Open Geospatial Consortium Inc.

Date: 2010-08-26

Reference number of this OGC® project document: OGC 10-167

OGC name of this OGC® project document: http://www.opengis.net/doc/IS/SOSGDA/1.0

Version: 1.0.0

Category: OpenGIS® Interface Standard

Editor: Johannes Echterhoff

OGC® SOS 2.0 – Get Data Availability Extension

To obtain additional rights of use, visit <http://www.opengeospatial.org/legal/>.

**Warning**

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

Document type: OpenGIS® Interface Standard

Document subtype: Extension

Document stage: Draft proposed version

Document language: English

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.

Contents Page

1 Scope 1

2 Conformance 1

2.1 Overview 1

2.2 Specification identifier 1

2.3 Conformance classes related to the Get Data Availability Extension 1

3 Normative references 2

4 Terms and definitions 3

5 Conventions 3

5.1 Abbreviated terms 3

5.2 UML notation 3

5.3 Platform-neutral and platform-specific standards 3

5.4 Data dictionary tables 4

5.5 Classes imported from other specifications with predefined XML encoding 4

5.6 Namespace Conventions 4

6 GetDataAvailability Operation Overview 6

7 Requirements Class: GetDataAvailability operation – Conceptual Model 9

7.1 Request Structure 9

7.2 Response Structure 13

7.3 Behavior and Use 19

7.4 Exceptions 24

7.5 Service Metadata 26

7.6 Example 26

8 Requirements Class: XML Encoding 29

9 Requirements Class: GetDataAvailability operation – Web Service Model 31

9.1 Request, Response and Exception Structure 31

9.2 Behavior and Use 32

9.3 Exceptions 32

9.4 Example 32

10 Requirements Class: GetDataAvailability operation – SOAP Web Service Model 35

10.1 Behavior and use 35

10.2 Exceptions 37

10.3 Example 37

11 Annex A – Abstract Test Suite (normative) 39

11.1 Conformance Test Class: GetDataAvailability operation – Service implementation 39

11.1.1 Structures 39

11.1.2 Behavior 40

11.1.3 Service Metadata 40

11.2 Conformance Test Class: XML Encoding 41

11.2.1 General Encoding 41

11.3 Conformance Test Class: GetDataAvailability operation – Web Service implementation 42

11.3.1 Structure 42

11.3.2 Behavior 43

11.4 Conformance Test Class: GetDataAvailability operation – SOAP Web Service implementation 44

11.4.1 General 44

**Figures** Page

[Figure 1: Conformance Classes of the SOS GetDataAvailability extension – and their dependencies 2](#_Toc270615013)

[Figure 2: O&M Observation with properties relevant for the GetDataAvailability operation 6](#_Toc270615014)

[Figure 3: How the SOS structures observations via offerings. 7](#_Toc270615015)

[Figure 4: Dependencies of the Get Data Availability operation 9](#_Toc270615016)

[Figure 5: Data types of the GetDataAvailability operation (request part) 10](#_Toc270615017)

[Figure 6: Primary information contained in the GetDataAvailability operation response (informative) 13](#_Toc270615018)

[Figure 7: The GetDataAvailability operation response model using association classes (informative) 14](#_Toc270615019)

[Figure 8: Data types of the GetDataAvailability operation – response part 15](#_Toc270615020)

[Figure 9: The AvailabilityInvestigator interface 19](#_Toc270615021)

**Tables** Page

[Table 1: Conformance classes related to the GetDataAvailability extension 2](#_Toc270615022)

[Table 2: XML Schema implementation of types from the SWE Service Model [OGC 09-001] 4](#_Toc270615023)

[Table 3: Prefixes and Namespaces used in this standard 5](#_Toc270615024)

[Table 4: Properties of the GetDataAvailability request data type 12](#_Toc270615025)

[Table 5: Properties of the GetDataAvailabilityResponse data type 16](#_Toc270615026)

[Table 6: Properties of the FeatureOfInterestInfo type 17](#_Toc270615027)

[Table 7: Properties of the ObservedPropertyInfo type 17](#_Toc270615028)

[Table 8: Properties of the ProcedureInfo type 18](#_Toc270615029)

[Table 9: Properties of the FeaturePropertyTemporalRelationship type 18](#_Toc270615030)

[Table 10: Properties of the ProcedureFeatureTemporalRelationship type 19](#_Toc270615031)

[Table 11: Properties of the ProcedurePropertyTemporalRelationship type 19](#_Toc270615032)

[Table 12: Exceptions for the GetDataAvailability operation 25](#_Toc270615033)

[Table 13: XML Schema implementation of types defined by the GetDataAvailability Extension 29](#_Toc270615034)

[Table 14: Implementation of primary conceptual types required by the GetDataAvailability operation 31](#_Toc270615035)

[Table 15: Action URIs for the GetDataAvailability operation 36](#_Toc270615036)

1. Preface

This document specifies an additional operation for the Sensor Observation Service interface standard to retrieve metadata about the observations stored by the service.

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.

1. Submitting organizations

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

1. International Geospatial Services Institute GmbH (iGSI)
2. …
3. Document contributor contact points

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

|  |  |  |
| --- | --- | --- |
| CONTACT | COMPANY | EMAIL |
| Johannes Echterhoff | iGSI | johannes.echterhoff@igsi.eu |
| … |  |  |

1. Revision history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Release | Editor | Primary clauses modified | Description |
| 10/07/21 | 0.1.0 | Johannes Echterhoff | throughout | initial draft |
| 10/08/26 | 0.9.0 | Johannes Echterhoff | throughout | revised requirements, completed missing sections, added abstract test suite |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

1. Changes to the OGC Abstract Specification

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

1. Future work

Usage of the operation will show if more information needs to be returned per association, e.g. the number of observations that contribute to the information given with the association.

Additional functionality may include enhanced filter functionality to further restrict the scope of featureOfInterest, procedure and observedProperty that are included in the availability study – currently the scope is defined only by their respective object identifier.

Foreword

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

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

Introduction

The Sensor Observation Service (SOS) is specialized for managing sensor observations. This standard defines an additional operation for the SOS: the GetDataAvailability operation. The purpose of this operation is to discover more specific metadata about the observations that are stored by a SOS. Information for which times observations with certain relationships of observed properties, features of interest and procedure exist can be determined using this new operation.

OpenGIS® SOS 2.0 – Get Data Availability Extension

# Scope

This standard defines the conceptual model of a service operation to retrieve metadata about observations stored by a Sensor Observation Service.

# Conformance

## Overview

This standard defines an additional operation for Sensor Observation Services to retrieve metadata about available observations.

Requirements for three standardization target types are considered:

* service implementations, i.e. all kinds of software entities that provide observation data
* web service implementations, i.e. all kinds of web services that provide observation data
* SOAP web service implementations, i.e. all kinds of web services that provide observation data using SOAP based message exchanges

## Specification identifier

All requirements-classes and conformance-classes described in this document are owned by the specification identified as <http://www.opengis.net/spec/SOSGDA/1.0>.

## Conformance classes related to the Get Data Availability Extension

Chapter 11 defines a set of conformance classes and tests that support various applications with a range of different requirements.

Table : Conformance classes related to the GetDataAvailability extension

|  |  |  |
| --- | --- | --- |
| **Conformance class** | **Description** | **Clause** |
| <http://www.opengis.net/spec/SOSGDA/1.0/conf/gda> | Service implements the GetDataAvailability operation. | 11.1 |
| <http://www.opengis.net/spec/SOSGDA/1.0/conf/xml> | The server encodes the data types from the conceptual model in XML as defined by this standard. | 11.2 |
| <http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws> | The service implements the POX binding of the GetDataAvailability operation, i.e. it uses plain XML representations of the operation facets and performs the operation via HTTP POST. | 11.3 |
| <http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws-soap> | The service implements the SOAP binding for the GetDataAvailability operation. | 11.4 |

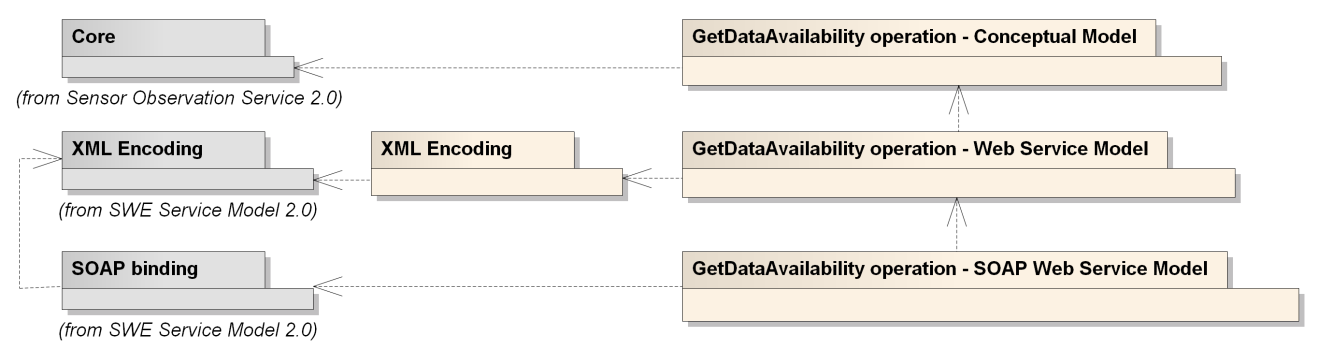


Figure : Conformance Classes of the SOS GetDataAvailability extension – and their dependencies

# Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 19105:2000, *Geographic information — Conformance and Testing*

ISO 19107:2003, *Geographic Information — Spatial schema*

ISO 19108:2002, *Geographic Information — Temporal schema*

ISO 19136:2007, *Geographic information — Geography Markup Language (GML)*

*(*see also: OpenGIS® Encoding Standard *Geography Markup Language*, OGC document 07-036)

ISO DIS 19156:2010, *Geographic information — Observations and Measurements*

OpenGIS® Implementation Standard, *SWE Service Model*, Version 2.0, OGC document 09-001

OpenGIS® Implementation Standard, *Sensor Observation Service*, Version 1.0, OGC document 10-037

OpenGIS® Implementation Specification, *Observations and Measurements - XML Implementation*, Version 2.0.0, OGC document 10-025

OpenGIS® Implementation Specification, *Web Services Common,* OGC document 06-121r3

# Terms and definitions

For the purposes of this standard, the definitions specified in Clause 4 of [OGC 06-121r3] shall apply as well as those from [OGC 10-037].

