Open GIS Consortium Inc.

Date: 2002-04-19

Reference number of this OpenGIS© Project Document: **OGC 02-028**

Version: 0.5.1

Category: OpenGIS© OGC Interoperability Program Report-Engineering Specification

Editor: Tom McCarty, (Science Applications International Corporation)

Sensor Collection Service

Document type: OpenGIS© Interoperability Program Report
Document subtype: Engineering Specification
Document stage: Draft
Document language: English

Table of Contents

i. Preface

ii. Submitting organizations

This Interoperability Program Report – Engineering Specification is being submitted to the OGC Interoperability Program by the following organizations:

Science Applications International Corporation

iii. Document Contributor Contact Points

All questions regarding this submission should be directed to the editor or the submitters:

CONTACT	COMPANY	ADDRESS	PHONE/FAX	EMAIL
Tom McCarty	SAIC	8301 Greensboro Dr. MS E-4-5 McLean, VA 22102	(703) 676-4738 O (703) 862-9408 C (703) 676-8705 F	mccartyt@saic.com
Michael Fene	SAIC	4770 Eastgate Mall Rd. MS C2 San Diego, CA 92121	(858) 826-6309 O (858) 826-6174 F	michael.fene@saic.com

iv. Revision history

Date	Release	Author	Paragraph modified	Description
10/29/01	0.1.0	Michael Fene		Initial draft of SCS DIPR

12/24/01	0.2.0	Tom McCarty, Michael Fene	Re-write	This version includes a slightly more detailed description of the SCS in the informative section with some examples. The normative section was populated with a SCS request schema and schemas from related SWE by others.
1/22/02	0.3.0	Tom McCarty		Update interface descriptions and schemas
1/30/02	0.4.0	Tom McCarty, Michael Fene, Jeff Lansing		Eliminated the GetFeature and GetCoverage Interface. Update keyword-value pairs. Updated schemas
3/26/02	0.5.0	Tom McCarty		Incorporated lessons learned and implementations from the OWS 1.1 testbed
2002-04-19	02-028 (.5.1)	HAN	Various	Final OWS-1 review and edit; minor changes. Produced OGC 02-028.

v. Changes to the OpenGIS® Abstract Specification

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

Foreword

Attention is drawn to the possibility that some of the elements of this part of OGC 02-028 may be the subject of patent rights. The Open GIS Consortium Inc. shall not be held responsible for identifying any or all such patent rights.

This specification was developed under the OGC Web Services (OWS) 1.1 initiative.

Introduction

The Sensor Collection Service (SCS) is one component of the OGC Sensor Web, as being developed under a series of Sensor Web Enablement (SWE) activities. SWE is one of many categories of application services that demonstrate OGC Web Services (OWS). OWS refers to those services that demonstrate the OGC vision for geospatial data and application interoperability. The objective of the OWS 1.1 SWE activity is to develop a set of components that *discover*, *find* and *bind* access to a single or constellation of real-time, in-situ sensors heretofore referred to as a *Sensor Collection Service* (SCS). Once accessed, a SWE client application receives readings from a sensor via the SCS where it processes and displays or processes the data in a variety of manners including points on a map, interpolations, charts, etc. Specifically, the intent in this phase of OWS is to access fixed, point source in-situ sensors that measure water quality and meteorological parameters.

The Sensor Collection Service Implementation Specification defines the basic operations the SCS are at a minimum **GetCapabilities** and **GetObservation**. The GetCabilities interface will follow the conventions set forth in the OWS 1 Common Architecture DIPR. The GetObservation operation discussed in this document specifies the interface for requesting and receiving readings from a wide variety of sensor configurations.

During the course of this specification development, there was much discussion about implementing additional *Web Feature Server (WFS)* [OGCWFS] and *Web Coverage Server (WCS)* *[OGCWCS]* "like" operations as part of the SCS specification. However, the demonstration focused solely on the implementation of the SCS described in this document. OWS1.2 may consider the WFS and WCS Sensor Collection Services

In addition to the operations supported by the SCS itself, this document also describes the relationship of the SCS to SCS clients, Sensor Registries, SensorML, Measurements and Observations and Physical Quantities Dictionaries.

Sensor Collection Service Engineering Specification

1 Scope

This document is concerned only with the interfaces/operations and structure of the Sensor Collection Service. As defined in the Sensor Model Language IPR[OWSSML], sensors are devices for the measurement of physical quantities. There are a great variety of sensor types from simple visual thermometers to complex electron microscopes and radiometers on-board earth orbiting satellites. In some cases, sensing may be accomplished by a person rather than a device, and the result of the "measurement" may be a category rather than a numeric quantity. The basic function of the *Sensor Collection Service* (SCS) is to provide a Web-enabled interface to a *sensor*, *collection of sensors* or *sensor proxy*. Figure 1-1 illustrates the components of the Sensor Web Architecture at a high level.

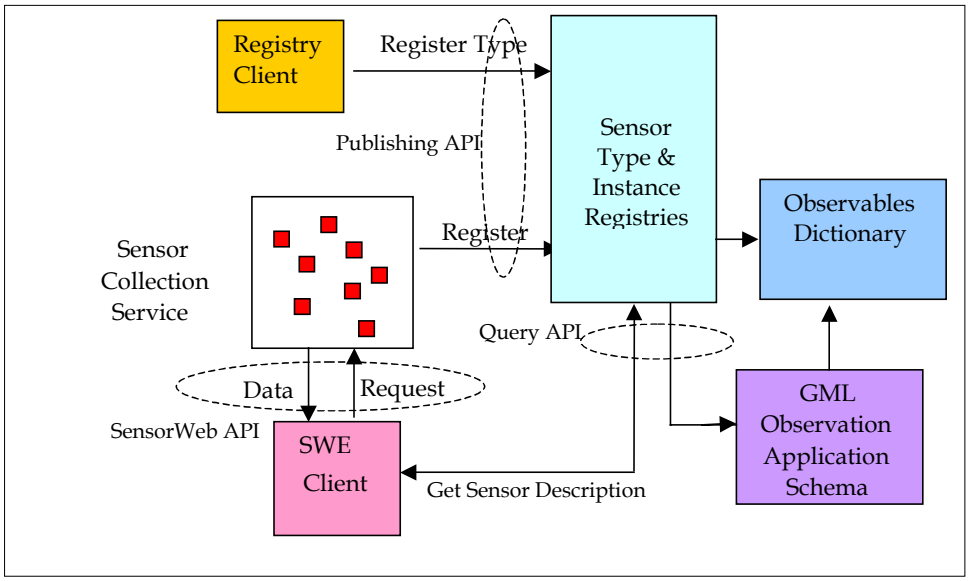


Figure 1-1 Sensor Web Component Architecture

The SCS is one of many interrelated components in the Sensor Web Environment. Figure 1-2 illustrates the role of the SCS. The SCS is a piece of software that accepts requests for information about sensors and provides access to and responds with information about a sensor, set of sensors or sensor proxy or sensor station. The SCS provides two basic types of information: 1) the specific capabilities of the sensors belonging to the SCS and 2) the observations or measurements supported by the SCS.

