North	c.hollmann<at>52north.org
Simon Cox	CSIRO	simon.cox<at>csiro.au

vii. Revision history

Date	Release	Editor	Primary clauses modified	Description
2012-01-06	1.0.0	Arne Bröring, Christoph Stasch, Johannes Echterhoff	-	final draft for adoption vote

viii. Changes to the OGC Abstract Specification

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

ix. Future work

This version of the SOS interface defines a SOAP binding for all specified operations as well as a KVP binding for the core operations, the *GetFeatureOfInterest* operation, as well as the result handling operations. Future versions or extensions of this standard may add a RESTful binding similar to what has been defined by Janowicz et al. (2011)[1].

[1] Krzysztof Janowicz, Arne Bröring, Christoph Stasch, Sven Schade, Thomas Everding & Alejandro Llaves (2011): *A RESTful proxy and data model for linked sensor data*. International Journal of Digital Earth. doi: 10.1080/17538947.2011.614698

x. Acknowledgements

This work has been financially supported by the ERDF program for NRW (contract number N 114/2008), as well as the EC funded projects “Earth Observation and Environmental Modelling for the Mitigation of Health Risks—EO2Heaven” (contract number 244100), “Emergency Support System—ESS” (contract number 217951), and “Uncertainty Enabled Model Web—UncertWeb“ (contract number 248488).

Foreword

This version of the SOS standard deprecates OGC standard [OGC 06-009r6].

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

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

Introduction

The Sensor Observation Service (SOS) provides a standardized interface for managing and retrieving metadata and observations from heterogeneous sensor systems. Sensor systems contribute the largest part of geospatial data used in geospatial systems today. Sensor systems include for example in-situ sensors (e.g. river gauges), moving sensor platforms (e.g. satellites or unmanned aerial vehicles) or networks of static sensors (e.g. seismic arrays). 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.

The SOS is part of the OGC Sensor Web Enablement (SWE) framework of standards [OGC 06-021r4]. The SWE activity aims at providing interfaces and protocols for enabling “Sensor Webs” through which applications and services are able to access sensors of all types. Sensor Webs can be accessed over networks such as the Internet with the same standard technologies and protocols that enable the Web.

SOS 2.0 relies on the OGC Observations and Measurements (O&M) [OGC Abstract Specification Topic 20 10-004r3/ISO 19156:2010] standard to encode data gathered by sensors.

OGC® Sensor Observation Service Implementation Standard

1 Scope

This OpenGIS® document specifies the interface standard of the Sensor Observation Service 2.0 (SOS).

This standard specifies how observations, sensor descriptions, as well as computational representations of observed features are accessed in an interoperable and standardized way. As such, this OGC document is applicable to use cases in which sensor data needs to be managed in an interoperable way.

Further, this standard defines means to register new sensors and to remove existing ones.

This standard also defines operations to insert new observations as well as to efficiently insert and retrieve observation result values.

This standard specifies SOS functionality in a binding independent way. Mappings for two bindings are specified: a KVP binding and a SOAP binding.

2 Compliance

The standardization targets of this standard are SOS 2.0 Web Server implementations.

The following tables list the requirements classes defined by this standard.

Annex A lists the conformance tests which shall be exercised on any software artifact claiming to adhere to the SOS 2.0 standard.

Table 1: Main Requirements class

Requirements class name	Requirements class identifier	Operation or behavior
Core	http://www.opengis.net/spec/SOS/2.0/req/core	The server implements the following operations: *GetCapabilities*, *DescribeSensor*, and *GetObservation*.

Table 2: Requirements classes in Transactional Extension

Requirements class name	Requirements class identifier	Operation or behavior
Insertion Capabilities	http://www.opengis.net/spec/SOS/2.0/req/insertionCap	The server provides metadata on its capabilities regarding the insertion of new sensors/observations.
Sensor Insertion	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion	The server implements the InsertSensor operation.
Sensor Deletion	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion	The server implements the DeleteSensor operation.
Observation Insertion	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion	The server implements the InsertObservation operation.

Table 3: Requirements classes in Result Handling Extension

Requirements class name	Requirements class identifier	Operation or behavior
Result Insertion	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion	The server implements the operations InsertResult and InsertResultTemplate.
Result Retrieval	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval	The server implements the operations GetResult and GetResultTemplate.

Table 4: Requirements classes in Enhanced Operations Extension

Requirements class name	Requirements class identifier	Operation or behavior
Observation Retrieval By Id	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval	The server implements the GetObservationById operation.
FeatureOfInterest Retrieval	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval	The server implements the GetFeatureOfInterest operation.

Table 5: Requirements classes in Spatial Filtering Profile

Requirements class name	Requirements class identifier	Operation or behavior
Spatial Filtering Profile	http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile	The server implements the Spatial Filtering Profile to improve spatial filtering of observations in GetObservation and GetResult (if supported) operations.

Table 6: Requirements classes in Binding Extension

Requirements class name	Requirements class identifier	Operation or behavior
XML Encoding	http://www.opengis.net/spec/SOS/2.0/req/xml	The server encodes the data types from the conceptual model in XML as defined by this standard.
SOAP Binding	http://www.opengis.net/spec/SOS/2.0/req/soap	The server wraps XML encoded requests and results within SOAP Envelopes for the operations it supports.
Core KVP Binding	http://www.opengis.net/spec/SOS/2.0/req/kvp-core	The server implements the key-value pair encoding for the operations *GetCapabilities*, *DescribeSensor*, and *GetObservation*.
GetFeatureOfInterest KVP Binding	http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval	The server implements the key-value pair encoding for the operation *GetFeatureOfInterest*.
Result Retrieval KVP Binding	http://www.opengis.net/spec/SOS/2.0/req/kvp-result	The server implements the key-value pair encoding for the *GetResult* and *GetResultTemplate* operations.

3 Normative references

This *OGC SOS 2.0* standard consists of the present document and an XML Schema. The complete standard is identified by OGC URI:

http://www.opengis.net/spec/SOS/2.0

the document has OGC URI

http://www.opengis.net/doc/IS/SOS/2.0.

The complete standard is available for download from http://www.opengeospatial.org/standards/sos. In addition, the XML Schema is posted online at http://schemas.opengis.net/sos/2.0 as part of the OGC schema repository. In the event of a discrepancy between bundled and schema repository versions of the XML Schema files, the schema repository is considered authoritative.

The following normative documents contain provisions that, 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.

IETF RFC 2396, Uniform Resource Identifiers (URI): Generic Syntax

ISO 8601:2000, Data elements and interchange formats — Information interchange — Representation of dates and times

ISO 19107:2003, Geographic information — Spatial schema

ISO 19108:2002, Geographic information — Temporal schema

ISO 19136:2007, Geographic information — Geography Markup Language (GML)[2]

ISO 19143:2010, Geographic information — Filter Encoding[3]

ISO 19156:2011, Geographic information — Observations and Measurements[4]

OGC Abstract Specification Topic 20 - Geographic Information: Observations and Measurements. Version 2.0.0. OGC document 10-004r3. http://www.opengis.net/doc/om/2.0

OGC® Policy Standard, The Specification Model - A Standard for Modular specifications, Version 1.0.0, OGC document 08-131r3. http://www.opengis.net/doc/modular-spec/1.0

OpenGIS® Encoding Standard, Geography Markup Language, Version 3.2.1, OGC document 07-036. http://www.opengis.net/doc/gml/3.2.1

OpenGIS® Encoding Standard, SWE Common Data Model, Version 2.0, OGC document 08-094. http://www.opengis.net/doc/swe/2.0

OpenGIS® Encoding Standard, SensorML Version 1.0 Schema - Corrigendum 1, Version 1.0.1, OGC document 07-022r2. http://www.opengis.net/doc/sensorml/1.0

OpenGIS® Implementation Standard, Observations and Measurements - XML Implementation, Version 2.0, OGC document 10-025. http://www.opengis.net/doc/omxml/2.0

OpenGIS® Implementation Standard, SWE Service Model, Version 2.0, OGC document 09-001. http://www.opengis.net/doc/swes/2.0

OpenGIS® Implementation Standard, OGC Web Services Common, Version 1.1.0, OGC document 06-121r3. http://www.opengis.net/doc/ows/1.1

OpenGIS® Best Practice, OGC® Sensor Web Enablement Architecture, Version 0.4, OGC document 06-021r4. http://www.opengis.net/doc/swe-architecture/0.4

4 Terms and definitions

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

4.1 Feature

Abstraction of real-world phenomena [OGC 10-004r3/ISO 19156].

---

[2] *Equivalent to:* OpenGIS® Geography Markup Language (GML) Encoding Standard, Geography Markup Language, Version 3.2.1, OGC document 07-036. http://www.opengis.net/doc/gml/3.2.1

[3] *Equivalent to*: OpenGIS Filter Encoding 2.0 Encoding Standard. Version 2.0.0. OGC document 09-026r1.

[4] *Equivalent to*: OGC Abstract Specification Topic 20 - Geographic Information: Observations and Measurements. Version 2.0.0. OGC document 10-004r3. http://www.opengis.net/doc/om/2.0

4.2 Measurement

Set of operations having the object of determining the value of a quantity [OGC 10-004r3/ISO 19156].

4.3 Observed Property

Facet or attribute of an object referenced by a name [OGC 10-004r3/ISO 19156] which is observed by a procedure.

4.4 Observation

Act of observing a property [OGC 10-004r3/ISO 19156].

4.5 Observation Offering

An Observation Offering groups collections of observations produced by one procedure, e.g., a sensor system, and lists the basic metadata for the associated observations including the observed properties of the observations.

4.6 Procedure

Method, algorithm, instrument, sensor, or system of these which may be used in making an observation. [OGC 10-004r3/ISO 19156]

NOTE: As the definition of *procedure* states, this standard uses that term as a generalization of, for example, the terms *sensor* and *sensor system*, but also for simulations or other calculations that may produce observations.

4.7 Sensor

Entity that provides information about an observed property as its output. A sensor uses a combination of physical, chemical or biological means in order to estimate the underlying observed property. At the end of the measuring chain electronic devices produce signals to be processed. [OGC 06-021r4]

4.8 Sensor System

System whose components are sensors. A sensor system as a whole may itself be referred to as a sensor with an own management and sensor output interface. In addition, the components of a sensor system are individually addressable. [OGC 06-021r4]

5 Conventions

5.1 Abbreviated terms

Most of the abbreviated terms listed in Subclause 5.1 of the OWS Common Implementation Specification [OGC 06-121r3] apply to this document, plus the following abbreviated terms.

GML	Geography Markup Language
ISO	International Organization for Standardization

OCL	Object Constraint Language
OGC	Open Geospatial Consortium
OWS	OGC Web Services
O&M	Observations and Measurements
SensorML	Sensor Model Language
SOS	Sensor Observation Service
SWE	Sensor Web Enablement
UML	Unified Modeling Language
XML	eXtensible Markup Language

5.2 UML notation

Diagrams that appear in this standard are presented using the Unified Modeling Language (UML) static structure diagram, as described in Subclause 5.2 of [OGC 06-121r3].

NOTE Packages and data types from foreign namespaces are shown with grey background.

5.3 Platform-neutral and platform-specific standards

For compliance with Clause 10 of [OGC Topic 12] and [ISO 19119], this standard follows the pattern defined in Subclause 5.4 of [OGC 06-121r3]. That is, model elements are specified in platform-neutral fashion first, using tables that serve as data dictionaries for the UML model (see Subclause 5.4 of this document). Platform-specific encodings of these model elements are provided in separate clauses or documents.

The XML Schema encoding has automatically been generated using the mapping rules defined in [OGC 09-001].

This document specifies a platform-specific encoding for a SOAP operation binding and a KVP binding over HTTP. The model as well as its XML Schema encoding (and other data) can also be used to create other bindings such as a REST(ful) or POX (Plain Old XML) over HTTP (using plain XML).

5.4 Data dictionary tables

The UML model data dictionary is specified herein in a series of tables. The contents of the columns in these tables are described in Table 1 of [OGC 06-121r3].

5.5 Classes imported from other specifications with predefined XML encoding

This standard uses an automatic mapping approach from the UML model to the XML Schema encoding. The approach is described in [OGC 09-001]. This standard uses types defined by other standards. For the mapping to XML Schema, the implementation instructions listed in Table D.2 of [OGC 07-036] are used together with the instructions listed in Table 7 to Table 11 in this standard.

Some of the properties defined in the conceptual model of this standard which point to objects rather than directly containing them are encoded as described in Subclause 24.2.4.11 of [OGC 09-001]. An XML Schema implementation for these types is therefore not needed in this section.

For an explanation of the table columns, see Subclause D.2.1 of [OGC 07-036].

Table 7: XML Schema implementation of types from the SWE Service Model [OGC 09-001]

UML class	object element	type	property type
AbstractContents	swes:AbstractContents	swes:AbstractContentsType	swes:AbstractContentsPropertyType
AbstractOffering	swes:AbstractOffering	swes:AbstractOfferingType	swes:AbstractOfferingPropertyType
ExtensibleRequest	swes:ExtensibleRequest	swes:ExtensibleRequestType	swes:ExtensibleRequestPropertyType
ExtensibleResponse	swes:ExtensibleResponse	swes:ExtensibleResponseType	swes:ExtensibleResponsePropertyType
InsertionMetadata	swes:InsertionMetadata	swes:InsertionMetadataType	swes:InsertionMetadataPropertyType

Table 8: Implementation of types from OWS Common [OGC 06-121r3]

UML class	object element	type	property type
GetCapabilities	-	ows:GetCapabilitiesType	-
OWSServiceMetadata	-	ows:CapabilitiesBaseType	-

Table 9: Implementation of types from SWE Common Data Model [OGC 08-094]

UML class	object element	type	property type
AbstractDataComponent	swe:AbstractDataComponent	swe:AbstractDataComponentType	swe:AbstractDataComponentPropertyType
AbstractEncoding	swe:AbstractEncoding	swe:AbstractEncodingType	swe:AbstractEncodingPropertyType

Table 10: Implementation of types from OGC 09-026r1/ISO 19143 (Filter Encoding Specification)

UML class	object element	type	property type
Filter_Capabilities	fes:Filter_Capabilities	-	-
SpatialOperator	fes:spatialOps	fes:SpatialOpsType	-
TemporalOperator	fes:temporalOps	fes:TemporalOpsType	-

Table 11: Implementation of types from OGC Abstract Specification Topic 20/ISO 19156

UML class	object element	type	property type
GFI_Feature	gml:AbstractFeature	gml:AbstractFeatureType	gml:AbstractFeaturePropertyType

5.6 Namespace Conventions

This standard uses a number of namespace prefixes throughout; they are listed in Table 12. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

Table 12: Prefixes and Namespaces used in this standard

Prefix	Namespace
fes	http://www.opengis.net/fes/2.0
gml	http://www.opengis.net/gml/3.2
om	http://www.opengis.net/om/2.0
ows	http://www.opengis.net/ows/1.1
soap11	http://schemas.xmlsoap.org/soap/
soap12	http://www.w3.org/2003/05/soap-envelope
sos	http://www.opengis.net/sos/2.0
swe	http://www.opengis.net/swe/2.0
swes	http://www.opengis.net/swes/2.0
wsa	http://www.w3.org/2005/08/addressing
xs	http://www.w3.org/2001/XMLSchema

6 Observation Model Overview

SOS is primarily designed to provide access to observations. The model for Observations and Measurements (O&M) is defined in the O&M standard [OGC 10-004r3/ISO 19156]. An XML encoding of this conceptual model is defined in the OGC O&M standard [OGC 10-025r1].

The basic Observation model is depicted in Figure 1.

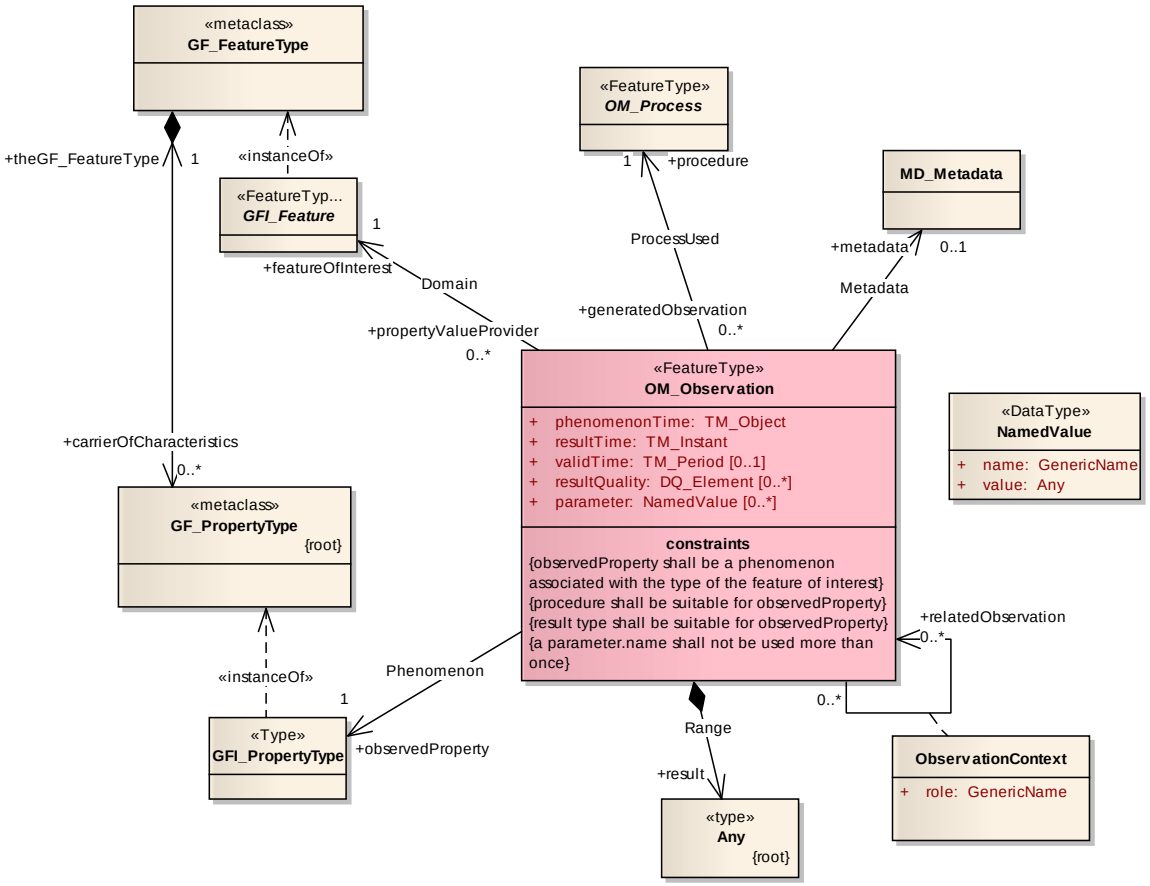


Figure 1: O&M Basic Observation Model

An Observation provides a *result* whose value is an estimate of a property of the observation target, the *feature of interest*; i.e. an observation is a property-value-provider for the feature of interest. An instance of an Observation is classified by its *phenomenonTime*, *featureOfInterest*, *observedProperty*, and the *procedure* used. The procedure is usually a sensor but can also be for example a computation or post-processing step. More detailed information about the observation data model can be found in [OGC 10-004r3/ISO 19156].

7 SOS Model Overview

As shown in Figure 2, the *Core* of the SOS 2.0 defined in Clause 8 builds up on certain specifications such as the OWS Common 1.1.0 [OGC 06-121r3], SWE Service Model 2.0 [OGC

09-001] and Filter Encoding [ISO 19143:2010] to define the service operations and O&M 2.0 [ISO 19156:2010] is used as a mandatory response format for observations.

Based on the Core, extensions can be defined to add further functionality. This document contains the following extensions, the *Transactional Extension* (Clause 10), *Result Handling Extension* (Clause 11), the *Enhanced Operations Extension* (Clause 8.4), and the *Binding Extension* (Clause 13). Further, this document contains the profile *Spatial Filtering Profile* (Clause 12).

Future extensions may specify additional functionality. Future profiles, in particular for certain domains (e.g. hydrology or oceanography) may further restrict the SOS to increase and facilitate interoperability in those communities.

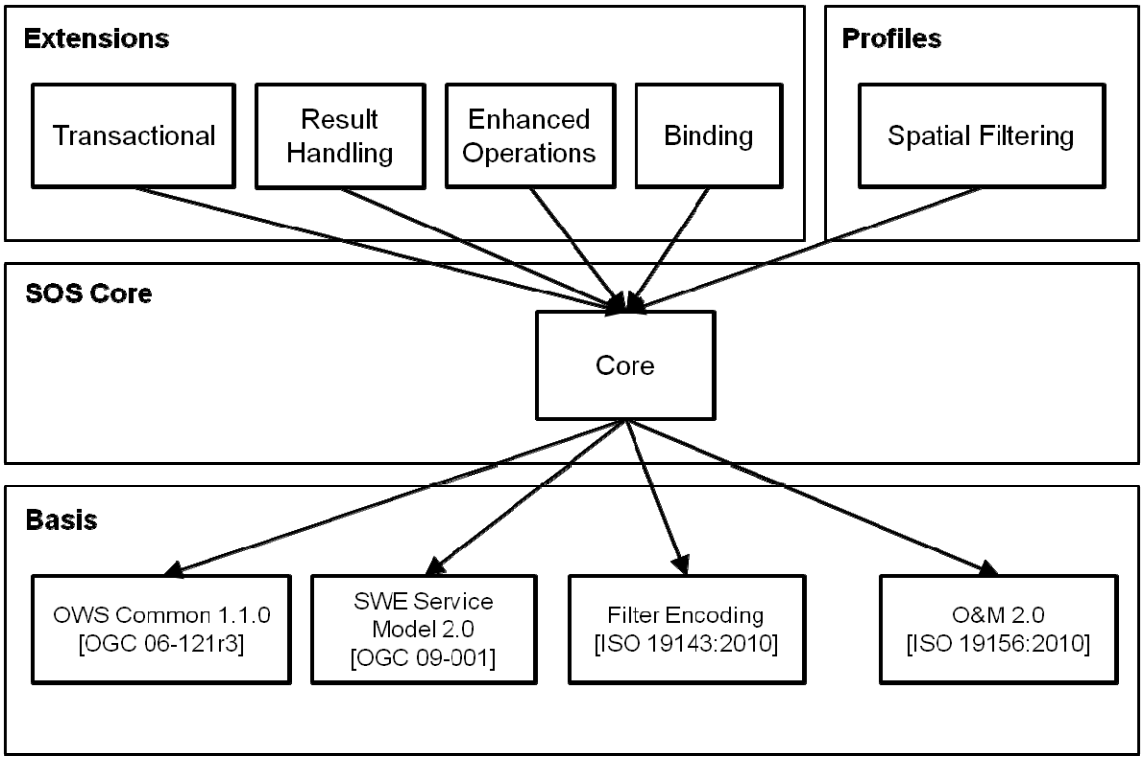


Figure 2: ***Informative:*** **Dependencies between SOS core, extensions and OGC specification basis**

The SOS Core requirements class defines three operations:

GetCapabilities - provides access to metadata and detailed information about the operations available by an SOS server.

DescribeSensor - enables querying of metadata about the sensors and sensor systems available by an SOS server.

GetObservation - provides access to observations by allowing spatial, temporal and thematic filtering.

Further, this document defines extensions of the SOS core which specify additional operations as listed below.

NOTE: The "extension" concept is an informative means used to structure this specification document. Extensions contain the formally defined requirements classes.

Enhanced Operations Extension:

GetObservationByID **-** provides access to observations from an SOS by passing only the ID of an observation.

GetFeatureOfInterest **-** provides direct access to the features of interest for which the SOS offers observations.

Transactional Extension:

InsertSensor – allows registration of new sensors at the SOS.

DeleteSensor – allows the deletion of registered sensors and all their associated observations.

InsertObservation **–** allows the insertion of observations in an SOS server.

Result Handling Extension:

InsertResult **-** allows the insertion of observation results in an SOS server. Before inserting, it is necessary that a template with observation metadata exists in the server.

InsertResultTemplate **-** allows the insertion of an observation template, containing the observation metadata and structure of the results. This operation is necessary for later insertion of observation results.

GetResultTemplate **-** provides access to a template containing the structure of results returned by later invocation of the GetResult operation.

GetResult **-** provides access to an observation result without the observation metadata and without the information about the structure of the results.

The following sections describe the interaction with the SOS to retrieve and insert observations, and to retrieve and insert results (no observation metadata). The operations to support these interactions are defined in the Core and extensions of this document.

7.1 Workflow of Observation Retrieval

Figure 3 and the following descriptions illustrate the sequence of operation calls to obtain and interpret observations from an SOS.

1. To obtain an up-to date listing of available data a client issues a *GetCapabilities* request to the server.

2. It may then issue a *DescribeSensor* request to find out further details about particular procedures associated with the SOS.

3. The client may also call the *GetFeatureOfInterest* operation to get the detailed description of a particular feature, or to get a list of all features of interest for specified spatial filters, observed properties or procedures. The information returned by the *GetFeatureOfInterest* operation may be used by the client to choose appropriate parameters for the *GetObservation* call.

4. The client issues the *GetObservation* request and retrieves the observations.

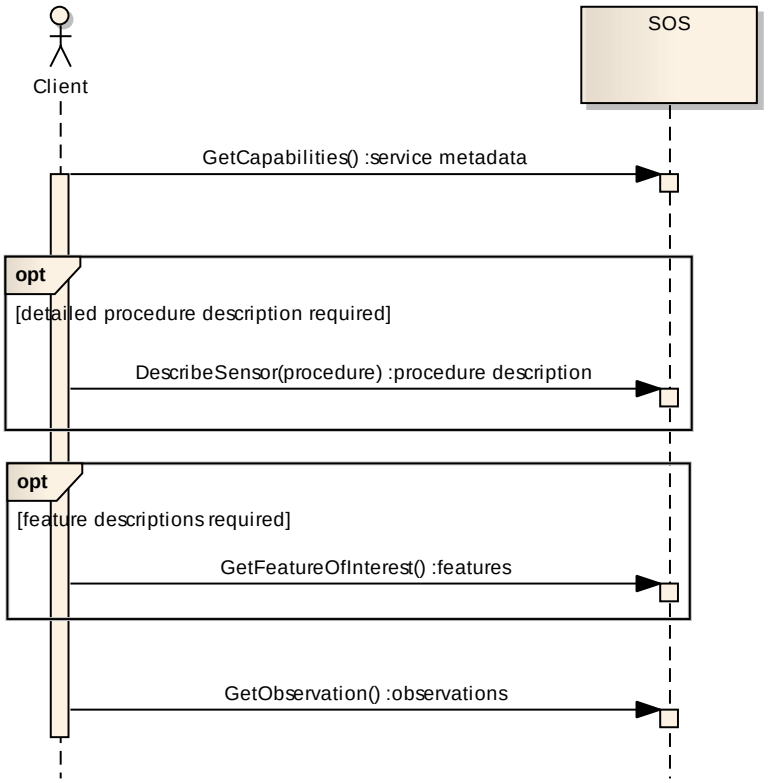
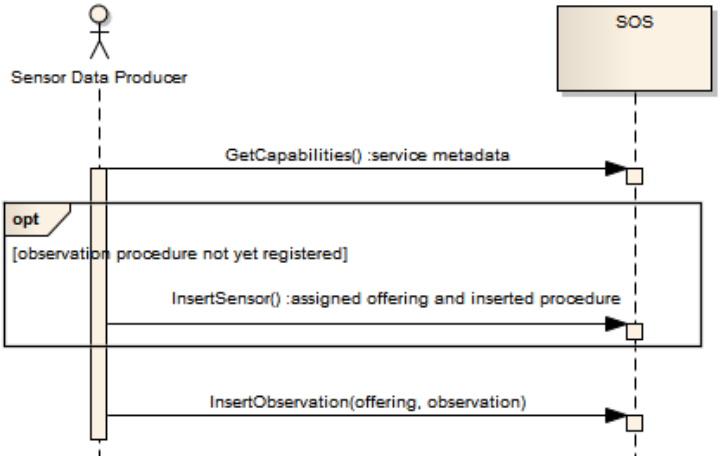


Figure 3: Operation sequence for observation retrieval

7.2 Workflow of Observation Insertion



As illustrated in Figure 4, a sensor data producer first requests the service metadata before new observations can be inserted into an SOS.

1. The service metadata is retrieved by invoking the *GetCapabilities* operation (Subclause 8.1). The returned Capabilities document contains, within the *Contents* section (Subclause 8.1.2.2), the procedures which are registered at the SOS. The *InsertionCapabilities* section (Clause10.1.1) lists the observation types and result types which are supported by the SOS for insertion.

2. If it has not been registered at the SOS, the *InsertSensor* operation (Subclause 10.2.1) is called to insert the procedure description of the observation procedure. The SOS returns the identifier of the *ObservationOffering* to which the procedure has been assigned as well as the identifier[5] which has been assigned for the procedure itself.

3. Finally, the observation can be inserted using the *InsertObservation* operation, by specifying the *ObservationOffering* to which it shall be uploaded.

[5] See Annex 15 for a detailed discussion of identifier handling.