# Conventions

## Abbreviated terms

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.

O&M Observations and Measurements

SOS Sensor Observation Service

SWE Sensor Web Enablement

## 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.

## Platform-neutral and platform-specific standards

For compliance with Clause 10 of OGC Topic 12 and ISO 19119, this specification 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 clause 7 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 rules defined in OGC 09‑001.

## 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].

## Classes imported from other specifications with predefined XML encoding

This specification 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 2 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 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 clause D.2.1 in OGC 07-036.

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

|  |  |  |  |
| --- | --- | --- | --- |
| **UML class** | **object element** | **type** | **property type** |
| ExtensibleRequest | swes:ExtensibleRequest | swes:ExtensibleRequestType | swes:ExtensibleRequestPropertyType |
| ExtensibleResponse | swes:ExtensibleResponse | swes:ExtensibleResponseType | swes:ExtensibleResponsePropertyType |

## Namespace Conventions

This standard uses a number of namespace prefixes throughout; they are listed in Table 3. Note that the choice of any namespace prefix is arbitrary and not semantically significant.

Table : Prefixes and Namespaces used in this standard

|  |  |
| --- | --- |
| **Prefix** | **Namespace** |
| gda | http://www.opengis.net/sosgda/1.0 |
| gml | http://www.opengis.net/gml/3.2 |
| soap11 | http://schemas.xmlsoap.org/soap/ |
| soap12 | http://www.w3.org/2003/05/soap-envelope |
| swes | http://www.opengis.net/swes/2.0 |
| wsa | http://www.w3.org/2005/08/addressing |
| xs | http://www.w3.org/2001/XMLSchema |

# GetDataAvailability Operation Overview

The GetDataAvailability operation enables clients to:

* discover for which times observations with certain property relationships are available at the service
* discover which procedures observed which properties at which times
* discover which procedures observed which features of interest at which times
* discover which properties of which features of interest were observed at which times
* narrow down the temporal scope for which these relationships are to be discovered to a certain period of time
* restrict the scope of observed properties, features of interest and procedures that are to be included in the result
* restrict the set of observations which are considered for the discovery to those from certain offerings
* generalize the information about temporal relationships to decrease the response size

In order to understand what this operation really is about, let us discuss a bit more how the SOS works. Basically, the SOS stores observations that are encoded in O&M and provides access to them. An observation has a few mandatory properties, the most important of which (for the GetDataAvailability operation) are the *featureOfInterest* (regarding which the observation was made), the *procedure* (the process that generated the observation), the *observedProperty* (basically what the procedure observed) and the *phenomenonTime* (the time at which the result value of the observation applies to the featureOfInterest) – see following figure.

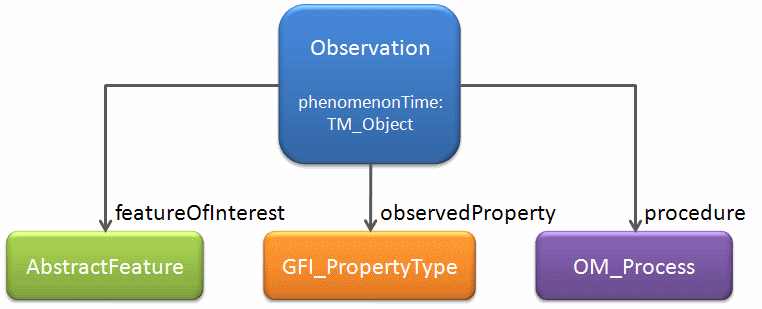


Figure : O&M Observation with properties relevant for the GetDataAvailability operation

The GetDataAvailability operation enables clients to explore these relationships. More specifically, the operation enables clients to estimate for which times observations with certain property relationships (procedure/foi, procedure/observedProperty, foi/observedProperty) exist.

Clients can more easily find out for example when a given procedure observed certain properties. A more intuitive example would be: “Tell me if and when sensor LV-426 observed air temperature during the last week”. With the GetDataAvailability operation this information can be retrieved without actually getting the whole observations. The operation can therefore be used as a precursor to subsequent GetObservation invocations. The operation also helps to only retrieve metadata on certain relationships, for example when only the information about which feature(s)OfInterest were observed by a certain procedure is needed.

Another concept that needs to be understood for applying the GetDataAvailability operation is that a SOS hosts a set of observations. This set, however, can be separated into a list of observation groups - so called ObservationOfferings (see OGC 10-037). Each offering represents a group of observations hosted by the SOS that were generated by one specific procedure – see Figure 3

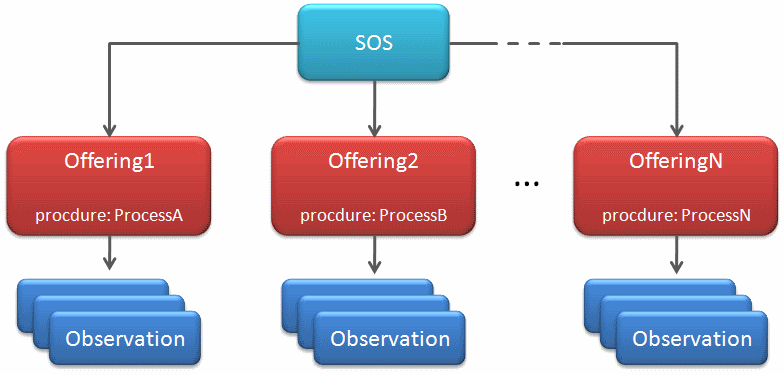


Figure : How the SOS structures observations via offerings.

Note that there is not a required 1:1 relationship between offerings and procedures – it is n:1. In other words: while each offering has exactly one procedure, the same procedure may be associated with multiple offerings. Furthermore, the relationship of observations to offerings is 1:n – meaning that two offerings having the same procedure may share observations. Usually offerings having the same procedure have different sets of observable properties or the implied spatial extent is different (e.g. one offering might host the observations gathered by a network of air quality stations for city A while another offering might host the observations gathered by that station network for city B).

The GetObservation operation provides access to these observations. There, it is possible to retrieve only observations from a certain list of offerings. The same is true for the GetDataAvailability operation – just that this operation does not return full observations. Instead, it returns information about the relationships of certain observation properties (those mentioned before, see Figure 2). The information provided is if observations are available at the Sensor Observation Service that have a given relationship. Furthermore, the GetDataAvailability operation tells clients which phenomenonTimes these observations have. This serves as an indicator for which time periods a GetObservation request asking for these observations would yield results.

# Requirements Class: GetDataAvailability operation – Conceptual Model

|  |  |
| --- | --- |
| Requirements Class | |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda | |
| Target Type | Service Implementation |
| **Dependency** | http://www.opengis.net/spec/SOS/2.0/req/core |

This standard defines the GetDataAvailability operation, an extension to the core set of operations defined by the Sensor Observation Service. The operation reuses data types defined by external conceptual models. Those data types are normatively referenced herein, instead of being repeated in this standard. In addition, the operation incorporates behavior defined by some of these external models.

Figure 4 shows a UML diagram summarizing the dependencies of the Get Data Availability operation to other models.

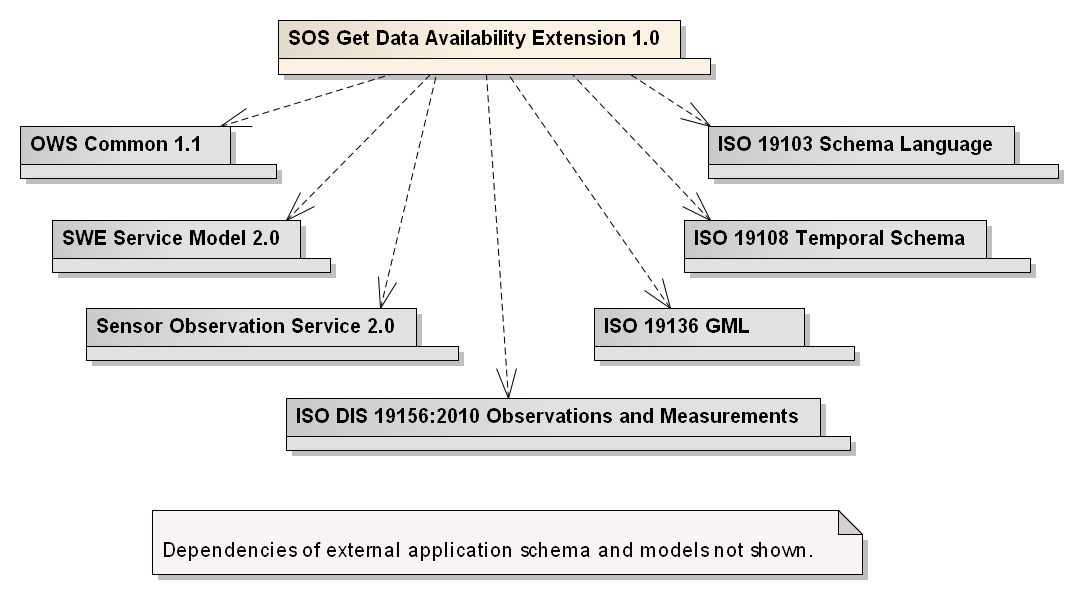


Figure : Dependencies of the Get Data Availability operation

The following clauses define both the structure of the GetDataAvailability operation’s request, response and exceptions as well as the operation behavior.

## Request Structure

The conceptual model of the GetDataAvailability operation request is shown in the following UML diagram.