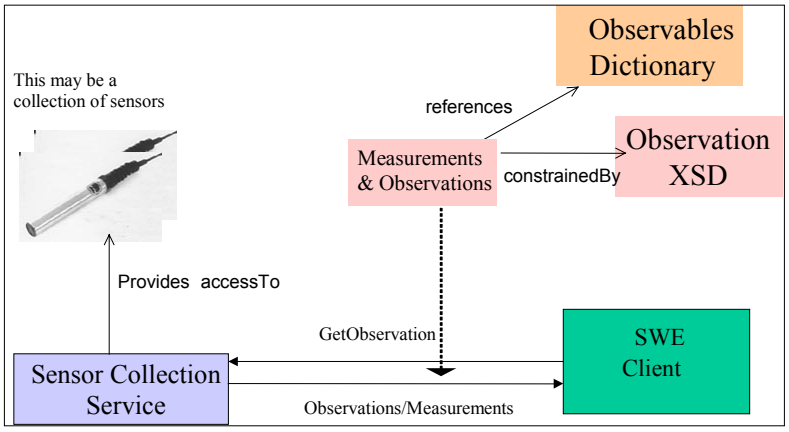


Figure 1-2Interactions with a Sensor Collection Service Instance

Metadata about SCS offerings and content are stored and accessed through an OGC service registry. Using the service registry interface, a Sensor Web client can register and obtain information about a particular *type* of sensor as well as specific instances of SCSs. Figure 1-3 illustrates the information and behaviour of the sensor registry.

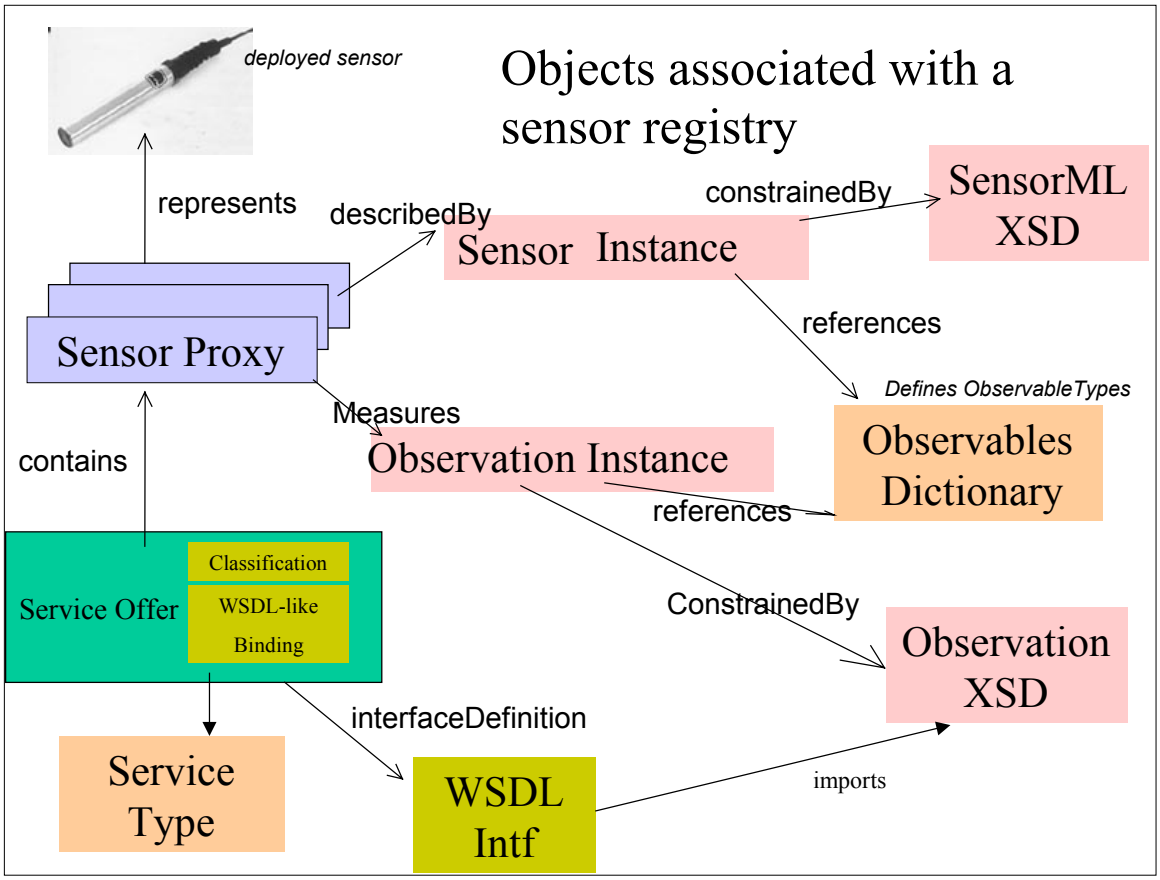


Figure 1-3 Objects Associated with a Sensor Registry

For the OWS 1.1 demonstration, a number of test bed participants implemented, tested and demonstrated various pieces of the SWE architecture, but were unable to demonstrate an end-to-end working model of the architecture. The following components were developed and demonstrated.

- Four SCS instances, three of which supported the R7 capabilities interface
- Two Registries that contained SCS service metadata
- One Registry that contained SensorML metadata
- The Observables Dictionary was documented, but not implemented

2 Conformance

Conformance and Interoperability Testing for this OGC Interoperability Program Report may be checked using all the relevant tests specified in Annex A (normative). The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in ISO 19105: Geographic information — Conformance and Testing.

3 Normative references

OGC Abstract Specification:

Topic 1 - Feature Geometry, Version 5

Topic 2 - Spatial Reference System, Version 4

Topic 5 - The OpenGIS Feature, Version 4

Topic 6 - The Coverage Type, Version 6

Topic 11 - Metadata. Same as ISO Metadata document 19115

Topic 12 - OGC Services Architecture, Version 4.1

OGC Implementation Specifications

OGC Grid Coverages Implementation Specification, Version 1.0

OGC Coordinate Transformation Services Implementation Specification, Version 1.0

OGC Geography Markup Language, Version 2.0

OGC Web Feature Services Implementation Specification, Version 0.0

Berners-Lee, T., Fielding, N., and Masinter, L., "Uniform Resource Identifiers (URI): Generic Syntax", IETF RFC 2396, August 1998, http://www.ietf.org/rfc/rfc2396.txt.
Bray, Paoli, Sperberg-McQueen, eds., "Extensible Markup Language (XML) 1.0", 2nd edition, W3C Recommendation, October 2000, http://www.w3.org/TR/2000/REC-xml.
European Petroleum Survey Group, "EPSG Geodesy Parameters", http://www.epsg.org/.
Fielding et. al., "Hypertext Transfer Protocol – HTTP/1.1," IETF RFC 2616, June 1999, http://www.ietf.org/rfc/rfc2616.txt.

4 Terms and definitions

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

4.1 Operation

Specification of an interaction that can be requested from an object to effect behavior (definition from ISO 19119).

4.2 Interface

An implementation of operations including the syntax of the interaction for a given distributed computing technology (definition from ISO 19119).

4.3 Service

A collection of operations, accessible through an interface, that allows a user to evoke behaviour of value to the user (definition from ISO 19119).

4.4 Service Instance

An actual implementation of a Service. Service Instance is synonymous with Server.

4.5 Client

A software component that can invoke an Operation from a Server.

4.6 Request

An invocation by a Client of an Operation.

4.7 Response

The result of an Operation returned from a Server to a Client.

4.8 Capabilities XML

Service-level metadata describing the operations and content available at a Service Instance.

5 Symbols (and Abbreviated Terms)

The following symbols and abbreviated terms are used in this document.