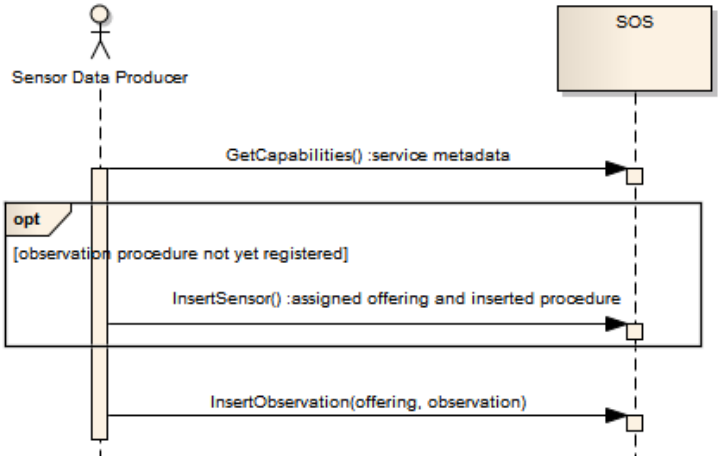


Figure 4: Operation sequence for observation insertion

7.3 Workflow of Result Insertion

An SOS may support the *Result Insertion* requirements class. It contains two operations, *InsertResultTemplate* (Subclause 11.1.1) and *InsertResult* (Subclause 11.1.2), which allow inserting sensor results into an SOS without the need to repeatedly transmit the complete set of observation metadata. The operations can be used, if the metadata contained in the produced observations remain the same. This is useful if the communication bandwidth and processing power of the sensor data producer is limited. Figure 5 depicts the sequence of operation calls for the insertion of results into an SOS.

1. The service metadata is retrieved by invoking the *GetCapabilities* operation (Subclause 8.1). The returned Capabilities document contains, within the *Contents* section (Subclause 8.1.2.2), the procedures which are registered at the SOS. The *InsertionCapabilities* section (Clause10.1.1) lists the observation types and result types which are supported by the SOS for insertion.

2. If it has not been registered at the SOS, the *InsertSensor* operation (Subclause 10.2.1) is called to insert the procedure description of the observation procedure. The SOS returns the *ObservationOffering* to which the procedure has been assigned.

3. A result template is sent to the SOS by using the *InsertResultTemplate* operation. This result template contains observation metadata for the results, the *ObservationOffering* to which the data shall be uploaded, and a structural description of the results.

4. The sensor data producer inserts the results without sending the whole observation metadata and by referencing the inserted result template.

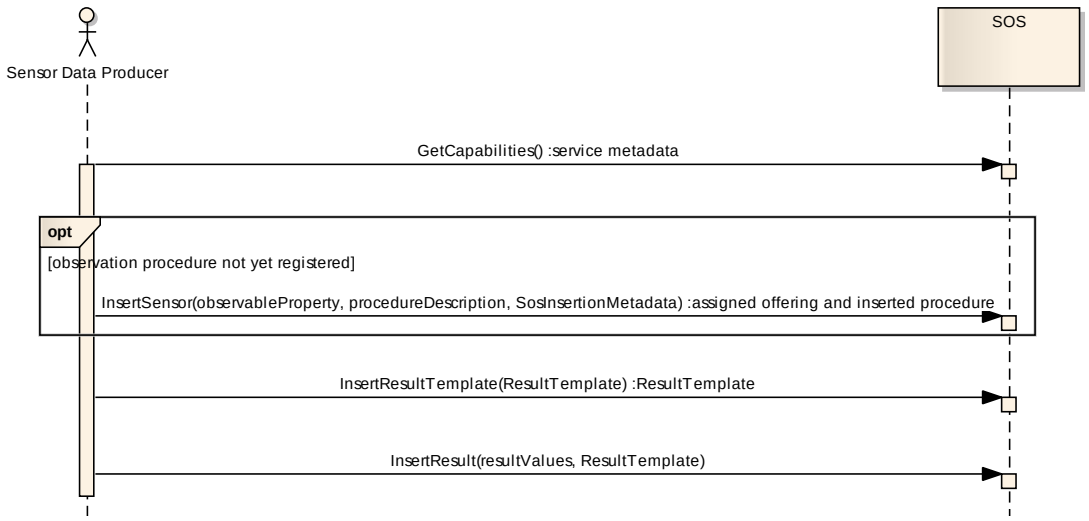


Figure 5: Sequence of operations for insertion of results

7.4 Workflow of Result Retrieval

An SOS may support the *Result Retrieval* requirements class. It contains two operations, *GetResultTemplate* (Subclause 11.2.1) and *GetResult* (Subclause 11.2.2), for the retrieval of observation results without the complete set of observation metadata. The purpose of the Result Retrieval requirements class is to allow clients to repeatedly obtain sensor data without having to send and receive requests and responses that largely contain the same data except for a new timestamp. A client can repeatedly request sensor data from one or more sensors on a recurring basis. This is in particular useful in scenarios with restricted bandwidth or clients with restricted processing power. Figure 6 and the following descriptions illustrate the operation sequence for result retrieval.

1. The client invokes the *GetResultTemplate* operation to retrieve the `resultStructure` and the `resultEncoding` which will be used by the SOS in later *GetResult* operation responses.

2. The client calls the *GetResult* operation and retrieves result values.

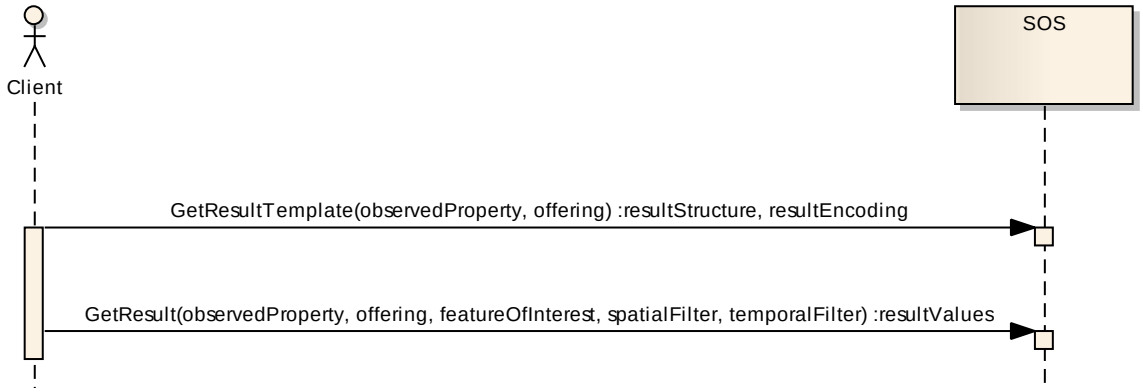


Figure 6: Sequence diagram of result retrieval

8 Requirements Class: Core

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/core	
Target Type	Web Service
Dependency	urn:iso:ts:iso:19156:clause:6
Dependency	http://www.opengis.net/doc/IS/OWS/1.1/clause/7
Dependency	http://www.opengis.net/doc/IS/SWES/2.0/clause/7
Dependency	http://www.opengis.net/doc/IS/SWES/2.0/clause/9
Dependency	http://www.opengis.net/doc/IS/SWES/2.0/clause/15
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**request-service**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**request-version**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-ows**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-version**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-sections**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-response**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-response-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-response-version**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-conf-class-listing**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-operation-listing**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-spatialFilter-listing**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**spatial-filter-minimum**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-temporalFilter-listing**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**temporal-filter-minimum**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-offerings-observations**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-contents**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-contents-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-observationoffering-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-offering-identifier**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-property-inheritance-mechanism**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-response-format-om20**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-observation-result-type**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**gc-exception**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**ds**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-parameters**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-omitting-parameters**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-filter-context**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-default-response-format**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-response-format**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-response-type**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-empty-response**

Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-response-elements-type**

Requirements Class (continued)	
http://www.opengis.net/spec/SOS/2.0/req/core	
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-observation-duplicates**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-exception**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/**go-too-many-obs-exception**

In this clause, the SOS core operations *GetCapabilities*, *DescribeSensor*, and *GetObservation* are specified.

GetCapabilities - provides access to metadata and detailed information about the operations available by an SOS server.

DescribeSensor - provides access to detailed information about the sensors and sensor systems available by an SOS server.

GetObservation - provides access to observations from sensors and sensor systems selected by spatial, temporal and thematic filtering.

The SOS operations follow the general pattern of other OGC Web Services and where appropriate inherit or re-use elements defined by the OWS Common standard [OGC 06-121r3] and the SWE Service Model standard [OGC 09-001].

All request and response types defined in this standard (except the *GetCapabilities* request and response which are based on [OGC 06-121r3]) are derived from the *ExtensibleRequest* or *ExtensibleRepsonse* type, respectively, as defined in OGC 09-001. Each request and response type defines an optional `extension` property, a container for request parameters that can be defined by an extension.

This standard defines the following general requirements applying to **ALL** operations defined in the following (except where explicitly negated):

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/request-service
Requirement 1 For **ALL** SOS request types defined in this standard, a mandatory `service` parameter specifies the OWS type abbreviation of the implementing service. It is of type `CharacterString` and shall have the fixed value "`SOS`".

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/request-version

Requirement 2	For **ALL** SOS request types defined in this standard except the request type of the GetCapabilities operation, a mandatory `version` parameter specifies the service type specification. It is of type `CharacterString` and shall have the fixed value "`2.0.0`".

8.1 GetCapabilities Operation

This operation allows clients to retrieve the service metadata (also called the "Capabilities" document) of an SOS server.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc
Requirement 3 Every SOS server shall support the *GetCapabilities* operation as defined in this Clause.

The conceptual model of the *GetCapabilities* operation is shown in the following UML diagram.

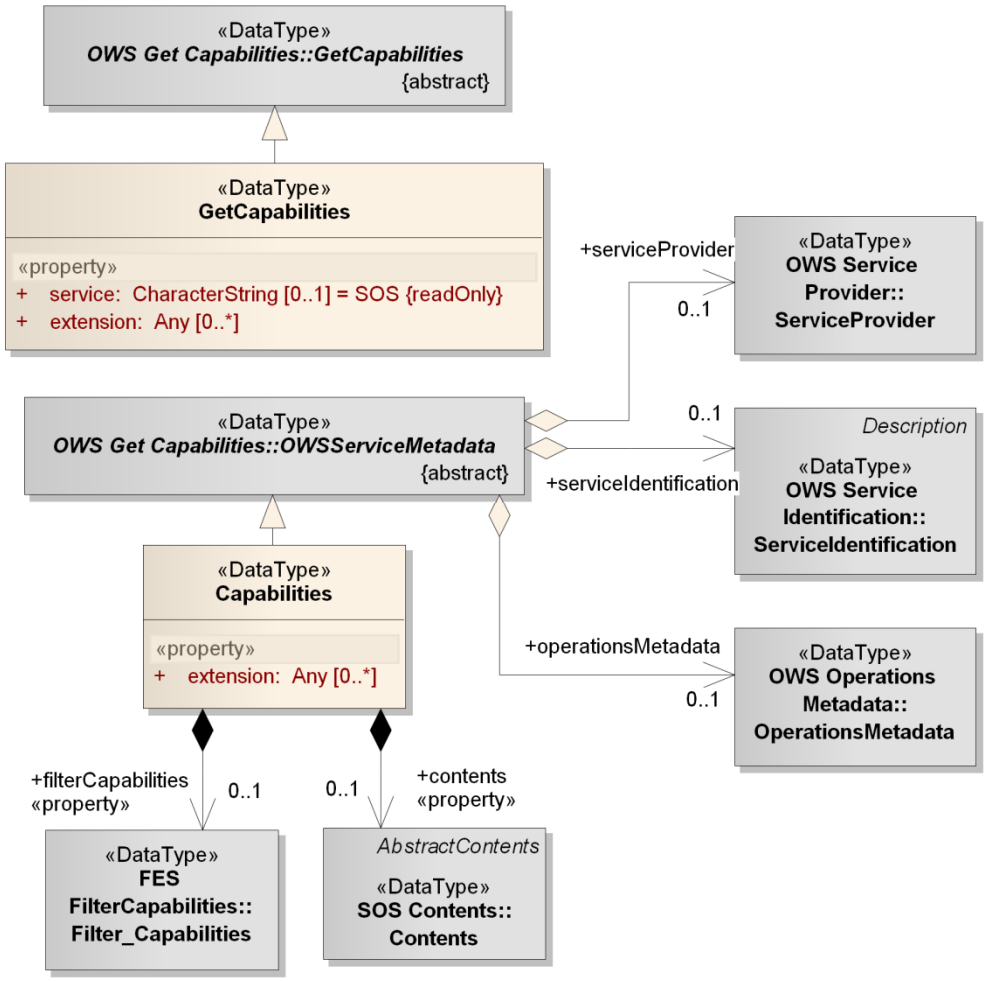


Figure 7: Data types of the GetCapabilities operation

8.1.1 Request

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-ows
Requirement 4 The *GetCapabilities* operation request shall be implemented as specified in Clause 7 of OWS Common [OGC 06-121r3].

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-version
Requirement 5 If the AcceptVersions parameter is contained in the request, it shall contain the character string "2.0.0".

The SOS *GetCapabilities* data type derives from the OWS Common *GetCapabilities* data type (listed in Table 3 of [06-121r3]) and thus inherits all the properties contained in that data type.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-request-structure
Requirement 6 In addition to the properties inherited from OWS Common *GetCapabilities*, the SOS *GetCapabilities* shall include the properties according to Table 13

Table 13: Properties in the GetCapabilities data type

Name	Definition	Data type and values	Multiplicity and use
extension	container for elements defined by extension specifications	Any type value is defined by the extension specification	Zero or more (optional)
service	service type identifier	Character String type, not empty value shall be "SOS"	Zero or one (optional) default value is "SOS"

NOTE: the request property – derived from OWS Common GetCapabilities type – is explicit or implied by each specific binding of the GetCapabilities operation, so is not necessarily part of the request representation defined by that binding.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-sections
Requirement 7 The allowed set of values for the *sections* parameter shall be as specified in Table 10 of [OGC 06-121r3], with the additions listed in Table 14 below.

NOTE: additional sections can be added by extensions; this is done for example by the InsertionCapabilities requirements class (clause 10.1).

Table 14: Additional section names for SOS Capabilities

Section name	Meaning
Contents	The Contents section of the SOS service metadata document contains information about the data offered by the SOS server (Subclause 8.1.2.2).
FilterCapabilities	The FilterCapabilties section of the SOS service metadata document contains information about the supported filters (Subclause 8.1.2.1)

8.1.2 Response

The response to a *GetCapabilities* operation request is also called the Capabilities document. This document provides clients with service metadata about a specific service instance, including metadata about the data served. This clause defines the required elements in a SOS 2.0 Capabilities document.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-response
Requirement 8 If the *GetCapabilities* request is valid, the server shall return an instance of the SOS `Capabilities` type else the server shall respond with an exception.

The SOS *Capabilities* data type derives from the OWS Common *OWSServiceMetadata* data type (as defined in clause 7.4.2 of [06-121r3]) and thus inherits all the properties contained in that data type.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-response-structure
Requirement 9 In addition to the properties inherited from OWS Common *OWSServiceMetadata*, the SOS *Capabilities* shall include the properties according to Table 15

Table 15: Properties in the Capabilities data type

Name	Definition	Data type and values	Multiplicity and use
contents	metadata about the observations and procedures hosted by the service	Contents, see Table 16	Zero or one (optional) inclusion depends on the values in the Sections parameter of the GetCapabilities operation request
extension	container for elements defined by extension specifications	Any type value is defined by the extension specification	Zero or more (optional)
filterCapabilities	metadata about the supported filter functionality	Filter_Capabilities, see OGC 09-026r1/ISO 19143	Zero or one (optional) inclusion depends on the values in the Sections parameter of the GetCapabilities operation request

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-response-version
Requirement 10 The default version of the Capabilities document returned by a service implementing this standard shall be "2.0.0".

The `ServiceIdentification`, `ServiceProvider`, and `OperationsMetadata` sections of the *GetCapabilities* response document are defined in [OGC 06-121r3].

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-conf-class-listing
Requirement 11 The Capabilities document shall advertise conformance classes which are supported by the server. Therefore, each value of the `Profile` property of the `ServiceIdentification` section shall be the pointer to a conformance class, and the server shall pass all tests defined for each listed conformance class.

NOTE Typically, such conformance classes will be specified in SOS extensions.

NOTE Conformance classes from implemented specification (e.g. OGC Filter Encoding Specification or OMXML) should also be listed.

NOTE Inherited conformance classes are listed in the `Profile` property as well. For example, even though the InsertObservation conformance class has a dependency on the InsertionCapabilities class, both conformance classes are listed rather than just the InsertObservation class.

The `OperationsMetadata` section specified in [OGC 06-121r3] lists the request types supported by an SOS server. For the SOS Core, these are *GetCapabilities*, *DescribeSensor*, and *GetObservation*; extensions may add further request types to this list.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-operation-listing
Requirement 12 The Capabilities document shall include an `OperationsMetadata` section which contains three `Operation` elements with case-sensitive name values "GetCapabilities", "DescribeSensor", and "GetObservation". All other operations supported by the SOS server and defined in extensions shall be listed in the same way.

The following subclauses define the sections of the SOS Capabilities document which are added by this standard, the `FilterCapabilities` and the `Contents` section.

8.1.2.1 FilterCapabilities Section

The `FilterCapabilities` section is imported from [OGC 09-026r1/ISO 19143] and used to state which filter operators and operands are supported by an SOS server. The operators and operands refer to the parameters of service operations that include OGC filter expressions, like the *GetObservation* operation.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-spatialFilter-listing
Requirement 13 The Capabilities document shall include a `FilterCapabilities` section which lists the spatial filter operators and operands that are supported by the service as defined in section 7.14.5 of [OGC 09-026r1/ISO 19143].

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/spatial-filter-minimum
Requirement 14 Each SOS shall at least support the spatial filter operator `BBOX` and shall advertise this in the Capabilities document.

Example 1 An SOS which supports the spatial operator `BBOX` for the geometry operand `gml:Point` would list this as shown in the example at http://schemas.opengis.net/sos/2.0/examples/core/GetCapabilities1_response.xml

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-temporalFilter-listing
Requirement 15 The Capabilities document shall include a `FilterCapabilities` section which lists the temporal filter operators and operands that are supported by the service as defined in section 7.14.6 of [OGC 09-026r1/ISO 19143].

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/temporal-filter-minimum
Requirement 16 Each SOS shall at least support the following temporal filter operators: a. `TEquals` operator in conjunction with the `TimeInstant` type. b. `During` operator in conjunction with the `TimePeriod` type. An SOS shall advertise the supported filter operators in the Capabilities document.

Example 2 An SOS which supports the temporal operators `TEquals` and `During` for the temporal operands `gml:TimeInstant` and `gml:TimePeriod` would list these as shown in the example at http://schemas.opengis.net/sos/2.0/examples/core/GetCapabilities1_response.xml

8.1.2.2 Contents Section

The Contents section of the *GetCapabilities* response describes the data offered by an SOS server. To group the offered observations the SOS defines the concept of *ObservationOfferings*.

The `Contents` type lists all `ObservationOfferings` of an SOS server. An ObservationOffering groups collections of observations produced by **one** procedure, e.g., a sensor system[6]. The ObservationOffering lists the basic metadata for the associated observations including the observed properties of the observations. An observation may belong to more than one ObservationOffering.

To summarize: there is a `1:n` relationship between procedures and ObservationOfferings; there is a `n:m` relationship between observations created by these procedures and ObservationOfferings.

NOTE: Observations are usually associated with ONE offering. However, observations from one sensor may be assigned to multiple offerings - a 1:n relationship is allowed. This may be done for the purpose of grouping pre-filtered observations. This way functionality is given which allows grouping of observations according to some thematic criteria. An example is the grouping of *severe* and *all* weather forecast observations from the same sensor system into two offerings.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-offerings-observations
Requirement 17 The observations grouped by an *ObservationOffering* shall not have been created by a procedure other than the one that is stated by the *ObservationOffering*.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-contents
Requirement 18 If requested the Capabilities document shall include a section of type `Contents` as defined in this sub-clause.

[6] A sensor system can be a simple thermometer, but can also consist of several sub-systems. So, a valid sensor system can be a system of sensors attached to a weather station, or it can be a network of spatially distributed sensors.

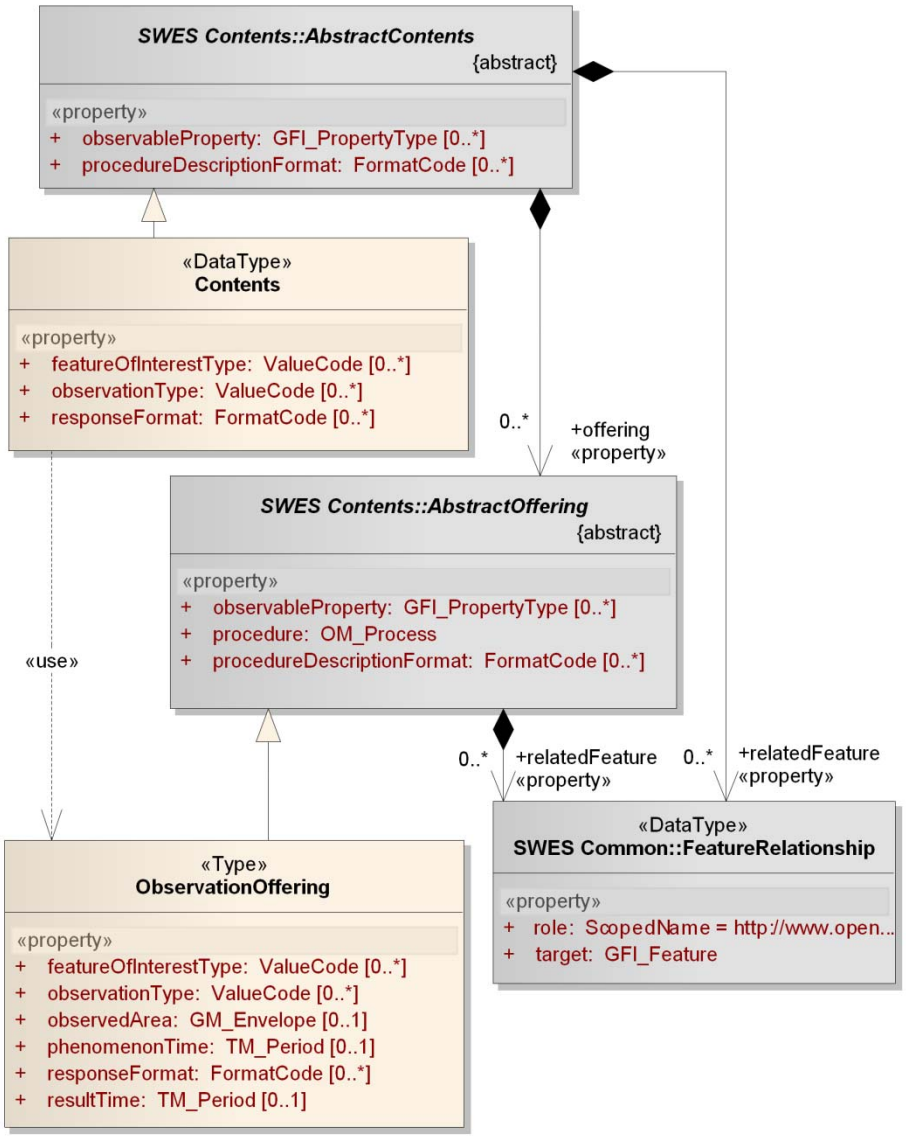


Figure 8: Data types of Contents section

The SOS `Contents` type derives from the SWES `AbstractContents` type defined in [OGC 09-001] and inherits its properties.

Those *inherited* properties are (see clause 7.2.1 in [OGC 09-001] for details):

- `procedureDescriptionFormat` - the identifier of a specific procedure/sensor description format.

- `observableProperty` - the pointer to a property that can be observed by a procedure, not necessarily a property that has already been observed.

- `relatedFeature` - feature that is directly or indirectly observed/observable by a procedure; can be any feature of which the service provider thinks the procedure can make valuable observations for.

- `offering` - contains metadata about a procedure/sensor hosted by the service.

NOTE: Related features are NOT necessarily features of interest (and associated with observations). The listing of a related feature can be used for discovery purposes by the service provider. The related feature can for example be the *Gulf of Mexico* in case an SOS serves data measured by sensors that are carried by marine drifters in the Gulf of Mexico area. Especially in such cases of mobile sensors, it is not reasonable to list all features of interest (= the sampling locations) as related features, since the Capabilities document would grow too big over time.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-contents-structure
Requirement 19 In addition to the properties inherited from SWES *AbstractContents*, the SOS *Contents* shall include the properties according to Table 16

Table 16: Properties of Contents data type

Name	Definition	Data type and values	Multiplicity
responseFormat	Identifies the response format supported for observation retrieval.	FormatCode see OGC 09-001	Zero or many (optional)
featureOfInterestType	Identifies the type of features of interest associated with the observations provided by the service.	ValueCode see clause 8.4.1	Zero or many (optional)
observationType	Identifies the type of observation (with unique result type) which is used by the service to encode observations.	ValueCode see clause 8.4.1	Zero or many[7] (optional)

The SOS `ObservationOffering` type derives from the SWES `AbstractOffering` type defined in OGC 09-001 and inherits all its properties.

Those *inherited* properties are (see clause 7.2.2 in [OGC 09-001] for details):

- `procedure` - Pointer to the procedure/sensor associated with this offering.

- As well as, `procedureDescriptionFormat`, `observableProperty`, and `relatedFeature` as already described above.

[7] The design decision for specifying the multiplicity as "zero or many" instead of "zero or one" is that there may be many observable properties per ObservationOffering. Each one could use a different observation type or result type. However, a known issue is that the relation between observable property and observation / result type cannot be reflected in the Capabilities. The description of the procedure associated with a given ObservationOffering may provide more detailed information on this relationship.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-observationoffering-structure
Requirement 20 In addition to the properties inherited from SWES *AbstractOffering*, the SOS *ObservationOffering* shall include the properties according to Table 17

Table 17: Properties of ObservationOffering data type

Name	Definition	Data type and values	Multiplicity and use
phenomenonTime	temporal bounding box of the phenomenon times of all observations belonging to the offering	TM_Period see [ISO 19108]	Zero or one (optional) omit only if the offering has no observations associated to it
resultTime	temporal bounding box of the result times of all observations belonging to the offering	TM_Period see [ISO 19108]	Zero or one (optional) omit only if the offering has no observations associated to it
observedArea	spatial bounding box of the spatial extent of all features of interest that belong to observations associated with the offering	GM_Envelope see [ISO 19107]	Zero or one (optional)
responseFormat	*As defined in Table 16.*		
featureOfInterestType	*As defined in Table 16.*		
observationType	*As defined in Table 16.*		

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-offering-identifier
Requirement 21 The service shall assign a unique identifier value to each of its ObservationOfferings[8]. This value shall be set in the swes:identifier property that each ObservationOffering automatically has (see [OGC 09-001] for further details).

For the purpose of reducing the amount of redundant information in the Capabilities document, the SOS supports a so-called *property inheritance mechanism* as defined in [OGC 09-001].

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-property-inheritance-mechanism

[8] See Annex 15 for a detailed discussion of identifier handling.

Requirement 22 The ObservationOffering is a property inheritor and the Contents its property provider as defined by OGC 09-001.

The service shall ensure that the number of values it provides for the properties of each offering is as stated in Table 18 when the *property inheritance mechanism* was applied.

The following table shows which of the properties of an ObservationOffering can be inherited and which cardinality is expected after the inheritance mechanism has been applied.

Clients need to apply the SWES *property inheritance mechanism* according to the inheritance categories defined for the ObservationOffering properties in Table 18 when determining their values (for a given offering).

Table 18: Inheritance of ObservationOffering properties (from Contents)

Property	Number[9]	Inheritance
observableProperty	1..*	replace
procedure	1	no
procedureDescriptionFormat	1..*	replace
relatedFeature	0..*	replace
featureOfInterestType	1..*	replace
observationType	1..*	replace
observedArea	0..1	no
phenomenonTime	0..1	no
responseFormat	1..*	replace
resultTime	0..1	no

Thus, even though the UML model and schema encoding define, for example, the `observationType` and `responseFormat` properties as optional, they are mandatory in each ObservationOffering. In other words, each offering has to include at least one value for these two properties after the property inheritance mechanism was applied.

As O&M v2.0 XML encoding (OMXML 2.0) is the only mandatory format for observation data at a SOS 2.0:

Requirement

[9] The design decisions for specifying those multiplicities are the following: The observableProperty, procedureDescriptionFormat, featureOfInterestType, and observationType are required as this information needs to be provided in the *InsertSensor* request and thus has to be provided even if no observations are assigned to the ObservationOffering yet (and even if the *InsertSensor* operation is not implemented). The responseFormat is required as Requirement 23 states that the default response format must be listed explicitly. The phenomenonTime/resultTime is omitted if no observation has been associated to the offering yet. This would for example be the case right after a new sensor was inserted at the service.

http://www.opengis.net/spec/SOS/2.0/req/core/gc-response-format-om20
Requirement 23 The responseFormat of each ObservationOffering shall have at least the value *http://www.opengis.net/om/2.0*.

Other response formats may be supported as well. However, the way how such formats are supported needs to be described in a specific extension to this document.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-observation-result-type
Requirement 24 The observation type identifiers listed in the observationType property of an ObservationOffering shall identify an observation type with unique result type, i.e. a unique combination of OM_Observation (or derived type) and one specific type that the result of the observation is provided in.

8.1.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/gc-exception
Requirement 25 When an SOS server encounters an error while performing a GetCapabilities operation, it shall return an exception message encoded as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in Listing 1. The meaning of each exception code shall be as defined in OGC 06-121r3 and OGC 09-001.

Listing 1: exception codes applicable to the GetCapabilities operation

- MissingParameterValue
- InvalidParameterValue
- VersionNegotiationFailed
- InvalidUpdateSequence
- OptionNotSupported
- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported

8.1.4 Examples

Example 3 An example request of the XML implementation of this operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/core/GetCapabilities1.xml

Example 4 An example response of the XML implementation of this operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/core/GetCapabilities1_response.xml

8.2 DescribeSensor Operation

The *DescribeSensor* operation enables the retrieval of metadata descriptions of procedures (or: *sensors*) associated with an SOS. While the response to a *GetCapabilities* request lists procedures associated with the SOS the *DescribeSensor* operation can be used subsequently to retrieve detailed definitions for the listed procedures. The definition of this operation can be found in Clause 11 of [OGC 09-001].