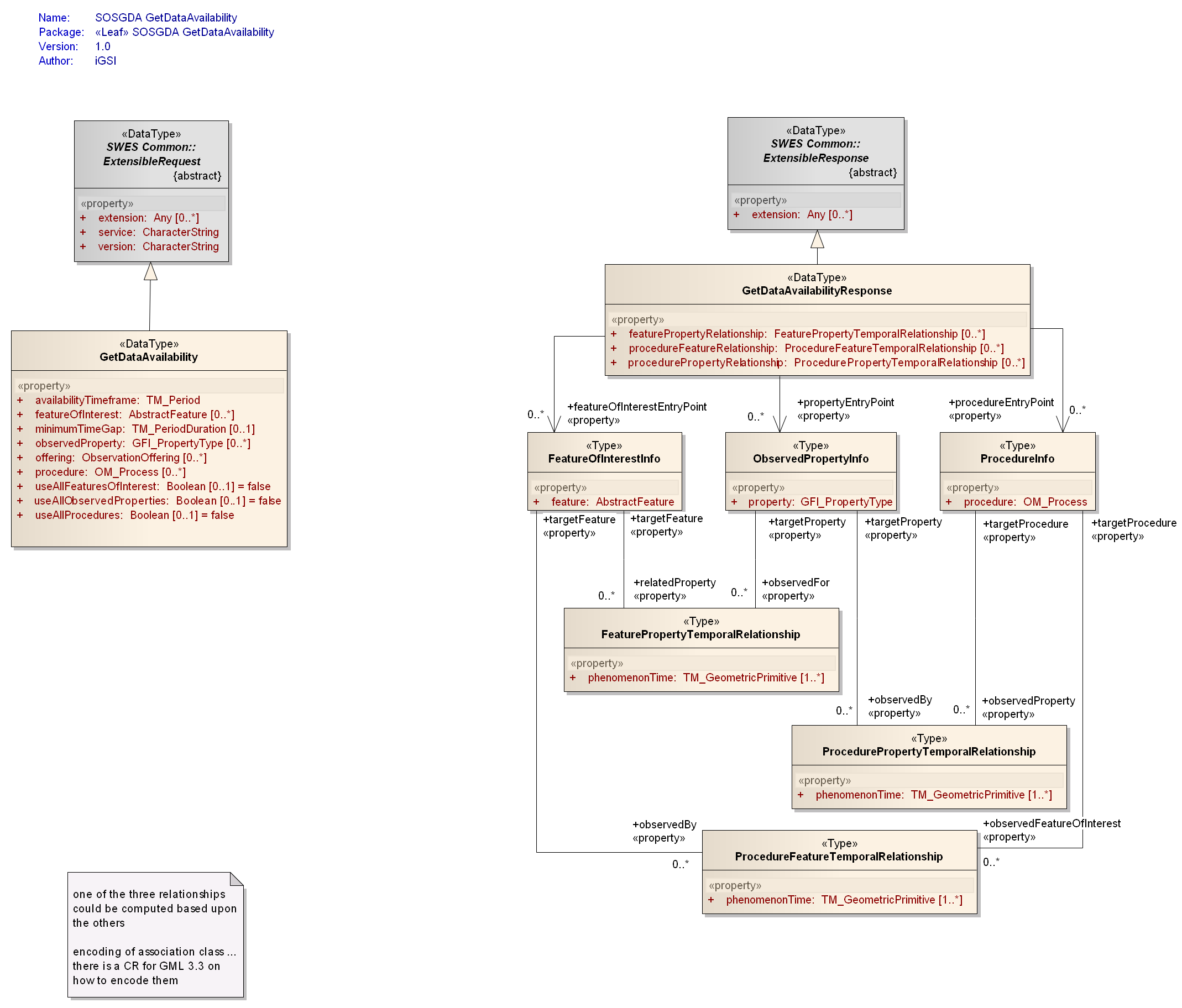


Figure : Data types of the GetDataAvailability operation (request part)

Sending an instance of the GetDataAvailability data type to the service performs an SOS GetDataAvailability operation request.

The *GetDataAvailability* data type is derived from the *ExtensibleRequest* data type specified in [OGC 09-001] and therefore inherits all the properties contained in that data type.

Table 4 specifies the properties of the *GetDataAvailability* data type in more detail. Rows marked with grey background define the properties inherited from the *ExtensibleRequest* data type.

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda/request-structure |
| REQ-001 The structure of the GetDataAvailability operation request shall be as defined in the UML diagram shown in . More specifically, the structure of the properties in the GetDataAvailability data type shall match the data types defined in the UML diagram of and . |

Table : Properties of the GetDataAvailability request data type

| **Name** | **Definition** | **Data type and values** | **Multiplicity** |
| --- | --- | --- | --- |
| service | service type identifier | CharacterString, not empty  value is OWS type abbreviation of the implementing service (“SOS”) | One (mandatory) |
| version | specification version for operation | CharacterString, not empty  value is specified by service type specification implemented by service (“2.0”) | One (mandatory) |
| extension | container for request parameters defined by extension | Any type  value is defined by the extension specification | Zero or more (optional) |
| availabilityTimeframe | Time interval that defines the temporal bounds of the data availability retrieval process. | TM\_Period (see ISO 19108 and OGC 07-036 Table D.2) | One (mandatory) |
| featureOfInterest | Pointer to a feature to include in the availability study. | AbstractFeature id  (see clause D.3.4 in ISO 19136 and clause 9.3 in [OGC 07-036]) | zero to many (optional) |
| minimumTimeGap | The minimum duration between the phenomenonTime(s) of two consecutive observations required for those times not to be aggregated into a single time period. | TM\_PeriodDuration  (see ISO 19108 and OGC 07-036 Table D.2) | zero or one (optional) |
| observedProperty | Pointer to a property to include in the availability study. | GFI\_PropertyType id  (see ISO DIS 19156) | zero to many (optional) |
| offering | Pointer to a service offering of which to include the associated observations in the availability study. | ObservationOffering id  (see OGC 10-037) | zero to many (optional) |
| procedure | Pointer to a process to include in the availability study. | OM\_Process id  (see ISO DIS 19156) | zero or more (optional) |
| useAllFeaturesOfInterest | True in order to include all featuresOfInterest known to the service in the availability study, else false (default). | Boolean  (see ISO 19103) | zero or one (optional) (default is *false*) |
| useAllObservedProperties | True in order to include all observedProperties known to the service in the availability study, else false (default). | Boolean  (see ISO 19103) | zero or one (optional) (default is *false*) |
| useAllProcedures | True in order to include all procedures known to the service in the availability study, else false (default). | Boolean  (see ISO 19103) | zero or one (optional) (default is *false*) |
| id) Note: the primary use of this property is to provide a pointer/identifier – see OGC 09-001 clause 16.3.1 for further details. | | | |

## Response Structure

The conceptual model of the GetDataAvailability operation response is shown in the following UML diagrams.

Figure 6 provides an overview of the primary information contained in the GetDataAvailability operation response. As we can see, the response provides information about the relationships of certain features of interest, observed properties and procedures. As described in section 6, these are properties contained in an observation. The diagram also shows that not only a relationship between certain procedures exists; also the phenomenon times of observations having the given relationship is provided.