CGI Common Gateway Interface

DCP Distributed Computing Platform

DTD Document Type Definition

EPSG European Petroleum Survey Group

GIF Graphics Interchange Format

GIS Geographic Information System

GML Geography Markup Language

HTTP Hypertext Transfer Protocol

IETF Internet Engineering Task Force

JPEG Joint Photographic Experts Group

MIME Multipurpose Internet Mail Extensions

OGC Open GIS Consortium

OWS OGC Web Service

PNG Portable Network Graphics

RFC Request for Comments

SensorML Sensor Model Language

SLD Styled Layer Descriptor

SVG Scalable Vector Graphics

URL Uniform Resource Locator

WebCGM Web Computer Graphics Metafile

WCS Web Coverage Service

WFS Web Feature Service

WMS Web Map Service

XML Extensible Markup Language

6 Basic Service Elements

This clause specifies aspects of the OGC Sensor Collection Server behaviour that are independent of particular operations or are common to several operations.

6.1 Version Numbering and Negotiation

6.1.1 Version Number Form

The published specification version number contains three positive integers, separated by decimal points, in the form "x.y.z". The numbers "y" and "z" will never exceed 99. Each OWS specification is numbered independently.

6.1.2 Version Changes

A particular specification's version number **shall** be changed with each revision. The number **shall** increase monotonically and **shall** comprise no more than three integers separated by decimal points, with the first integer being the most significant. There may be gaps in the numerical sequence. Some numbers may denote experimental or interim versions. Service instances and their clients need not support all defined versions, but **shall** obey the negotiation rules below.

6.1.3 Appearance in Requests and in Service Metadata

The version number appears in at least two places: in the Capabilities XML describing a service, and in the parameter list of client requests to that service. The version number used in a client's request of a particular service instance **shall** be equal to a version number which that instance has declared it supports (except during negotiation as described below). A service instance may support several versions, whose values clients may discover according to the negotiation rules.

6.1.4 Version Number Negotiation

An OWS Client may negotiate with a Service Instance to determine a mutually agreeable specification version. Negotiation is performed using the GetCapabilities operation (described in Section 7.5) according to the following rules.

All Capabilities XML **shall** include a protocol version number. In response to a GetCapabilities request containing a version number, an OGC Web Service **shall** either respond with output that conforms to that version of the specification, **or** negotiate a mutually agreeable version if the requested version is not implemented on the server. If no version number is specified in the request, the server **shall** respond with the highest version it understands and label the response accordingly.

Version number negotiation occurs as follows:

1) If the server implements the requested version number, the server **shall** send that version.

2a) If a version unknown to the server is requested, the server **shall** send the highest version less than the requested version.

2b) If the client request is for a version lower than any of those known to the server, then the server **shall** send the lowest version it knows.

3a) If the client does not understand the new version number sent by the server, it **may** either cease communicating with the server **or** send a new request with a new version number that the client does understand but which is less than that sent by the server (if the server had responded with a lower version).

3b) If the server had responded with a higher version (because the request was for a version lower than any known to the server), and the client does not understand the proposed higher version, then the client **may** send a new request with a version number higher than that sent by the server.

The process is repeated until a mutually understood version is reached, or until the client determines that it will not or cannot communicate with that particular server.

Example 1: Server understands versions 1, 2, 4, 5 and 8. Client understands versions 1, 3, 4, 6, and 7. Client requests version 7. Server responds with version 5. Client requests version 4. Server responds with version 4, which the client understands, and the negotiation ends successfully.

Example 2: Server understands versions 4, 5 and 8. Client understands version 3. Client requests version 3. Server responds with version 4. Client does not understand that version or any higher version, so negotiation fails and client ceases communication with that server.

6.2 General HTTP Request Rules

At present, the only distributed computing platform (DCP) explicitly supported by OGC Web Services is the World Wide Web itself, or more specifically Internet hosts implementing the Hypertext Transfer Protocol (HTTP) [IETF RFC 2616]. Thus the Online Resource of each operation supported by a service instance is an HTTP Uniform Resource Locator (URL). The URL may be different for each operation, or the same, at the discretion of the service provider. Each URL **shall** conform to the description in [IETF RFC 2616] (Section 3.2.2 "HTTP URL") but is otherwise implementation-dependent; only the parameters comprising the service request itself are mandated by the OGC Web Services specifications.

HTTP supports two request methods: GET and POST. One or both of these methods may be defined for a particular OGC Web Service type and offered by a service instance, and the use of the Online Resource URL differs in each case.

6.2.1 Reserved characters in HTTP GET URLs

The URL specification [IETF RFC 2396] reserves particular characters as significant and requires that these be escaped when they might conflict with their defined usage. The present SCS specification explicitly reserves several of these characters for use in the query portion of HTTP

GET requests. When the characters "?", "&", "=", "/", ":" and "," appear in one of the roles defined in Table 1, they are to appear literally in the URL. When such characters appear elsewhere (for example, in the value of a parameter), they are to be encoded as defined in [IETF RFC 2396].

Table 1 — Reserved Characters in HTTP GET Query

Character	Reserved Usage
?	Separator indicating start of query string.
&	Separator between parameters in query string.
=	Separator between name and value of parameter.
/	Separator between MIME type and subtype in format parameter value.
:	Separator between Namespace and Identifier in SRS parameter value.
,	Separator between individual values in list-oriented parameters.

6.2.2 HTTP GET

An Online Resource URL intended for HTTP GET requests is in fact only a URL prefix to which additional parameters are appended in order to construct a valid Operation request. A URL prefix is defined as an opaque string including the protocol, hostname, optional port number, path, a question mark '?', and, **optionally**, one or more server-specific parameters ending in an ampersand '&'. The prefix uniquely identifies the particular service instance. A client appends the necessary request parameters as name/value pairs in the form "name=value&". The resulting URL **shall** be valid according to the HTTP Common Gateway Interface standard [CGI], which mandates the presence of '?' before the sequence of query parameters and the '&' between each parameter.

The URL prefix **shall** end in either a '?' (in the absence of additional server-specific parameters) or a '&'. In practice, however, Clients **should** be prepared to add a necessary trailing '?' or '&' before appending the Operation parameters defined in this specification in order to construct a valid request URL.

Table 2 summarizes the components of an operation request URL.

Table 2 — A general OGC Web Service Request

URL Component	Description
http://host[:port]/path?{name[=value]&}	URL prefix of service operation. [ ] denotes 0 or 1 occurrence of an optional part; {} denotes 0 or more occurrences. The prefix is entirely at the discretion of the service provider.
name=value&	One or more standard request parameter name/value pairs defined by an OGC Web Service. The actual list of required and optional parameters is mandated for each operation by the appropriate OWS specification.

6.2.3 HTTP POST

An Online Resource URL intended for HTTP POST requests is a complete and valid URL to which Clients transmit request parameters in the body of the POST request. A SCS **shall not** require additional parameters to be appended to the URL in order to construct a valid target for the Operation request.

6.3 General HTTP Response Rules

Upon receiving a valid request, the service **shall** send a response corresponding exactly to the request as detailed in the appropriate specification. Only in the case of Version Negotiation (described above) may the server offer a differing result.

Upon receiving an invalid request, the service **shall** issue a Service Exception as described in Section 6.7.

NOTE: As a practical matter, in the WWW environment a client should be prepared to receive either a valid result, or nothing, or any other result. This is because the client may itself have formed a non-conforming request that inadvertently triggered a reply by something other than an OGC Web Service, because the Service itself may be non-conforming, etc.

Response objects **shall** be accompanied by the appropriate Multipurpose Internet Mail Extensions (MIME) type [IETF RFC 2045] for that object. Allowable types for operation responses and service exceptions are discussed below.