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/ds
Requirement 26 Every SOS server shall support the *DescribeSensor* operation as defined in Clause 11 of [OGC 09-001]. When targetting procedures hosted by the SOS, the `service` attribute in the DescribeSensor request shall have the value "SOS" and the `version` attribute shall have the value "2.0.0".

8.2.1 Examples

Example 5 An example request of the XML implementation of this operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/core/DescribeSensor1.xml

8.3 GetObservation Operation

The *GetObservation* operation is designed to query an SOS to retrieve observation data structured according to the O&M specification. Other response formats are possible; however, O&M is the default and mandatory format for every SOS server. Several parameters of the *GetObservation* operation allow extensive filtering of the observations requested from the SOS server.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go
Requirement 27 Every SOS server shall support the *GetObservation* operation as defined here.

The conceptual model of the *GetObservation* operation is shown in the following UML diagram.

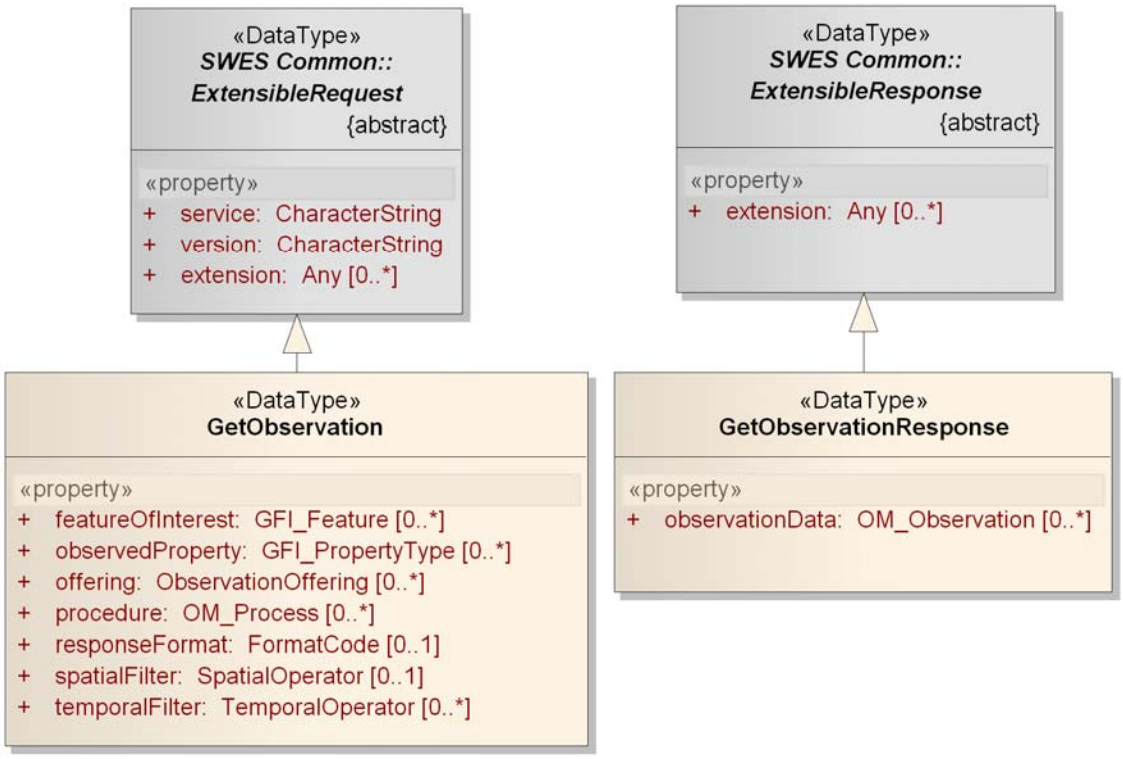


Figure 9: Data types of GetObservation operation

8.3.1 Request

A GetObservation operation request contains parameters that constrain the observations to be retrieved from a SOS. This sub-clause describes the requirements to make a valid GetObservation request.

The GetObservation data type is derived from the SWES ExtensibleRequest data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-request-structure
Requirement 28 In addition to the properties inherited from SWES ExtensibleRequest, the SOS GetObservation operation request type shall include the properties according to Table 19.

The concrete representation of this structure depends on the chosen protocol binding.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-parameters
Requirement 29 The SOS returns all observations that match the specified parameter values. The filter parameters (e.g., `observedProperty`, `procedure`, or `temporalFilter`) shall be connected with an implicit **AND**. The values of each of the parameters shall be connected with an implicit **OR**.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-omitting-parameters
Requirement 30 If an optional parameter of a *GetObservation* request is not included in the request, the filter (represented by the parameter) shall not be applied to the observation set which will be returned by the SOS server.

For example, in consequence of Requirement 30, an SOS server returns observations of all time to the client if the temporal filter is omitted.

NOTE: An implementation of an SOS server may return an exception message as specified in Clause 15 of [OGC 09-001] if the response of a *GetObservation* request would be too big to be reasonably sent to a client.

Example 6 Resulting from Requirement 29 and Requirement 30 an abstract *GetObservation* request looks like this:

GetObservation (`featureOfInterest` := *weatherstation_in_my_backyard*
AND `observedProperty` := *temperature*
AND `procedure` := *thermometerX* **OR** *anemometerY*)

This request returns the observations of all offerings, all time and all spatial extent, and which were made for the feature of interest "weather_station_in_my_backyard", and which carry results for the observed property "temperature" and were made by the sensor "thermometerX" or "anemometerY".

Table 19: Properties of GetObservation request data type

Name	Definition	Data type and values	Multiplicity and use
procedure	Pointer to a procedure for which observations are requested. It defines a filter for the procedure property of the observations.	OM_Process [id] see [OGC 10-004r3/ISO 19156]	Zero or many (Optional)
offering	Pointer to an ObservationOffering advertised in the Capabilities document for which observations are requested.	ObservationOffering [id] see Table 17	Zero or many[10] (Optional)
observedProperty	Pointer to an observedProperty for which observations are requested.	GFI_PropertyType [id] see [OGC 10-004r3/ISO 19156]	Zero or many (optional)
temporalFilter	Specifies a filter for a time property of requested observations. This property is defined in the valueReference element of the TemporalOperator.	TemporalOperator see [OGC 09-026r1/ISO 19143]	Zero or many (Optional)

[10] The design decision for specifying the multiplicity "zero or many" instead of "one or many" is that a client might want to request all observations for a particular observedProperty (or featureOfInterest, procedure etc.).

Name	Definition	Data type and values	Multiplicity and use
featureOfInterest	Pointer to a feature of interest for which observations are requested.	GFI_Feature [id] see [OGC 10-004r3/ISO 19156]	Zero or many (Optional)
spatialFilter	Specifies a filter[11] which applies to a spatial property of an observation. This property is defined in the valueReference element of the SpatialOperator.	SpatialOperator see [OGC 09-026r1/ISO 19143]	Zero or one (Optional)
responseFormat	Identifier of desired responseFormat for the requested observations. The supported responseFormats are listed in the ObservationOffering.	FormatCode see OGC 09-001 Default is O&M 2.0 [OGC 10-004r3/ISO 19156] identified by the value `http://www.opengis.net/om/2.0`	Zero or one (optional) If not provided, default value is assumed.
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details.			

NOTE Observations returned by the SOS might be encoded with different observation types and result types. For the client, such responses are difficult to parse. This issue is not addressed here. It might be solved in future through an extension which allows requesting only certain observation / result types. However, such a mechanism would put the burden of observation / result type transformation on the SOS server.

It is necessary to establish a linkage between the GetObservation request model and the underlying SOS data model.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-filter-context
The default context for the ValueReferences used in the expression of a filter operator (see OGC 09-026r1/ISO 19143 for further details) for the GetObservation request parameters that do not have a pre-defined target property and operator for filtering shall be as defined in **Requirement 31** Table 20. The default context for GetObservation request parameters that have a pre-defined target property shall be as defined in Table 21.

Table 20: default context for GetObservation request parameters without pre-defined target property

Filter Property	Context for ValueReference in filter operator
temporalFilter	OM_Observation
spatialFilter	OM_Observation

[11] A profile of this generic spatialFilter is given in Clause 12. This profile restricts the spatialFilter so that it is applied to the sampling location parameter of the observations. Those observations need to conform to the requirements class "Spatial Observation Data" defined in [OGC 10-025] Subclause 7.13.

Table 21: default context for GetObservation request parameters with pre-defined target property

Filter Property	Target property for PropertyIsEqualTo comparison operator
offering	Capabilities.contents.offering
observedProperty	OM_Observation.observedProperty
procedure	OM_Observation.procedure
featureOfInterest	OM_Observation.featureOfInterest

NOTE in an XML based SOS implementation the filter context can be defined via XPath expressions – see Table 48 in Subclause 13.1.

Example 7 The SOS receives a GetObservation request that filters the returned observations based upon a given offering by pointing to the object *Offering12849* (of type ObservationOffering). The service determines whether that specific object is contained in the set of offerings listed in the Capabilities' contents section of the service. If the offering is contained, then the service will include the observations that belong to it in the GetObservationResponse.

Example 8 The SOS receives a GetObservation request that filters returned observations based upon a given features of interest by pointing to *FeatureA* and *FeatureB* (both of type GFI_Feature). The service will include only those observations in the response which have FeatureA or FeatureB as value of their featureOfInterest property.

8.3.2 Response

This sub-clause describes the requirements on a SOS 2.0 service in responding to a GetObservation request.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-default-response-format
Requirement 32 An SOS server shall be able to return the requested observations encoded as O&M 2.0 [OGC 10-004r3/ISO 19156]. O&M 2.0 is the default response format for the *GetObservation* operation and shall be returned in the response if no other specific `responseFormat` was specified in the request.

NOTE: in SOS version 1.0, the GetObservation operation was sometimes used to not only retrieve observations that match the request criteria – which is the default behavior of the operation also in SOS 2.0 - but also to subset the observation result, only listing the (typically SWE Common encoded) components that match the observedProperty values provided in the request. As the SOS 2.0 is not tied to a specific result format, it does not perform such subsetting and re-structuring. According behavior can be defined via additional conformance classes, as an extension to the SOS 2.0 core. The extension points offered in SWES ExtensibleRequest and ExtensibleResponse data types can support such functionality.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-response-format
Requirement 33 The SOS server shall respond in the specified `responseFormat`. If the response format specified by the client is not supported an exception message shall be returned with the exception code “InvalidParameterValue” and the locator value ”responseFormat”.

The SOS `GetObservationResponse` data type derives from the SWES `ExtensibleResponse` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-response-type
Requirement 34 If O&M 2.0 [OGC 10-004r3/ISO 19156] is returned by an SOS server, the response shall contain an instance of the `GetObservationResponse` type. In addition to the properties inherited from SWES `ExtensibleResponse`, the SOS `GetObservationResponse` shall include the properties according to Table 22.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-empty-response
Requirement 35 An instance of `GetObservationResponse` type shall be empty if none of the observations associated with the SOS fulfill the *GetObservation* parameters specified by the client.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-response-elements-type
Requirement 36 If the *GetObservation* response contains observations, the `observationData` elements shall be of type `OM_Observation` as defined in [OGC 10-004r3/ISO 19156] or subtypes of `OM_Observation`.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-observation-duplicates
Requirement 37 A response to a *GetObservation* request shall not contain observation duplicates. Even if multiple `ObservationOfferings` are requested and an observation belongs to more than one `ObservationOffering`, then the *GetObservation* response shall not list such an observation twice.

Table 22: Properties of GetObservationResponse data type

Name	Definition	Data type and values	Multiplicity and use
observationData	Observation which fulfills the *GetObservation* request.	OM_Observation (see [OGC 10-004r3/ISO 19156])	Zero or many (optional)

8.3.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-exception
Requirement 38 When an SOS server encounters an error while performing a GetObservation operation, it shall return an exception message as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in Listing 2. The meaning of each exception code shall be as defined in OGC 06-121r3, OGC 09-001 and Table 23.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/go-too-many-obs-exception
Requirement 39 A *GetObservation* request can result in a huge observation result set. If an SOS server implementation needs to prevent the delivery of such result sets to the client, since it would result in a response that is too big, the SOS server shall return an exception message as specified in Clause 15 of [OGC 09-001] with exception code, meaning and "locator" value as defined in Table 23.

Note: If the client receives a ResponseExceedsSizeLimit exception as response to a GetObservation request, it should try to split the set of requested observations into smaller chunks and query them via a number of GetObservation requests with according parameters. For example, the request could apply a temporal partitioning via the GetObservationRequest.temporalFilter property. The service may also support additional operations to retrieve metadata about the stored observations, for example for which time ranges observations were generated by a given procedure.

Table 23: ResponseExceedsSizeLimit Exception

exceptionCode value	Meaning of code	"locator" value
ResponseExceedsSizeLimit	The service determined that the requested result set exceeds the response size limit of the service and thus cannot be delivered.	None, omit "locator" parameter

Listing 2: exception codes applicable to the GetObservation operation

- MissingParameterValue
- InvalidParameterValue
- OptionNotSupported

- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported
- ResponseExceedsSizeLimit

8.3.4 Examples

Example 9 An example request and response of the XML implementation of the *GetObservation* operation with a specified `observedProperty` and `temporalFilter` can be found here:

http://schemas.opengis.net/sos/2.0/examples/core/GetObservation1_obsProps.xml

http://schemas.opengis.net/sos/2.0/examples/core/GetObservation1_obsProps_response.xml

Example 10 An example request and response of the XML implementation of the *GetObservation* operation with a specified `observedProperty` and `procedure` can be found here:

http://schemas.opengis.net/sos/2.0/examples/core/GetObservation2_obsProps_Procedure.xml

http://schemas.opengis.net/sos/2.0/examples/core/GetObservation2_obsProps_Procedure_response.xml

Example 11 An example request and response of the XML implementation of the *GetObservation* operation with a specified `observedProperty`, `procedure` and `featureOfInterest` can be found here:

http://schemas.opengis.net/sos/2.0/examples/core/GetObservation3_foiIDFilter.xml

http://schemas.opengis.net/sos/2.0/examples/core/GetObservation3_foiIDFilter_response.xml

Example 12 An example request and response of the XML implementation of the *GetObservation* operation with a specified `observedProperty`, `procedure` and `spatialFilter` can be found here:

http://schemas.opengis.net/sos/2.0/examples/core/GetObservation4_spatialFilter.xml

http://schemas.opengis.net/sos/2.0/examples/core/GetObservation4_spatialFilter_response.xml

8.4 Codes

SOS makes use of code values in several places, for example for identifying the format that observations can be provided in but also which conceptual types for representing features of interest, observations and observation results are supported.

This standard reuses code types defined by the SWE Service Model [OGC 09-001] but also uses additional code types that are defined in the following.

The conceptual model of the *Codes* package is shown in the following UML diagram.

Figure 10: Data types of Codes package

The ValueCode code list shown in the diagram is empty because it defines an unlimited set of possible code values. The details of each code list contained in the package as well as some example code values will be explained in the following.

8.4.1 ValueCode

This code list is modeled after the code lists defined in the Common Codes package of the SWE Service Model [OGC 09-001]. As such, all code values assigned to it are URIs.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/core/co-valuecodes
Requirement 40 All code values identifying some type defined in a given conceptual model (either directly or via a direct mapping) shall be added to the *ValueCode* code list. These code values shall be URIs.

There are several ways to identify a type that is defined in a conceptual model:

- The model itself could have assigned a unique identifier to that type. Example: *http://www.opengis.net/def/observationType/OGC-OM/2.0/OM_ComplexObservation*

- The type can be uniquely identified via a combination of the namespace of the model, the full path to the package that contains the type and the type's name. Example: *http://www.isotc211.org/19103/2003/BasicTypes/Derived/UnitsOfMeasure/Measure*, *http://www.isotc211.org/19103/2003/BasicTypes/Primitive/Truth/Boolean*

- There is some direct mapping between the conceptual model and some implementation encoding, for example an XML encoding, and the mapped type can be uniquely identified in that encoding. In an XML encoding this could then be the combination of the XML namespace and the element name (or, if that is not available, the complex/simple type name) assigned to the type. Example: *http://www.opengis.net/swe/2.0/Quantity*, *http://www.opengis.net/gml/3.2/measure*

The following table lists some code values for the *ValueCode* list together with their definition and meaning. Extensions to this specification can define additional codes.

Table 24 – List of some code values used for identifying types defined in a conceptual model

Value code value	Definition/Meaning
http://www.opengis.net/def/observationType/OGC-OM/2.0/OM_Measurement	O&M Observation whose result type is a measure NOTE: OGC 10-025 Table 5 lists URIs to identify other observation types (most of which have a well-defined result type) defined in the O&M model
http://www.opengis.net/swe/2.0/DataArray	the DataArray type defined by SWE Common [OGC 08-094]
http://www.opengis.net/def/samplingFeatureType/OGC-OM/2.0/SF_SamplingPoint	the SF_SamplingPoint type defined by O&M NOTE: OGC 10-025 Table 6 lists URIs to identify other sampling feature types defined in the O&M model

9 Enhanced Operations Extension

9.1 Requirements Class: Feature of Interest Retrieval

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/**request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/**parameters**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/**omitting-parameters**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/**filter-context**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/**response-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/**empty-response**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/**exception**

9.1.1 GetFeatureOfInterest Operation

The *GetFeatureOfInterest* operation allows clients to retrieve certain features of interest of observations.

NOTE: the GetFeatureOfInterest operation resembles the GetResult operation in the sense that it only returns a certain property of selected observations – the feature of interest.

The conceptual model of the *GetFeatureOfInterest* operation is shown in the following UML diagram.

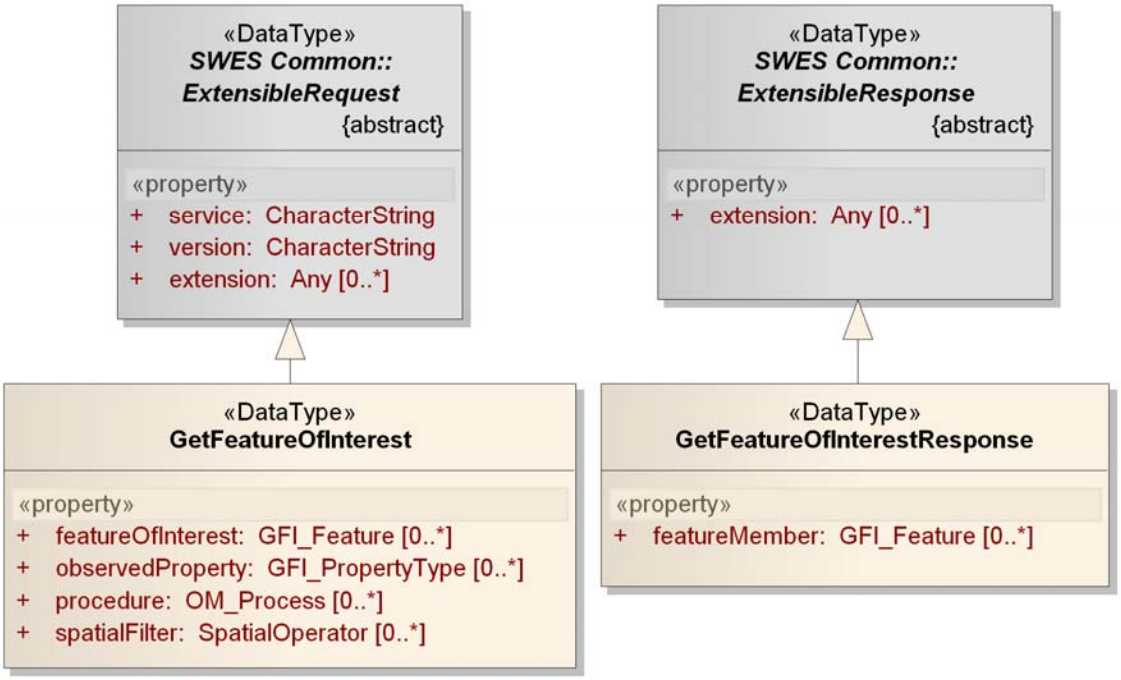


Figure 11: Data types of GetFeatureOfInterest operation

9.1.1.1 Request

The SOS `GetFeatureOfInterest` data type derives from the SWES `ExtensibleRequest` type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/request-structure
Requirement 41 **Requirement 42** **Requirement 43** Table 25

Requirement
http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/parameters
Requirement 44 Upon receipt of a GetFeatureOfInterest request, the SOS shall return the features of interest of all observations that match the specified request parameter values. The request parameters shall be connected with an implicit **AND**. The values of each of the parameters are connected with an implicit **OR**.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/omitting-parameters
Requirement 45 If an optional parameter of a *GetFeatureOfInterest* request is not specified by the client, the filter (represented by that parameter) shall not be applied to the feature set which will be returned by the SOS server.

: An implementation of an SOS server may return an exception message as specified in Clause 15 of [OGC 09-001] if the response of a *GetFeatureOfInterest* request would be too big to be reasonably sent to a client.

Example 13 Resulting from Requirement 44 and Requirement 45 an example abstract *GetFeatureOfInterest* request looks like this:

GetFeatureOfInterest (`observedProperty` := *temperature*
AND `procedure` := *thermometerX* **OR** *anemometerY*)

This request returns all features of interest of all spatial extent, which carry the property "temperature" and are observed by the sensor "thermometerX" or "anemometerY".

Table 25: Properties of GetFeatureOfInterest data type

Name	Definition	Data type and values	Multiplicity and use
procedure	Pointer to a procedure which is observing the feature of interest.	OM_Process [id] see [OGC 10-004r3/ISO 19156]	Zero or many (optional)
observedProperty	Pointer to a property that was observed regarding the feature of interest.	GFI_PropertyType [id] see [OGC 10-004r3/ISO 19156]	Zero or many (optional)
featureOfInterest	Pointer to a specific feature of interest (used to restrict the set of all possibly returned features of interest to a specific set)	GFI_Feature [id] see [OGC 10-004r3/ISO 19156]	Zero or many (optional)
spatialFilter	Specifies a filter which applies to a spatial property of the requested features. This property is defined in the valueReference element of the SpatialOperator.	SpatialOperator see [OGC 09-026r1/ISO 19143]	Zero or many (optional)
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details on identifier handling.			

It is necessary to establish a linkage between the GetFeatureOfInterest request model and the underlying SOS data model.

Requirement

http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/filter-context
Requirement 46 The default context for the ValueReferences used in the expression of a filter operator (see OGC 09-026r1/ISO 19143 for further details) for the GetFeatureOfInterest request parameters that do not have a pre-defined target property and operator for filtering shall be as defined in Table 26. The default context for GetFeatureOfInterest request parameters that have a pre-defined target property shall be as defined in Table 27.

Table 26: default context for GetFeatureOfInterest request parameters without pre-defined target property

Filter Property	Context for ValueReference in filter operator
spatialFilter	OM_Observation

Table 27: default context for GetFeatureOfInterest request parameters with pre-defined target property

Filter Property	Target property for PropertyIsEqualTo comparison operator
observedProperty	OM_Observation.observedProperty
procedure	OM_Observation.procedure
featureOfInterest	OM_Observation.featureOfInterest

NOTE: in an XML based SOS implementation the filter context can be defined via XPath expressions – see Table 48 in Subclause 13.1.

9.1.1.2 Response

The SOS `GetFeatureOfInterestResponse` data type derives from the SWES `ExtensibleResponse` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/response-structure
Requirement 47 In addition to the properties inherited from SWES `ExtensibleResponse`, the SOS `GetFeatureOfInterestResponse` shall include the property according to Table 28.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/empty-response

Requirement 48 An instance of `GetFeatureOfInterestResponse` type shall be empty if none of the (features of interest associated to) observations stored by the SOS fulfill the *GetFeatureOfInterest* parameters specified by the client.

Table 28: Properties of GetFeatureOfInterestResponse data type

Name	Definition	Data type and values	Multiplicity and use
featureMember	Feature of interest of observations matching the request parameters specified by the client.	GFI_Feature (see [OGC 10-004r3/ISO 19156])	Zero or many (optional)

9.1.1.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/exception
Requirement 49 When an SOS server encounters an error while performing a *GetFeatureOfInterest* operation, it shall return an exception message as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in Listing 3. The meaning of each exception code shall be as defined in OGC 06-121r3, OGC 09-001 and Table 23 in this document.

Listing 3: exception codes applicable to the GetFeatureOfInterest operation

- OperationNotSupported
- MissingParameterValue
- InvalidParameterValue
- OptionNotSupported
- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported
- ResponseExceedsSizeLimit

9.1.1.4 Examples

Example 14 An example request of the XML implementation of the *GetFeatureOfInterest* operation with a specified `observedProperty` and `procedure` can be found here:

http://schemas.opengis.net/sos/2.0/examples/enhancedOperations/GetFOI1.xml

Example 15 An example request of the XML implementation of the *GetFeatureOfInterest* operation with specified `featureOfInterest` can be found here:

http://schemas.opengis.net/sos/2.0/examples/enhancedOperations/GetFOI2.xml

Example 16 An example request of the XML implementation of the *GetFeatureOfInterest* operation with a specified `observedProperty` and `spatialFilter` can be found here:

http://schemas.opengis.net/sos/2.0/examples/enhancedOperations/GetFOI3.xml

9.2 Requirements Class: Observation Retrieval By ID

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/**request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/**filter-context**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/**response-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/**response-behavior**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/**empty-response**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/**exception**

9.2.1 GetObservationById operation

The *GetObservationByID* operation allows the client to retrieve an observation by passing a pointer[12] to that observation.

The conceptual model of the *GetObservationByID* operation is shown in the following UML diagram.

[12] See Annex 15 for a discussion on pointer / identifier handling.

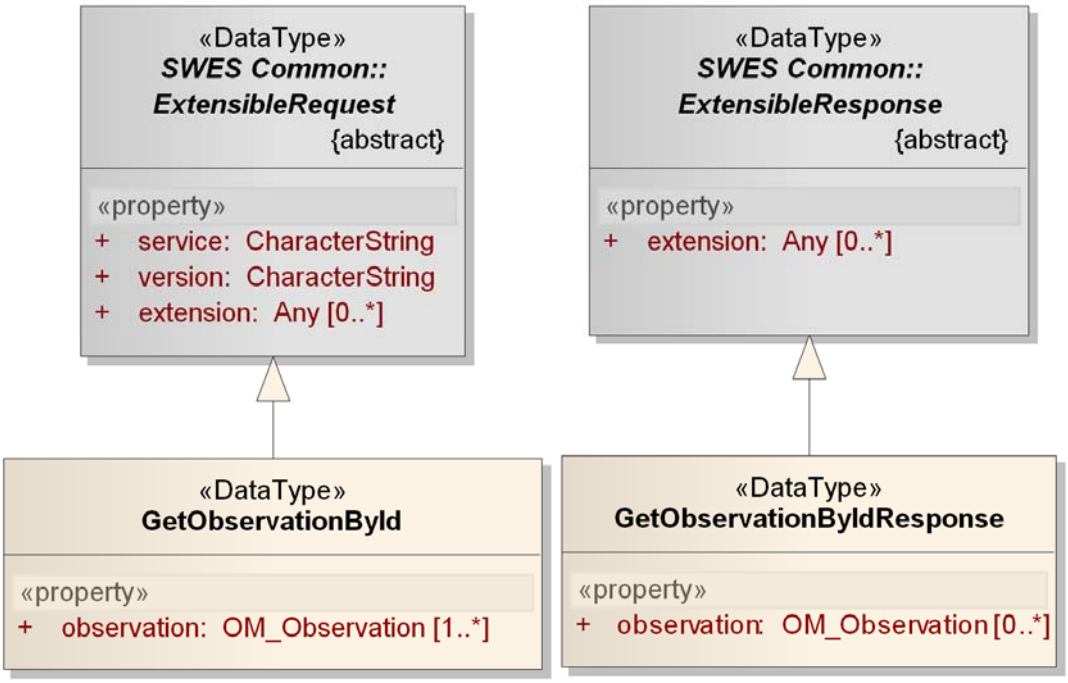


Figure 12: Data types of GetObservationById operation

9.2.1.1 Request

The SOS `GetObservationById` data type derives from the SWES `ExtensibleRequest` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/request-structure
Requirement 50 In addition to the properties inherited from SWES `ExtensibleRequest`, the SOS `GetObservationById` shall include the property according to Table 29.

Table 29: Properties of GetObservationById data type

Name	**Definition**	**Data type and values**	**Multiplicity and use**
observation	Pointer to the observation which shall be returned.	OM_Observation [id] see [OGC 10-004r3/ISO 19156]	One or many (mandatory)
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details on identifier handling.			

It is necessary to establish a linkage between the GetObservationById request model and the underlying SOS data model.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/gobi/filter-context
Requirement 51 The default context for the ValueReferences used in the expression of a filter operator (see OGC 09-026r1/ISO 19143 for further details) for the GetObservationById request parameters that have a pre-defined target property shall be as defined in Table 30.

Table 30: default context for GetObservationById request parameters with pre-defined target property

Filter Property	Target property for PropertyIsEqualTo comparison operator
observation	OM_Observation

NOTE: in an XML based SOS implementation the filter context can be defined via XPath expressions – see Table 48 in Subclause 13.1.

9.2.1.2 Response

The SOS `GetObservationByIdResponse` data type derives from the SWES `ExtensibleResponse` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/response-structure
Requirement 52 In addition to the properties inherited from SWES `ExtensibleResponse`, the SOS `GetObservationByIdResponse` shall include the property according to Table 31.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/response-behavior
Requirement 53 The GetObservationById operation response shall contain O&M observations that have a `gml:identifier` value equal to one of the values of the `observation` parameter provided in the operation request.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/empty-response
Requirement 54 An instance of `GetObservationByIdResponse` type shall be empty if none of the observations associated with the SOS fulfill the *GetObservationById* parameter specified by the client.