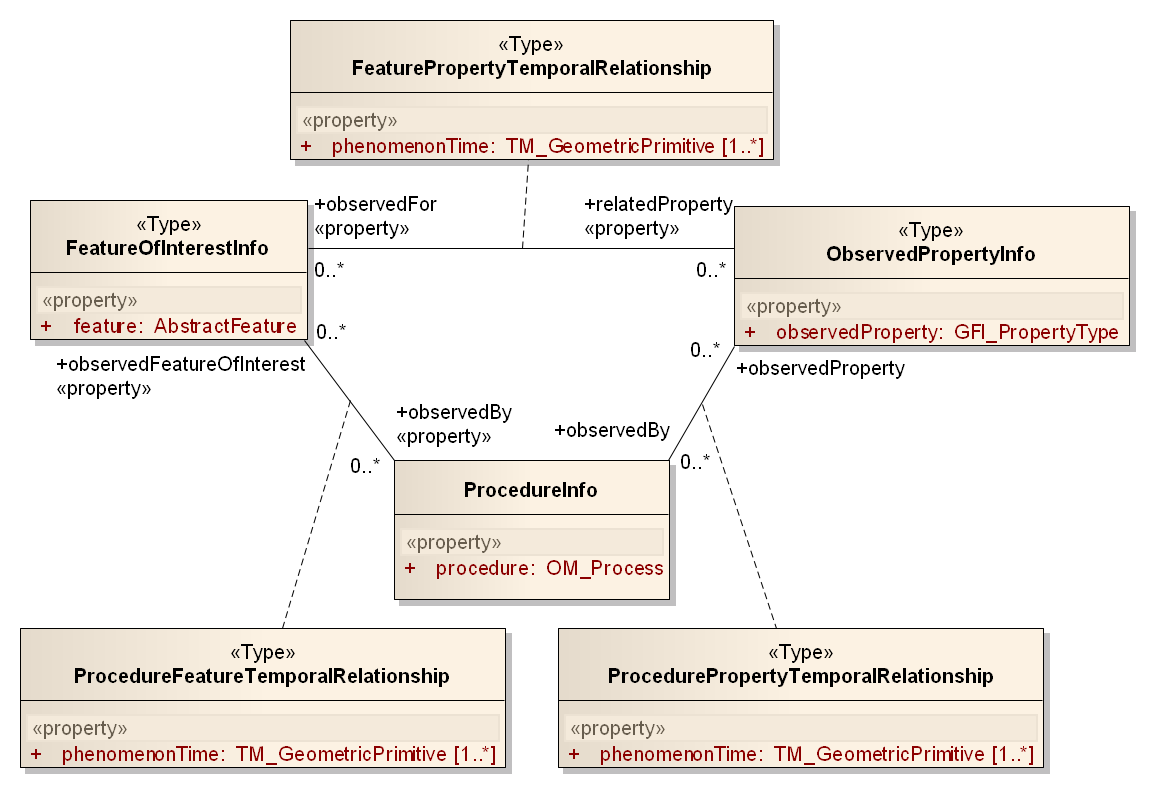
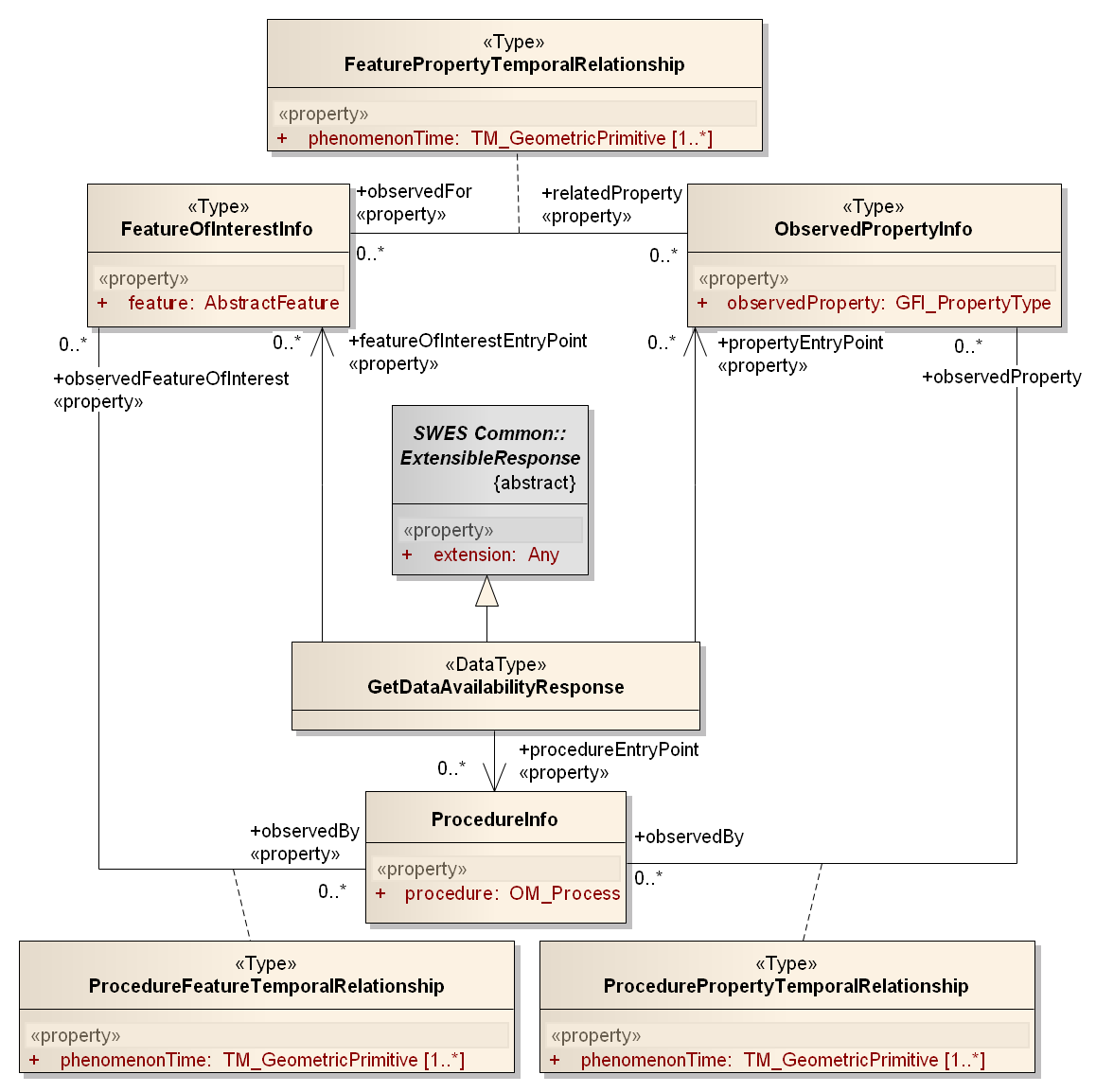


Figure : Primary information contained in the GetDataAvailability operation response (informative)

The cardinality of the association roles are set to zero to many – instead of one to many – because the availability study may be limited in scope. This can be done by setting the availabilityTimeframe in the request (see section ) to a certain time frame but also by providing a list of feature, process and / or property identifiers.

Figure 6 only provides an overview of the primary information contained in the GetDataAvailability response (and as such is informative). The actual model of the response is shown in the following figure.



addition to previous figure

Figure : The GetDataAvailability operation response model using association classes (informative)

Note: this figure is informative – the final model of the GetDataAvailability operation response is shown in .

Here we see that the GetDataAvailabilityResponse is associated to the three info data types. Because in many implementation technologies association classes cannot be represented, the model shown in Figure 7 needs to be modified: association classes need to be converted into explicit data types. Furthermore, in order to have a convenient way to store and access the temporal relationship types, they better be listed in the GetDataAvailabilityResponse type directly. The resulting model is shown in Figure 8.

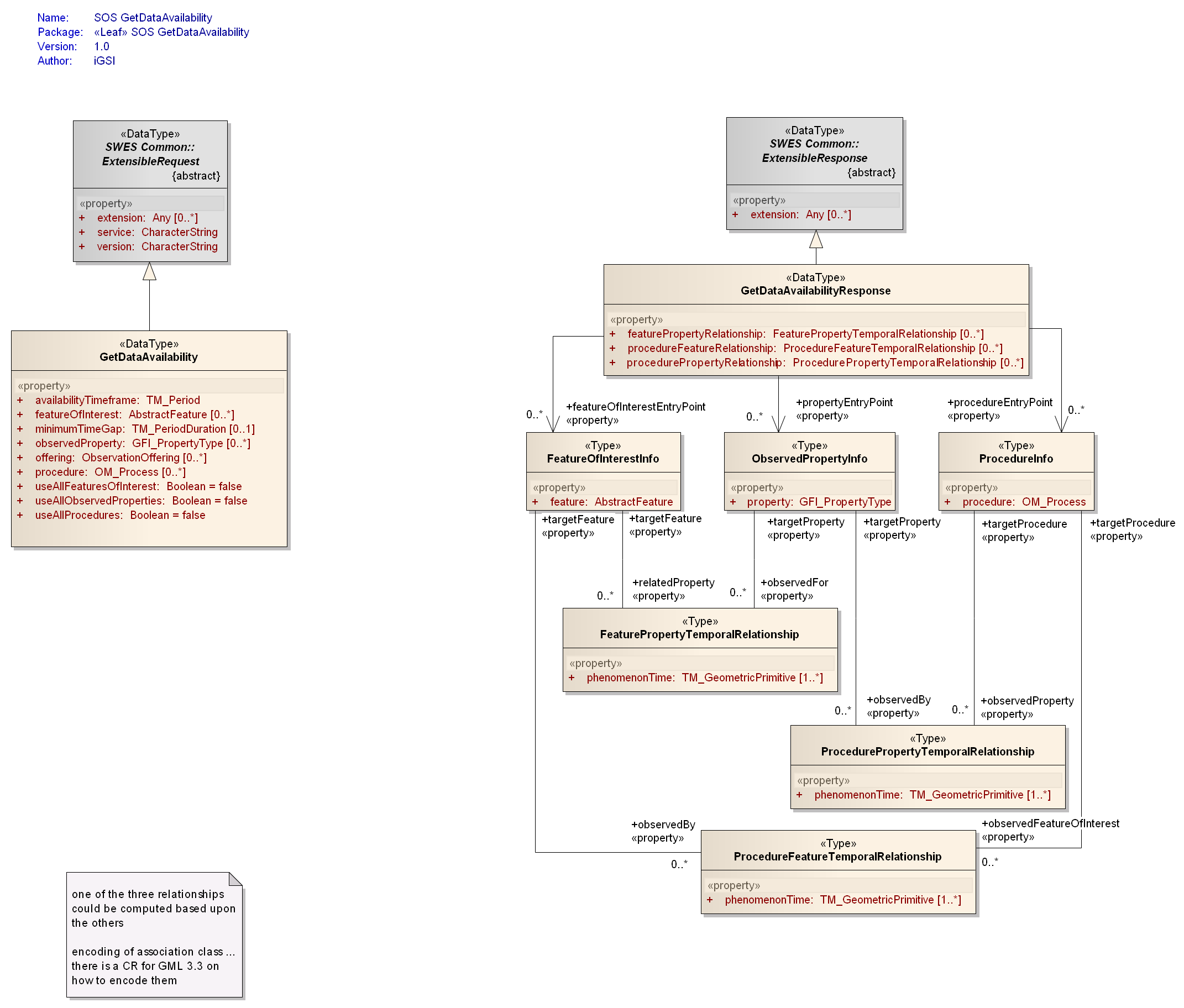


Figure : Data types of the GetDataAvailability operation – response part

Figure 8 shows the final structure of the GetDataAvailability operation response.

The *GetDataAvailabilityResponse* data type represents the response to a GetDataAvailability operation request. It is derived from the *ExtensibleResponse* data type specified in OGC 09-001 and therefore inherits all the properties contained in that data type.

Table 5 specifies the properties of the *GetDataAvailabilityResponse* data type in more detail. Rows marked with grey background define the properties inherited from the *ExtensibleResponse* data type.

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda/response-structure |
| REQ-002 The structure of the GetDataAvailability operation response shall be as defined in the UML diagram shown in . More specifically, the structure of the properties in the *GetDataAvailabilityResponse* data type shall match the data types defined in the UML diagram of and to . |

Table : Properties of the GetDataAvailabilityResponse data type

| **Name** | **Definition** | **Data type and values** | **Multiplicity** |
| --- | --- | --- | --- |
| extension | container for response parameters defined by extension | Any type  value is defined by the extension specification | Zero or more (optional) |
| featureOfInterestEntryPoint | Information on a specific featureOfInterest and its relationships to observedProperties and procedures. | FeatureOfInterestInfo  (see Table 6) | Zero or more (optional) |
| propertyEntryPoint | Information on a specific observedProperty and its relationships to featuresOfInterest and procedures. | ObservedPropertyInfo  (see Table 7) | Zero or more (optional) |
| procedureEntryPoint | Information on a specific procedure and its relationships to observedProperties and featuresOfInterest. | ProcedureInfo  (see Table 8) | Zero or more (optional) |
| featurePropertyRelationship | Additional information about the relationship between a given featureOfInterest and observedProperty. | FeaturePropertyTemporalRelationship  (see Table 9) | Zero or more (optional) |
| procedureFeatureRelationship | Additional information about the relationship between a given procedure and featureOfInterest. | ProcedureFeatureTemporalRelationship  (see Table 10) | Zero or more (optional) |
| procedurePropertyRelationship | Additional information about the relationship between a given procedure and observedProperty. | ProcedurePropertyTemporalRelationship  (see ) | Zero or more (optional) |

Table 6 specifies the properties of the *FeatureOfInterestInfo* type in more detail. This type provides information on a specific featureOfInterest and its relationships to observedProperties and procedures.