Response objects **should** be accompanied by other HTTP entity headers as appropriate and to the extent possible. In particular, the Expires and Last-Modified headers provide important information for caching; Content-Length may be used by clients to know when data transmission is complete and to efficiently allocate space for results, and Content-Encoding or Content-Transfer-Encoding may be necessary for proper interpretation of the results.

6.4 Request Parameter Rules

6.4.1 Parameter Ordering and Case

Parameter names **shall not** be case sensitive, but parameter values **shall** be case sensitive. In this document, parameter names are typically shown in uppercase for typographical clarity, not as a requirement.

Parameters in a request **may** be specified in any order.

A SCS **shall** be prepared to encounter parameters that are not part of this specification. In terms of producing results per this specification, a SCS **shall not** require such parameters.

6.4.2 Parameter Lists

Parameters consisting of lists (for example, the SENSOR_ID and PROPERTY_NAME in SCS GetObservation) **shall** use the comma (",") as the separator between items in the list. Additional white space **shall not** be used to delimit list items. If a parameter value includes a space or comma, it **shall** be escaped using the URL encoding rules [IETF RFC 2396].

Individual entries in a list **may** be empty, as represented by two successive commas (",,").

6.5 Common Request Parameters

6.5.1 VERSION

The VERSION parameter specifies the protocol version number. The format of the version number, and version negotiation, are described in Section 6.1.

6.5.2 REQUEST

The REQUEST parameter indicates which service operation is being invoked. The value **shall** be the name of one of the operations offered by the SCS.

6.5.3 FORMAT

The FORMAT parameter specifies the output format of the response to an operation other than the capabilities operation.

An OGC Web Service **may** offer only a subset of the formats known for that type of operation, but the server **shall** advertise in its Capabilities XML those formats it does support and **shall** accept requests for any format it advertises. A Service Instance **may** optionally offer a new format not previously offered by other instances, with the recognition that clients are not required to accept or process an unknown format. If a request contains a Format not offered by a particular server, the server **shall** throw a Service Exception (with code "InvalidFormat").

A Client **may** accept only a subset of the formats known for that type of operation. If a Client and Service do not support any mutually agreeable formats, the Client may, at its discretion, cease communicating with that service, or search for an intermediary service provider that performs format conversion, or allow the user to choose other disposition methods (e.g., saving to local storage or passing to helper application).

Formats are expressed in both Capabilities XML and in operation requests using MIME types. Each Operation has a distinct list of supported formats. Some formats may be offered by several operations, and are then duplicated as needed in each list.

Generally, OGC Web Service MIME types are chosen from among those in common use on the Internet [9]. However, additional OGC-specific types have been adopted to distinguish among different types of XML-formatted content (the generic XML MIME types being text/xml and application/xml), as listed in Table 3.

Table 3 — OGC-Specific MIME Types

MIME Type	Document Content
application/vnd.ogc.wms_xml	WMS Capabilities XML
application/vnd.ogc.gml	Geography Markup Language XML [1]
application/vnd.ogc.se_xml	Service Exception XML
application/vnd.ogc.se_inimage	Image overwritten with Exception message.

application/vnd.ogc.se_blank	Blank image because Exception occurred.

6.5.4 EXCEPTIONS

The EXCEPTIONS parameter states the format in which to report errors. See Section 6.7 on Service Exceptions, below.

6.5.5 Spatial Reference System

The Spatial Reference System (SRS) is a text parameter that names a coordinate reference system code. The name includes a namespace prefix, a colon, a numeric identifier, and possibly a comma followed by additional parameters. This specification defines one namespace, EPSG, which is discussed below.

NOTE: The use of the term SRS is in keeping with WMS 1.0.0. A more modern definition uses Coordinate Reference System (CRS) when referring to spatial referencing by coordinates, and SRS when referring to spatial referencing by addresses or indexes.

OGC Web Services are **not** required to support all possible SRSes, but **shall** advertise in their Capabilities XML those projections which they do offer and **shall** accept requests for all advertised projections. If a request contains an SRS not offered by a particular server, the server **shall** throw a Service Exception (code = "InvalidSRS").

Clients are **not** required to support all possible SRSes. If a Client and Service do not support any mutually agreeable SRS, the Client may, at its discretion, cease communicating with that service, or search for an intermediary service provider that performs coordinate transformations, or allow the user to choose other disposition methods.

6.5.5.1 EPSG Namespace for SRS

The EPSG namespace makes use of the European Petroleum Survey Group tables [EPSG], which define numeric identifiers (the EPSG "CRS code," corresponding to the field "COORD_REF_SYS_CODE" in the EPSG database) for many common projections and which associate projection or coordinate metadata (such as measurement units or central meridian) for each identifier. An SRS name in the EPSG namespace includes only the prefix and the identifier, not any additional parameters. This format is used both as the value of the SRS parameter in a service request and as the value of an <SRS> element in the Capabilities XML.

The BBOX request parameter (Section 7.2.3.6) values for such a coordinate reference system **shall** be specified in the order minimum longitude, minimum latitude, maximum longitude, maximum latitude. The BBOX parameter values **shall** use the coordinate reference system units.

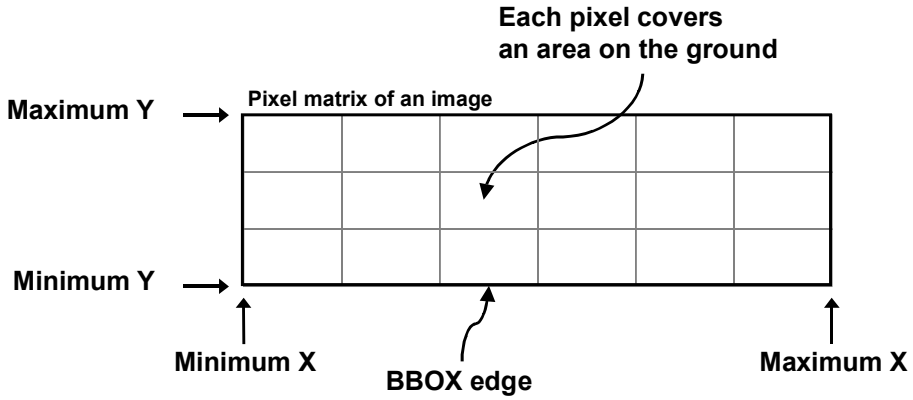
Some Projected Coordinate Reference Systems, e.g., "EPSG:30800" ("RT38 2.5 gon W", used in Sweden), have axes order other than X=East, Y=North. The BBOX request parameter values for such a coordinate system **shall** be specified in the order minimum Easting, minimum Northing, maximum Easting, maximum Northing. The BBOX parameters **shall** use the coordinate reference system units.

The BBOX parameters **shall** be specified using decimal or floating-point notation. In particular, sexagesimal degrees **shall** be specified as decimal degrees.

6.5.6 Bounding Box

The Bounding Box (BBOX) is a set of four comma-separated decimal, scientific notation, or integer values (if integers are provided where floating point is needed, the decimal point is assumed at the end of the number). These values specify the minimum X, minimum Y, maximum X, and maximum Y ranges, in that order, expressed in units of the SRS of the request, such that a rectangular area is defined in those units. When the SRS is a Platte Carrée projection of longitude and latitude coordinates, X refers to the longitudinal axis and Y to the latitudinal axis. The four bounding box values indicate the outside edges of a rectangle, as in Figure 5: minimum X is the left edge, maximum X the right, minimum Y the bottom, and maximum Y the top. The relation of the Bounding Box to the image pixel matrix is shown in the figure: the bounding box goes around the "outside" of the pixels of the image rather than through the centers of the border pixels. In this context, individual pixels have an area.