Table 31: Properties of GetObservationByIdResponse data type

Name	Definition	Data type and values	Multiplicity and use
observation	O&M Observation matching one of the values of the observation request parameter specified by the client.	OM_Observation see [OGC 10-004r3/ISO 19156]	Zero or many (optional)

9.2.1.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/gobi/exception
Requirement 55 When an SOS server encounters an error while performing a *GetObservationById* operation, it shall return an exception message as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in Listing 4. The meaning of each exception code shall be as defined in OGC 06-121r3 and OGC 09-001.

Listing 4: exception codes applicable to the GetObservationById operation

- OperationNotSupported
- MissingParameterValue
- InvalidParameterValue
- OptionNotSupported
- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported

9.2.1.4 Examples

Example 17 An example request of the XML implementation of the *GetObservationById* operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/enhancedOperations/GetObservationById.xml

10 Transactional Extension

10.1 Requirements Class: InsertionCapabilities

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/insertionCap	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Requirement	http://www.opengis.net/spec/SOS/2.0/req/insertionCap/**structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/insertionCap/**capabilities-inclusion**

10.1.1 InsertionCapabilities Section

The `InsertionCapabilities` section states which feature types, observation types, and result types are supported by the SOS server for the insertion of new data. It is needed by the *Sensor Insertion* (Subclause 10.2), *Observation Insertion* (Subclause 10.4), and *Result Insertion* (Subclause 11.1) requirements classes.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/insertionCap/structure
Requirement 56 The `InsertionCapabilities` section shall be structured as defined in Figure 13 and Table 32.

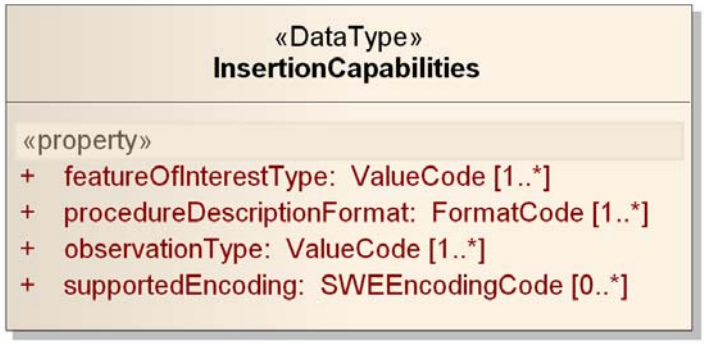


Figure 13: Data type of InsertionCapabilities section

Table 32: Properties of InsertionCapabilities data type

Name	Definition	Data type and values	Multiplicity and use
procedureDescriptionFormat	identifier of a procedure description format that is supported by the service Listed procedure descriptions can be used when inserting a new sensor or when inserting an observation (or result template with an observation) that references a procedure encoded in a given description format.	FormatCode see OGC 09-001	One or many (mandatory)
featureOfInterestType	identifier of the feature type that is supported by the service Listed feature types can be used when adding a new feature of interest which is associated with an observation inserted through *InsertObservation* or *InsertResultTemplate*.	ValueCode see clause 8.4.1	One or many (mandatory)
observationType	identifier of the observation type (with unique result type) supported by the service Listed observation types can be used when adding new observations through *InsertObservation* or *InsertResultTemplate*.	ValueCode see clause 8.4.1	One or many (mandatory)
supportedEncoding	identifier of the result encoding supported by the service Listed result encodings can be used for new results of observations which are added to the SOS.	SWEEncoding Code see OGC 09-001	Zero or many (optional)

Requirement
http://www.opengis.net/spec/SOS/2.0/req/insertionCap/capabilities-inclusion
Requirement 57 If the service lists the `InsertionCapabilities` in its `OperationsMetadata` as a supported value for the `sections` parameter of the *GetCapabilities* operation and a *GetCapabilities* request includes the `sections` parameter with this value or the "all" value or the request was made without the `sections` parameter, then the `InsertionCapabilities` shall be provided in the `Capabilities` document that is returned by the service. More specifically, it shall then be placed in the `sos:Capabilities/sos:extension` property.

10.2 Requirements Class: Sensor Insertion

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core

Dependency	http://www.opengis.net/spec/SOS/2.0/req/insertionCap
Dependency	http://www.opengis.net/doc/IS/SWES/2.0/clause/12
Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/**request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/**response**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/**exception**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/**exception-unsupported-types**

10.2.1 InsertSensor Operation

The *InsertSensor* operation allows a client to register a new sensor system at the SOS. The operation is part of the Sensor Insertion requirements class. Sensor observations can only be inserted for sensors that have first been inserted in the SOS.

The conceptual model of the *InsertSensor* operation is shown in the following UML diagram.

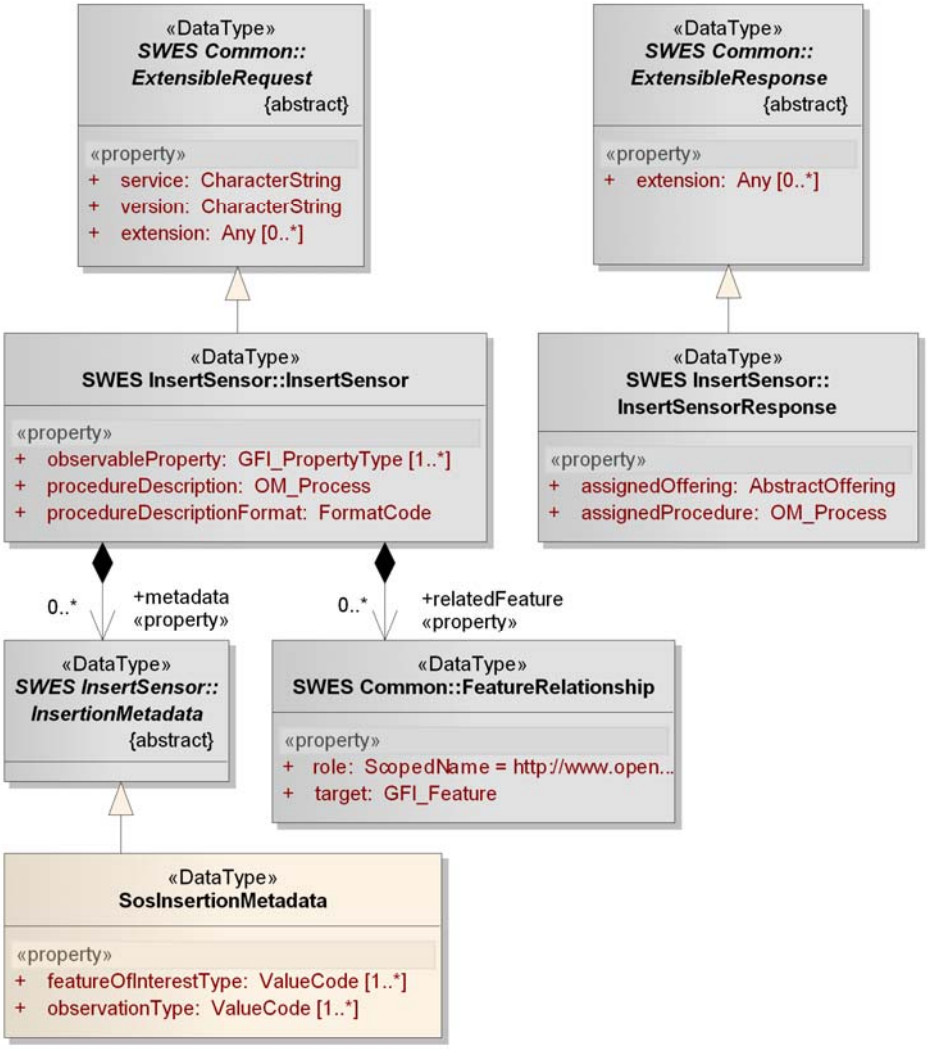


Figure 14: Data types of InsertSensor operation

The `InsertionCapabilities` section in the Capabilities document of the service lists the types and formats which can be used by the client when inserting a new sensor at the SOS server.

10.2.1.1 Request

The base structure of the *InsertSensor* operation is defined in Clause 13 of [OGC 09-001].

Requirement
http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/request-structure
Requirement 58 For the SOS the *InsertSensor* operation shall be used as specified in Clause 13 of [OGC 09-001]. Additionally, the SOS extends the abstract `InsertionMetadata` data type through the type `SosInsertionMetadata`, with properties as defined in Table 33. This type shall be included in `InsertSensor` requests sent to the SOS server. The `featureOfInterestTypes` and `observationTypes` (with unique result type) in the `SosInsertionMetadata` element of the request shall determine which are used by the sensor to encode observations. The `service` attribute in the `InsertSensor` request that is sent to the SOS server shall have the value "SOS" and the `version` attribute shall have the value "2.0.0".

Table 33 Properties of SosInsertionMetadata data type

Name	Definition	Data type and values	Multiplicity and use
featureOfInterestType	identifier of feature of interest type associated with observation produced by the sensor	ValueCode see clause 8.4.1	One or many (mandatory)
observationType	identifier of observation type (with unique result type) which is produced by the sensor	ValueCode see clause 8.4.1	One or many (mandatory)

10.2.1.2 Response

Requirement
http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/response
Requirement 59 The *InsertSensor* operation shall return a response as defined in Subclause 13.2.3 of [OGC 09-001].

10.2.1.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/exception
Requirement 60 When an SOS server encounters an error while performing an *InsertSensor* operation, it shall return an exception message as specified in Subclause 13.3 of [OGC 09-001].

Requirement
http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/exception-unsupported-types
Requirement 61 If one of the `featureOfInterestTypes` or `observationTypes` specified by the client in the `SosInsertionMetadata` element provided in the `InsertSensor` request are not supported by the SOS server (supported values are listed in the InsertionCapabilities section of the Capabilities document) then an exception shall be returned with the exception code "InvalidParameterValue" and locator value "featureOfInterestType", or "observationType".

10.2.1.4 Examples

Example 18 An example of request and response of the XML implementation of the *InsertSensor* operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/transactional/InsertSensor1.xml

http://schemas.opengis.net/sos/2.0/examples/transactional/InsertSensor1_response.xml

10.3 Requirements Class: Sensor Deletion

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/**request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/**obsoffering-deletion**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/**observation-deletion**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/**exception**

10.3.1 DeleteSensor Operation

The *DeleteSensor* operation allows a client to delete a sensor system from the SOS. The operation is part of the Sensor Deletion requirements class.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/request-structure
Requirement 62 The *DeleteSensor* request shall be implemented as defined in Clause 14 of the OGC SWE Service Model [OGC 09-001] specification.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/obsoffering-deletion

Requirement 63 After a successful deletion of a sensor, all `ObservationOfferings` which have been associated with the deleted sensor shall no longer be listed in the Capabilities of the SOS server.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/observation-deletion
Requirement 64 When the sensor is successfully deleted by this operation, all observations which have been produced by the sensor shall no longer be accessible or discoverable through the SOS.

10.3.1.1 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/exception
Requirement 65 When an SOS server encounters an error while performing a *DeleteSensor* operation, it shall return an exception message as specified in Subclause 14.3 of [OGC 09-001].

10.3.1.2 Examples

Example 19 An example of request and response of the XML implementation of the *DeleteSensor* operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/transactional/DeleteSensor1.xml

http://schemas.opengis.net/sos/2.0/examples/transactional/DeleteSensor1_response.xml

10.4 Requirements Class: Observation Insertion

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/obsInsertion	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Dependency	http://www.opengis.net/spec/SOS/2.0/req/insertionCap
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/**request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/**supported-types**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/**property-constellation**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/**multiple-offerings**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/**response-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/**exception**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/**exception-supported-types**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/**exception-property-constellation**

10.4.1 InsertObservation Operation

The *InsertObservation* operation allows clients to insert new observations for a registered sensor system.

The conceptual model of the *InsertObservation* operation is shown in the following UML diagram.

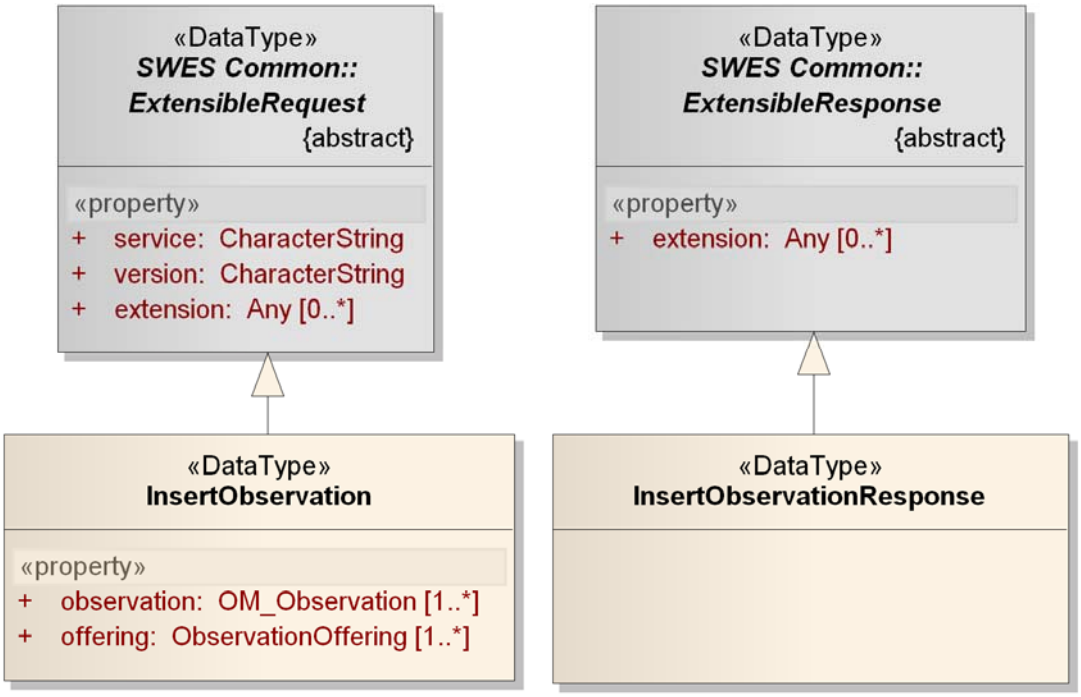


Figure 15: Data types of InsertObservation operation

The `InsertionCapabilities` section lists the types (of observations and features of interest) which can be used by the client when inserting observations at the SOS server.

10.4.1.1 Request

The SOS `InsertObservation` data type derives from the SWES `ExtensibleRequest` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/request-structure
Requirement 66 In addition to the properties inherited from SWES `ExtensibleRequest`, the SOS `InsertObservation` operation request shall include the properties according to Table 34.

Requirement

http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/supported-types
Requirement 67 The type of the inserted observation (with unique result type) shall be supported by the SOS server (and hence listed in the `InsertionCapabilities` section) AND shall be one of the types defined for (each of) the `ObservationOffering`(s) to which the observation is added.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/property-constellation
Requirement 68 For a particular constellation of `procedure`, `observedProperty` and `ObservationOffering`, observations shall always be encoded in the same `observationType` . The *InsertObservation* operation shall check whether a differing `observationType` is inserted for the same constellation of `procedure`, `observedProperty` and `ObservationOffering` that was inserted before.

NOTE If this requirement was missing a client could, for example, insert observations of type `OM_TruthObservation` today and of type `OM_Measurement` tomorrow for the same `procedure`, `observedProperty`, `ObservationOffering` constellation. In subsequent calls of *GetObservation* to request data for today and tomorrow it would be unclear how to encode the response.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/multiple-offerings
Requirement 69 If multiple offerings are specified for the sensor of the observations which should be inserted, all specified observations shall be added to all specified offerings.

Table 34: Properties of InsertObservation data type

Name	**Definition**	**Data type and values**	**Multiplicity and use**
offering	Pointer to an ObservationOffering to which the observation(s) shall be added.	ObservationOffering [id] see Subclause 8.1.2.2	One or many (mandatory)
observation	Observation to insert	OM_Observation see [OGC 10-004r3/ISO 19156]	One or many (mandatory)
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details on identifier handling.			

10.4.1.2 Response

The SOS `InsertObservationResponse` data type derives from the SWES `ExtensibleResponse` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/response-structure

Requirement 70 The SOS `InsertObservationResponse` shall contain the properties inherited from SWES `ExtensibleResponse`. No other properties are defined by this standard for the InsertObservationResponse type. The return of an instance of the InsertObservationResponse indicates successful insertion of the observations.

10.4.1.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/exception
Requirement 71 When an SOS server encounters an error while performing an *InsertObservation* operation, it shall return an exception message as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in Listing 5. The meaning of each exception code shall be as defined in OGC 06-121r3 and OGC 09-001.

Listing 5: exception codes applicable to the InsertObservation operation

- OperationNotSupported
- MissingParameterValue
- InvalidParameterValue
- OptionNotSupported
- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/exception-supported-types
Requirement 72 If the `observationType` of the observation which should be inserted is not supported by the SOS or the type is not listed in the `ObservationOfferings` of the sensor in the `Capabilities`, an exception shall be returned with the `ExceptionCode` "InvalidParameterValue" and `locator` value "observationType".

Requirement
http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/exception-property-constellation

Requirement 73 If an `observationType` is inserted for the same constellation of `procedure`, `observedProperty` as well as `ObservationOffering` and that `observationType` is different than in previous insertions of observations with that property constellation, an exception shall be returned with the `ExceptionCode` "InvalidParameterValue" and `locator` value "observationType".

10.4.1.4 Examples

Example 20 An example of request and response of the XML implementation of the *InsertObservation* operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/transactional/InsertObservation1.xml

http://schemas.opengis.net/sos/2.0/examples/transactional/InsertObservation1_response.xml

11 Result Handling Extension

11.1 Requirements Class: Result Insertion

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Dependency	http://www.opengis.net/spec/SOS/2.0/req/insertionCap
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-supported-types**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-property-constellation**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-obs-template-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-result-structure-phenomenonTime**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-result-structure-resultTime**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-observation-time-provisioning**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-response-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-exception**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-exception-supported-types**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**irt-exception-property-constellation**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**ir-request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**ir-resultValues-content**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**ir-response**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**ir-exception**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/**ir-exception-unknown-result-structure**

11.1.1 InsertResultTemplate Operation

The *InsertResultTemplate* operation allows clients to upload a template for result values. Result values which conform to this template can be inserted into the SOS using subsequent calls of the *InsertResult* operation. The *InsertResultTemplate* request includes the pointer to an ObservationOffering into which the results will be inserted. The inserted result template contains not only the description of the result structure and encoding but also an observation template with the complete observation metadata such as procedure, feature of interest and observed property for the observations corresponding to the results.

The conceptual model of the *InsertResultTemplate* operation is shown in the following UML diagram.

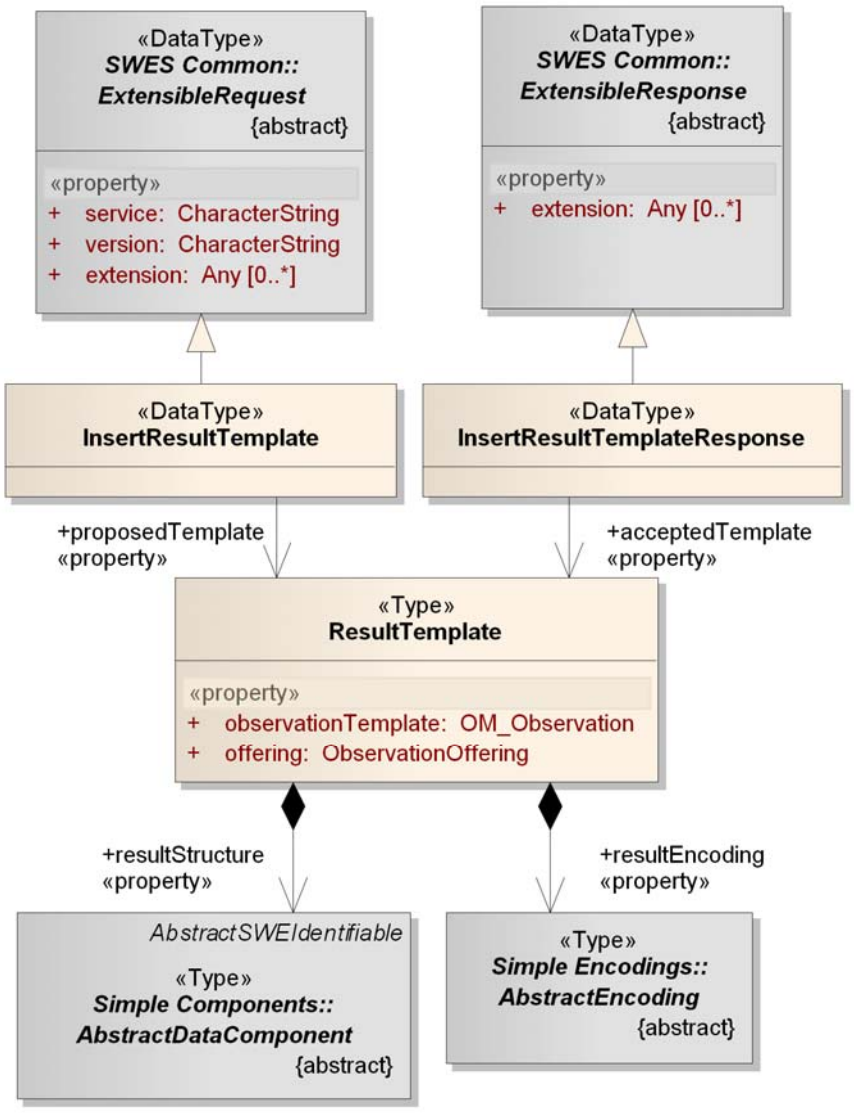


Figure 16: Data types of InsertResultTemplate operation

The `InsertionCapabilities` section lists the types (of observations and features of interest) which can be used by the client when inserting observations – which, ultimately, is the purpose of invoking the InsertResultTemplate operation - at the SOS server.

11.1.1.1 Request

The SOS `InsertResultTemplate` data type derives from the SWES `ExtensibleRequest` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-request-structure
Requirement 74 In addition to the properties inherited from SWES `ExtensibleRequest`, the SOS `InsertResultTemplate` operation request shall include the property according to Table 35.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-supported-types
Requirement 75 The type of the registered observation (template), the type of its result and the specified `resultEncoding` shall be supported by the SOS server as advertised in the `InsertionCapabilities` section). The type of the observation shall match the observationType as listed in the `ObservationOffering` to which the observation , built from the results which will be inserted, is added.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-property-constellation
Requirement 76 Result values inserted for a particular `procedure` shall not be given in different result structures for the same `observedProperty` and `ObservationOffering`. While processing an *InsertResultTemplate* request, an SOS server shall check that no differing `resultStructure` is used for a constellation of `procedure`, `observedProperty` and `ObservationOffering` that was inserted before.

NOTE Differing `resultStructure` can only be used for the same constellation of `observedProperty` and `procedure` if different `ObservationOffering`s are used.

NOTE If Requirement 76 were missing a client could upload today's results structured, for example, as a `DataArray` [OGC 08-094] and tomorrow as a `DataRecord` [OGC 08-094] for the same `procedure`, `observedProperty`, `ObservationOffering` constellation. In subsequent calls of *GetResult* to request data for today and tomorrow it would be unclear how to encode the response.

Table 35: Properties of InsertResultTemplate data type

Name	Definition	Data type and values	Multiplicity and use
proposedTemplate	Specifies the observation metadata and also information about the structure and encoding of the result, but no result value	ResultTemplate see Table 36	One (mandatory)

Table 36: Properties of ResultTemplate data type

Name	Definition	Data type and values	Multiplicity and use
offering	Pointer to ObservationOffering to which the results and corresponding observations shall be added.	ObservationOffering [id] see Subclause 8.1.2.2	One (mandatory)
observationTemplate	template which contains observation metadata that is used to form complete observations with result values that are inserted later on	OM_Observation see [OGC 10-004r3/ISO 19156] Requirement 77 defines further value constraints	One (mandatory)
resultStructure	Specifies the structure of the results which will be inserted in subsequent *InsertResult* calls for the observationTemplate	AbstractDataComponent see [OGC 08-094]	One (mandatory)
resultEncoding	Specifies the encoding of the results which will be inserted in subsequent *InsertResult* calls for the observationTemplate	AbstractEncoding see [OGC 08-094]	One (mandatory)
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details on identifier handling.			

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-obs-template-structure
Requirement 77 The observation that is provided by the client in the ResultTemplate shall have null as value of `om:phenomenonTime`, `om:resultTime` and `om:result`. For the first two properties, the `nilReason` shall be set to the value 'template'. The `procedure`, `featureOfInterest` and `observedProperty` of the observation template shall not be empty. Other observation properties can be set by the client.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-result-structure-phenomenonTime
Requirement 78 The `resultStructure` in the `ResultTemplate` shall have at least a `swe:Time` or `swe:TimeRange` component with definition property set to the value "http://www.opengis.net/def/property/OGC/0/PhenomenonTime". The value of this component shall

be used by the service to populate the om:phenomenonTime property of the observation template for each new result block the client is going to insert via the InsertResult operation.

Note: a swe:Time defines a specific point in time and thus maps to a TM_Instant / gml:TimeInstant which is a valid value of the om:phenomenonTime and om:resultTime. Likewise, a swe:TimeRange defines an interval of time and thus maps to a TM_Period / gml:TimePeriod, which is a valid value of the om:phenomenonTime as well – but not of the om:resultTime.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-result-structure-resultTime
Requirement 79 If the resultStructure in the ResultTemplate has a swe:Time component with definition property set to the value "http://www.opengis.net/def/property/OGC/0/ResultTime" then the value of this component shall be used by the service to populate the om:resultTime property of the observation template for each new result block the client is going to insert via the InsertResult operation. If no such component is contained in the resultStructure then the service shall use the om:phenomenonTime as value of the om:resultTime (at least the phenomenon time has to be provided in each ResultTemplate). In case the om:phenomenonTime is not a TimeInstant, an InvalidParameterValue exception shall be returned, with locator 'resultTime'.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-observation-time-provisioning
Requirement 80 A client shall encode the om:phenomenonTime as a swe:Time or swe:TimeRange component with definition "http://www.opengis.net/def/property/OGC/0/PhenomenonTime". in the resultStructure that it proposes to the service in the *InsertResultTemplate* operation request. If any of the observation results that the client intends to send to the service via the *InsertResult* operation is going to have a resultTime that is different to the phenomenonTime then the resultStructure of the ResultTemplate shall also have a swe:Time component with definition "http://www.opengis.net/def/property/OGC/0/ResultTime".

11.1.1.2 Response

The SOS InsertResultTemplateResponse data type derives from the SWES ExtensibleResponse data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-response-structure
Requirement 81 In addition to the properties inherited from SWES ExtensibleResponse, the SOS InsertResultTemplateResponse shall include the property according to Table 37.

Table 37: Properties of InsertResultTemplateResponse data type

Name	Definition	Data type and values	Multiplicity and use

acceptedTemplate	Pointer to the ResultTemplate which has been accepted and registered at the SOS server. This ResultTemplate can be used in subsequent *InsertResult* requests.	ResultTemplate [id]	One (mandatory)
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details on identifier handling.			

11.1.1.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-exception
Requirement 82 When an SOS server encounters an error while performing an *InsertResultTemplate* operation, it shall return an exception message as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in Listing 6. The meaning of each exception code shall be as defined in OGC 06-121r3 and OGC 09-001.

Listing 6: exception codes applicable to the InsertResultTemplate operation

- OperationNotSupported
- MissingParameterValue
- InvalidParameterValue
- OptionNotSupported
- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-exception-supported-types
Requirement 83 If the `observationType` of the observation template is not supported by the SOS or the type is not listed in the `ObservationOfferings` of the sensor in the `Capabilities`, an exception shall be returned with `ExceptionCode` "InvalidParameterValue" and `locator` value "observationType".

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-exception-property-constellation
Requirement 84 If a result template with differing `observationType` or (SWE Common

encoded) result structure is inserted for the same constellation of `procedure`, `observedProperty` and `ObservationOffering` (for which observations already exist) an exception shall be returned with the `ExceptionCode` "InvalidParameterValue" and `locator` value "proposedTemplate".

11.1.1.4 Examples

Example 21 An example of request and response of the XML implementation of the *InsertResultTemplate* operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/resultHandling/InsertResultTemplate1.xml

http://schemas.opengis.net/sos/2.0/examples/resultHandling/InsertResultTemplate1_response.xml

11.1.2 InsertResult Operation

The *InsertResult* operation allows a client to insert new observations for a sensor system by inserting only the results of the observations. The operation is useful if most of the metadata contained in the observations remain the same and/or the communication bandwidth and processing power of the client is limited.

Before invoking the *InsertResult* operation, the sensor has to be associated with the SOS and has to be listed in the Capabilities document. This can be done by invoking the *InsertSensor* operation.

Also, before invoking the *InsertResult* operation the *InsertResultTemplate* operation has to be invoked once for defining the structure of the result elements which are inserted afterwards.

The conceptual model of the *InsertResult* operation is shown in the following UML diagram.