Table : Properties of the FeatureOfInterestInfo type

| **Name** | **Definition** | **Data type and values** | **Multiplicity** |
| --- | --- | --- | --- |
| feature | Pointer to the featureOfInterest for which availability information is provided. | AbstractFeature id  (see clause D.3.4 in ISO 19136 and clause 9.3 in [OGC 07-036]) | one (mandatory) |
| relatedProperty | Information about the phenomenon times of observations that have the given feature as featureOfInterest and the related property as observedProperty. | FeaturePropertyTemporalRelationship  (see Table 9) | Zero or more (optional) |
| observedBy | Information about the phenomenon times of observations that have the given feature as featureOfInterest and were observed by the related procedure. | ProcedureFeatureTemporalRelationship  (see Table 10) | Zero or more (optional) |
| id) Note: the primary use of this property is to provide a pointer/identifier – see OGC 09-001 clause 16.3.1 for further details. | | | |

Table 7 specifies the properties of the *ObservedPropertyInfo* type in more detail. This type provides information on a specific observedProperty and its relationships to featuresOfInterest and procedures.

Table : Properties of the ObservedPropertyInfo type

| **Name** | **Definition** | **Data type and values** | **Multiplicity** |
| --- | --- | --- | --- |
| property | Pointer to the (observed) property for which availability information is provided. | GFI\_PropertyType id  (see ISO DIS 19156) | one (mandatory) |
| observedFor | Information about the phenomenon times of observations that have the given property as observedProperty and the related feature as featureOfInterest. | FeaturePropertyTemporalRelationship  (see Table 9) | Zero or more (optional) |
| observedBy | Information about the phenomenon times of observations that have the given property as observedProperty and were observed by the related procedure. | ProcedurePropertyTemporalRelationship  (see ) | Zero or more (optional) |
| id) Note: the primary use of this property is to provide a pointer/identifier – see OGC 09-001 clause 16.3.1 for further details. | | | |

Table 8 specifies the properties of the *ProcedureInfo* type in more detail. This type provides information on a specific procedure and its relationships to observedProperties and featuresOfInterest.

Table : Properties of the ProcedureInfo type

| **Name** | **Definition** | **Data type and values** | **Multiplicity** |
| --- | --- | --- | --- |
| procedure | Pointer to the procedure for which availability information is provided. | OM\_Process id  (see ISO DIS 19156) | one (mandatory) |
| observedProperty | Information about the phenomenon times of observations that were generated by the given procedure and have the related property as observedProperty. | ProcedurePropertyTemporalRelationship  (see ) | Zero or more (optional) |
| observedFeatureOfInterest | Information about the phenomenon times of observations that were generated by the given procedure and have the related feature as featureOfInterest. | ProcedureFeatureTemporalRelationship  (see Table 10) | Zero or more (optional) |
| id) Note: the primary use of this property is to provide a pointer/identifier – see OGC 09-001 clause 16.3.1 for further details. | | | |

Table 9 specifies the properties of the *FeaturePropertyTemporalRelationship*type in more detail. This type provides additional information about the relationship between a given featureOfInterest and observedProperty.

Table : Properties of the FeaturePropertyTemporalRelationship type

| **Name** | **Definition** | **Data type and values** | **Multiplicity** |
| --- | --- | --- | --- |
| phenomenonTime | Time (exact or generalized) for which observations are available that have the related feature as featureOfInterest and the related property as observedProperty. | TM\_GeometricPrimitive  (see ISO 19108 and OGC 07-036 Table D.2) | one to many (mandatory) |
| targetFeature | The related feature. | FeatureOfInterestInfo  (see Table 6) | one (mandatory) |
| targetProperty | The related property. | ObservedPropertyInfo  (see Table 7) | one (mandatory) |

Table 10 specifies the properties of the *ProcedureFeatureTemporalRelationship* type in more detail. This type provides additional information about the relationship between a given procedure and featureOfInterest.

Table : Properties of the ProcedureFeatureTemporalRelationship type

| **Name** | **Definition** | **Data type and values** | **Multiplicity** |
| --- | --- | --- | --- |
| phenomenonTime | Time (exact or generalized) for which observations are available that have the related feature as featureOfInterest and were generated by the related procedure. | TM\_GeometricPrimitive  (see ISO 19108 and OGC 07-036 Table D.2) | one to many (mandatory) |
| targetFeature | The related feature. | FeatureOfInterestInfo  (see Table 6) | one (mandatory) |
| targetProcedure | The related procedure. | ProcedureInfo  (see Table 8) | one (mandatory) |

Table 11 specifies the properties of the *ProcedurePropertyTemporalRelationship* type in more detail. This type provides additional information about the relationship between a given procedure and observedProperty.

Note: this does NOT provide information on which procedure is generally capable of observing certain properties. That information is available in the SOS Capabilities document’s contents section.

Table : Properties of the ProcedurePropertyTemporalRelationship type

| **Name** | **Definition** | **Data type and values** | **Multiplicity** |
| --- | --- | --- | --- |
| phenomenonTime | Time (exact or generalized) for which observations are available that have the related property as observedProperty and were generated by the related procedure. | TM\_GeometricPrimitive  (see ISO 19108 and OGC 07-036 Table D.2) | one to many (mandatory) |
| targetProperty | The related property. | ObservedPropertyInfo  (see Table 7) | one (mandatory) |
| targetProcedure | The related procedure. | ProcedureInfo  (see Table 8) | one (mandatory) |

## Behavior and Use

The GetDataAvailability operation is part of the *AvailabilityInvestigator* interface, the conceptual model of which is shown in the following UML diagram.

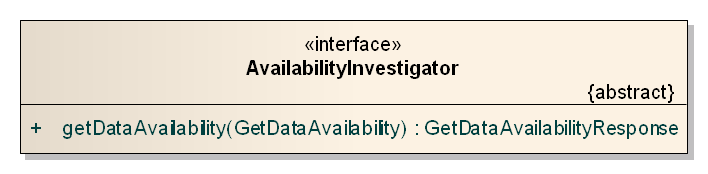


Figure : The AvailabilityInvestigator interface

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda/normal-behavior |
| REQ-003 When receiving a GetDataAvailability request, a service shall – in the normal case, i.e. without encountering an exception – create an according GetDataAvailabilityResponse as defined in and return it. |
| Note: behavior in case of exception is specified in section 7.4. |

Listing – (pseudo) algorithm to create a GetDataAvailabilityResponse

1. Create a GetDataAvailabilityResponse object (gdar).
2. If the request contains pointers to one or more offerings then put all observations from these offerings into the working set. Otherwise put all observations available to the service into the working set.
3. Remove any duplicate observations from the working set[[1]](#footnote-1).
4. Remove those observations from the working set whose phenomenonTime is not before and not after the *availabilityTimeframe* given in the request[[2]](#footnote-2).

Note: the availabilityTimeframe may be a time period that is not only located in the past – this is useful to also include observations in the availability study that are the result of simulations or forecasts.

1. Determine which features of interest, observed properties and procedures are targeted by the request:
   1. Create an empty proc\_set to store procedure values (processes) in – the set automatically removes duplicate values. If *useAllProcedures* is set to true in the request, gather all procedures associated to the observations contained in the working set and put them in the proc\_set. Otherwise get all *procedure* values contained in the request and put those into the proc\_set that are also contained in one or more of the observations of the working set.
   2. Create an empty foi\_set to store featureOfInterest values (features) in – the set automatically removes duplicate values. If *useAllFeaturesOfInterest* is set to true in the request, gather all featuresOfInterest associated to the observations contained in the working set and put them in the foi\_set. Otherwise get all *featureOfInterest* values contained in the request and put those into the foi\_set that are also contained in one or more of the observations of the working set.
   3. Create an empty obsProp\_set to store observedProperty values (properties) in – the set automatically removes duplicate values. If *useAllObservedProperties* is set to true in the request, gather all procedures associated to the observations contained in the working set and put them in the obsProp\_set. Otherwise get all *observedProperty* values contained in the request and put those into the obsProp\_set that are also contained in one or more of the observations of the working set.
2. For each value foi\_X in the foi\_set: go through the list of values in the obsProp\_set and for each combination of value obsProp\_X from the obsProp\_set and foi\_X do the following:
   1. Create a new FeaturePropertyTemporalRelationship object (fptr).
   2. Get all observations from the working set which have both foi\_X as value of the featureOfInterest and obsProp\_X as value of the observedProperty.
   3. If such observations exist:
      1. Add their phenomenonTime values to fptr.
      2. Create a FeatureOfInterestInfo object (foiInfo\_X) for foi\_X.
      3. Create an ObservedPropertInfo object (obsPropInfo\_X) for obsProp\_X.
      4. Add fptr to both foiInfo\_X and obsPropInfo\_X.
      5. Add both foiInfo\_X and obsPropInfo\_X to fptr.
      6. Add both foiInfo\_X and obsPropInfo\_X to gdar.
      7. Add fptr to gdar.
3. For each value obsProp\_Y in the obsProp\_set: go through the list of values in the proc\_set and for each combination of value proc\_Y from the proc\_set and obsProp\_Y do the following:
   1. Create a new ProcedurePropertyTemporalRelationship object (pptr).
   2. Get all observations from the working set which have both obsProp\_Y as value of the observedProperty and proc\_Y as value of the procedure.
   3. If such observations exist:
      1. Add their phenomenonTime values to pptr.
      2. Create a ProcedureInfo object for proc\_Y (procInfo\_Y).
      3. If an ObservedPropertInfo object has already been created for obsProp\_Y in step 6) get it; otherwise create a new ObservedPropertyInfo object. In both cases the object is in the following referred to as obsPropInfo\_Y.
      4. Add pptr to both procInfo\_Y and obsPropInfo\_Y.
      5. Add both procInfo\_Y and obsPropInfo\_Y to pptr.
      6. Add procInfo\_Y to gdar.
      7. Add obsPropInfo\_Y to gdar (unless this has already been done in step 6).
      8. Add pptr to gdar.
4. For each value proc\_Z in the proc\_set: go through the list of values in the foi\_set and for each combination of value foi\_Z from the foi\_set and proc\_Z do the following:
   1. Create a new ProcedureFeatureTemporalRelationship object (pftr).
   2. Get all observations from the working set which have both proc\_Z as value of the procedure and foi\_Z as value of the featureOfInterest.
   3. If such observations exist:
      1. Add their phenomenonTime values to pftr.
      2. If a FeatureOfInterestInfo object has already been created for foi\_Z in step 6) get it; otherwise create a new FeatureOfInterestInfo object. In both cases the object is in the following referred to as foiInfo\_Z.
      3. If a ProcedureInfo object has already been created for proc\_Z in step 7) get it; otherwise create a new ProcedureInfo object. In both cases the object is in the following referred to as procInfo\_Z.
      4. Add pftr to both procInfo\_Z and foiInfo\_Z.
      5. Add both procInfo\_Z and foiInfo\_Z to pftr.
      6. Add foiInfo\_Z to gdar (unless this has already been done in step 6).
      7. Add procInfo\_Z to gdar (unless this has already been done in step 7).
      8. Add pftr to gdar.
5. If the GetDataAvailability request contains a value for the *minimumTimeGap* property:
   1. Get all FeaturePropertyTemporalRelationship as well as ProcedurePropertyTemporalRelationship and ProcedureFeatureTemporalRelationship objects contained in gdar. For each of these relationship objects (rel), do the following:
      1. Get the list of phenomenonTime values (pt\_list\_A) in rel.
      2. Create a new empty list of phenomenonTime values (pt\_list\_B).
      3. Sort pt\_list\_A in ascending order.
      4. Create temporary variable elem\_A1.
      5. Get the first element of the sorted pt\_list\_A, remove it from that list and set elem\_A1 to its value.
      6. While pt\_list\_A is not empty, repeat:
         1. Get the first element of the sorted pt\_list\_A (called elem\_A2) and remove it from that list.
         2. Check the duration between elem\_A1 and elem\_A2:
            1. If it is smaller than the given *minimumTimeGap* duration: create the temporal bounding box of elem\_A1 and elem\_A2 and set the value of elem\_A1 to the resulting time period.
            2. Otherwise create a new entry in pt\_list\_B with the value of elem\_A1 and set elem\_A1 to the value of elem\_A2.
      7. Create a new entry in pt\_list\_B with the value of elem\_A1.
      8. Replace pt\_list\_A in rel with pt\_list\_B.