Figure 5 — Bounding Box Representation



A Bounding Box **should not** be a single point (area equals zero).

If a request contains an invalid Bounding Box (e.g., one whose minimum X is greater than or equal to the maximum X, or whose minimum Y is greater than or equal to the maximum Y) the server **shall** throw an exception.

If a request contains a Bounding Box whose area does not overlap at all with the BoundingBox advertised in the Capabilities XML for the requested geospatial data object, the server **should** return empty content (e.g., null observation set) for that element. Any elements that are partly or entirely contained in the Bounding Box **should** be returned in the appropriate format.

If the Bounding Box values are not defined for the given SRS (e.g., latitudes greater than 90 degrees in EPSG:4326), the server **should** return empty content for areas outside the valid range of the SRS.

In the particular case of longitude, the following behaviour **may** apply regarding the anti-meridian at 180 degrees of longitude. There is a legitimate desire for maps that span the anti-meridian (for example, a map centred on the Pacific Ocean). However, a strict interpretation of the previous paragraph suggests that areas beyond 180 degrees should be shown as empty content; this corresponds to the STRICT constraint below. A server **may** choose to relax this behaviour by instead applying the LOOSE constraint below. If **`minx`** is the west-most longitude in degrees and **`maxx`** is the east-most, then:

STRICT longitude constraint (default):

```
-180 <= minx < maxx <= 180
```

LOOSE longitude constraint (optional):

```
-180 <= minx < maxx < minx + 360 < 540
```

EXAMPLES: minx, maxx values and the corresponding scope of the bounding box:
-180,180 = Earth centred at Greenwich
0,360 = Earth with Greenwich at left edge
120,250 = Pacific Ocean

6.5.7 Time Dimension

Some geospatial information may be available at multiple times (for example, an hourly weather map). An OGC Web Service may announce available times in its Capabilities XML, and some operations include a parameter for requesting a particular time. The format of a time string is specified in Annex B. When providing temporal information, Servers **should** declare a default value in Capabilities XML unless there is compelling reason to behave otherwise, and Servers **shall** respond with the default value if one has been declared and the Client request does not include a value.

6.5.8 Additional Request Parameters

Most service requests require additional parameters (beyond REQUEST) to unambiguously state what result to construct. Each OGC Web Service specification defines the required and optional parameters for its operation(s).

6.5.9 Vendor-Specific Parameters

Finally, the requests allow for optional vendor-specific parameters (VSPs) that will enhance the results of a request. Typically, these are used for private testing of non-standard functionality prior to possible standardization. A generic client is **not** required or expected to make use of these VSPs.

An OGC Web Service **shall** produce a valid result even if VSPs are missing or malformed (i.e., the Service **shall** supply a default value), or if VSPs are supplied that are not known to the Service (i.e., the Service **shall** ignore unknown request parameters).

An OGC Web Service **may** choose not to advertise some or all of its VSPs. If VSPs are included in the Capabilities XML, then they **shall** be defined within an internal DTD section of that XML document. (An internal DTD comprises declarations enclosed in square brackets [] within the <DOCTYPE> element of the XML [see ref. 9].) In the absence of such parameters, the internal DTD is absent.

Clients **may** read the internal DTD and formulate requests using any VSPs advertised therein.

Vendors **should** choose vendor-specific parameter names with care to avoid clashes with standard parameters.

6.6 Service Result

The return value of a valid Service request **shall** correspond to the type requested in the FORMAT parameter. In an HTTP environment, the Content-type header of the response **shall** be exactly the MIME type given in the request.

Several OGC-specific MIME types have been defined in Table 3 for various XML document types (all of which would traditionally be labeled "text/xml"). To be compliant with this Specification a server **shall** return the appropriate OGC MIME type if defined, and the client **shall** be able to accept it, but it is **recommended** that the client also be prepared to accept the MIME type "text/xml" and deduce the specific content type through other means.

6.7 Service Exceptions

Upon receiving a request that is invalid according to the rules of the Distributed Computing Platform (DCP) in use, the service **may** issue an exception of a type valid in that DCP. For example: in the HTTP DCP, if the URL prefix is incorrect an HTTP 404 status code [IETF RFC 2616] is sent.

Upon receiving a request that is invalid according to the relevant OGC Web Services specification, the service **shall** issue a Service Exception Report as defined here and in Annex

A.3. The Report is meant to describe to the client application or its human user the reason(s) that the request is invalid.

The EXCEPTIONS parameter in a request indicates the format in which the Client wishes to be notified of Service Exceptions. The only value of the EXCEPTIONS parameter that is defined for all OGC Web Services is "`application/vnd.ogc.se_xml`", which means "Service Exception XML." Particular services may define other formats; this specification defines additional Exception formats for Web Map Servers in Section 7.2.3.11.

NOTE: A client should also be prepared for other returned values and types since there is a possibility that the Service instance is poorly behaved or that a request was directed at a non-compliant OGC Web Service.

Service Exception Report XML **shall** be valid according to the Service Exception DTD in Annex A.3. In an HTTP environment, the MIME type of the returned XML **shall** be "`application/vnd.ogc.se_xml`". Individual error messages appear as <ServiceException> elements within the <ServiceExceptionReport>. The messages can be formatted either as chunks of plain text or, if included in a character data (CDATA) section, as XML-like text containing angle brackets ("<" and ">"), as shown in the example Service Exception Report in Annex A.4.

Service Exceptions **may** include exception codes as indicated in Annex A.3. Services **shall not** use these codes for meanings other than those specified. This specification defines several exception codes. The specific codes and semantics of allowed exceptions may be extended by other OGC Web Service implementation specifications. Clients **may** use these codes to automate responses to Service Exceptions.

7 Sensor Collection Service Operations

7.1 Overview

During the OWS 1.1 test bed there was considerable thought and effort put in to the notion of unified services, whereby, the general behaviour interface to sensor information was common between services that returned raw measurements, features and coverages. This unified service notion was not achieved in this test bed but may be considered in future work. What was achieved was the implementation of a straight-forward interface for collecting observations from individual sensors, sensor constellations, sensor proxies and sensor stations.

7.2 Future

The following is a list of suggested improvements that should be considered for the next version of the SCS

7.2.1 SENSOR_ID vs STATION_ID

The test bed dialog most commonly referred to a SCS as a collection of sensors each having a unique SENSOR_ID. In practice, it was revealed that a client would more likely want to identify a unique STATION_ID and request observations from individual sensors by PROPERTY_NAME, not SENSOR_ID. It is recommended that SENSOR_ID be replaced by STATION_ID.

7.2.2 FORMAT

Not much analysis was done to define a sensible set of FORMAT values to be returned by a SCS. Up to the implementation and demonstration, the only defined format for a SCS was GML. The test bed starkly revealed that two implementations interpreted the Measurements and Observations and Measurements IPR [OWSOM] in two different ways to satisfy two different requirements. Two of the implementations were oriented toward getting the latest measurements from a geographically dispersed constellation of sensors. The data returned from these SCSs follows the schemas defined in the Observations and Measurements IPR, section 7.3, for *Measurement Collection* and *Measurement Array*. In the future, the FORMAT parameter should support all types of measurements and observations as defined in the Observations and Measurements IPR and should respond to specific requests of that format type.