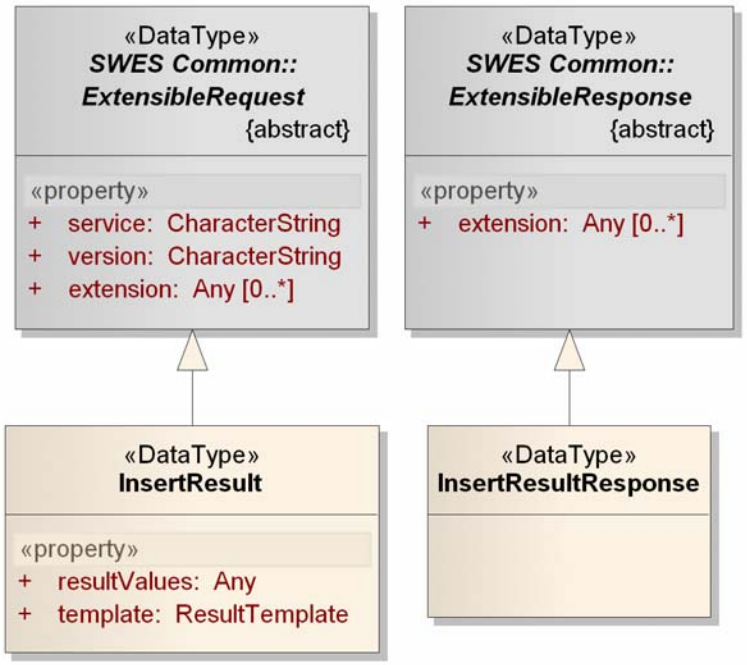


Figure 17: Data types of InsertResult operation

11.1.2.1 Request

The SOS `InsertResult` data type derives from the SWES `ExtensibleRequest` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-request-structure
Requirement 85 In addition to the properties inherited from SWES `ExtensibleRequest`, the SOS `InsertResult` operation request shall include the properties according to Table 38.

The request includes a pointer to the before uploaded template which describes the structure and encoding of the results. Further, it includes an element which contains the results of observations which shall be inserted.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-resultValues-content
Requirement 86 The client shall encode the values of the observation result that is to be inserted via the *InsertResult* operation according to the `resultStructure` of the `ResultTemplate` it points to

in the `InsertResult` request.

Table 38: Properties of InsertResult data type

Name	Definition	Data type and values	Multiplicity and use
template	Pointer to the template defining the structure and encoding of the results.	ResultTemplate [id] see Table 36	One (mandatory)
resultValues	The results of observations which shall be inserted.	Any type	One (mandatory)
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details on identifier handling.			

11.1.2.2 Response

The SOS `InsertResultResponse` data type derives from the SWES `ExtensibleResponse` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-response
Requirement 87 The SOS `InsertResultResponse` shall contain the properties inherited from SWES ExtensibleResponse. No other properties are defined by this standard for the InsertResultResponse type. The return of an instance of the `InsertResultResponse` shall indicate a successful insertion of the result values.

11.1.2.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-exception
Requirement 88 When an SOS server encounters an error while performing an *InsertResult* operation, it shall return an exception message as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in Listing 7. The meaning of each exception code shall be as defined in OGC 06-121r3 and OGC 09-001.

Listing 7: exception codes applicable to the InsertResult operation

- OperationNotSupported
- MissingParameterValue
- InvalidParameterValue
- OptionNotSupported
- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-exception-unknown-result-structure
Requirement 89 If the structure and encoding of the inserted results does not adhere to the `resultStructure` and `resultEncoding` defined for the `ResultTemplate` pointed to in the `InsertResult` request (and inserted beforehand through an *InsertResultTemplate* call) an exception shall be thrown with the exceptionCode "InvalidParameterValue" with locator value "template".

11.1.2.4 Examples

Example 22 An example of request and response of the XML implementation of the *InsertResult* operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/resultHandling/InsertResult1.xml

http://schemas.opengis.net/sos/2.0/examples/resultHandling/InsertResult1_response.xml

11.2 Requirements Class: Result Retrieval

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Dependency	http://www.opengis.net/spec/SOS/2.0/req/insertionCap
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**grt-request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**grt-filter-context**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**grt-response-structure**

Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**grt-exception**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**gr-request-structure**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**gr-parameters**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**gr-omitting-parameters**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**gr-filter-context**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**gr-response-empty**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/**gr-exception**

The functionality defined in this requirements class enables clients to efficiently retrieve pure SWE Common encoded observation result values from a SOS. The concept of the two operations defined in this requirements class is as follows:

- *GetResultTemplate* operation – With this operation, the client retrieves information on the exact structure used by a specific procedure to generate a new observation result. The result structure is unique per combination of procedure, offering and observed property. Furthermore, the operation response also includes the definition of the encoding that the service will use to return result values in a GetResult operation response.

- *GetResult* operation – With this operation, the client identifies the observations of which the pure SWE Common encoded result values are to be retrieved. The request contains parameters to identify the offering as well as observed property of observations that the client is interested in. Optional request parameters enable the client to further subset these observations. With a temporal filter, for example, the client is able to retrieve the result values of all new observations (with specific observed property) that were added to the offering since the last invocation of the GetResult operation.

The information returned in both the GetResultTemplate and GetResult operation responses enable the client to construct a complete SWE Common DataArray that aggregates the result values of observations retrieved in a single GetResult request – see following figure.

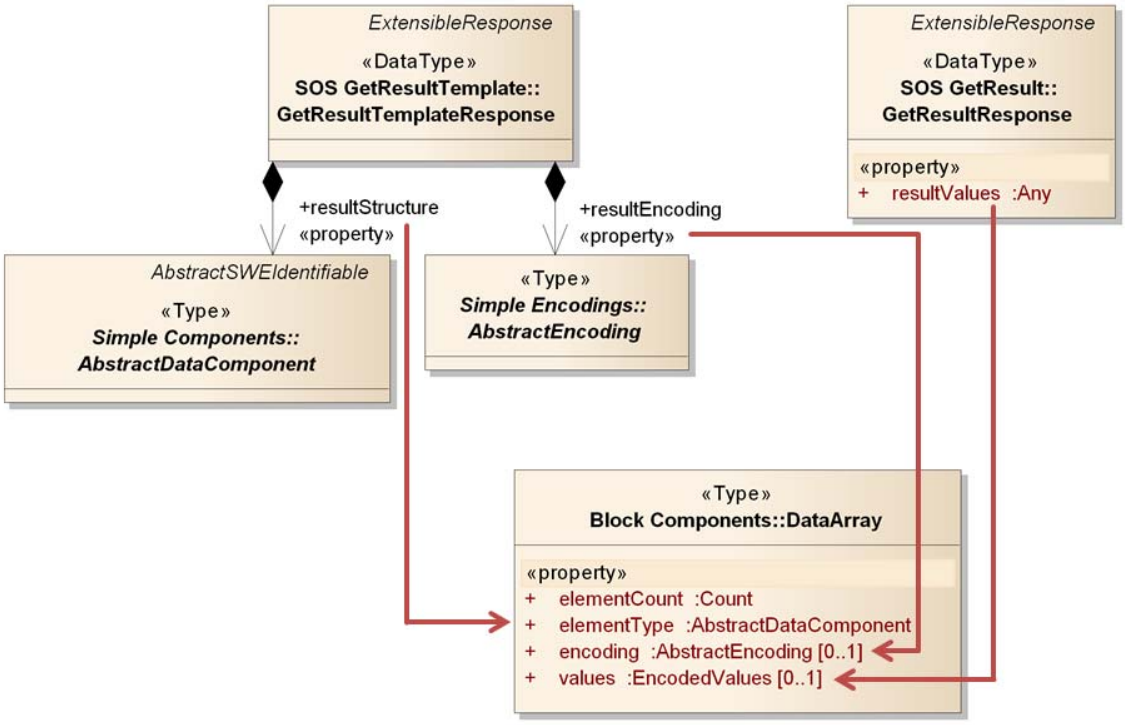


Figure 18: Constructing a SWE Common DataArray with the contents of a GetResultTemplateResponse and GetResultResponse

The figure shows that:

- the GetResultTemplateResponse.resultStructure is used to populate the DataArray.elementType,
- the GetResultTemplateResponse.resultEncoding is used to populate the DataArray.encoding and
- the GetResultResponse.resultValues is used to populate the DataArray.values.

The DataArray.elementCount is variable because it depends on the actual number of result values returned in a GetResultResponse. It is not directly provided as an element of the GetResultResponse. However, following requirement http://www.opengis.net/spec/SWE/2.0/req/general-encoding-rules/array-size-encoding-rule from [OGC 08-094], the exact array size of a variable size DataArray is given in the encoded values.

With the DataArray thus defined, a client can decode and use the information that is contained in the aggregated (via the GetResultResponse) observation result values.

11.2.1 GetResultTemplate Operation

The *GetResultTemplate* operation allows the client to retrieve the structure and encoding of the results which will be returned during later *GetResult* operations for specified observed property

and ObservationOffering. By requesting this information from the SOS server, the client is enabled to interpret the result values retrieved from subsequent *GetResult* calls.

NOTE: there is an inherent assumption that the SWE Common defined result structure of observations belonging to one observation offering is the same if the observed property of these observations is the same. In other words, observations generated by one procedure for a certain observed property may not have a different SWE Common defined result structure unless they are associated with different offerings.

The conceptual model of the *GetResultTemplate* operation is shown in the following UML diagram.

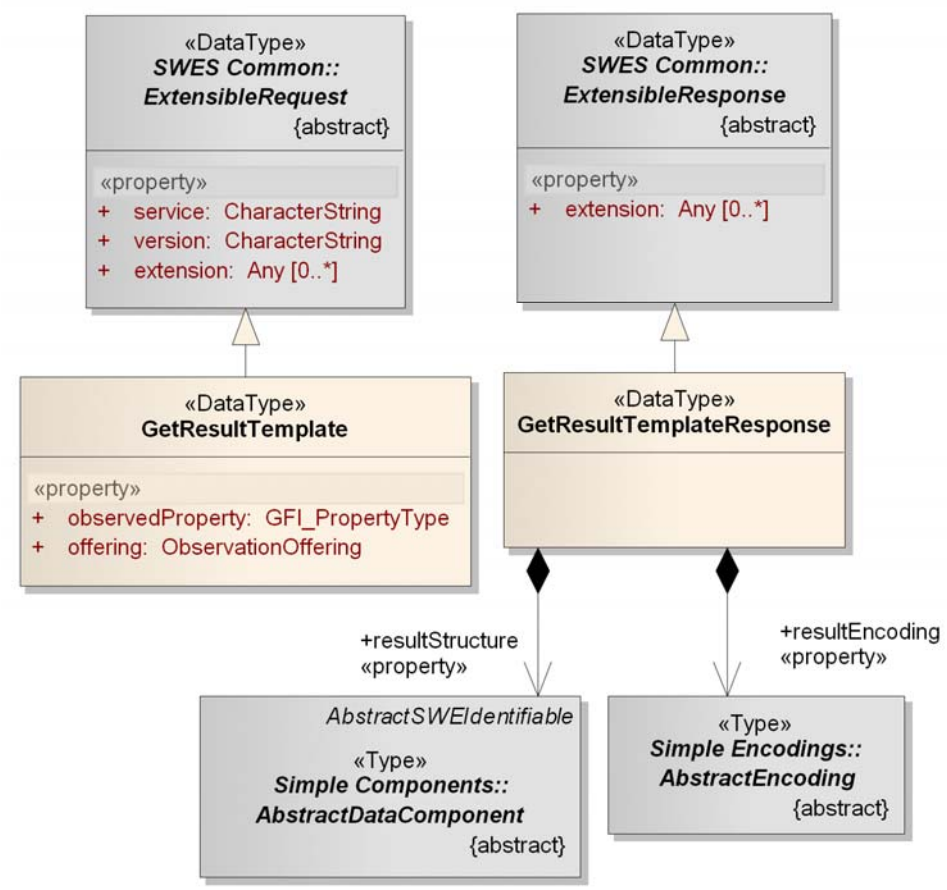


Figure 19: Data types of GetResultTemplate operation

11.2.1.1 Request

The SOS `GetResultTemplate` data type derives from the SWES `ExtensibleRequest` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/grt-request-structure
Requirement 90 In addition to the properties inherited from SWES `ExtensibleRequest`, the SOS `GetResultTemplate` operation request shall include the properties according to Table 39.

Table 39: Properties of GetResultTemplate data type

Name	Definition	Data type and values	Multiplicity and use
offering	Pointer to an ObservationOffering for which results will be requested in subsequent *GetResult* calls. Since an ObservationOffering is associated with exactly one procedure, this parameter indirectly identifies the procedure for which results will be requested.	ObservationOffering [id] see Subclause 8.1.2.2	One (mandatory)
observedProperty	Pointer to an observed property for which the results serve values.	GFI_PropertyType [id] see [OGC 10-004r3/ISO 19156]	One (mandatory)
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details on identifier handling.			

The `offering` parameter is used in the request to identify a specific procedure. This is due to the fact that a procedure may be associated with multiple offerings and that the encoding for the same `observedProperty` may be different per offering. For example, in one offering the service may provide access to OM_Measurement data while in another offering the service may use a simple SWEScalarObservation as defined in clause 7.11 of [OGC 10-025r1] with a SWE Quantity as result value. Thus, the combination of `offering` and `observedProperty` uniquely identifies both the procedure and the result structure used by the procedure to encode the results of observations that are associated with the offering.

In addition, the GetResultTemplate and GetResult operations are designed to only allow retrieval of SWE Common encoded result values from the observations of one specific offering at a time. This design prevents potential confusion if observations from different offerings use different result structures.

It is necessary to establish a linkage between the GetResultTemplate request model and the underlying SOS data model.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/grt-filter-context
Requirement 91 The default context for the ValueReferences used in the expression of a filter operator (see OGC 09-026r1/ISO 19143 for further details) for the GetResultTemplate request parameters that have a pre-defined target property shall be as defined in Table 40.

Table 40: default context for GetResultTemplate request parameters with pre-defined target property

Filter Property	Target property for PropertyIsEqualTo comparison operator
offering	Capabilities.contents.offering
observedProperty	OM_Observation.observedProperty

NOTE: in an XML based SOS implementation the filter context can be defined via XPath expressions – see Table 48 in Subclause 13.1.

11.2.1.2 Response

The SOS `GetResultTemplateResponse` data type derives from the SWES `ExtensibleResponse` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/grt-response-structure
Requirement 92 In addition to the properties inherited from SWES `ExtensibleResponse`, the SOS `GetResultTemplateResponse` shall include the properties according to Table 41.

Table 41: Properties of GetResultTemplateResponse data type

Name	Definition	Data type and values	Multiplicity and use
resultStructure	The structure of the results which may be requested in subsequent *GetResult* calls.	AbstractDataComponent see [OGC 08-094]	One (mandatory)
resultEncoding	The encoding of the results which may be requested in subsequent Get*Result* calls.	AbstractEncoding see [OGC 08-094]	One (mandatory)

Note: different to the result insertion extension, in the result retrieval extension the intention is not to construct complete observations with the result values returned via the `GetResultResponse`. Consequently, there is no requirement on SOS services to include components in the `sos:GetResultResponse/sos:resultStructure` to provide the phenomenon or result time of observations in. Such components may of course be part of an observation result. As such, SWE Common defined observation results can easily be retrieved via the GetResultTemplate/GetResult operations.

11.2.1.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/grt-exception
Requirement 93 When an SOS server encounters an error while performing a *GetResultTemplate* operation, it shall return an exception message as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in

Requirement 94 Table 42: InvalidPropertyOfferingCombination Exception

exceptionCode value	Meaning of code	"locator" value
InvalidPropertyOfferingCombination	The service is unable to fulfill the request because the observations with the specific combination of observedProperty and offering as requested by the client do NOT have a SWE Common encoded result.	None, omit "locator" parameter

Requirement 95 Listing 8. The meaning of each exception code shall be as defined in OGC 06-121r3,OGC 09-001 and Table 42 in this standard.

Table 42: InvalidPropertyOfferingCombination Exception

exceptionCode value	Meaning of code	"locator" value
InvalidPropertyOfferingCombination	The service is unable to fulfill the request because the observations with the specific combination of observedProperty and offering as requested by the client do NOT have a SWE Common encoded result.	None, omit "locator" parameter

Listing 8: exception codes applicable to the GetResultTemplate operation

- OperationNotSupported
- MissingParameterValue
- InvalidParameterValue
- OptionNotSupported
- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported
- InvalidPropertyOfferingCombination

11.2.1.4 Examples

Example 23 An example of request and response of the XML implementation of the *GetResultTemplate* operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/resultHandling/GetResultTemplate1.xml

http://schemas.opengis.net/sos/2.0/examples/resultHandling/GetResultTemplate1_response.xml

http://schemas.opengis.net/sos/2.0/examples/resultHandling/GetResult1_resultingDataArray.xml

11.2.2 GetResult Operation

The *GetResult* operation allows retrieving just the result values of observations without the entire metadata of the observation. It offers five parameters: ObservationOffering, feature of interest, observed property, as well as temporal and spatial filter, which can be used to filter the observations of which result values are returned by the SOS. For being able to interpret the returned result values, the client can invoke the *GetResultTemplate*, before calling *GetResult*, to retrieve the structure and encoding of the results returned for the specified ObservationOffering and observed property (the other parameters do not influence the structure or encoding of the results).

The conceptual model of the *GetResult* operation is shown in the following UML diagram.

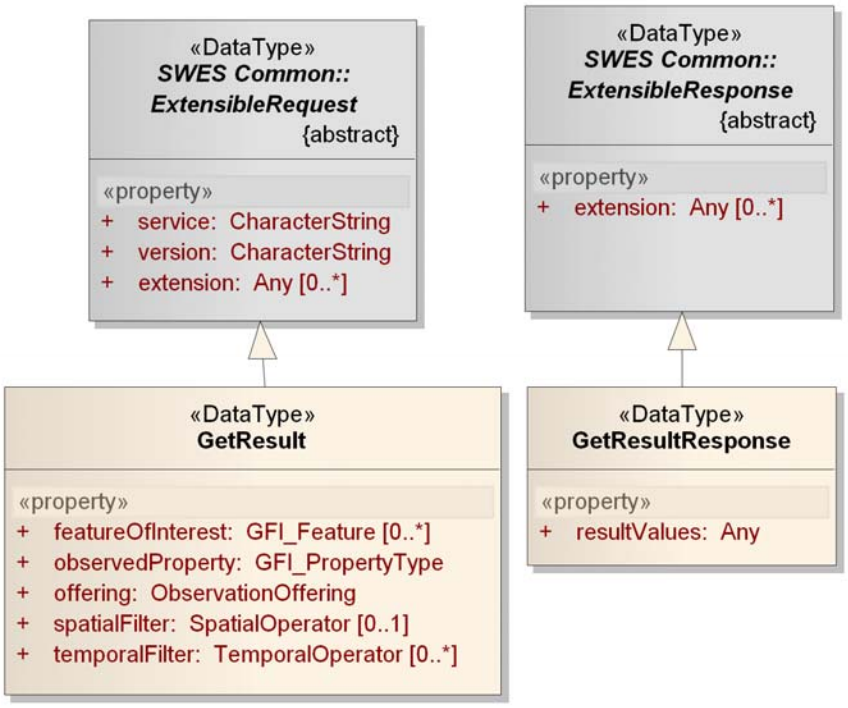


Figure 20: Data types of GetResult operation

11.2.2.1 Request

The SOS `GetResult` data type derives from the SWES `ExtensibleRequest` data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-request-structure
Requirement 96 Table 43 Table 43.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-parameters
Requirement 97 The SOS returns the results of all observations that match the specified parameter values. The request parameters shall be connected with an implicit **AND**. The values of each of the parameters are connected with an implicit **OR**.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-omitting-parameters
Requirement 98 If an optional parameter of a *GetResult* request is not specified by the client, the filter (represented by the parameter) shall not be applied to the result set which will be returned by the SOS server.

NOTE: An implementation of an SOS server may return an exception message as specified in Clause 15 of [OGC 09-001] if the response of a *GetResult* request would be too big to be reasonably sent to a client.

Example 24 Resulting from Requirement 97 and Requirement 98 an example abstract *GetResult* request looks like this:

GetResult (`offering` := *weatherstation_in_my_backyard*
AND `observedProperty` := *temperature*)

This request returns the results of the observations in ObservationOffering “weatherstation_in_my_backyard”, of all spatial extent and all features of interest, which were measured for the property “temperature”.

Table 43: Properties of GetResult data type

Name	Definition	Data type and values	Multiplicity and use
offering	Pointer to an ObservationOffering advertised in the Capabilities document for which results are requested.	ObservationOffering [id] see Table 17	One (mandatory)

observedProperty	Pointer to an observedProperty of the observations whose results are requested.	GFI_PropertyType [id] see [OGC 10-004r3/ISO 19156]	One (mandatory)
temporalFilter	Specifies a filter for a time property of observations whose results are requested.[1] This property is defined in the valueReference element of the TemporalOperator.	TemporalOperator see [OGC 09-026r1/ISO 19143]	Zero or more (optional)
featureOfInterest	Pointer to a feature of interest of the observations whose results are requested.	GFI_Feature [id] see [OGC 10-004r3/ISO 19156]	Zero or many (optional)
spatialFilter	Specifies a filter[2] which applies to a spatial property of an observation (or one of its properties). This property is defined in the valueReference element of the SpatialOperator.	SpatialOperator see [OGC 09-026r1/ISO 19143]	Zero or one (optional)
id) NOTE: the primary use of this property is to provide a pointer/identifier – see Annex 15 and OGC 09-001 clause 16.3.1 for further details on identifier handling. 1) The supported time ranges for the phenomenonTime as well as resultTime properties of observations are listed in the selected ObservationOffering. The supported temporal operands and operators are listed in the FilterCapabilities section of the Capabilities document. 2) A profile of this generic spatialFilter is given in clause 12. This profile defines requirements that allow the spatialFilter to be applied to the sampling location parameter of observations.			

It is necessary to establish a linkage between the GetResult request model and the underlying SOS data model.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-filter-context
Requirement 99 The default context for the ValueReferences used in the expression of a filter operator (see OGC 09-026r1/ISO 19143 for further details) for the GetResult request parameters that do not have a pre-defined target property and operator for filtering shall be as defined in Table 44. The default context for GetResult request parameters that have a pre-defined target property shall be as defined in Table 45.

Table 44: default context for GetResult request parameters without pre-defined target property

Filter Property	Context for ValueReference in filter operator
temporalFilter	OM_Observation
spatialFilter	OM_Observation

Table 45: default context for GetResult request parameters with pre-defined target property

Filter Property	Target property for PropertyIsEqualTo comparison operator
offering	Capabilities.contents.offering
observedProperty	OM_Observation.observedProperty
featureOfInterest	OM_Observation.featureOfInterest

NOTE: in an XML based SOS implementation the filter context can be defined via XPath expressions – see Table 48 in Subclause 13.1.

11.2.2.2 Response

The SOS GetResultResponse data type derives from the SWES ExtensibleResponse data type defined in OGC 09-001 and inherits its properties.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-response
Requirement 100 In addition to the properties inherited from SWES ExtensibleResponse, the SOS GetResultResponse shall include the properties according to Table 46

Table 46: Properties of GetResultResponse data type

Name	Definition	Data type and values	Multiplicity and use
resultValues	Encoded value blocks representing the result values of the observations targeted by the GetResult request.	Any	One (mandatory)

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-response-empty
Requirement 101 If no observations match the parameters of the `GetResult` request then the `resultValues` property of the `GetResultResponse` shall be empty.

11.2.2.3 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-exception

Requirement 102 When an SOS server encounters an error while performing a *GetResult* operation, it shall return an exception message as specified in Clause 15 of [OGC 09-001] with applicable exception code as defined in Listing 9. The meaning of each exception code shall be as defined in OGC 06-121r3 and OGC 09-001.

Listing 9: exception codes applicable to the GetResult operation

- OperationNotSupported
- MissingParameterValue
- InvalidParameterValue
- OptionNotSupported
- NoApplicableCode
- InvalidRequest
- RequestExtensionNotSupported

11.2.2.4 Examples

Example 25 An example of request and response of the XML implementation of the *GetResult* operation can be found here:

http://schemas.opengis.net/sos/2.0/examples/resultHandling/GetResult1.xml

http://schemas.opengis.net/sos/2.0/examples/resultHandling/GetResult1_response.xml

12 Spatial Filtering Profile

In this profile of the SOS 2.0, the observations offered by an SOS server are restricted to spatial observations which provide a well-defined parameter for carrying the sampling geometry of the observation. If an SOS server follows this profile, spatial filters can target the sampling geometries of observations.

12.1 Requirements Class: Spatial Filtering Profile

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Dependency	http://www.opengis.net/spec/OMXML/2.0/req/SpatialObservation
Requirement	http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile/**observationRestriction**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile/**observedAreaRestriction**

The requirements class "Spatial Observation Data" (http://www.opengis.net/spec/OMXML/2.0/req/SpatialObservation) defined in [OGC 10-025] Subclause 7.13 constrains the encoding of an observation to have one parameter which contains the sampling geometry of the observation. The sampling geometry represents the spatial extent where the observation result applies to. This is usually the extent of the observation's feature of interest but may also be computed or determined by other means.

Example 26 An example of such a spatial observation as defined by [OGC 10-025] with sampling location parameter is shown here:

```
<?xml version="1.0" encoding="UTF-8"?>
<om:OM_Observation>
  …
  <om:parameter>
    <om:NamedValue>
      <om:name
      xlink:href="http://www.opengis.net/def/param-name/OGC-
OM/2.0/samplingGeometry"/>
      <om:value>
        <gml:Point gml:id="SamplingPoint">
          <gml:pos srsName="urn:ogc:def:crs:EPSG::4326">52.9 7.52</gml:pos>
        </gml:Point>
      </om:value>
    </om:NamedValue>
  </om:parameter>
  …
</om:OM_Observation>
```

The following requirement restricts the observations served by an SOS server to be such spatial observations. The sampling geometry of an observation is encoded as a parameter of the observation.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile/observationRestriction
Requirement 103 All observations provided by the SOS server shall be encoded compliant to the requirements class http://www.opengis.net/spec/OMXML/2.0/req/SpatialObservation as defined in [OGC 10-025] Subclause 7.13.

The `observedArea` provided in the `Capabilities` is restricted to represent the minimum bounding box of the sampling geometries of the observations.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile/observedAreaRestriction
Requirement 104 The `observedArea` of `ObservationOfferings` provided in the `Contents` section of the Capabilities document shall contain a geometry representing the minimum bounding box of the sampling geometries of the observations that belong to the offering.

The spatial filter of a *GetObservation* or, if supported, of a *GetResult* operation request can target the sampling geometries of the observations by choosing the following value for the `valueReference` of the `spatialFilter`[13]:

`http://www.opengis.net/req/omxml/2.0/data/samplingGeometry`

Example 27 An example of request of the XML implementation of the *GetObservation* operation conformant to this profile can be found at:

http://schemas.opengis.net/sos/2.0/examples/spatialFilteringProfile/GetObservation1_spatialFilteringProfile.xml

13 Binding Extension

This Clause specifies concrete encodings and bindings for the operations of the SOS 2.0 standard. Those bindings describe how SOS 2.0 clients and servers can communicate with each other.

13.1 Requirements Class: XML Encoding

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/xml	
Target Type	XML Instances
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Dependency	http://www.opengis.net/spec/SWE/2.0/req/xsd-simple-components
Dependency	http://www.opengis.net/spec/SWE/2.0/req/xsd-simple-encodings
Dependency	http://www.opengis.net/spec/OMXML/2.0/req/observation
Dependency	http://www.opengis.net/doc/IS/SWES/2.0/clause/AnnexB
Dependency	http://www.opengis.net/doc/IS/OWS/1.1/clause/AnnexB
Dependency	urn:iso:ts:iso:19143:clause:AnnexB
Dependency	urn:iso:ts:iso:19136:clause:AnnexC
Requirement	http://www.opengis.net/spec/SOS/2.0/req/xml/**GeneralEncodingRules**

In addition to this document, this standard includes several normative XML Schema Documents. These XML Schema Documents are bundled in a zip file with the present document. After OGC acceptance of this standard, these XML Schema Documents will also be posted online at the URL http://schemas.opengis.net/sos/2.0. In the event of a discrepancy between the bundled and online versions of the XML Schema Documents, the online files shall be considered authoritative.

The UML model has been mapped to its XML Schema encoding using the rules described in OGC 09-001, resulting in the following XML Schema documents:

sos.xsd (includes the other schema through xs:include statements)

[13] The value of the value reference acts as a shortcut for the XPath expression pointing to the samplingGeometry property of the observations (om:parameter/om:NamedValue[om:name/@xlink:href= 'http://www.opengis.net/def/param-name/OGC-OM/2.0/samplingGeometry']/om:value).

sosContents.xsd

sosGetCapabilities.xsd

sosGetFeatureOfInterest.xsd

sosGetObservation.xsd

sosGetObservationById.xsd

sosGetResult.xsd

sosGetResultTemplate.xsd

sosInsertionCapabilities.xsd

sosInsertObservation.xsd

sosInsertResult.xsd

sosInsertResultTemplate.xsd

sosInsertSensor.xsd

The following table provides an overview how each of the conceptual model types defined by this standard has been realized in the XML Schema implementation.

NOTE: as the types defined in the *SOS Codes* package are not intended to be encoded as XML elements, an XML Schema file for that package is not needed and thus not available.