Implications of the algorithm described in

Listing 1:

* The client can define the scope of observations to include in the availability study to a certain time frame by setting the availabilityTimeframe to a suitable value (e.g. from 2010-05-10T00:00:00Z to 2010-05-11T00:00:00Z). In addition the scope can further be defined by including some offerings in the request which results in the availability study only being performed upon the observations associated with these offerings.
* All observations that have a phenomenonTime which does not temporally interact with the given time interval are ignored when computing the operation response.
* If no featureOfInterest values are provided in the request and useAllFeaturesOfInterest = false, the response does not include any FeatureOfInterestInfo element and thus also no FeaturePropertyTemporalRelationship or ProcedureFeatureTemporalRelationship (likewise for observedProperty and procedure).
* If all useAllXX properties are set to false and the request does not contain values for either the featureOfInterest, observedProperty or procedure, the response will be empty.
* If only one specific observation property - either the featureOfInterest, the procedure or the observedProperty – is targeted by the request (this could be done by including one or more values for the according property or setting its useAllXX property to true; values for the other properties would then be omitted and their useAllXX properties not set to true), the response will be empty.
* If a specific observation property was targeted in the request, for example a certain procedure, but no according ProcedureInfo element exists in the response, then no relationship to one of the featuresOfInterest or observedProperties represented in the response could be found. This depends upon the relationships the client wants to discover as well as the scope the client defined via the *availabilityTimeframe* and *offering* properties in the request.
* There can be no XXInfo element that does not have at least one XXTemporalRelationship.
* Via the minimumTimeGap request property, a client can considerably decrease the amount of phenomenonTime values in XXTemporalRelationship objects containd in the response – and thus the size of the response.

## Exceptions

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda/exception-structure |
| REQ-004 The structure of an exception returned by a service upon a GetDataAvailability request shall be an OWS Common Exception as specified in section 8.2 of [OGC 06-121r3]. |

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda/exception-behavior |
| REQ-005 When a service encounters an error while performing the GetDataAvailability operation, it shall return an OWS Common Exception with exception code (and implied meaning) as defined in . |
| Note: the normal operation behavior is specified in section 7.3. |

Table : Exceptions for the GetDataAvailability operation

|  |  |  |
| --- | --- | --- |
| **Exception Code** | | **Meaning of code & Locator value** |
| **Name / Value** | **Defined by** |
| OperationNotSupported | OGC  06-121r3 | As defined in table 25 of OGC 06-121r3. Further:  If a client invokes the GetDataAvailability operation on a service that does not implement the operation, an exception with this code shall be raised. |
| MissingParameterValue | As defined in table 25 of OGC 06-121r3. |
| InvalidParameterValue | As defined in table 25 of OGC 06-121r3. Further:  If the value of one of the request parameters does not conform to its type specification, an exception with this code shall be raised.  The service shall ignore unknown pointers to feature(s)OfInterest, observedProperties, offerings and procedures.  Note: in that case the according property will automatically be ignored by the algorithm described in . |
| OptionNotSupported | As defined in table 25 of OGC 06-121r3. Further:  Although no options are defined for the operation at this time, extensions could define such options. |
| NoApplicableCode | As defined in table 25 of OGC 06-121r3. |
| InvalidRequest | OGC 09-001 | As defined in table 29 of OGC 09-001. |
| RequestExtensionNotSupported | As defined in table 29 of OGC 09-001. |
| Note: the VersionNegotiationFailed as well as InvalidUpdateSequence exception codes defined by OGC 06-121r3 are not listed because they are not applicable to this operation. | | |

## Service Metadata

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda/service-metadata-listing |
| REQ-006 A Sensor Observation Service that implements the GetDataAvailability operation shall list this operation in the OperationsMetadata section of its Capabilities document. |

## Example

Consider the set of offerings and associated observations shown in the following listing.

Listing – Example of SOS offerings and observations

offering: http://service.com/sos/offering1

observation: http://service.com/sos/observation1\_1

procedure = http://my.org/sensors/process1

phenomenonTime = 2010-07-21T08:10:00Z

featureOfInterest = http://my.org/fois/foi1

observedProperty = http://authority.org/dictionary/def/property/airquality

observation: http://service.com/sos/observation1\_2

procedure = http://my.org/sensors/process1

phenomenonTime = 2010-07-21T08:30:00Z

featureOfInterest = http://my.org/fois/foi1

observedProperty = http://authority.org/dictionary/def/property/airquality

observation: http://service.com/sos/observation1\_3

procedure = http://my.org/sensors/process1

phenomenonTime = 2010-07-21T08:50:00Z

featureOfInterest = http://my.org/fois/foi1

observedProperty = http://authority.org/dictionary/def/property/airquality

observation: http://service.com/sos/observation1\_4

procedure = http://my.org/sensors/process1

phenomenonTime = 2010-07-21T10:00:00Z

featureOfInterest = http://my.org/fois/foi1

observedProperty = http://authority.org/dictionary/def/property/airquality

observation: http://service.com/sos/observation1\_5

procedure = http://my.org/sensors/process1

phenomenonTime = 2010-07-21T10:10:00Z

featureOfInterest = http://my.org/fois/foi1

observedProperty = http://authority.org/dictionary/def/property/airquality

offering: http://service.com/sos/offering2

observation: http://service.com/sos/observation2\_1

procedure = http://my.org/sensors/process2

phenomenonTime = 2010-07-21T08:12:00Z

featureOfInterest = http://my.org/fois/foi2

observedProperty = http://authority.org/dictionary/def/property/humidity

offering: http://service.com/sos/offering3

observation: http://service.com/sos/observation3\_1

procedure = http://my.org/sensors/process1

phenomenonTime = 2010-07-20T01:00:00Z

featureOfInterest = http://my.org/fois/foi3

observedProperty = http://authority.org/dictionary/def/property/airquality

offering: http://service.com/sos/offering4

observation: http://service.com/sos/observation4\_1

procedure = http://my.org/sensors/process4

phenomenonTime = 2010-07-210T11:59:00Z - 2010-07-21T12:01:00Z

featureOfInterest = http://my.org/fois/foi2

observedProperty = http://authority.org/dictionary/def/property/ozone\_concentration

Now assume that the client would like to know for which times observations are available that:

* have a phenomenonTime that is not before 2010-07-21T06:00:00Z or after 2010-07-21T12:00:00Z
* do not belong to offering http://service.com/sos/offering4
* have a featureOfInterest with identifier http://my.org/fois/foi1, http://my.org/fois/foi2, http://my.org/fois/foi3 or http://my.org/fois/foi4444
* contain any observed property

Further, the client is not interested to know about specific procedure relationships, so the response shall not contain any procedure information.

The client also wants to have the phenomenon times of two consecutive observations that are part of a specific feature/property relationship combined if the duration between them is less than fifteen minutes.

The XML encoding of the according request and response are provided in section 9.4 (Listing 3 and Listing 4).

# Requirements Class: XML Encoding

|  |  |
| --- | --- |
| Requirements Class | |
| http://www.opengis.net/spec/SOSGDA/1.0/req/xml | |
| Target Type | XML Instance |
| **Dependency** | http://www.opengis.net/doc/IS/SWES/2.0/clause/AnnexB |

In addition to this document, this standard includes a normative XML Schema document. This XML Schema document is bundled in a zip file with the present document. After OGC acceptance of a version 1.0.0 of this standard, this XML Schema document will also be posted online at the URL http://schemas.opengis.net/sosgda/1.0.0. In the event of a discrepancy between the bundled and online versions of the XML Schema document, the online file shall be considered authoritative.