7.3 Illustration Example

To illustrate the behaviour of the Sensor Collection Service Interface, consider the following example. The text below is the result of a query against the USGS Real- time National Water Information System (NWISWeb) http://water.usgs.gov/usa/nwis. In our terminology, the NWIS site is a *Proxy Sensor Collection System.* The record shows real-time sensor readings from one water quality *sensor package* in the NWIS Proxy SCS. This particular water quality sensor measures:

1. Gage Height (Feet)

2. Discharge (Cubic Feet Per Second)

3. Water Temperature (Degrees Celsius)

The temporal frequency reported by this sensor package is 1 hour. The following additional metadata is available for this site.

USGS 01362230 DIVERSION FROM SCHOHARIE RESERVOIR

LOCATION.--Lat 42°06'52", long 74°21'51", Ulster County, Hydrologic Unit 02020006, on left bank at outlet of Shandaken tunnel on Esopus Creek, 70 ft upstream from State Route 28 bridge, and 3.3 mi northwest of Phoenicia. Water-quality sampling site at discharge station.
PERIOD OF RECORD.--February 1924 to September 1950 and October 1960 to September 1996 (monthly and yearly discharge only), December 1996 to current year. (Prior to October 1950, published in WSP 1302, October 1960 to September 1970, in WSP 2102.) Records for October 1950 to September 1960 are unpublished and available in files of the Geological Survey.
GAGE.--Water-stage recorder. Concrete control since May 8, 1998. Elevation of gage is 800 ft above sea level, from topographic map.
REMARKS.--No estimated daily discharges. Records good. Flow completely regulated by Schoharie Reservoir. Records prior to October 1996 provided by Department of Environmental Protection, City of New York. Telephone gage-height and temperature telemeter at station.
EXTREMES FOR PERIOD OF RECORD.--Maximum discharge, 933 ft^3/s, Apr. 22, 23, 24, 25, 1999, gage height, 5.58 ft; minimum discharge, 0.14 ft^3/s, part or all of each day Dec. 30, 1996, Jan. 1, 3-11, 15-16, 18-22, 1997; minimum gage height since concrete control, 1.91 ft, July 3, 1998.
EXTREMES FOR 1999 WATER YEAR.--Maximum discharge, 933 ft3/s, Apr. 22, 23, 24, 25, gage height, 5.58 ft; minimum, 0.68 ft3/s, May 6, 7, Sept. 21, gage height, 1.96 ft.

00065 physical property GAGE HEIGHT, FEET

00060 physical property DISCHARGE, CUBIC FEET PER SECOND

00010 physical property WATER TEMPERATURE, DEGREES CELSIUS

Date / Time	GAGE HEIGHT (FEET) (DD 03)	Stream-flow (CFS) (DD 02)	TEMPER-ATURE WATER (DEG C) (DD 05)
12/21/2001 01:00	3.32	96.0	3.6

12/21/2001 02:00	3.31	94.0	3.7
12/21/2001 03:00	3.31	94.0	3.6
12/21/2001 04:00	3.30	93.0	3.6
12/21/2001 05:00	3.31	94.0	3.6
12/21/2001 06:00	3.31	94.0	3.5
12/21/2001 07:00	3.30	93.0	3.4
12/21/2001 08:00	3.32	96.0	3.3
12/21/2001 09:00	3.31	94.0	3.2
12/21/2001 10:00	3.32	96.0	3.2
12/21/2001 11:00	3.33	97.0	3.1
12/21/2001 12:00	3.31	94.0	3.0
12/21/2001 13:00	3.31	94.0	2.9
12/21/2001 14:00	3.32	96.0	2.9
12/21/2001 15:00	3.32	96.0	2.8
12/21/2001 16:00	3.32	96.0	2.7
12/21/2001 17:00	3.33	97.0	2.6
12/21/2001 18:00	3.32	96.0	2.5
12/21/2001 19:00	3.32	96.0	2.4
12/21/2001 20:00	3.32	96.0	2.3
12/21/2001 21:00	3.33	97.0	2.3
12/21/2001 22:00	3.32	96.0	2.4
12/21/2001 23:00	3.33	97.0	2.4
12/21/2001 24:00	3.32	96.0	2.4

7.4 Sensor Collection Service Interfaces

The Observations and Measurements DIPR [OWSOM] suggests that the SCS operations request and responses in the manner illustrated in Table 7-1. ***Note: This current specification will only address the GetObservation Interface.***

Table 7-1 Observations and Measurements DIPR Interfaces

Request	Response Type
GetObservation	*Observed Values*, *Value Array*, *Measurement Array* or *Measurement Collection* from a single sensor (or several sensors) to a client directly. The client would also need access to the related sensor and location information: either directly through other interfaces onto the sensor service, or indirectly from a registry. The client must specify the timing of the measurement.
GetFeature	Package information from the description of a single sensor (location, timing, sensor description) together with an *Observed Value* into a *Measurement Feature* or as a homogeneous *Feature collection* (*MeasurementArray*).
GetCoverage	Multiple measurements from a single sensor in a time-series as a *Coverage* or aggregate *Observed Values* from (a collection of) individual sensors, and add the contextual information, such as location, and then present these to the client as a *Coverage.*

7.5 GetCapabilities (required)

7.5.1 General

The purpose of the GetCapabilities operation is described in the Basic Service Elements section, above. In the particular case of a Sensor Collection Service, the response of a GetCapabilities request is general information about the service itself and specific information about the available sensors and observables.

7.5.2 GetCapabilities Request Overview

The general form of a GetCapabilities request is defined in the Basic Service Elements section. When making this request of a SCS, which may offer other OGC Web Services as well, it is necessary to indicate that the client seeks information about the SCS in particular. Thus, the SERVICE parameter of the request **shall** have the value "SCS" as shown in Table 4 below.

Table 7-2The parameters of a GetCapabilities request URL

Request Parameter	Required/ Optional	Description

VERSION=version	O	Request version
SERVICE=WMS	R	Service type
REQUEST=GetCapabilities	R	Request name
UPDATESEQUENCE=string	O	Sequence number or string for cache control

7.5.3 Request Parameters

7.5.3.1 VERSION

The **optional** VERSION parameter, and its use in version negotiation, is specified in the Basic Service Elements section.

7.5.3.2 SERVICE

The **required** SERVICE parameter indicates which of the available service types at a particular service instance is being invoked. This parameter allows the same URL prefix to offer Capabilities XML for multiple OGC Web Services.

When invoking GetCapabilities on a SCS that implements this version of the specification or a later one, the service_name value "SCS" **shall** be used.

7.5.3.3 REQUEST

This nature of the **required** REQUEST parameter is specified in the Basic Service Elements section. To invoke the GetCapabilities operation, the value "GetCapabilities" **shall** be used..

7.5.3.4 UPDATESEQUENCE

The **optional** UPDATESEQUENCE parameter is for maintaining cache consistency. Its value can be an integer, a timestamp in [ISO 8601:1988(E)] format (see Appendix B), or any other number or string. The server **may** include an UpdateSequence value in its Capabilities XML. If present, this value **should** be increased when changes are made to the Capabilities (e.g., when sensors, observables or observable values are added to the service). The server is the sole judge of lexical ordering sequence. The client **may** include this parameter in its GetCapabilities request. The response of the server based on the presence and relative value of UpdateSequence in the client request and the server metadata **shall** be according to Table 5:

Table 7-3 Use of UpdateSequence Parameter

Client Request UpdateSequence Value	Server Metadata UpdateSequence Value	Server Response
none	any	most recent Capabilities XML
any	none	most recent Capabilities XML
equal	equal	Exception: code=CurrentUpdateSequence
lower	higher	most recent Capabilities XML
higher	lower	Exception: code=InvalidUpdateSequence

7.5.4 GetCapabilities Response

The Basic Service Elements section specifies general rules about the GetCapabilities response.