Table 47: XML schema implementation of types defined by this standard

UML class	object element	type	property type
SOS GetCapabilities			
Capabilities	sos:Capabilities	sos:CapabilitiesType	sos:CapabilitiesPropertyType
Contents	sos:Contents	sos:ContentsType	sos:ContentsPropertyType
GetCapabilities	sos:GetCapabilities	sos:GetCapabilitiesType	sos:GetCapabilitiesPropertyType
ObservationOffering	sos:ObservationOffering	sos:ObservationOfferingType	sos:ObservationOfferingPropertyType
SOS GetFeatureOfInterest			
GetFeatureOfInterest	sos:GetFeatureOfInterest	sos:GetFeatureOfInterestType	sos:GetFeatureOfInterestPropertyType
GetFeatureOfInterestResponse	sos:GetFeatureOfInterestResponse	sos:GetFeatureOfInterestResponseType	sos:GetFeatureOfInterestResponsePropertyType

UML class	object element	type	property type
SOS GetObservation			
GetObservation	sos:GetObservation	sos:GetObservationType	sos:GetObservationPropertyType
GetObservationResponse	sos:GetObservationResponse	sos:GetObservationResponseType	sos:GetObservationResponsePropertyType
SOS GetObservationById			
GetObservationById	sos:GetObservationById	sos:GetObservationByIdType	sos:GetObservationByIdPropertyType
GetObservationByIdResponse	sos:GetObservationByIdReponse	sos:GetObservationByIdResponseType	sos:GetObservationByIdResponsePropertyType
SOS GetResult			
GetResult	sos:GetResult	sos:GetResultType	sos:GetResultPropertyType
GetResultResponse	sos:GetResultResponse	sos:GetResultResponseType	sos:GetResultResponsePropertyType
SOS GetResultTemplate			
GetResultTemplate	sos:GetResultTemplate	sos:GetResultTemplateType	sos:GetResultTemplatePropertyType
GetResultTemplateResponse	sos:GetResultTemplateResponse	sos:GetResultTemplateResponseType	sos:GetResultTemplateResponsePropertyType
SOS InsertObservation			
InsertObservation	sos:InsertObservation	sos:InsertObservationType	sos:InsertObservationPropertyType
InsertObservationResponse	sos:InsertObservationResponse	sos:InsertObservationResponseType	sos:InsertObservationResponsePropertyType
SOS InsertResult			
InsertResult	sos:InsertResult	sos:InsertResultType	sos:InsertResultPropertyType
InsertResultResponse	sos:InsertResultResponse	sos:InsertResultResponseType	sos:InsertResultResponsePropertyType
SOS InsertResultTemplate			
InsertResultTemplate	sos:InsertResultTemplate	sos:InsertResultTemplateType	sos:InsertResultTemplatePropertyType
InsertResultTemplateResponse	sos:InsertResultTemplateResponse	sos:InsertResultTemplateResponseType	sos:InsertResultTemplateResponsePropertyType
ResultTemplate	sos:ResultTemplate	sos:ResultTemplateType	sos:ResultTemplatePropertyType

UML class	object element	type	property type
SOS InsertSensor			
SosInsertionMetadata	sos:SosInsertionMetadata	sos:SosInsertionMetadataType	sos:SosInsertionMetadataPropertyType
SOS InsertionCapabilities			
InsertionCapabilities	sos:InsertionCapabilities	sos:InsertionCapabilitiesType	sos:InsertionCapabilitiesPropertyType

Requirement
http://www.opengis.net/spec/SOS/2.0/req/xml/GeneralEncodingRules
Requirement 105 The XML encoding of the conceptual types defined in this standard shall be as defined by the XML Schema files listed and referenced in Subclause 13.1. More specifically, the XML encoding of each conceptual type shall be valid against the XML Schema definition of the according mapping as defined in Table 47.

0, Requirement 46, Requirement 51, Requirement 91 and **Error! Reference source not found.** define the context for the filter properties contained in several SOS operation requests. The context is given in the form of OCL like expressions that point to the property in the SOS data model that is used for filtering (in case that a simple object comparison is performed) or that is used as the default context for a `valueReference` which is used as expression of a filter operator (see [ISO 19143] / [OGC 09-026r1] for further details).

In an XML encoding of the SOS operation facets (requests, responses and exceptions), object comparison needs to be performed by specifying certain elements of an object to use for the comparison operation. For example, a feature of interest is identified via its `gml:identifier` value[14]. The OCL like expressions therefore map to XPath statements that identify the relevant elements or attributes whose value shall be used for comparison or that provides the default context. The following table defines the according mapping.

Table 48 – Mapping of OCL like expression to identify the target of a filter property to the according XPath expression for the XML encoding

OCL	XPath [1)]
Capabilities.contents.offering	sos:Capabilities/sos:contents/sos:Contents/swes:offering/sos:ObservationOffering/swes:identifier

[14] See Annex 15 for details on identifier handling.

OM_Observation.featureOfInterest	{observation}/om:featureOfInterest/*/gml:identifier
OM_Observation.observedProperty	{observation}/om:observedProperty/@xlink:href
OM_Observation.procedure	{observation}/om:procedure/@xlink:href
OM_Observation	{observation}/gml:identifier

1) {observation} is an observation stored by the SOS (encoded as om:OM_Observation according to ISO 19156 - or a subtype thereof)

NOTE: the filter context for temporalFilter and spatialFilter properties in various operation requests is always the {observation} itself, not the {observation}/gml:identifier

As we can see, to respond to a GetObservation request that asks for all observations that belong to offering http://www.my_namespace.org/water_gage_1_observations, the service looks up the `swes:identifier` value in its `sos:ObservationOffering` (which are listed in the `sos:Contents` section of its `sos:Capabilities`) and uses the value of this element to determine which offering matches the requested one. Internally, the service then needs to get all observations that are associated with that offering.

Likewise, if a GetObservation request asks for all observations that have the observed property *urn:ogc:def:phenomenon:OGC:water_level* then the service compares this URI with the value of the `xlink:href` attribute on the `om:observedProperty` element contained in its XML encoded observations to determine which observations match the request.

13.2 Requirements Class: Core KVP Binding

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/kvp-core	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Dependency	http://www.opengis.net/spec/SOS/2.0/req/xml
Dependency	urn:ietf:2396
Dependency	urn:iso:ts:iso:8601
Dependency	http://www.opengis.net/doc/IS/OWS/1.1/clause/10.2.3
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**general**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**url-encoded-values**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**case-sensitivity**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**namespaces**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**gc-request**

Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**gc-response**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**ds-request**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**ds-time-encoding**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**ds-response**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**go-request**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**go-bbox-encoding**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**go-temporalFilter-encoding**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/**go-response**

This requirements class defines how to invoke SOS 2.0 operations over HTTP GET with key/value pair (KVP) encoding. This KVP binding is defined for the operations *GetCapabilities*, *DescribeSensor*, and *GetObservation*.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/general
Requirement 106 For this KVP binding, the general rules defined in Subclause 11.5 of [OGC 06-121r3] apply if not superseded by the here defined requirements.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/url-encoded-values
Requirement 107 Special characters of parameter values of the KVP request shall be URL-encoded as defined in [IETF 2396].

Example 28 Use "`%20`" to represent a whitespace, " ".

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/case-sensitivity
Requirement 108 Keys shall be case insensitive, values shall be case sensitive.

Any extension property contained in the conceptual model of SOS operation request and response types can be realized as simple additional key-value pairs that are appended to the parameters of the original operation. Such extension kvp parameters can be defined in extensions to this requirements class.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/namespaces
Requirement 109 KVP requests with a parameter that includes a `valueReference` as defined in ISO 19143/OGC 09-026r1 which uses XML prefixes in its path expression value shall include a `namespaces` parameter.

This parameter shall specify the mappings of the set of XML prefixes that occur in the `valueReference` to their respective XML namespace URIs.

The format shall be `xmlns(prefix,namespace_uri)`. Multiple namespaces shall be bound by specifying a comma separated list of xmlns() values.

Note that the value of the `namespaces` parameter is URL-encoded by default (Requirement 107).

Example 29 URL encoded:
namespaces=xmlns(ns1%2Chttp%3A%2F%2Fwww.opengis.net%2FsamplingSpatial%2F2.0)%2Cxmlns(ns2%2Chttp%3A%2F%2Fwww.opengis.net%2Fom%2F2.0)

URL decode:
namespaces=xmlns(ns1,http://www.opengis.net/samplingSpatial/2.0),xmlns(ns2,http://www.opengis.net/om/2.0)

NOTE: The `namespaces` parameter is not defined for XML encoded requests because XML has its own mechanism for asserting namespaces.

13.2.1 GetCapabilities KVP Binding

This KVP binding is a realization of the conceptual model of the *GetCapabilities* operation as defined in Subclause 8.1. In general, all requirements defined for the conceptual model of the *GetCapabilities* operation apply for its realization as a KVP binding.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/gc-request
Requirement 110 The KVP encoding of the *GetCapabilities* operation request shall be as specified in Table 5 in Subclause 7.2.3 of [OGC 06-121r3]. The fixed value of the service parameter shall be "`SOS`".

Example 30 To request a Capabilities document, a client can issue the following *GetCapabilities* operation request encoded as KVP:

```
http://hostname:port/path?service=SOS&request=GetCapabilities&AcceptVersions
=2.0.0
```

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/gc-response
Requirement 111 The response behaviour of the *GetCapabilities* operation shall be implemented as defined in Subclause 8.1.2 of this document and encoded as defined in Subclause 13.1.

13.2.2 DescribeSensor KVP Binding

This KVP binding is a realization of the conceptual model of the *DescribeSensor* operation as defined in Subclause 8.2. In general, all requirements defined for the conceptual model of the

DescribeSensor operation apply for its realization as a KVP binding. However, due to the limitations of a KVP binding, the parameterization of the operation needs to be restricted and further defined as described below.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/ds-request
Requirement 112 The KVP encoding of a *DescribeSensor* request shall be as defined in Table 49.

Table 49: DescribeSensor request KVP encoding

Name	**Definition and format**	**Optionality**
service	Identifier of the OGC service. Fixed value: "`SOS`".	Mandatory
version	Request protocol version. Fixed value "`2.0.0`".	Mandatory
request	Request type name. Fixed value: "`DescribeSensor`".	Mandatory
procedure	URL-encoded URI pointing to the procedure for which a metadata description shall be retrieved.	Mandatory
procedureDescriptionFormat	URL-encoded URI pointing to the requested procedure description format. Recommended by this standard is SensorML SensorML version 1.0.1 [OGC 07-022r2] is identified by the value "`http%3A%2F%2Fwww.opengis.net%2FsensorML%2F1.0.1`" (the URL encoded namespace)	Mandatory
validTime	Time instance or time interval encoded conformant to [ISO 8601] for which the sensor description shall be retrieved. If omitted the currently valid sensor description is returned. If end time of time interval is in the future, all descriptions from start time to now - including the current description - shall be returned	Optional

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/ds-time-encoding
Requirement 113 The `validTime` request parameter shall be encoded compliant to [ISO 8601].

Example 31 The following KVP structure requests metadata about a procedure[15]:

```
http://www.myserver.org:port/path
?service=SOS
&version=2.0.0
&request=DescribeSensor
&procedure=urn:ogc:object:Sensor:MyOrg:thermometer1
&procedureDescriptionFormat=http://www.opengis.net/sensorML/1.0.1
&validTime=2010-01-01T18:31:42Z
```

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/ds-response
Requirement 114 The response behaviour of the *DescribeSensor* operation shall be as defined in Subclause 8.2 of this document. The operation response shall be encoded as defined in Annex B of OGC 09-001

13.2.3 GetObservation KVP Binding

This KVP binding is a realization of the conceptual model of the *GetObservation* operation as defined in Subclause 8.3. In general, all requirements defined for the conceptual model of the *GetObservation* operation apply for its realization as a KVP binding. However, due to the limitations of a KVP binding, the parameterization of the operation needs to be restricted and further defined as described below.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/go-request
Requirement 115 The KVP encoding of a *GetObservation* request shall be as defined in Table 50.

Table 50: GetObservation request KVP encoding

Name	Definition and format	Optionality
service	Identifier of the OGC service. Fixed value: "`SOS`".	Mandatory
version	Request protocol version. Fixed value "`2.0.0`".	Mandatory
request	Request type name. Fixed value: "`GetObservation`".	Mandatory
offering	Comma-separated unordered list of one or more URL-encoded URIs pointing to the requested observation offerings. [1]	Optional
observedProperty	Comma-separated unordered list of one or more URL-encoded URIs pointing to the observed properties of the	Optional

[15] For the purpose of better readability, this example contains URL-decoded query parameter values.

	requested observations. [1]	
procedure	Comma-separated unordered list of one or more URL-encoded URIs pointing to procedures of the requested observations. [1]	Optional
featureOfInterest	Comma-separated unordered list of one or more URL-encoded URIs pointing to specific features of interest of observations stored by the service. [1]	Optional
namespaces	Defines the mapping of XML prefixes used in valueReferences included in other request parameters to their respective XML namespace URIs. See Requirement 109 for further details.	Conditional; include if spatialFilter and/or temporalFilter is contained in request and uses XML prefixes in path expression of valueReference
spatialFilter	Specifies a bounding box used as a spatial filter which applies to a spatial property (identified by its valueReference) of the requested observations. [1] The bounding box is encoded as defined in Requirement 116.	Optional
temporalFilter	Specifies a temporal filter which applies to a temporal property (identified by its valueReference) of the requested observations. [1] The filter is encoded as defined in Requirement 117.	Optional
responseFormat	Specifies the desired response format for transport of the observations. The supported output formats are listed in the ObservationOffering in the Capabilities document. By default this is O&M 2.0 [OGC 10-025] identified by the value "`http%3A%2F%2Fwww.opengis.net%2Fom%2F2.0`" (URL-encoded namespace).	Optional
1 The context of this filter parameter is specified in 0.		

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/go-bbox-encoding
Requirement 116 For the KVP binding, a bounding box shall be used as the `spatialFilter` as defined in the conceptual model of the *GetObservation* operation (Subclause 8.3). The encoding of the bounding box shall be a list of comma separated values. The first value shall be the `valueReference` of the spatial property of the observations to which this bounding box, as a spatial filter, is applied – following the requirements on ValueReference as defined by [ISO 19143] / [OGC 09-026r1]. The following values shall be as defined in Subclause 10.2.3 of [OGC 06-121r3].

This results in the following encoding:
`valueReference,minCoordinate1,minCoordinate2,...,minCoordinateN,maxCoordinate1,maxCoordinate2,...,maxCoordinateN,crsURI`

The `crsURI` is optional. An example value for `crsURI` is "`urn:ogc:def:crs:EPSG::4326`".

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/go-temporalFilter-encoding
Requirement 117 As defined in the conceptual model of the *GetObservation* operation (Subclause 8.3), the `temporalFilter` parameter shall be used to filter on a temporal property of the requested observations (e.g., the `phenomenonTime` property).The value of the `temporalFilter` parameter shall be encoded compliant to [ISO 8601]. Instances and periods of time shall be supported. Periods of time (start and end) are separated by "/". If the `temporalFilter` consists of a time instant, it shall apply the semantics of the `TEquals` operator. If the `temporalFilter` consists of a time period, it shall apply the semantics of the `During` operator. The encoding of the `temporalFilter` shall be a list of two comma separated values. The first value shall be the `valueReference` of the temporal property of the observations to which the temporal filter is applied – following the requirements on ValueReference as defined by [ISO 19143] / [OGC 09-026r1]. This results in the following encoding: `valueReference,iso8601Time`

Example 32 The following example shows a KVP-encoded *GetObservation* request[16]:

```
   http://myserver.org:port/path
   ?service=SOS
   &version=2.0.0
   &request=GetObservation
   &offering=http://www.my_namespace.org/thermometer1_observations
   &observedProperty=http://sweet.jpl.nasa.gov/2.0/atmoThermo.owl#EffectiveTemp
erature
   &procedure=http://www.my_namespace.org/sensors/thermometer1
   &featureOfInterest=http://wfs.example.org?request=getFeature&amp;featureid=b
uilding1
   &namespaces=xmlns(sams,http://www.opengis.net/samplingSpatial/2.0),xmlns(om,
http://www.opengis.net/om/2.0)
   &spatialFilter=om:featureOfInterest/*/sams:shape,22.32,11.2,32.32,22.2,urn:o
gc:def:crs:EPSG::4326
   &temporalFilter=om:phenomenonTime,2009-01-10T10:00:00Z/2009-01-10T11:00:00Z
```

[16] For the purpose of better readability, this example contains URL-decoded query parameter values.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-core/go-response
Requirement 118 The response behaviour of the *GetObservation* operation shall be as defined in Subclause 8.3.2 of this document. If the response format chosen in the request requires that a `GetObservationResponse` be returned then it shall be encoded as defined in subclause 13.1

13.3 Requirements Class: GetFeatureOfInterest KVP Binding

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/kvp-core
Dependency	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval/**request**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval/**bbox-encoding**

This requirements class defines a KVP binding for the *GetFeatureOfInterest* operation.

The here defined KVP binding of the *GetFeatureOfInterest* operation is a realization of the conceptual model of that operation as defined in Subclause 9.1.1. In general, all requirements defined for the conceptual model of the *GetFeatureOfInterest* operation apply for its realization as a KVP binding. However, due to the limitations of a KVP binding, the parameterization of the operation needs to be restricted and further defined as described below.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval/request
Requirement 119 The KVP encoding of a *GetFeatureOfInterest* request shall be as defined in Table 51.

Table 51: GetFeatureOfInterest request KVP encoding

Name	Definition and format	Optionality
service	Identifier of the OGC service. Fixed value: "`SOS`".	Mandatory
version	Request protocol version. Fixed value "`2.0.0`".	Mandatory
request	Request type name. Fixed value: "`GetFeatureOfInterest`".	Mandatory
featureOfInterest	Comma-separated unordered list of one or more URL-encoded URIs pointing to requested features of interest. [1]	Optional
observedProperty	Comma-separated unordered list of one or more URL-encoded URIs pointing to observed properties which are properties of the requested features of interest. [1]	Optional
procedure	Comma-separated unordered list of one or more URL-encoded URIs pointing to procedures which observe the requested features of interest. [1]	Optional
namespaces	Defines the mapping of XML prefixes used in	Conditional;

	valueReferences included in other request parameters to their respective XML namespace URIs. See Requirement 109 for further details.	include if spatialFilter is contained in request and uses XML prefixes in path expression of valueReference
spatialFilter	Specifies a bounding box used as a spatial filter which applies to a spatial property (identified by its valueReference) of the requested features. [1] The bounding box is encoded as defined in Requirement 120.	Optional
1 The context of this filter parameter is specified in Requirement 46.		

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval/bbox-encoding
Requirement 120 For the KVP binding, a bounding box shall be used as `spatialFilter` as defined in the conceptual model of the *GetFeatureOfInterest* operation (Subclause 9.1.1). The encoding of the bounding box shall be a list of comma separated values. The first value shall be the `valueReference` that identifies the spatial property of the features to which this bounding box, as a spatial filter, is applied – following the requirements on ValueReference as defined by ISO 19143/OGC 09-026r1. The following values shall be as defined in Subclause 10.2.3 of [OGC 06-121r3]. This results in the following encoding: `valueReference,minCoordinate1,minCoordinate2,...,minCoordinateN,maxCoordinate1,maxCoordinate2,...,maxCoordinateN,crsURI` The `crsURI` is optional. An example value for `crsURI` is "`urn:ogc:def:crs:EPSG::4326`".

Example 33 The following example shows a KVP-encoded *GetFeatureOfInterest* request[17]:

```
   http://myserver.org:port/path
   ?service=SOS
   &version=2.0.0
   &request=GetFeatureOfInterest
   &observedProperty=http://sweet.jpl.nasa.gov/2.0/atmoThermo.owl#EffectiveTemperature
   &procedure=http://www.my_namespace.org/sensors/thermometer1
   &namespaces=xmlns(sams,http://www.opengis.net/samplingSpatial/2.0),xmlns(om,http://www.opengis.net/om/2.0)
```

[17] For the purpose of better readability, this example contains URL-decoded query parameter values


```
   &spatialFilter=om:featureOfInterest/*/sams:shape,22.32,11.2,32.32,22.2,urn:o
gc:def:crs:EPSG::4326
```

13.4 Requirements Class: KVP Binding for Result Retrieval

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/kvp-result	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/kvp-core
Dependency	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-result/**grt-request**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-result/**grt-response**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-result/**gr-request**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-result/**gr-response-xml**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-result/**gr-response-raw**

This requirements class defines a KVP binding for the *GetResult* and *GetResultTemplate* operations.

The here defined KVP binding of the *GetResult* and *GetResultTemplate* operations is a realization of the conceptual model of these operations as defined in Subclause 11.2. In general, all requirements defined for the conceptual model of the *GetResult* and *GetResultTemplate* operations apply for its realization as a KVP binding. However, due to the limitations of a KVP binding, the parameterization of the operations needs to be restricted and further defined as described below.

13.4.1 GetResultTemplate KVP Binding

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-result/grt-request
Requirement 121 The KVP encoding of a *GetResultTemplate* request shall be as defined in Table 52.

Table 52: GetResultTemplate request KVP encoding

Name	**Definition and format**	**Optionality**
service	Identifier of the OGC service. Fixed value: "`SOS`".	Mandatory
version	Request protocol version. Fixed value "`2.0.0`".	Mandatory
request	Request type name. Fixed value: "`GetResultTemplate`".	Mandatory
offering	A URL-encoded URI pointing to the requested ObservationOffering. [1]	Mandatory
observedProperty	A URL-encoded URI pointing to the observed property for which to retrieve observation results. [1]	Mandatory
1 The context of this filter parameter is specified in Requirement 91.		

Example 34 The following example shows a KVP-encoded *GetResultTemplate* request:

```
   http://myserver.org:port/path
   ?service=SOS
   &version=2.0.0
   &request=GetResultTemplate
   &offering=http://www.my_namespace.org/thermometer1_observations
   &observedProperty=http://sweet.jpl.nasa.gov/2.0/atmoThermo.owl#EffectiveTemp
erature
```

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-result/grt-response
Requirement 122 The response behaviour of the *GetResultTemplate* operation shall be as defined in Subclause 11.2.1.2 of this document and encoded in XML as defined by the "XML Encoding" requirements class.

13.4.2 GetResult KVP Binding

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-result/gr-request
Requirement 123 The KVP encoding of a *GetResult* request shall be as defined in Table 53.

Table 53: GetResult request KVP encoding

Name	Definition and format	Optionality
service	Identifier of the OGC service. Fixed value: "`SOS`".	Mandatory
version	Request protocol version. Fixed value "`2.0.0`".	Mandatory
request	Request type name. Fixed value: "`GetResult`".	Mandatory
offering	A URL-encoded URI pointing to the requested ObservationOffering. [1]	Mandatory
observedProperty	A URL-encoded URI pointing to the observed property for which to retrieve observation results. [1]	Mandatory
featureOfInterest	Comma-separated unordered list of one or more URL-encoded URIs pointing to features of interest of observations for which result values are requested. [1]	Optional
namespaces	Defines the mapping of XML prefixes used in valueReferences included in other request parameters to their respective XML namespace URIs. See Requirement 109 for further details.	Conditional; include if spatialFilter and/or temporalFilter is contained in request and uses XML prefixes in path expression of

		valueReference
spatialFilter	Specifies a bounding box used as a spatial filter which applies to a spatial property (identified by its valueReference) of the requested observations. [1] The bounding box is encoded as defined in Requirement 116.	Optional
temporalFilter	Specifies a temporal filter which applies to a temporal property (identified by its valueReference) of the requested observations. [1] The filter is encoded as defined in Requirement 117.	Optional
xmlWrapper	Specifies whether the response should be wrapped into an XML element or if the SWE Common encoded results should be provided raw. Default is: "false" *NOTE: Raw binary encoded results can only be sent if xmlWrapper is set to false.*	Optional
1 The context of this filter parameter is specified in **Error! Reference source not found.**.		

Example 35 The following example shows a KVP-encoded *GetResult* request[18]:

```
   http://myserver.org:port/path
   ?service=SOS
   &version=2.0.0
   &request=GetResult
    &offering=http://www.my_namespace.org/thermometer1_observations
   &observedProperty=http://sweet.jpl.nasa.gov/2.0/atmoThermo.owl#EffectiveTemp
erature
   &featureOfInterest=http://wfs.example.org?request=getFeature&amp;featureid=b
uilding1
   &namespaces=xmlns(sams,http://www.opengis.net/samplingSpatial/2.0),xmlns(om,
http://www.opengis.net/om/2.0)
   &spatialFilter=om:featureOfInterest/*/sams:shape,22.32,11.2,32.32,22.2,urn:o
gc:def:crs:EPSG::4326
   &temporalFilter=om:phenomenonTime,2009-01-10T10:00:00Z/2009-01-10T11:00:00Z
```

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-result/gr-response-xml
Requirement 124 When the *xmlWrapper* request argument is set to true, the response behaviour of the *GetResult* operation shall be as defined in Subclause 11.2.2.2 of this document and encoded in XML as defined by the "XML Encoding" requirements class.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/kvp-result/gr-response-raw

[18] For the purpose of better readability, this example contains URL-decoded query parameter values.

Requirement 125 When the *xmlWrapper* request argument is set to false, the response of the *GetResult* operation shall consist of result values only, directly encoded using the encoding defined in the corresponding *GetResultTemplate* response.

13.5 Requirements Class: SOAP Binding

Requirements Class	
http://www.opengis.net/spec/SOS/2.0/req/soap	
Target Type	Web Service
Dependency	http://www.opengis.net/spec/SOS/2.0/req/core
Dependency	http://www.opengis.net/spec/SOS/2.0/req/xml
Dependency	http://www.opengis.net/doc/IS/SWES/2.0/clause/18
Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/**exceptions**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/**ResponseExceedsSizeLimit**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/**InvalidPropertyOfferingCombination**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/**action-uris**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/**action-uris-exceptions**
Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/**message-body**

This requirements class defines the realization of functionality defined by the SOS 2.0 standard for a service using SOAP. This standard does not prescribe usage of either SOAP 1.1 or SOAP 1.2. It also does not prescribe WSDL 1.1 or WSDL 2.0.

This standard does not define any specific policy statements to be included in a WSDL document or in service requests and responses for defining certain established, available or desired behavior. If the need for such policies arises in the future, necessary policy statements can be included in the standard.

13.5.1 Exceptions

Requirement
http://www.opengis.net/spec/SOS/2.0/req/soap/exceptions
Requirement 126 The operations defined in this standard shall use the exception codes defined by Clause 8 in [OGC 06-121r3] and Clause 15 in [OGC 09-001] as well as clauses 8.3.3 and 11.2.1.3 in this standard. The encoding of these exceptions for the operations defined by this standard (in a SOAP binding) shall be as defined in clause 19.2 of [OGC 09-001].

The following subclauses define the SOAP fault encoding of the SOS exceptions that are introduced in clauses 8.3.3 and 11.2.1.3 . The definitions are provided using abstract (SOAP) fault properties as described in OGC 09-001 chapter 19.2.1. These abstract fault properties are mapped to the properties of SOAP 1.1/1.2 faults as defined in sections 19.2.2 and 19.2.3 of OGC 09-001.

13.5.1.1 ResponseExceedsSizeLimit exception

The meaning of this exception (code) is defined in clause 8.3.3 of this standard.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/soap/ResponseExceedsSizeLimit
Requirement 127 The abstract fault properties for the ResponseExceedsSizeLimit exception shall be as follows: • [Code] The *QName* soap11:Service (SOAP 1.1) or soap12:Receiver (SOAP 1.2) • [Subcode] The *QName sos:ResponseExceedsSizeLimit* • [Reason] the string: "The requested result set exceeds the response size limit of this service and thus cannot be delivered." • [Details] An *ows:Exception* element as defined in clause 8.2 of [OGC 06-121r3] with exception code and locator value as defined in Table 23.

13.5.1.2 InvalidPropertyOfferingCombination exception

The meaning of this exception (code) is defined in clause 11.2.1.3 of this standard.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/soap/InvalidPropertyOfferingCombination
Requirement 128 The abstract fault properties for the InvalidPropertyOfferingCombination exception shall be as follows: • [Code] The *QName* soap11:Client (SOAP 1.1) or soap12:Sender (SOAP 1.2) • [Subcode] The *QName sos:InvalidPropertyOfferingCombination* • [Reason] the string: "Observations for the requested combination of observedProperty and offering do not use SWE Common encoded results." • [Details] An *ows:Exception* element as defined in clause 8.2 of [OGC 06-121r3] with exception code and locator value as defined in Table 42

13.5.2 Action URIs

For the SOAP binding, a standard needs to define action URIs for the following features:

- as SOAPAction HTTP header field of a SOAP 1.1 request

- as action parameter in a SOAP 1.2 request (SOAP 1.2 feature: “http://www.w3.org/2003/05/soap/features/action/”)
- as WS-Addressing [action] message addressing property

NOTE If and how a service instance makes use of one or more of these features depends upon the chosen SOAP and WSDL version as well as on the requirements of the service instance.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/soap/action-uris
Requirement 129 Action URIs for the message facets (requests and responses of operations) defined by this standard shall be as defined by Table 54 in this document.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/soap/action-uris-exceptions
Requirement 130 Action URIs for exceptions / fault message types, of which SOS operations make use of, shall be as defined in Table 36 (Subclause 19.3) of [OGC 09-001], while the action URI for SOS specific exceptions / fault message types defined in this standard shall be as defined in Table 55.

Table 54: Action URIs for SOS message facets

Message Facet [a]	Action URI [a]	Applicable in feature (Y=yes, N=no)		
		SOAP 1.1 SOAP Action	SOAP 1.2 action	WS-Addressing [action]
GetCapabilities request	http://www.opengis.net/def/serviceOperation/sos/core/2.0/GetCapabilities	Y	Y	Y
GetCapabilities response	http://www.opengis.net/def/serviceOperation/sos/core/2.0/GetCapabilitiesResponse	N	N	Y
GetObservation request	http://www.opengis.net/def/serviceOperation/sos/core/2.0/GetObservation	Y	Y	Y
GetObservation response	http://www.opengis.net/def/serviceOperation/sos/core/2.0/GetObservationResponse	N	N	Y
GetFeatureOfInterest request	http://www.opengis.net/def/serviceOperation/sos/foiRetrieval/2.0/GetFeatureOfInterest	Y	Y	Y
GetFeatureOfInterest response	http://www.opengis.net/def/serviceOperation/sos/foiRetrieval/2.0/GetFeatureOfInterestResponse	N	N	Y
GetObservationById request	http://www.opengis.net/def/serviceOperation/sos/obsByIdRetrieval/2.0/GetObservationById	Y	Y	Y
GetObservationById response	http://www.opengis.net/def/serviceOperation/sos/obsByIdRetrieval/2.0/GetObservationByIdResponse	N	N	Y
InsertObservation request	http://www.opengis.net/def/serviceOperation/sos/obsInsertion/2.0/InsertObservation	Y	Y	Y
InsertObservation response	http://www.opengis.net/def/serviceOperation/sos/obsInsertion/2.0/InsertObservationResponse	N	N	Y
InsertResultTemplate request	http://www.opengis.net/def/serviceOperation/sos/resultInsertion/2.0/InsertResultTemplate	Y	Y	Y

InsertResultTemplate response	http://www.opengis.net/def/serviceOperation/sos/resultInsertion/2.0/InsertResultTemplateResponse	N	N	Y
InsertResult request	http://www.opengis.net/def/serviceOperation/sos/resultInsertion/2.0/InsertResult	Y	Y	Y
InsertResult response	http://www.opengis.net/def/serviceOperation/sos/resultInsertion/2.0/InsertResultResponse	N	N	Y
GetResultTemplate request	http://www.opengis.net/def/serviceOperation/sos/resultRetrieval/2.0/GetResultTemplate	Y	Y	Y
GetResultTemplate response	http://www.opengis.net/def/serviceOperation/sos/resultRetrieval/2.0/GetResultTemplateResponse	N	N	Y
GetResult request	http://www.opengis.net/def/serviceOperation/sos/resultRetrieval/2.0/GetResult	Y	Y	Y
GetResult response	http://www.opengis.net/def/serviceOperation/sos/resultRetrieval/2.0/GetResultResponse	N	N	Y

[a] Although some values listed in the column appear to contain spaces, they shall not contain spaces.