This XML Schema document matches the UML package that contains the types described in the main section of this document and is named:

sosGetDataAvailability.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.

Table : XML Schema implementation of types defined by the GetDataAvailability Extension

|  |  |  |  |
| --- | --- | --- | --- |
| **UML class** | **object element** | **type** | **property type** |
| FeatureOfInterestInfo | gda:FeatureOfInterestInfo | gda:FeatureOfInterestInfoType | gda:FeatureOfInterestInfoPropertyType |
| FeaturePropertyTemporalRelationship | gda:FeaturePropertyTemporalRelationship | gda:FeaturePropertyTemporalRelationshipType | gda:FeaturePropertyTemporalRelationshipPropertyType |
| GetDataAvailability | gda:GetDataAvailability | gda:GetDataAvailabilityType | gda:GetDataAvailabilityPropertyType |
| GetDataAvailabilityResponse | gda:GetDataAvailabilityResponse | gda:GetDataAvailabilityResponseType | gda:GetDataAvailabilityResponsePropertyType |
| ObservedPropertyInfo | gda:ObservedPropertyInfo | gda:ObservedPropertyInfoType | gda:ObservedPropertyInfoPropertyType |
| ProcedureFeatureTemporalRelationship | gda:ProcedureFeatureTemporalRelationship | gda:ProcedureFeatureTemporalRelationshipType | gda:ProcedureFeatureTemporalRelationshipPropertyType |
| ProcedureInfo | gda:ProcedureInfo | gda:ProcedureInfoType | gda:ProcedureInfoPropertyType |
| ProcedurePropertyTemporalRelationship | gda:ProcedurePropertyTemporalRelationship | gda:ProcedurePropertyTemporalRelationshipType | gda:ProcedurePropertyTemporalRelationshipPropertyType |

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/xml/GeneralEncodingRule |
| REQ-007 The XML encoding of the conceptual types defined in this standard shall be as defined by the XML Schema files listed and referenced in clause . More specifically, the XML encoding of each conceptual type shall be valid against the XML Schema definition of the according mapping as defined in . |

# Requirements Class: GetDataAvailability operation – Web Service Model

|  |  |
| --- | --- |
| Requirements Class | |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws | |
| Target Type | Web Service Implementation |
| **Dependency** | http://www.opengis.net/spec/SOSGDA/1.0/req/gda |
| **Dependency** | http://www.opengis.net/doc/IS/SWES/2.0/clause/AnnexB |

## Request, Response and Exception Structure

An operation based web service that intends to implement the GetDataAvailability operation requires that the operation request and response be encoded in XML. As such, an XML Schema implementation of the operation request, response and exception report is needed.

Table : Implementation of primary conceptual types required by the GetDataAvailability operation

|  |  |
| --- | --- |
| **UML type** | **implementing XML Schema element** |
| GetDataAvailability (see section 7.1) | gda:GetDataAvailability  (see clause 8) |
| GetDataAvailabilityResponse (see section 7.2) | gda:GetDataAvailabilityResponse  (see clause 8) |
| Exception (see section 7.4) | ows:ExceptionReport  (see owsExceptionReport.xsd – OGC 06-121r3 Annex B) |

Note: provides a full mapping of all the conceptual types defined in this standard to their XML Schema implementation.

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws/structures |
| REQ-008 The XML implementation of the GetDataAvailability operation request, response and exception report conceptual model types shall follow the XML Schema elements as defined in . |

## Behavior and Use

Simple web service communication is established by sending HTTP POST requests to the service which generates an according response (or exception) and returns it in the HTTP POST response.

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws/normal-behavior |
| REQ-009 The GetDataAvailability operation implemented by a web service shall be invoked by sending an XML encoded GetDataAvailability request via HTTP POST to the service. The web service shall create an XML encoded GetDataAvailabilityResponse as defined in and return it in the HTTP POST response. |
| Note: behavior in case of exception is specified in section 9.3. |

## Exceptions

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws/exception-behavior |
| REQ-010 When a web service encounters an error while performing the GetDataAvailability operation, it shall return an XML encoded exception report (in the HTTP POST response) with exception code (and implied meaning) as defined in . |
| Note: the normal operation behavior is specified in section 9.2. |

## Example

The XML implementation of the example request given in section 7.5 is shown in the following listing.

Listing – GetDataAvailability operation request example – XML encoding

<gda:GetDataAvailability service="SOS" version="2.0" xmlns:gda="http://www.opengis.net/sosgda/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:gml="http://www.opengis.net/gml/3.2">

<gda:availabilityTimeframe>

<gml:TimePeriod gml:id="ID-01">

<gml:beginPosition>2010-07-21T06:00:00Z</gml:beginPosition>

<gml:endPosition>2010-07-21T12:00:00Z</gml:endPosition>

</gml:TimePeriod>

</gda:availabilityTimeframe>

<gda:featureOfInterest>http://my.org/fois/foi1</gda:featureOfInterest>

<gda:featureOfInterest>http://my.org/fois/foi2</gda:featureOfInterest>

<gda:featureOfInterest>http://my.org/fois/foi3</gda:featureOfInterest>

<gda:featureOfInterest>http://my.org/fois/foi4444</gda:featureOfInterest>

<gda:minimumTimeGap>PT15M</gda:minimumTimeGap>

<gda:offering>http://service.com/sos/offering1</gda:offering>

<gda:offering>http://service.com/sos/offering2</gda:offering>

<gda:offering>http://service.com/sos/offering3</gda:offering>

<gda:useAllObservedProperties>true</gda:useAllObservedProperties>

</gda:GetDataAvailability>

The response from the web service – given the constellation of offerings and observations shown in Listing 2– is provided in Listing 4.

Listing – GetDataAvailability operation response example – XML encoding

<gda:GetDataAvailabilityResponse xmlns:gda="http://www.opengis.net/sosgda/1.0" xmlns:swes="http://www.opengis.net/swes/2.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:xlink="http://www.w3.org/1999/xlink">

<gda:featurePropertyRelationship>

<gda:FeaturePropertyTemporalRelationship swes:id="FPTR\_ID-01">

<gda:phenomenonTime>

<gml:TimePeriod gml:id="TID-1">

<gml:beginPosition>2010-07-21T08:10:00Z</gml:beginPosition>

<gml:endPosition>2010-07-21T08:50:00Z</gml:endPosition>

</gml:TimePeriod>

</gda:phenomenonTime>

<gda:phenomenonTime>

<gml:TimePeriod gml:id="TID-2">

<gml:beginPosition>2010-07-21T10:00:00Z</gml:beginPosition>

<gml:endPosition>2010-07-21T10:10:00Z</gml:endPosition>

</gml:TimePeriod>

</gda:phenomenonTime>

<gda:targetFeature xlink:href="#FID-01"/>

<gda:targetProperty xlink:href="#PID-01"/>

</gda:FeaturePropertyTemporalRelationship>

</gda:featurePropertyRelationship>

<gda:featurePropertyRelationship>

<gda:FeaturePropertyTemporalRelationship swes:id="FPTR\_ID-02">

<gda:phenomenonTime>

<gml:TimePeriod gml:id="TID-3">

<gml:beginPosition>2010-07-210T11:59:00Z</gml:beginPosition>

<gml:endPosition>2010-07-21T12:01:00Z</gml:endPosition>

</gml:TimePeriod>

</gda:phenomenonTime>

<gda:targetFeature xlink:href="#FID-02"/>

<gda:targetProperty xlink:href="#PID-02"/>

</gda:FeaturePropertyTemporalRelationship>

</gda:featurePropertyRelationship>

<gda:propertyEntryPoint>

<gda:ObservedPropertyInfo swes:id="PID-01">

<gda:property>http://authority.org/dictionary/def/property/airquality</gda:property>

<gda:observedFor xlink:href="#FPTR\_ID-01"/>

</gda:ObservedPropertyInfo>

</gda:propertyEntryPoint>

<gda:propertyEntryPoint>

<gda:ObservedPropertyInfo swes:id="PID-02">

<gda:property>http://authority.org/dictionary/def/property/humidity</gda:property>

<gda:observedFor xlink:href="#FPTR\_ID-02"/>

</gda:ObservedPropertyInfo>

</gda:propertyEntryPoint>

<gda:featureOfInterestEntryPoint>

<gda:FeatureOfInterestInfo swes:id="FID-01">

<gda:feature>http://my.org/fois/foi1</gda:feature>

<gda:relatedProperty xlink:href="#FPTR\_ID-01"/>

</gda:FeatureOfInterestInfo>

</gda:featureOfInterestEntryPoint>

<gda:featureOfInterestEntryPoint>

<gda:FeatureOfInterestInfo swes:id="FID-02">

<gda:feature>http://my.org/fois/foi2</gda:feature>

<gda:relatedProperty xlink:href="#FPTR\_ID-02"/>

</gda:FeatureOfInterestInfo>

</gda:featureOfInterestEntryPoint>

</gda:GetDataAvailabilityResponse>

# Requirements Class: GetDataAvailability operation – SOAP Web Service Model

This chapter defines the realization of the GetDataAvailability operation for a web 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.

|  |  |
| --- | --- |
| Requirements Class | |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws-soap | |
| Target Type | SOAP Web Service Implementation |
| **Dependency** | http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws |
| **Dependency** | http://www.opengis.net/doc/IS/SWES/2.0/clause/19 |

## Behavior and use

The essential behavior of the GetDataAvailability operation in a SOAP based web service implementation is not different to that of a simple web service (see clause 9) or a pure service implementation (see clause 7). However, wrapping the XML encoded request and response elements in SOAP messages offers rich functionality to augment the delivery of these messages; adding addressing information to route the message to its final destination as well as securing the communication through the application of content encryption are only two examples.