7.5.4.1 Names vs. Titles

A number of elements have both a <Name> and a <Title>. Typically, the Name is a single word used for machine-to-machine communication while the Title is for the benefit of humans. For example, a dataset might have the Title "National Water Information System" and be requested using the Name "NWIS".

7.5.4.2 General Service Metadata

The first part of the Capabilities XML is a <Service> element providing general metadata for the service as a whole. It **shall** include a Name, Title, and Online Resource URL. Optionally, Abstract, Keyword List, Contact Information, Fees, and Access Constraints **may** be provided. The meaning of most of these elements is defined in [ISO 19115].

The Service Name **shall** be "OGC:SCS" in the case of a Sensor Collection Service.

The Service Title is at the discretion of the provider, and **should** be brief yet descriptive enough to identify this server in a menu with other servers.

The Abstract element allows a descriptive narrative providing more information about the enclosing object.

The OnlineResource element within the Service element can be used, for example, to point to the web site of the service provider. There are other OnlineResource elements used for the URL prefix of each supported operation.

A list of keywords or keyword phrases **should** be included to help catalog searching. Currently, no controlled vocabulary has been defined.

Contact Information **should** be included.

The reserved word "none" (case-insensitive) **shall** be used if there are no fees or access constraints, as follows: <Fees>none</Fees>, <AccessConstraints>none</AccessConstraints>. When constraints are imposed, no precise syntax has been defined for the place-holder elements.

7.5.4.3 Capability Metadata

The <Capability> element of the Capabilities XML names the actual operations that are supported by the service instance, the output formats offered for those operations, and the URL prefix for each operation. The XML DTD includes placeholders for Distributed Computing Platforms other than HTTP, and request methods other that HTTP GET, but currently only HTTP GET is defined for a basic SCS.

7.5.4.4 Sensor IDs

7.5.4.5 Property Types

7.5.4.6 SRS

7.5.4.7 BoundingBox

7.5.5 Output Formats

Format specifiers appear in several places in Capabilities XML: as valid output formats for an operation, as supported Exception formats, and as the format of content at external URLs.

The following example shows what a capabilities document might look like for a basic Sensor Collection Service. The GetCapabilities operation supports a variety of use cases including an early binding scenario where the developer or user has a priori knowledge of the existence and general content of the SCS, as well as the a late binding scenario where the capabilities of a SCS is discovered and consumed as a "behind-the-scenes" operation of a "smart" SCS client.

7.5.5.1 GetCapabilities Request Example

To request a capabilities document, a client would issue a GetCapabilities request

http://www.usgs.gov/usa/nwis/scs?request=GetCapabilities

7.5.5.2 GetCapabilities Response Example

The response to the GetCapabilities request would follow the information framework defined above. In general, the SCS service offer will specify the types of operations offered by the service including GetObservation and GetCapabilities. Additionally, the service offer will describe the content offered by the service including the Property Names that can be queried against. An example capabilities response for a SCS consisting of one sensor package would appear as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.3 U (http://www.xmlspy.com) by Mark Fisher (SAIC) -->
<OGC_Capabilities xmlns="http://www.opengis.net/ows" xmlns:ows="http://www.opengis.net/ows"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="0.0.1" service="SCS">
    <ServiceType serviceTypeId="http://www.opengis.net/ows/scs/0.0.1">
```

```
        <ows:typeWSDL xmlns="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="http://www.opengis.net/ows/servicetypes/scs/0.0.1"
xmlns:tns="http://www.opengis.net/ows/servicetypes/scs/0.0.1" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <types>
                <schema xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.opengis.net/ows" xmlns:ows="http://www.opengis.net/ows">
                    <include schemaLocation="WSDLTypesSection.xsd"/>
                    <complexType name="GetObservationRequestType">
                        <sequence>
                            <element name="PropertyName" type="ows:PropertyNameType"/>
                            <element name="SensorID" type="ows:SensorIDType"/>
                            <element name="BoundingBox" type="gml:BoxType"/>
                            <element name="Time" type="ows:ObservationTimeType"/>
                            <element name="Output">
                                <complexType>
                                    <sequence>
                                        <element name="Format" type="ows:ObservationsFormatType"/>
                                    </sequence>
                                </complexType>
                            </element>
                            <element name="Exceptions" type="ows:ExceptionFormatType" minOccurs="0"/>
                        </sequence>
                        <attribute name="version" type="ows:VersionValueType" use="required"/>
                        <attribute name="service" type="ows:ServiceValueType" use="required"/>
                    </complexType>
                    <simpleType name="PropertyNameType">
                        <annotation>
                            <appinfo>warning</appinfo>
                            <documentation>needs work. supposed to be a comma separated list of observable
names</documentation>
                        </annotation>
                        <restriction base="string">
                            <pattern value="*"/>
                        </restriction>
                    </simpleType>
                    <simpleType name="SensorIDType">
                        <restriction base="token">
                            <pattern value="(\d)(,\d)*"/>
                        </restriction>
                    </simpleType>
                    <simpleType name="ObservationTimeType">
                        <annotation>
                            <appinfo>warning</appinfo>
                            <documentation>needs work. either a single ISOdate or a comma separated list of
them or a beginning one, a slash, an ending one, a slash, and a "1" or a "01:00"</documentation>
                        </annotation>
                        <restriction base="string">
                            <pattern value="*"/>
                        </restriction>
                    </simpleType>
                </schema>
            </types>
            <message name="ExceptionResponse">
                <part name="body" element="ows:ServiceExceptionReport"/>
            </message>
            <message name="TextExceptionResponse">
                <part name="body" type="xsd:string"/>
            </message>
            <message name="GetCapabilitiesResponse">
                <part name="body" element="ows:OGC_Capabilities"/>
            </message>
            <message name="GetCapabilitiesRequest">
                <part name="body" type="ows:GetCapabilitiesRequestType"/>
            </message>
            <message name="GetObservationRequest">
```