NOTE The action URIs for the messages defined by the SWE Service Model and only extended in this specification (such as DescribeSensor, InsertSensor and DeleteSensor) are not listed here – they can be found in Table 35 of [OGC 09-001].

Table 55: Action URI for SOS exceptions / fault types

Exception / fault type	WS-Addressing [action] message addressing property value
Exception defined by SOS	http://www.opengis.net/def/serviceOperation/sos/core/2.0/Exception

13.5.3 SOAP Message Body

When implementing the SOAP Binding Extension, the XML encoded requests and responses are transferred in the body of a SOAP message.

Requirement
http://www.opengis.net/spec/SOS/2.0/req/soap/message-body
Requirement 131 The body of the SOAP messages that represent requests and responses of operations defined in this standard shall be encoded as defined in Table 47.

13.5.4 Example

Example 36 The example shows a *GetCapabilities* request enclosed by a SOAP 1.2 envelope.

```
<soap12:Envelope
       xmlns:soap12=http://www.w3.org/2003/05/soap-envelope
       xsi:schemaLocation="http://www.w3.org/2003/05/soap-envelope http://www.w3.org/2003/05/soap-envelope/soap-
envelope.xsd http://www.opengis.net/sos/2.0 http://schemas.opengis.net/sos/2.0/sos.xsd"
       xmlns:sos="http://www.opengis.net/sos/2.0"
       xmlns:wsa="http://www.w3.org/2005/08/addressing"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:ows="http://www.opengis.net/ows/1.1">
 <soap12:Header>
  <wsa:To>http:/my.organization.org/services/sos</wsa:To>
  <wsa:Action>http://www.opengis.net/def/serviceOperation/sos/core/2.0/GetCapabilities</wsa:Action>
  <wsa:ReplyTo>
    <wsa:Address>http://my.client.com/client/myReceiver</wsa:Address>
  </wsa:ReplyTo>
  <wsa:MessageID>http://my.client.com/uid/msg-0010</wsa:MessageID>
 </soap12:Header>
 <soap12:Body>
  <sos:GetCapabilities>
    <ows:Sections>
      <ows:Section>serviceIdentification</ows:Section>
      <ows:Section>serviceProvider</ows:Section>
      <ows:Section>contents</ows:Section>
    </ows:Sections>
  </sos:GetCapabilities>
 </soap12:Body>
</soap12:Envelope>
```

14 Annex A – Abstract test suite (normative)

14.1 SOS Core Tests

This section defines conformance tests for the SOS Core. All tests are defined in one conformance class defined below

14.1.1 Conformance Class: SOS Core

There are dependencies on conformance classes of OGC Web Services Common 1.1 [OGC 06-121r3], OGC SWE Common Service Model [OGC 09-001], and ISO Geographic Information — Observations and Measurements [OGC 10-004r3/ISO 19156].

<table>
<tr><td colspan="3">http://www.opengis.net/spec/SOS/2.0/conf/core</td></tr>
<tr><td>Requirements</td><td colspan="2">http://www.opengis.net/spec/SOS/2.0/req/core</td></tr>
<tr><td>Dependency</td><td colspan="2">http://www.opengis.net/doc/IS/OWS/1.1/clause/A4.2</td></tr>
<tr><td>Dependency</td><td colspan="2">http://www.opengis.net/doc/IS/SWES/2.0/clause/A19.1.1.1</td></tr>
<tr><td>Dependency</td><td colspan="2">http://www.opengis.net/doc/IS/SWES/2.0/clause/A19.1.1.2</td></tr>
<tr><td>Dependency</td><td colspan="2">http://www.opengis.net/doc/IS/SWES/2.0/clause/A19.1.7.1</td></tr>
<tr><td>Dependency</td><td colspan="2">http://www.opengis.net/doc/IS/SWES/2.0/clause/A19.1.7.2</td></tr>
<tr><td>Dependency</td><td colspan="2">urn:iso:ts:iso:19156:clause:A1</td></tr>
<tr><td>Test</td><td colspan="2">http://www.opengis.net/spec/SOS/2.0/conf/core/request-service</td></tr>
<tr><td></td><td>Requirement</td><td>http://www.opengis.net/spec/SOS/2.0/req/core/request-service</td></tr>
<tr><td></td><td>Test purpose</td><td>Verify that the server checks and accepts the <code>service</code> parameter value “SOS”.</td></tr>
<tr><td></td><td>Test method</td><td>Send several valid operation requests to the SOS server and verify that the SOS server answers with correct responses. Send invalid operation request with missing service attribute and service attribute with incorrect value to SOS server and verify that the server responds with appropriate exceptions.</td></tr>
<tr><td></td><td>Test type</td><td>Conformance</td></tr>
<tr><td>Test</td><td colspan="2">http://www.opengis.net/spec/SOS/2.0/conf/core/request-version</td></tr>
</table>

	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/request-version
	Test purpose	For all SOS request types defined in this specification, verify that the `version` parameter is checked by the server and the value "2.0.0" is accepted.
	Test method	Send one valid request per request type to the service and verify that the operation result is as expected. Send invalid operation request with missing version attribute and version attribute with incorrect value to SOS server and verify that the server responds with appropriate exceptions.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/mandatory-operations**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/ds
	Test purpose	Check whether the mandatory operations *GetCapabilities*, *GetObservations* and *DescribeSensor* are supported by the service.
	Test method	Execute a *GetCapabilities*, *DescribeSensor*, and *GetObservation* request. Verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-sections**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-sections
	Test purpose	Check whether the service accepts `section` parameters as defined in clause 8.1.1 of this specification and in clause 7 of OWS Common [OGC 06-121r3].
	Test method	Submit GetCapabilities operation requests containing various values and combinations of values of the `section` parameter. Verify that the server provides the correct response to each

		request.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-version**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-version
	Test purpose	Check whether the service checks the AcceptVersions parameter of *GetCapabilties* requests and accepts the value "2.0.0".
	Test method	Send valid *GetCapabilities* request containing the AcceptVersions parameter with value "2.0.0" to the service and verify that the `Capabilities` document is returned. Send a GetCapabilities request with AcceptVersions parameter set to a value other than "2.0.0" which is not supported by the service (e.g. "99.0.0") to the service and verify that an appropriate exception is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-ows**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-ows
	Test purpose	Check that the service accepts *GetCapabilties* requests as defined in clause 7 of OWS Common [OGC 06-121r3].
	Test method	Send valid *GetCapabilities* request to service and verify that a valid `Capabilities` document is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-request-structure
	Test purpose	Check whether the service accepts *GetCapabilities* requests as defined in Subclause 8.3.1 of this specification.

	Test method	Send valid *GetCapabilities* request to service and verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-response**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-response
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-response-structure
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-contents-structure
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-observationoffering-structure
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-contents
	Test purpose	Check whether the service returns an instance of the `Capabilities` type as defined in the conceptual model of clause 8.1 of this specification.
	Test method	Send valid *GetCapabilities* request to service and verify that response is conformant to `Capabilities` model.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-response-version**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-response-version
	Test purpose	To verify that the default version of the `Capabilities` is "2.0.0".
	Test method	Send valid *GetCapabilities* request to service without AcceptVersions parameter and verify that default version of the `Capabilities` returned is "2.0.0".
	Test type	Conformance

Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-operation-listing**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-operation-listing
	Test purpose	Check whether the service supports the operations listed in the `OperationsMetadata` section of the `Capabilities` document.
	Test method	Send valid operation request for each operation listed in `OperationsMetadata` section of `Capabilities` document and verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-conf-class-listing**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-conf-class-listing
	Test purpose	To verify that the `Capabilities` document advertises conformance classes which are supported by the server in addition to the SOS Core conformance class.
	Test method	Retrieve `Capabilities` document, more specifically the Capabilities document including the `ServiceIdentification` section, and verify that the server passes all conformance tests of the conformance classes listed in the `Profile` list of the `ServiceIdentification` section.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-property-inheritance-mechanism**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-property-inheritance-mechanism
	Test purpose	Check whether the properties of the `Contents` element in the `Capabilities` are inherited for all `ObservationOfferings` and check whether properties are contained as described in Table 18 of this specification.

	Test method	Send a valid *GetCapabilities* request to the service and verify that the properties of the `Contents` element in the `Capabilities` response are inherited for all `ObservationOfferings` as described in Table 18 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-spatialFilter-listing**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-spatialFilter-listing
	Test purpose	Check whether the spatial operators and operands are listed in the `FilterCapabilities` section of the `Capabilities`.
	Test method	Get a `Capabilities` document from the service and check that the spatial operators and operands are listed in the FilterCapabilities section of the Capabilities. Execute a *GetObservation* request for each combination of spatial operators and operands listed in the `FilterCapabilities`. Verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/spatial-filter-minimum**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/spatial-filter-minimum
	Test purpose	Check whether the spatial operator `BBOX` is listed in the `FilterCapabilities` of the `Capabilities` document and whether it is supported for the *GetObservation* operation.
	Test method	Query a `Capabilities` document from the service and check whether the `BBOX` operator is listed in the `FilterCapabilities` section. Send a *GetObservation* request containing the `BBOX` filter to the server and verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance

Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-temporalFilter-listing**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-temporalFilter-listing
	Test purpose	Check whether the temporal operators and operands are listed in the `FilterCapabilities` section of the `Capabilities` and are supported by the server.
	Test method	Query a `Capabilities` document from the service and check whether the temporal operators and operands are listed in the FilterCapabilities section of the Capabilities. Execute a *GetObservation* request for each temporal operator and operand listed in the `FilterCapabilities`. Verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/temporal-filter-minimum**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-temporal-filter-minimum
	Test purpose	Check whether the temporal operators `TEquals` and `During` are listed as temporalFilters as well as `TimeInstant` and `TimePeriod` as temporal operands in the `FilterCapabilities` of the `Capabilities` document and verify that these are supported for the *GetObservation* operation.
	Test method	Query a `Capabilities` document from the service and check that the operators and operands above are listed in the `FilterCapabilities` section. Send *GetObservation* requests for each combination of temporal operators and operands and verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-offerings-observations**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-offerings-observations

	Test purpose	Check whether the observations that an offering provides information about have not been created by another procedure than the one that is stated by the offering.
	Test method	Send *GetCapabilities* request and cache the `Capabilities` response. Send a *GetObservation* request for each offering and verify that the observations returned have not been created by another procedure than the one that is stated by the offering.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-offering-identifier**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-offering-identifier
	Test purpose	Check whether the server assigned a unique identifier to each `ObservationOffering` in the `Capabilities`.
	Test method	Send *GetCapabilities* request containing a `Sections` parameter with value "Contents" to the server and verify that the server has assigned a unique identifier to each `ObservationOffering` in the `Capabilities`.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-response-format-om20**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-response-format-om20
	Test purpose	Check whether each `ObservationOffering` in the `Capabilities` contains at least one `responseFormat` parameter with value "http://www.opengis.net/om/2.0".
	Test method	Send *GetCapabilities* request containing a `Sections` parameter with value "Contents" to the server and verify that each ObservationOffering in the Capabilities contains at least one responseFormat parameter with value "http://www.opengis.net/om/2.0".
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/gc-exception**	

	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/gc-exception
	Test purpose	Check whether the server returns appropriate exception messages in case of an error while executing the operation.
	Test method	Send invalid *GetCapabilities* requests to the server and verify that the server returns appropriate exception messages according to section 8.1.3 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/ds**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/ds
	Test purpose	Check that the service supports the *DescribeSensor* operations as specified in Section 11 of [OGC 09-001] and accepts such requests when their `service` property is set to "SOS" and their `version` property is set to "2.0.0"
	Test method	Execute conformance tests as described in Subclause 20.1.7 of [OGC 09-001]. Verify that the service accepts valid DescribeSensor requests with `service` property set to "SOS" and `version` property set to "2.0.0"
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-request-structure
	Test purpose	Check whether the service accepts *GetObservation* requests as defined in Subclause 8.3.1 of this specification.
	Test method	Send valid *GetObservation* request to service and verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-parameters**	

	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-parameters
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-omitting-parameters
	Test purpose	Check whether the observations returned match the specified parameter values of the *GetObservation* request.
	Test method	Send several *GetObservation* requests containing several valid parameters and check whether the observations returned match the request parameters.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-default-response-format**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-default-response-format
	Test purpose	Check whether the format of observations returned in the response is O&M 2.0 [OGC 10-004r3/ISO 19156].
	Test method	Send a *GetObservation* request without `responseFormat` parameter and check whether the response is a `GetObservationResponse` containing observations encoded as O&M 2.0. To check the observations, use the conformance tests described in section A.1 of [OGC 10-004r3/ISO 19156].
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-response-format**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-response-format
	Test purpose	Check that the format of the *GetObservation* response is valid according to the `responseFormat` parameter defined in the request.
	Test method	Send valid *GetObservation* requests for each `responseFormat` listed in the `Contents` section of the `Capabilities` and verify that the SOS responds with the correct response format.

	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-response-type**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-response-type
	Test purpose	To check that the type of the *GetObservation* response is valid according to the type of the `GetObservationResponse` as defined in subclause 8.3.2 of this specification.
	Test method	Send valid *GetObservation* request to server and verify that the SOS responds with the correct response format.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-empty-response**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-empty-response
	Test purpose	To check that an instance of GetObservationResponse type is empty if none of the observations associated with the SOS fulfil the GetObservation parameters specified by the client.
	Test method	Send valid *GetObservation* request with request parameters that do not match any observations to server and verify that the SOS responds with an empty GetObservation response.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-response-elements-type**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-response-elements-type
	Test purpose	To check that the observationData elements in an GetObservation response are of type OM_Observation as defined in [OGC 10-004r3/ISO 19156] or subtypes of OM_Observation, if the GetObservation response contains observations
	Test method	Send valid *GetObservation* request to server and verify that the GetObservation response contains observationData elements that are of type OM_Observation as defined in [OGC 10-

		004r3/ISO 19156] or subtypes of OM_Observation.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-observation-duplicates**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-observation-duplicates
	Test purpose	Check that there are no duplicate observations in a *GetObservation* response.
	Test method	Send several *GetObservation* requests to the service with offering parameter targeting all offerings with the same procedure but also targeting all observations and verify that there are no observation duplicates in the response.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/core/go-exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-exception
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/core/go-too-many-obs-exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *GetObservation* operation.
	Test method	Send invalid *GetObservation* requests to the server and verify that the server returns appropriate exception messages according to section 8.3.3 of this specification.
	Test type	Conformance

14.2 SOS Enhanced Operations Extension Tests

This section contains the conformance classes for the SOS Enhanced Operations Extension.

14.2.1 Conformance Class: SOS Feature of Interest Retrieval

This conformance class defines conformance tests for the retrieval of features of interest.

http://www.opengis.net/spec/SOS/2.0/conf/foiRetrieval		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/foiRetrieval/request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/request-structure
	Test purpose	Check that the service accepts *GetFeatureOfInterest* requests as defined in Subclause 9.1.1.1 of this specification.
	Test method	Send valid *GetFeatureOfInterest* request to service and verify that the server sends appropriate responses as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/foiRetrieval/parameters**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/parameters
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/omitting-parameters
	Test purpose	Check that the features returned match the specified parameter values of the *GetFeatureOfInterest* request.
	Test method	Send several *GetFeatureOfInterest* requests containing several valid parameters and verify that the features returned match the request parameters.
	Test type	Conformance

Test	**http://www.opengis.net/spec/SOS/2.0/conf/foiRetrieval/filter-context**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/filter-context
	Test purpose	Check that the filter context of each request parameter is correct.
	Test method	Send a *GetFeatureOfInterest* request for each filter parameter to the SOS and check that only those features are returned that match the filter context of the particular filter parameter.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/foiRetrieval/response-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/response-structure
	Test purpose	Check that the structure of the response is conformant to the structure defined in Subclause 9.1.1.2 of this specification.
	Test method	Send a *GetFeatureOfInterest* request to the service and check that the response is returned as defined in Subclause 9.1.1.2 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/foiRetrieval/empty-response**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/empty-response
	Test purpose	Check that an instance of `GetFeatureOfInterestResponse` type is empty if none of the features associated with the SOS fulfill the *GetFeatureOfInterest* parameters specified by the client.
	Test method	Send valid *GetFeatureOfInterest* requests to the server that contains parameters that none of the features associated with the SOS fulfil. Verify that the server returns an empty GetFeatureOfInterestResponse.
	Test type	Conformance

Test	**http://www.opengis.net/spec/SOS/2.0/conf/foiRetrieval/exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/foiRetrieval/exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *GetFeatureOfInterest* operation.
	Test method	Send invalid *GetFeatureOfInterest* requests to the server and verify that the server returns appropriate exception messages according to section 9.1.1.3 of this specification.
	Test type	Conformance

14.2.2 Conformance Class: SOS Observation Retrieval By ID

This conformance class defines conformance tests for the retrieval of observations by ID.

http://www.opengis.net/spec/SOS/2.0/conf/obsByIdRetrieval		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsByIdRetrieval/request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/request-structure
	Test purpose	Check that the service accepts *GetObservationById* requests as defined in Subclause 9.2.1.1 of this specification.
	Test method	Send valid *GetObservationById* request to service and verify that the server sends an appropriate response as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsByIdRetrieval/response-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/res

		ponse-structure
	Test purpose	Check that the structure of the response is conformant to the structure defined in Subclause 9.2.1.2 of this specification.
	Test method	Send a *GetObservationById* request to the service and check that the response is returned as defined in Subclause 9.2.1.2 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsByIdRetrieval/response-behavior**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval-response-behavior
	Test purpose	Check that the server returns appropriate observations containing `gml:identifier` values as passed in the request.
	Test method	Send valid *GetObservationById* request to the server and verify that the server returns observations containing `gml:identifier` values as passed in the request.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsByIdRetrieval/exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *GetObservationById* operation.
	Test method	Send invalid *GetObservationById* requests to the server and verify that the server returns appropriate exception messages according to section 9.2.1.3 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsByIdRetrieval/empty-response**	

	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsByIdRetrieval/empty-response
	Test purpose	Check that the server returns an empty response if no observation was found matching one or more of the identifiers provided in the request.
	Test method	Send *GetObservationById* with identifier value that is not associated with an observation hosted by the service and verify that an empty response is returned as defined above.
	Test type	Conformance

14.3 SOS Transactional Extension Tests

This section contains the conformance classes for the SOS Transactional Extension. It comprises four conformance classes which are defined below.

14.3.1 Conformance Class: SOS Insertion Capabilities

This conformance class defines conformance tests for the InsertionCapabilities section in the Capabilities of the service.

http://www.opengis.net/spec/SOS/2.0/conf/insertionCap		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/insertionCap	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/insertionCap/structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/insertionCap/structure
	Test purpose	Check that the `InsertionCapabilities` is structured as defined in Subclause 10.1.1 of this specification.
	Test method	Query Capabilities of the service and check that `InsertionCapabilities` are valid according to the model defined in Subclause 10.1.1 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/insertionCap/capabilities-**	

	inclusion	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/insertionCap/capabilities-inclusion
	Test purpose	Check that the `InsertionCapabilities` is contained in a `Capabilities` response, if it is requested.
	Test method	Send *GetCapabilities* request with Sections parameter containing value "InsertionCapabilities" and verify that `InsertionCapabilities` are contained in the response.
	Test type	Conformance

14.3.2 Conformance class: SOS Sensor Insertion

This conformance class defines conformance tests for the insertion of sensors in the service.

http://www.opengis.net/spec/SOS/2.0/conf/sensorInsertion		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/insertionCap	
Dependency	http://www.opengis.net/doc/IS/SWES/2.0/clause/A19.1.8	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/sensorInsertion/request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/request-structure
	Test purpose	Check that the service accepts an *InsertSensor* request as defined in Subclause 10.2.1.1 of this specification.
	Test method	Send a valid *InsertSensor* request to the service and verify that an appropriate response as defined in this specification is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/sensorInsertion/response**	

	Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/supported-types
	Test purpose	Check that the response is created as defined in Subclause 10.2.1.2
	Test method	Execute the conformance test defined in Subclause 20.1.8 of [OGC 09-001]
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/sensorInsertion/exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *InsertSensor* operation.
	Test method	Send invalid *InsertSensor* requests to the server and verify that the server returns appropriate exception messages according to section 10.2.1.3 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/sensorInsertion/exception-unsupported-types**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorInsertion/exception-unsupported-types
	Test purpose	Check that an exception is returned with the `ExceptionCode` "InvalidParameterValue" and `locator` value "featureOfInterestType" or "observationType" if one of the `featureOfInterestType` or `observationType` specified by the client in the `SosInsertionMetadata` element provided in the InsertSensor request are not supported by the SOS server (supported values are listed in the `InsertionCapabilities` section of the `Capabilities` document)
	Test method	Send valid *InsertSensor* request containing an element of a type as described above with value that is not listed in the `Capabilities` to the server and verify that the server returns an exception with the `ExceptionCode` "InvalidParameterValue" and `locator` value

		"featureOfInterestType" or "observationType"
	Test type	Conformance

14.3.3 Conformance Class: SOS Sensor Deletion

This conformance class defines conformance tests for the deletion of sensors in the SOS.

http://www.opengis.net/spec/SOS/2.0/conf/sensorDeletion		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Dependency	http://www.opengis.net/doc/IS/SWES/2.0/clause/A19.1.6	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/sensorDeletion/request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/request-structure
	Test purpose	Check that the service accepts *DeleteSensor* requests as defined in Subclause 10.3 of this specification.
	Test method	Execute conformance test defined in Subclause 20.1.6 of [OGC 09-001]
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/sensorDeletion/obsoffering-deletion**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/obsoffering-deletion
	Test purpose	Check that the `ObservationOfferings` of the sensor which has been deleted are no more listed in the `Capabilities` of the service.
	Test method	Delete a sensor and query the `Capabilities` from the service. Check that the `ObservationOfferings` of the sensor are no longer listed in the `Contents` section of the `Capabilities`.

	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/sensorDeletion/observation-deletion**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/observation-deletion
	Test purpose	Check that the observations of the sensor which has been deleted are no more provided by the service.
	Test method	Delete a sensor and send a *GetObservation* request for the sensor which has been deleted to the service. Verify that no observations but an `Exception` with code `InvalidParameterValue` is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/sensorDeletion/exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/sensorDeletion/exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *DeleteSensor* operation.
	Test method	Send invalid *DeleteSensor* requests to the server and verify that the server returns appropriate exception messages according to section 10.3.1.1 of this specification.
	Test type	Conformance

14.3.4 Conformance Class: SOS Observation Insertion

This conformance class defines conformance tests for the insertion of observations in a SOS instance.

http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion	
Requirements	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/insertionCap

Dependency	urn:iso:ts:iso:19156:clause:A1	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion/request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/request-structure
	Test purpose	Check that the service accepts *InsertObservation* requests as defined in Subclause 10.4.1.1 of this specification.
	Test method	Send a valid *InsertObservation* request to the service and verify that an appropriate response according to this specification is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion/supported-types**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/supported-types
	Test purpose	Check that the `observationTypes` listed in the `Capabilities` are supported by the SOS and are also listed in the `ObservationOffering` to which the observations are added.
	Test method	Query the `Capabilities`. Send valid InsertObservation requests for each `observationType`- to the service and verify that an appropriate response according to this specification is returned. Retrieve a new Capabilities document and check that the `observationTypes` are also listed in the `ObservationOfferings` to which the observations have been added.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion/property-constellation**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/request-structure
	Test purpose	To verify that a particular constellation of `procedure`, `observedProperty` and `ObservationOffering` for observations is always encoded in the same `observation`- and `resultType`. Verify that the service checks that the

		`observation`- and `resultType` is valid for a particular constellation.
	Test method	Send an *InsertObservation* request with an incorrect `observation`- and `resultType` to the service. Verify that the observations are not inserted and an `Exception` is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion/multiple-offerings**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/multiple-offerings
	Test purpose	Check that the observations are inserted in all offerings that are specified in an *InsertObservation* request..
	Test method	Send an *InsertObservation* request that specifies multiple offerings to the server and verify that the all observations are inserted in all offerings that are specified in the request..
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion/response-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/response-structure
	Test purpose	Check that the service returns an *InsertObservation* response as defined in Subclause 10.4.1.2 of this specification.
	Test method	Send a valid *InsertObservation* request to the service. Verify that service returns a response as defined in Subclause 10.4.1.2 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion/exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *InsertObservation*

		operation.
	Test method	Send invalid *InsertObservation* requests to the server and verify that the server returns appropriate exception messages according to section 10.4.1.3 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion/exception-supported-types**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/exception-supported-types
	Test purpose	Check that an exception is returned with the `ExceptionCode` "InvalidParameterValue" and `locator` value "observationType" or "resultType", if the `observationType` and `resultType` of the observation which should be inserted is not supported by the SOS or the types are not listed in the `ObservationOfferings` of the sensor in the `Capabilities`.
	Test method	Send valid *InsertObservation* request containing an observation of invalid type to the server and verify that the server returns appropriate exception messages as described above.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/obsInsertion/exception-property-constellation**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/obsInsertion/exception-property-constellation
	Test purpose	Check that an exception is returned with the `ExceptionCode` "InvalidParameterValue" and `locator` value "observationType", if an `observationType` is inserted for the same constellation of `procedure`, `observedProperty` as well as `ObservationOffering` and that `observationType` is different than in previous insertions of observations with that property constellation.
	Test method	Send valid *InsertObservation* request containing an observation of invalid type to the server and verify that the server returns appropriate exception messages as described above.

	Test type	Conformance

14.4 SOS Result Handling Extension

This section describes conformance tests for the Result Handling Extension of the SOS.

14.4.1 Conformance Class: Result Insertion

This conformance class defines conformance tests for result insertion in a SOS.

http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/insertionCap	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-request-structure
	Test purpose	Check that the service accepts *InsertResultTemplate* requests as defined in Subclause 11.1.1.1 of this specification.
	Test method	Send a valid *InsertResultTemplate* request to the service and verify that an appropriate response according to this specification is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-supported-types**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-supported-types
	Test purpose	Check whether the `observationTypes` and `resultEncodings` listed in the `InsertionCapabilities` are supported for the SOS and that the `observationTypes` are also listed in the `ObservationOffering` to which the observation results are added.

	Test method	Query the `Capabilities`. Send valid *InsertResultTemplate* requests for each constellation of `observationType`, `featureOfInterestType`, `resultStructure` and `resultEncoding` to the service and verify that an appropriate response according to this specification is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-property-constellation**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-property-constellation
	Test purpose	For a particular constellation of `procedure`, `observedProperty` and `ObservationOffering`, results shall always be given in the same `resultStructure`. Verify that the service checks that no different `resultStructure` can be inserted for a particular constellation.
	Test method	Send two *InsertResultTemplate* requests with different `resultStructures` for the same constellation of `procedure`, `observedProperty` and `ObservationOffering` to the service. Verify that the service returns an `Exception` as response to the second *InsertResultTemplate* call.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-obs-template-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-obs-template-structure
	Test purpose	Verify that the server checks whether the observation that is provided by the client in the ResultTemplate has `om:phenomenonTime`, `om:resultTime` and `om:result` with null value. For the first two properties, the `nilReason` has to be set to the value 'template'. The `procedure`, `featureOfInterest` and `observedProperty` of the observation template shall not be empty.
	Test method	Send a valid *InsertResultTemplate* request to the server and check whether it responds with an appropriate response. Afterwards, send an invalid *InsertResultTemplate* request to the server which does not fulfil the conditions defined above and

		verify that the server returns an exception message.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-result-structure-phenomenonTime**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-result-structure-phenomenonTime
	Test purpose	The `resultStructure` in the `ResultTemplate` shall have at least a `swe:Time` or `swe:TimeRange` component with definition property set to the value "http://www.opengis.net/def/property/OGC/0/PhenomenonTime". Verify that the server checks the existence of this component and that the value of this component is used by the service to populate the `om:phenomenonTime` property of the observation template for each new result block the client is going to insert via the *InsertResult* operation.
	Test method	Send a valid *InsertResultTemplate* request to the server and check whether it responds with an appropriate response. Then, send *InsertResult* requests for the template and verify that the value of the component with definition property set to the value http://www.opengis.net/def/property/OGC/0/PhenomenonTime is used by the service to populate the `om:phenomenonTime` property of the observation template for each new result block (e.g. by retrieving the full observations just inserted via a GetObservation request with appropriate procedure parameter and temporalFilter). Afterwards, send an invalid *InsertResultTemplate* request to the server which does not fulfil the conditions defined above and verify that the server returns an exception message.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-result-structure-resultTime**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-result-structure-resultTime
	Test purpose	If the `resultStructure` in the `ResultTemplate` has a `swe:Time` component with definition property set to the value "http://www.opengis.net/def/property/OGC/0/ResultTime", verify that the value of this component is used by the service to populate the `om:resultTime` property of the observation template for each new result block the client is going to insert

		via the *InsertResult* operation.
	Test method	Send a valid *InsertResultTemplate* request containing a component definition as defined above to the server and check that it responds with an appropriate response. Then, send *InsertResult* requests for the template and verify that the value of the component with definition property set to the value http://www.opengis.net/def/property/OGC/0/ResultTime is used by the service to populate the `om:resultTime` property of the observation template for each new result block (e.g. by retrieving the full observations just inserted via a GetObservation request with appropriate procedure and temporalFilter).
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-observation-time-provisioning**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-observation-time-provisioning
	Test purpose	Verify that the service checks that a client encodes the `om:phenomenonTime` as a `swe:Time` or `swe:TimeRange` component with definition "http://www.opengis.net/def/property/OGC/0/PhenomenonTime" in the resultStructure that it proposes to the service in the *InsertResultTemplate* operation request.
	Test method	Send a valid *InsertResultTemplate* request containing a component definition as defined above to the server and check that it responds with an appropriate response. Then, send an invalid *InsertResultTemplate* request without the component defined above to the server and verify that an exception message is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-response-structure-irt**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-response-structure
	Test purpose	Check that the service returns an `InsertResultTemplateResponse` as defined in Subclause 11.1.1.2 of this specification.