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws-soap/normal-behavior |
| REQ-011 The GetDataAvailability operation implemented by a SOAP based web service shall be invoked by sending a GetDataAvailability request via the chosen communication protocol to the service.  The SOAP based web service shall create a GetDataAvailabilityResponse as defined in and return it via the communication protocol indicated by the request.  Both the GetDataAvailability request and –Response shall be encoded as defined in and be put into the body element of a SOAP 1.1 OR SOAP 1.2 envelope (as advertised by the service). |
| Note: a SOAP based web service usually uses a WSDL description file to indicate which SOAP version it supports and which communication protocol it implements. Additional information, for example whether WS-Addressing is supported/required, can also be advertised through WSDL. |

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 (requires a URI for both the request and the response message)

Table : Action URIs for the GetDataAvailability operation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Message  Facet a** | **Action URI a** | **Applicable in feature (Y=yes, N=no)** | | |
| **SOAP 1.1 SOAPAction** | **SOAP 1.2 action** | **WS-Addressing [action]** |
| GetDataAvailability request | http://www.opengis.net/sosgda/1.0/GetDataAvailability | Y | Y | Y |
| GetDataAvailability response | http://www.opengis.net/sosgda/1.0/GetDataAvailabilityResponse | N | N | Y |

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws-soap/action-URI |
| REQ-012 A SOAP based web service shall require (in case of request messages) and use (in case of response messages) action URIs for the SOAP as well as WS-Addressing actions as defined in  Table 15. |
| Note: Which feature is appropriate for a given service implementation depends upon the service interface. A specific service can support SOAP 1.1 and / or SOAP 1.2 based communication and can also support WS-Addressing. This standard does not define any further requirements in this regard. |

## Exceptions

|  |
| --- |
| Requirement |
| http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws-soap/exception-behavior |
| REQ-013 When a SOAP based web service encounters an error while performing the GetDataAvailability operation, it shall return a SOAP fault as defined in OGC [09-001] clause 19 via the communication protocol indicated by the request.  The exception report included in the SOAP fault shall use the exception code (and implied meaning) as defined in . |

## Example

The example GetDataAvailability request that was shown in Listing 3 has been augmented with a SOAP 1.2 envelope that contains addressing information – the result is shown in the following listing.

Listing – GetDataAvailability operation request example – SOAP 1.2 encoding using WS-Addressing

<soap12:Envelope xmlns:soap12="http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:gda="http://www.opengis.net/sosgda/1.0" xmlns:gml="http://www.opengis.net/gml/3.2">

<soap12:Header>

<wsa:MessageID>http://my.client.com/uid/msg-0010</wsa:MessageID>

<wsa:ReplyTo>

<wsa:Address>http://my.client.com/client/myReceiver</wsa:Address>

</wsa:ReplyTo>

<wsa:To>http://service.com/sos</wsa:To>

<wsa:Action>http://www.opengis.net/sosgda/1.0/GetDataAvailability</wsa:Action>

</soap12:Header>

<soap12:Body>

<gda:GetDataAvailability service="SOS" version="2.0">

<!-- details omitted for brevity -->

</gda:GetDataAvailability>

</soap12:Body>

</soap12:Envelope>

The GetDataAvailabilityResponse example from Listing 4 has been augmented in a similar way – which is shown in the following listing.

Listing – GetDataAvailability operation request example – SOAP 1.2 encoding using WS-Addressing

<soap12:Envelope xmlns:soap12="http://www.w3.org/2003/05/soap-envelope" xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:gda="http://www.opengis.net/sosgda/1.0" xmlns:gml="http://www.opengis.net/gml/3.2">

<soap12:Header>

<wsa:RelatesTo>http://my.client.com/uid/msg-0010</wsa:RelatesTo>

<wsa:To>ttp://my.client.com/client/myReceiver</wsa:To>

<wsa:Action>http://www.opengis.net/sosgda/1.0/GetDataAvailabilityResponse</wsa:Action>

</soap12:Header>

<soap12:Body>

<gda:GetDataAvailabilityResponse>

<!-- details omitted for brevity -->

</gda:GetDataAvailabilityResponse>

</soap12:Body>

</soap12:Envelope>

# Annex A – Abstract Test Suite (normative)

## Conformance Test Class: GetDataAvailability operation – Service implementation

|  |  |
| --- | --- |
| Conformance Test Class | |
| Name | http://www.opengis.net/spec/SOSGDA/1.0/conf/gda |
| **Requirements** | **http://www.opengis.net/spec/SOSGDA/1.0/req/gda** |
| **Dependency** | http://www.opengis.net/spec/SOS/2.0/conf/core |

### Structures

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/gda/exception-structure** | |
| Requirement(s) | REQ-004 |
| Test Purpose | To verify that the service returns exceptions that are structured correctly. |
| Test Method | Whenever an exception is returned by the service upon invocation of the GetDataAvailability operation, check that it is structured as defined in section 8.2 of [OGC 06-121r3]. |
| Test Type | Capability |

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/gda/structure** | |
| Requirement(s) | REQ-001, REQ-002, REQ-005 |
| Test Purpose | To verify that the structure of a GetDataAvailability request is checked by the service for validity and that a normal response by the service is also valid. |
| Test Method | Devise an invalid GetDataAvailability request and execute it. Verify that the service returns an InvalidRequest exception.  Devise a valid GetDataAvailability request and execute it. Verify that the service returns a valide GetDataAvailabilityResponse. |
| Test Type | Capability |

### Behavior

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/gda/behavior** | |
| Requirement(s) | REQ-003, REQ-005 |
| Test Purpose | To verify that the service correctly handles a GetDataAvailability invocation. |
| Test Method | For each exception code appropriate to the GetDataAvailability operation (see Table 12), design and execute a GetDataAvailability request that results in that exception. Verify that the service returns a valid exception with the according code and locator value.  Devise several GetDataAvailability requests that check various options of the algorithm with which the response is computed (see Listing 1). Verify that the response is as expected. |
| Test Type | Capability |

### Service Metadata

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/gda/service-metadata** | |
| Requirement(s) | REQ-006 |
| Test Purpose | To verify that the service lists the GetDataAvailability operation in its operations metadata. |
| Test Method | Design and execute a GetCapabilities request to retrieve the operations metadata section of the SOS’s Capabilities document. Verify that the GetDataAvailability operation is listed there as a supported operation. |
| Test Type | Capability |

## Conformance Test Class: XML Encoding

|  |  |
| --- | --- |
| Conformance Test Class | |
| Name | http://www.opengis.net/spec/SOSGDA/1.0/conf/xml |
| **Requirements** | **http://www.opengis.net/spec/SOSGDA/1.0/req/xml** |
| **Dependency** | http://www.opengis.net/spec/SWES/2.0/conf/XMLEncoding |

### General Encoding

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/xml/encoding** | |
| Requirement(s) | REQ-007 |
| Test Purpose | To verify that the XML instance of a GetDataAvailability request/response is encoded correctly. |
| Test Method | Verify that each GetDataAvailability request/response is valid according to its XML Schema definition defined in Table 13. |
| Test Type | Capability |

## Conformance Test Class: GetDataAvailability operation – Web Service implementation

|  |  |
| --- | --- |
| Conformance Test Class | |
| Name | http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws |
| **Requirements** | **http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws** |
| **Dependency** | http://www.opengis.net/spec/SOSGDA/1.0/conf/gda |
| **Dependency** | http://www.opengis.net/spec/SOSGDA/1.0/conf/xml |

### Structure

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws/structures** | |
| Requirement(s) | REQ-008 |
| Test Purpose | To verify that the XML instances of GetDataAvailability request/response and of service exceptions received as result of a GetDataAvailability invocation are encoded correctly. |
| Test Method | Verify that each GetDataAvailability request/response and exceptions returned by the service are valid according to its XML Schema definition as defined in Table 14. |
| Test Type | Capability |

### Behavior

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws/behavior** | |
| Requirement(s) | REQ-008 |
| Test Purpose | To verify that the web service correctly handles a GetDataAvailability invocation. |
| Test Method | For each exception code appropriate to the GetDataAvailability operation (see Table 12), design an XML encoded GetDataAvailability request that results in that exception and send it to the service via an HTTP POST request. Verify that the service returns an XML encoded exception (with the according code and locator value) via the HTTP response.  Devise several XML encoded GetDataAvailability requests that check various options of the algorithm with which the response is computed (see Listing 1). Send them to the service via HTTP POST. Verify that the service returns an XML encoded GetDataAvailabilityResponse via the HTTP response and that its value is as expected. |
| Test Type | Capability |

## Conformance Test Class: GetDataAvailability operation – SOAP Web Service implementation

|  |  |
| --- | --- |
| Conformance Test Class | |
| Name | http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws-soap |
| **Requirements** | **http://www.opengis.net/spec/SOSGDA/1.0/req/gda-ws-soap** |
| **Dependency** | http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws |
| **Dependency** | http://www.opengis.net/spec/SWES/2.0/conf/SOAPBinding |

### General

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws-soap/action-URI** | |
| Requirement(s) | REQ-012 |
| Test Purpose | To verify that the service recognizes and uses correct action URIs for the GetDataAvailability operation request and response. |
| Test Method | Depending upon the SOAP binding available at the service, execute a request for the GetDataAvailability operation. Verify that the client uses the correct SOAP action in the request as defined in Table 15. If WS-Addressing is used, verify that client/service use the correct WS-Addressing action URI in request/response as defined in Table 15. |
| Test Type | Capability |

|  |  |
| --- | --- |
| **Conformance Test** | |
| **http://www.opengis.net/spec/SOSGDA/1.0/conf/gda-ws-soap/behavior** | |
| Requirement(s) | REQ-011, REQ-012 |
| Test Purpose | To verify that the service handles a GetDataAvailability invocation made via SOAP correctly. |
| Test Method | Devise a valid XML encoded GetDataAvailability request that is wrapped in a SOAP envelope (in a version supported by the service). Send the request to the service via the communication protocol supported in the SOAP binding of the service (usually HTTP). Verify that the response is encoded correctly, meaning that a normal response (computed as defined by the algorithm shown in Listing 1) is returned with correct XML encoding in an appropriate SOAP envelope while an exception is returned encoded as a SOAP fault as defined in OGC [09-001] clause 19. |
| Test Type | Capability |

1. keep in mind that one observation may be associated with more than one offering [↑](#footnote-ref-1)
2. see TM\_RelativePosition in ISO 19108 for further details on the temporal relationships „before“ and „after“ [↑](#footnote-ref-2)