	Test method	Send a valid *InsertResultTemplate* request to the service. Verify that service returns a response as defined in Subclause 11.1.1.2 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *InsertResultTemplate* operation.
	Test method	Send invalid *InsertResultTemplate* requests to the server and verify that the server returns appropriate exception messages according to section 11.1.1.3of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-exception-supported-types**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-exception-supported-types
	Test purpose	Check that an exception is returned with the `ExceptionCode` "InvalidParameterValue" and `locator` value "observationType", if the `observationType` of the observation template which should be inserted is not supported by the SOS or the type is not listed in the `ObservationOfferings` of the sensor in the `Capabilities`.
	Test method	Send invalid *InsertResultTemplate* request containing an observation of invalid type to the server and verify that the server returns appropriate exception messages as described above.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/irt-exception-property-constellation**	

	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/irt-exception-property-constellation
	Test purpose	Check that an exception is returned with the `ExceptionCode` "InvalidParameterValue" and `locator` value "proposedTemplate" if an `observationType` or result structure is inserted for the same constellation of `procedure`, `observedProperty` as well as `ObservationOffering` and that `observationType` / result structure is different than in previous insertions of result templates with that property constellation.
	Test method	Send invalid *InsertResultTemplate* request containing an observation template of invalid type to the server and verify that the server returns appropriate exception messages as described above.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/ir-request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-request-structure
	Test purpose	Check that the service accepts *InsertResult* requests as defined in Subclause 11.1.2.1 of this specification.
	Test method	Send a valid *InsertResult* request to the service and verify that an appropriate response according to this specification is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/ir-resultValues-content**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-resultValues-content
	Test purpose	The client shall encode the values of the observation result that is to be inserted via the *InsertResult* operation according to the `resultStructure` of the `ResultTemplate` it points to in the `InsertResult` request.
	Test method	Send a valid *InsertResult* request to the service and verify that an appropriate response according to this specification is returned. Afterwards, send an *InsertResult* request to the

		service where the resultValues are not encoded as defined in the referenced result template (which defines both the structure and encoding of the values), and verify that the service responds with an appropriate exception message.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/ir-response**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-response
	Test purpose	Check that the service returns an `InsertResultResponse` as defined in Subclause 11.1.2.2 of this specification.
	Test method	Send a valid *InsertResultTemplate* request to the service. Verify that service returns a response as defined in Subclause 11.1.2.2 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/ir-exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *InsertResult* operation.
	Test method	Send invalid *InsertResult* requests to the server and verify that the server returns appropriate exception messages according to section 11.1.2.3 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultInsertion/ir-exception-unknown-result-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultInsertion/ir-exception-unknown-result-structure
	Test purpose	Check that an `Exception` is returned with `exceptionCode` value “InvalidParameterValue” and `locator` value “template”, if the structure and encoding of the inserted results does not adhere to the `resultStructure` and `resultEncoding` defined in the

		`ResultTemplate` inserted beforehand.
	Test method	Send an invalid *InsertResult* request to the service containing results encoded in another structure and encoding as defined in the template inserted beforehand. Verify that an `Exception` is returned with `exceptionCode` value "InvalidParameterValue" and `locator` value "template".
	Test type	Conformance

14.4.2 Conformance Class: Result Retrieval

This conformance class defines conformance tests for result retrieval from a SOS.

http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval/grt-request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/grt-request-structure
	Test purpose	Check that the service accepts *GetResultTemplate* requests as defined in Subclause 11.2.1.1 of this specification.
	Test method	Send a valid *GetResultTemplate* request to the service and verify that an appropriate response according to this specification is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval/grt-response-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/grt-response-structure
	Test purpose	Check that the service returns a `GetResultResponse` as defined in Subclause 11.2.1.2 of this specification.
	Test method	Send a valid *GetResultTemplate* request to the service. Verify that the service returns a response as defined in Subclause

		11.2.1.2 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval/grt-exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/grt-exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *GetResultTemplate* operation.
	Test method	Send invalid *GetResultTemplate* requests to the server and verify that the server returns appropriate exception messages according to section 11.2.1.3 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval/gr-request-structure**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-request-structure
	Test purpose	Check that the service accepts *GetResult* requests as defined in Subclause 11.2.2.1 of this specification.
	Test method	Send a valid *GetResult* request to the service and verify that an appropriate response according to this specification is returned.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval/gr-parameters**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-parameters
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-omitting-parameters
	Test purpose	Check that the service returns result values from all observations that match the specified parameter values of the *GetResult* request.

	Test method	Send several valid *GetResult* requests to the service and verify that the results returned match the specified parameter values of the *GetResult* request.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval/gr-response**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-response
	Test purpose	Check whether the service response is returned as defined in Subclause 11.2.2.2 of this specification.
	Test method	Send a valid *GetResult* request to the service and verify that the response is valid according to the model defined in Subclause 11.2.2.2 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval/gr-response-empty**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-response-empty
	Test purpose	Check that the service response contains a `resultValues` property with "null" value, if no observation matches the parameters of the *GetResult* request.
	Test method	Send a valid *GetResult* request to the service containing parameters that do not match the observations provided by the SOS (e.g. by choosing a temporal filter that targets observations from the year 40000). Verify that the response contains an empty `resultValues` property.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/resultRetrieval/gr-exception**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/resultRetrieval/gr-exception
	Test purpose	Check that the server returns appropriate exception messages in case of an error while executing the *GetResult* operation.

	Test method	Send invalid *GetResult* requests to the server and verify that the server returns appropriate exception messages according to section 11.2.2.3 of this specification.
	Test type	Conformance

14.5 SOS Spatial Filtering Profile

This section describes conformance test for the spatial filtering profile.

14.5.1 Conformance Class: Spatial Filtering Profile

http://www.opengis.net/spec/SOS/2.0/conf/spatialFilteringProfile		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Dependency	http://www.opengis.net/spec/OMXML/2.0/conf/spatialObservation	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/spatialFilteringProfile/observationRestriction**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile/observationRestriction
	Test purpose	Check that the response of a *GetObservation* request only contains observations which are conformant to the `OM_SpatialObservation` defined in Section A.11 of [OGC 10-025].
	Test method	Send a valid *GetObservation* request to the service and verify that the observations returned are conformant to Spatial observation data defined in Section A.11 of [OGC 10-025].
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/spatialFilteringProfile/observedAreaRestriction**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/spatialFilteringProfile/observedAreaRestriction
	Test purpose	Check that the `observedArea` of `ObservationOfferings` provided in the `Capabilities` document contain a geometry representing the minimum bounding box of the sampling geometries of

		observations that belong to that offering.
	Test method	Query the `Capabilities` from the service. Afterwards, query the observations for each `ObservationOffering` from the service and check that the `observedArea` is the minimum bounding box of the sampling geometries of these observations.
	Test type	Conformance

14.6 SOS Binding Extension Tests

This section defines the conformance tests for the different bindings defined in the Binding Extension in section 14.

14.6.1 Conformance Class: XML Encoding

This conformance class defines test for the XML Encoding requirements class.

http://www.opengis.net/spec/SOS/2.0/conf/xml-encoding		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/xml	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Dependency	http://www.opengis.net/spec/SWE/2.0/conf/xsd-simple-components	
Dependency	http://www.opengis.net/spec/SWE/2.0/conf/xsd-simple-encodings	
Dependency	http://www.opengis.net/spec/OMXML/2.0/conf/observation	
Dependency	http://www.opengis.net/spec/SWES/2.0/conf/XMLEncoding	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/xml/validation**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/xml/GeneralEncodingRules
	Test purpose	Verify that XML implementations of the conceptual types defined in this specification are valid according to their XML Schema implementation.
	Test method	For all XML instance documents received from the service or XML instance child elements that are in the namespace http://www.opengis.net/sos/2.0, verify that they are valid

		according to their XML Schema definition listed in Table 47. NOTE: the sos.xsd can be used for validating any such XML instance against its schema definition.
	Test type	Conformance

14.6.2 Conformance Class: KVP Binding Extension

http://www.opengis.net/spec/SOS/2.0/conf/kvp-core		
Requirements	http://www.opengis.net/spec/SOS/2.0/req/kvp-core	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/url-encoded-values**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/url-encoded-values
	Test purpose	Check that the server accepts URL encoded values.
	Test method	Send a valid *GetCapabilities* KVP request to the service and verify that the server returns an appropriate response as specified in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/case-sensitivity**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/case-sensitivity
	Test purpose	Verify that the server checks case-sensitivity of parameter values.
	Test method	Send two *GetCapabilities* KVP requests to the service, one with a case-sensitive encoding of parameter values and one without (with then incorrect values). Check that the server responds with an `Exception` in the latter case.
	Test type	Conformance

Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/gc-request**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/gc-request
	Test purpose	Check that the server accepts a *GetCapabilities* KVP request as defined in section 13.2.1.
	Test method	Send a valid *GetCapabilities* KVP request to the service and verify that the server returns an appropriate response as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/gc-response**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/gc-response
	Test purpose	Check that the server returns a *GetCapabilities* response as defined in section 8.1.2.
	Test method	Send a valid *GetCapabilities* KVP request to the service and verify that the server returns a valid Capabilities response.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/ds-request**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/ds-request
	Test purpose	Check that the server accepts a *DescribeSensor* KVP request as defined in section 13.2.2.
	Test method	Send a valid *DescribeSensor* KVP request to the service and verify that the server returns an appropriate response as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/ds-time-encoding**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/ds-time-encoding

	Test purpose	Check that the server accepts validTime parameter values compliant to [ISO 8601].
	Test method	Send a valid *DescribeSensor* KVP request containing a validTime parameter value compliant to [ISO 8601] to the service and verify that the server returns an appropriate response as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/ds-response**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/ds-response
	Test purpose	Check that the server returns a *DescribeSensor* response as defined in section 13.2.2.
	Test method	Send a valid *DescribeSensors* KVP request to the service and verify that the server returns a valid response.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/go-request**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/go-request
	Test purpose	Check that the server accepts a *GetObservation* KVP request as defined in section 13.2.3.
	Test method	Send a valid *GetObservation* KVP request to the service and verify that the server returns an appropriate response as defined in this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/go-BBOX-encoding**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/go-BBOX-encoding
	Test purpose	Check that the `BBOX` parameter is encoded according to the definition in section 13.2.3.

	Test method	Send a valid *GetObservation* KVP request containing a `BBOX` parameter to the service and verify that the server returns observations matching the `BBOX` of the request.
	Test type	Conformance
Test	http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/go-temporalFilter-encoding	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/go-temporalFilter-encoding
	Test purpose	Check that the `temporalFilter` parameter is used to filter on the temporal property of requested observations specified in the `valueReference`. Separated by commas, the first value of the temporalFilter defines the valueReference and the second value shall be encoded compliant to [ISO 8601]. Instances and periods of time shall be supported.
	Test method	Send a valid *GetObservation* KVP request containing a temporal filter as defined above to the service and verify that the server returns observations with values for the temporal property (as specified in the `valueReference`) that match the `temporalFilter`.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/go-response**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-core/go-response
	Test purpose	Check that the server returns a *GetObservation* response as defined in section 13.2.3.
	Test method	Send a valid *GetObservation* KVP request to the service and verify that the server returns a valid *GetObservation* response.
	Test type	Conformance

14.6.3 Conformance Class: GetFeatureOfInterest KVP Binding

http://www.opengis.net/spec/SOS/2.0/conf/kvp-gfoi

Requirements	http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval	
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/core	
Test	**http://www.opengis.net/spec/SOS/2.0/conf/kvp-core/foiRetrieval-request**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval/gfoi-request
	Test purpose	Check that the server accepts a *GetFeatureOfInterest* KVP request as defined in section 13.3.
	Test method	Send a valid *GetFeatureOfInterest* KVP request to the service and verify that the server returns an appropriate response as defined in this specification.
	Test type	Conformance
Test	http://www.opengis.net/spec/SOS/2.0/conf/kvp-foiRetrieval/BBOX-encoding	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/kvp-foiRetrieval/BBOX-encoding
	Test purpose	Check whether the `spatialFilter` parameter is encoded according to the definition in section 13.3.
	Test method	Send a valid *GetFeatureOfInterest* KVP request containing a `spatialFilter` parameter to the service and verify that the server returns features of interest of observations whose spatial properties (identified via the valueReference in the `spatialFilter` parameter) match the `spatialFilter` of the request.
	Test type	Conformance

14.6.4 Conformance Class: SOAP Binding

http://www.opengis.net/spec/SOS/2.0/conf/soap	
Requirements	http://www.opengis.net/spec/SOS/2.0/req/soap
Dependency	http://www.opengis.net/spec/SOS/2.0/conf/xml
Dependency	http://www.opengis.net/spec/SWES/2.0/conf/SOAPBinding

Test	**http://www.opengis.net/spec/SOS/2.0/conf/soap/exceptions**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/exceptions
	Test purpose	Check that the service returns exception codes defined by Subclause 8 in [OGC 06-121r3] and Subclause 15 in [OGC 09-001] as well as clauses 8.3.3 and 11.2.1.3 in this standard and that the Exceptions are encoded as defined in Subclause 19.2 of [OGC 09-001].
	Test method	Send invalid SOAP requests to the service and verify that the server returns valid exception codes encoded as defined in Subclause 19.2 of [OGC 09-001].
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/soap/action-uris**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/action-uris
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/action-uris-exceptions
	Test purpose	Check that the service accepts and returns valid action URIs as defined in Subclause 0 of this specification.
	Test method	Send valid SOAP requests to the service and verify that the server accepts and returns valid action URIs as defined in Subclause 0 of this specification.
	Test type	Conformance
Test	**http://www.opengis.net/spec/SOS/2.0/conf/soap/message-body**	
	Requirement	http://www.opengis.net/spec/SOS/2.0/req/soap/message-body
	Test purpose	Check that the service accepts and returns valid XML encoded requests and responses contained in the body of the SOAP messages.
	Test method	Send valid SOAP requests for each supported operation to the service and verify that the service returns valid XML encoded responses. Validate the requests and responses with the schema defined in section 13.1.

	Test type	Conformance

15 Annex B - Identifier Handling (informative)

This Annex describes the handling of identifiers for the various resources associated with an SOS. The general usage of identifiers in SWE services is described in Clause 16 of the SWE Service Model standard [OGC 09-001], which serves as a basis for this specification.

The SWE Service Model defines that each object type (e.g. types for procedure, observed property, or observations) has identity. Unlike in GML, a SWES object does not necessarily have an XML attribute that provides identity within an XML instance document (see clause 24.2.4.1 in [OGC 09-001]), since SWES cannot rely on a single data encoding for those objects. Instead, SWES defines that it is the responsibility of the service to assign and maintain identifiers to all elements that are referenceable (using `xlink:href` from other elements).

Furthermore - as explained in [OGC 09-001], Clause 16.3.1 - each SWES object has an identifier property that is defined as of type `ScopedName` in the UML model, and is encoded as type `xs:anyURI` in the XML Schema. Therefore, a SWES object can be uniquely identified via a single URI.

However, note that this is not the case for GML feature identifiers, which are encoded (in XML) as a `gml:identifier` element with a `codeSpace` (of type `xs:anyURI`) attribute and an element value of type `xs:string`. This is explained in more detail in [OGC 09-001] section 16.3.1. Due to the nature of mapping SOS operation request properties that identify / point to a GML feature from the UML model to XML Schema, clients can only use a single URI value to identify a GML feature in a SOS operation request. The default behavior of a SOS therefore is to ignore the `codeSpace` of the `gml:identifier` and to just perform an equality check of the given URI with the element value of the `gml:identifier`. A SOS should ensure that the `gml:identifier` element values of the GML features it manages (e.g. observations and their features of interest) are unique – at least in the scope of the service.

B.1 Identifying a Procedure

The procedure of an observation in the O&M model is of type `OM_Process`. In the service model of the SOS, the `OM_Process` type is used within the observation offering to list the procedures that are associated with the observations hosted by the SOS.

The actual encoding of an `OM_Process` is unknown and can be anything, from SensorML over any GML Application Schema to some new sensor description format. Hence, we cannot rely on finding an identifier for the procedure within its encoding. I.e., it is impossible for an SOS implementation to do a generic search for the identifier value inside of a given `OM_Process` encoding (unless the SOS follows domain specific restrictions of the possible encodings). Thus, after successfully inserting a procedure/sensor into the SOS using the *InsertSensor* operation, the SOS returns a newly created pointer for this procedure (see Clause 13.2.3 in [OGC 09-001]).

This newly created pointer is the identifier (a URI) used by the SOS to address the procedure. Hence, when inserting new observations into the SOS for this new procedure, its identifier needs to be specified. Thereby, the procedure identifier is specified as the value of the `xlink:href` attribute of the `procedure` element of an observation. In consequence, the value stored in `OM_Observation/procedure/@xlink:href` shall be the same as the identifier of the procedure of

the observation offering to which the observations are inserted. This identifier is given in the element `ObservationOffering/procedure` in the Capabilities of an SOS.

The identifier of the procedure can but does not need to be contained as identifier in the actual procedure description. The identifier value could be contained, for example, in the `gml:identifier` or some other field - this depends on the actual encoding of a given procedure description. Due to the variety of possible encodings for the procedure description, a containment of the identifier in the encoding may be not possible.

B.2 Identifying an Observation Offering

As Requirement 21 of this document states, an SOS shall assign a unique identifier value to each of its observation offerings. A new offering is created, when a sensor/procedure gets inserted with the *InsertSensor* operation. In the XML encoding, the offering identifier (a URI) is contained in the `swes:identifier` property of the `sos:ObservationOffering` contained in the contents section of the Capabilities.

B.3 Identifying an Observed Property

The identifier handling of an (observed) property resource of type `GFI_PropertyType` is similar to the identifier handling of `OM_Process` objects. The encoding of the property is unknown and hence the identifier in the encoding of a `GFI_PropertyType` can be anything. Thus, it is impossible to search the identifier in a given `GFI_PropertyType` instance.

However, the difference is that the SOS does not need to manage the actual resources. The value of each property that is of type `GFI_PropertyType` is always the identifier for / pointer to the actual `GFI_PropertyType` resource.

The identifier (the URI) of an instance of the `GFI_PropertyType` is used as the value of:

- the `observedProperty` element in an `sos:ObservationOffering`
- the `xlink:href` attribute of the `observedProperty` element in an `OM_Observation`

B.4 Identifying a Feature of Interest

For features of interest, there is a big difference to the other identifiable types. Since a feature of interest is always of type `GFI_Feature` (which is in XML encoded a subtype of `AbstractFeature`), its encoding supports a `gml:identifier` property. Hence, an SOS can find the identifier of a feature of interest when it is inserted into the SOS via the *InsertObservation* or *InsertResultTemplate* operations.

Note, however, that features of interest do not necessarily contain a `gml:identifier` value. It is in the responsibility of observation publishers to provide these identifiers. If a feature of interest does not have a `gml:identifier` value and a client request – for example a GetObservation request – contains a list of feature of interest identifiers, then that feature of interest will not match the request (as it cannot match the identifier filter) and thus will not be included in the response.

While inserting an observation into the SOS, the feature of interest of the observation can be encoded inline of the observation instance or given as a reference.

Whenever the feature of interest is given by reference, it is not recommended that the SOS implementation stores the feature instance, but instead it should store the reference as given in the attribute `OM_Observation/featureOfInterest/@xlink:href`. Then, if the SOS needs to search for a specific feature of interest identifier in consequence of an operation request, the service would need to resolve the reference and check the `gml:identifier` value of the referenced feature. This is necessary as the feature is governed outside of the SOS and may thus be altered without the SOS knowing.

B.5 Identifying an Observation

In the XML implementation of the O&M data model, the identifier of an observation is stored in the `gml:identifier` property (ignoring the `codeSpace` attribute). Whenever an observation is requested by identifier (e.g., via `GetObservationById` operation), the SOS implementation has to search in the `gml:identifier` field of all available observations to respond to that request.

16 Annex C - Phenomena and Units of Measure (informative)

C.1 Identifying and referencing Phenomena and Units of Measure

A critical issue for interoperability is defining a standard way to refer to the phenomena that are measured by sensors and the units of measure for those phenomena. This is important for both (i) discovery of SOS service instances in a catalog and (ii) to parameterize a request to a given service instance that offers a choice of observed properties. Because SOS is intended to be used in a 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. Observable phenomena include most properties of all feature types in all application domains (see O&M). The range of different phenomena and units of measure is large, unknown *a-priori*, and in fact both unknowable and incomputable. Phenomena and units of measure are often 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 mechanism for identifying phenomena and units of measure must be flexible enough to handle this.

The solution for identifying phenomena and units of measure is to use external references. These may resolve to resources expressed in a variety of forms, utilizing various technologies including semantic web representations. GML dictionaries provide a relatively lightweight format which is compatible with the GML representation of data and web-based addressing patterns used by SOS. Services and clients use URIs to refer to specific entries in a particular GML dictionary.

The URI might be 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. URN values using OGC as the authority must follow the format specified in OGC document 06-023r1 - Definition identifier URNs in OGC namespace.

A URL may be used when the reference is to a new or non-standard definition, for example in the case that a service provides its own dictionary or uses a third-party dictionary that is not well-known. If the specific definition is a sub-element within a dictionary provided as a single resource, then its URL must include a fragment identifier or XPointer to locate the definition within the dictionary.

C.2 Describing and defining Phenomena and Units of Measure

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. Concepts like phenomena and units of measure occupy a different meta-level in the information modeling hierarchy, and their definitions are usually subject to more rigorous governance arrangements, compared with “instance” level data, such as observations and sensor instances. Hence, they will ideally be managed in a registry environment. The GML Dictionary representation may be thought of as a “static” view of such a collection of resources that would usually be provided by a service, such as a register or catalogue.

The SWE initiative relies on the existing GML support for identifying or defining units of measure. This is based on the usual hierarchy of base, derived and “other” (or “conventional”) units, such as defined by the Systeme International. The mechanism for deriving units is well-defined and can be done automatically using software as long as the base units are commonly understood. Ideally, though, UCUM symbols are used.

A GML conformant schema for describing phenomena derived by combination and/or constraining base phenomena was developed as part of the SWE initiative. This schema allows for the definition of base phenomena in much the same way 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. SensorML 1.0.1 contains a schema for describing such phenomena.

17 Annex D - Relationship to Other OGC Web Service Standards (informative)

This Annex describes the relationships of the SOS to other OGC service specifications which are also used to distribute geospatial data. Right now only the relationship to the Web Feature Service (WFS) is described.

D.1 Relationship to Web Feature Service

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. SOS leverages the standard properties of these two data types (sensors, observations) to provide specialized operation signatures for observation data.

This may be contrasted with the approach taken in the Web Feature Service (WFS). WFS is based on a generic definition of a geographic feature that is flexible enough to encompass any real-world entity, and uses GML application schemas to define the feature type exposed by a specific service instance. Hence, the WFS "get data" request is highly parameterized since it must be fully generic. With this approach, interoperability requires organizations to agree on domain-specific GML application schemas. Clients that access a WFS for rich processing in a particular domain must have a-priori knowledge of the application schemas used in that domain.

The SOS defines a common model for all sensors, sensor systems and their observations. This model is "horizontal" since it applies to all domains that use sensors to collect data. The domain-specific details are encapsulated in the second layer (features-of-interest, observed properties, sensor descriptions) allowing the basic "observation" to be processed by a generic client. In that sense, the SOS also provides feature rich access to observation data and metadata.

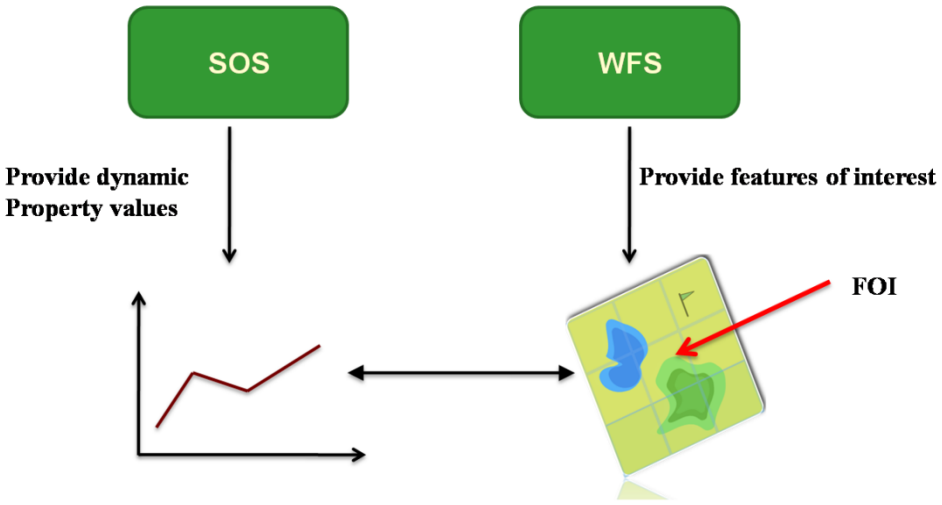


Figure 17-1. Relationship between SOS and WFS where WFS is providing features of interest

Figure 17-1 and Figure 17-2 contain two simple examples how to couple SOS and WFS. In the first figure, the SOS is providing dynamic property values encoded in O&M observations for certain features of interest. These features are provided by an external WFS. For example, the

SOS is providing surface temperature for a certain lake. Then the dynamic surface temperature values are provided by the SOS whereas the feature of interest, the lake, is provided by a WFS instance. As described above, the advantage for using an SOS is that it offers the surface temperature values and its metadata in the well-defined O&M format and that pre-defined filters (such as for time, result quality or producing procedure) can be used instead of the generic GetFeature operation of the WFS for retrieving the observations.

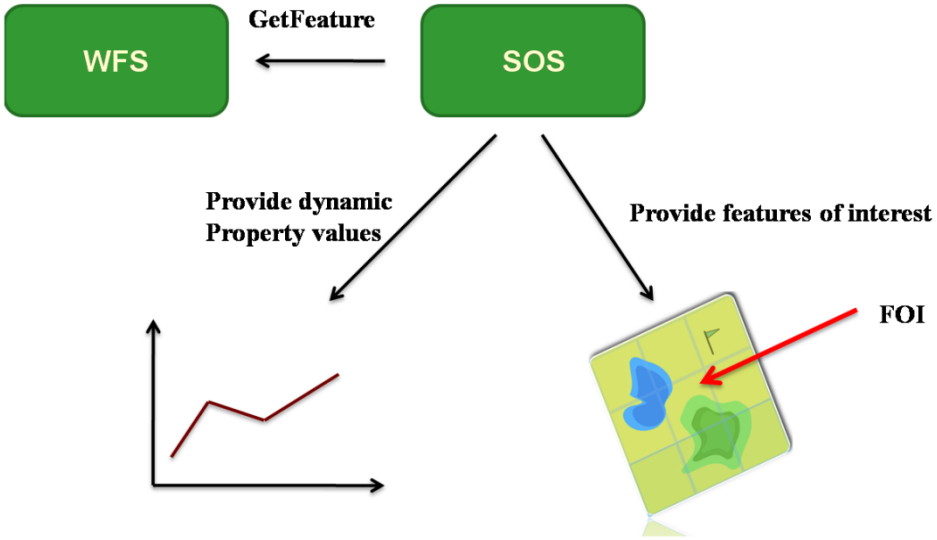


Figure 17-2. Relationship between SOS and WFS where SOS is encapsulating WFS

Figure 17-2 shows another possibility for coupling SOS and WFS instances. In this case, the SOS is using a WFS at the backend for handling the features of interest. The SOS offers both the observations as well as the features of interest. In the example above, the client can retrieve the surface temperature observations as well as the feature of interest, the lake, from the same SOS instance. The operation for retrieving features of interest through the SOS interface is kept simple, as there is only a limited set of query parameters.

The two examples above point out that it is not a question whether you want to use either SOS or WFS, but a question of how to combine or couple the two services. As stated above, the SOS describes a WFS profile with pre-defined observation feature types from the O&M specification and specialized operation signatures for retrieving these observations. The specialization on O&M feature data also enables the SOS to provide a more detailed summary of the observations stored by the service in the service metadata document. This enables registries to harvest the information, improving the discovery of observation data. It also enables clients to generate and perform more informed requests to retrieve observation data. Usually, the SOS is used for providing dynamic property values for certain features of interest in the form of time series. These features of interest can be served either by the SOS instance itself (via GetFeatureOfInterest operation) or by an external WFS.