```
                <part name="body" type="ows:GetObservationRequestType"/>
            </message>
            <portType name="GetCapabilitiesPortType">
                <operation name="GetCapabilities">
                    <input message="tns:GetCapabilitiesRequest"/>
                    <output message="tns:GetCapabilitiesResponse"/>
                    <fault name="exception" message="tns:TextExceptionMessage"/>
                    <fault name="exception" message="tns:SCSExceptionMessage"/>
                </operation>
            </portType>
            <portType name="GetObservationPortType">
                <operation name="GetObservation">
                    <input message="tns:GetObservationRequest"/>
                    <output message="tns:GetObservationResponse"/>
                    <fault name="exception" message="tns:TextExceptionMessage"/>
                    <fault name="exception" message="tns:SCSExceptionMessage"/>
                </operation>
            </portType>
        </ows:typeWSDL>
    </ServiceType>
    <ServiceOffer serviceInstanceId="com:saic:ysi:scs:0.0.1" contentTypeId="http://www.opengis.org/scslayer"
majorVersion="1" minorVersion="0">
        <ows:instanceWSDL xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:http="http://schemas.xmlsoap.org/wsdl/http/" xmlns:mime="http://schemas.xmlsoap.org/wsdl/mime/"
xmlns:tns="http://www.opengis.net/ows/servicetypes/scs/0.0.1">
            <binding name="GetCapabilitiesGETBinding" type="tns:GetCapabilitiesPortType">
                <http:binding verb="GET"/>
                <operation name="GetCapabilities">
                    <http:operation location="scs"/>
                    <input>
                        <mime:content type="application/vnd.ogc.kvp"/>
                    </input>
                    <output>
                        <mime:content type="application/vnd.ogc.scs_xml"/>
                    </output>
                    <fault name="exception">
                        <mime:content type="application/vnd.ogc.se_xml"/>
                    </fault>
                </operation>
            </binding>
            <binding name="GetObservationGETBinding" type="tns:GetObservationPortType">
                <http:binding verb="GET"/>
                <operation name="GetObservation">
                    <http:operation location="scs"/>
                    <input>
                        <mime:content type="application/vnd.ogc.kvp"/>
                    </input>
                    <output>
                        <mime:content type="image/jpeg"/>
                        <mime:content type="image/png"/>
                    </output>
                    <fault name="exception">
                        <mime:content type="application/vnd.ogc.se_xml"/>
                    </fault>
                </operation>
            </binding>
            <service name="SCS">
                <port name="GetCapabilitiesGETPort" binding="tns:GetCapabilitiesGETBinding">
                    <http:address location="http://cats.saic.com:8080/SCSYSI/"/>
                </port>
                <port name="GetObservationGETPort" binding="tns:GetObservationGETBinding">
                    <http:address location="http://cats.saic.com:8080/SCSYSI/"/>
                </port>
            </service>
        </ows:instanceWSDL>
```

```
            <ISO19119 xmlns="http://www.opengis.net/iso19119">
                <serviceType>
                    <nameValue>Sensor Collection Server</nameValue>
                    <nameNameSpace>SCS</nameNameSpace>
                </serviceType>
                <serviceTypeVersion>0.0.1</serviceTypeVersion>
                <citation>
                    <title>SAIC YSI SCS</title>
                </citation>
                <abstract>Sensor collection service</abstract>
            </ISO19119>
            <bindingTemplate>
                <operation>GetObservation</operation>
                <parameter name="version">
                    <enumeration>
                        <value>0.0.1</value>
                    </enumeration>
                </parameter>
            </bindingTemplate>
        </ServiceOffer>
        <ContentTypeList>
            <ContentType name="SCSLayer" contentTypeId="http://www.opengis.org/scslayer" id="scslayer"
majorVersion="1" minorVersion="0">
                <definition markupLanguage="##other"/>
            </ContentType>
        </ContentTypeList>
        <ContentInstanceList>
            <ContentInstance contentInstanceId="com:saic:scs:ysi" title="YSI Sensor collection"
contentTypeId="http://www.opengis.org/scslayer" serviceInstanceId="com:saic:ysi:scs:0.0.1">
                <VendorSpecific/>
                <authority url="http://www.saic.com" name="saic.com"/>
                <contact>
                    <contactInfo xmlns="http://www.opengis.net/iso19119">
                        <phone>
                            <voice>(703)-676-6030</voice>
                        </phone>
                        <address/>
                        <onLineResource>
                            <linkage xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
xlink:href="http://www.saic.com/"/>
                        </onLineResource>
                    </contactInfo>
                    <roleCode xmlns="http://www.opengis.net/iso19119" value="pointOfContact"/>
                </contact>
                <extent srsName="EPSG:4326">
                    <gml:coord xmlns:gml="http://www.opengis.net/gml">
                        <gml:X>-73.9219</gml:X>
                        <gml:Y>40.7858</gml:Y>
                    </gml:coord>
                    <gml:coord xmlns:gml="http://www.opengis.net/gml">
                        <gml:X>-73.9220</gml:X>
                        <gml:Y>40.7859</gml:Y>
                    </gml:coord>
                </extent>
                <nativeSRS>EPSG:4326</nativeSRS>
                <keywords>
                    <iso19119:keyword
xmlns:iso19119="http://www.opengis.net/iso19119">OWS</iso19119:keyword>
                    <iso19119:keyword
xmlns:iso19119="http://www.opengis.net/iso19119">Demo</iso19119:keyword>
                    <iso19119:keyword
xmlns:iso19119="http://www.opengis.net/iso19119">Data</iso19119:keyword>
                </keywords>
                <bindingTemplate>
                    <operation>GetObservation</operation>
```

```
                <parameter name="sensorid">
                    <enumeration>
                        <value>0741-SONDE-0</value>
                        <value>0741-SONDE-1</value>
                        <value>0741-MET</value>
                    </enumeration>
                </parameter>
            </bindingTemplate>
            <bindingTemplate>
                <operation>GetObservation</operation>
                <parameter name="SENSORID">
                    <enumeration>
                        <value>0741-SONDE-0</value>
                        <value>0741-SONDE-1</value>
                        <value>0741-MET</value>
                    </enumeration>
                </parameter>
                <parameter name="TIME">
                    <range>
                        <min>2002-02-01T12:00:00Z</min>
                        <max>PRESENT</max>
                    </range>
                </parameter>
                <parameter name="PROPERTYNAME">
                    <enumeration>
                        <value>WSPDMH</value>
                        <value>WSPDMH</value>
                        <value>WDIR</value>
                        <value>RH</value>
                        <value>ATEMPF</value>
                        <value>RAINRI</value>
                        <value>RAINI</value>
                        <value>SOLRL</value>
                        <value>BARI</value>
                        <value>DATMDY</value>
                        <value>THETIME</value>
                        <value>TEMPC</value>
                        <value>COND</value>
                        <value>SALP</value>
                        <value>DOXP</value>
                        <value>DOXC</value>
                        <value>PRESM</value>
                        <value>BATV</value>
                        <value>TEMPC</value>
                        <value>CONDU</value>
                        <value>SALP</value>
                        <value>DOXP</value>
                        <value>DOXC</value>
                        <value>PRESF</value>
                        <value>TURB</value>
                        <value>Chlorophyl</value>
                        <value>BATV</value>
                    </enumeration>
                </parameter>
            </bindingTemplate>
        </ContentInstance>
    </ContentInstanceList>
</OGC_Capabilities>
```

7.6 GetObservation (required)

Table 7-1 describes the Map Request. The request is typically encoded as a URL that is invoked on the WMS using the HTTP GET operation.

Table 7-4 - The Parameters of a GetObservation Request

Request Parameter	Required/ Optional	Description
REQUEST=GetObservation	R	The name of the SCS request.. Other valid value for SCS **GetCapabilities**.
VERSION=version	R	Request version. This DIPR defines version 0.5.0
PROPERTY_NAME=property_name_list	O	This is a comma separated list of the observable parameters as provided by the SCS capabilities.
SENSOR_ID=sensor_id_list	O	A comma separated list of sensor or sensor station identifiers (string) within a SCS. All sensor s within the SCS returned by default.
BBOX= minx, miny, maxx, maxy	O	Bounding box corners (lower left, upper right) in SRS units.
SRS=namespace:identifier	O	Spatial Reference System only applies with use of BBOX.
TIME=time	O	Temporal envelope of desired observations.
FORMAT=output_format	O	Output format of Observation data file. Default is GML3.
EXCEPTIONS=OGC_XML	O	The format in which exceptions are to be reported by the Map Server. *Optional*.

7.6.1 Request Parameters

7.6.1.1 VERSION

The **required** VERSION parameter is specified in the Basic Service Elements section.

7.6.1.2 REQUEST

The nature of the **required** REQUEST parameter is specified in the Basic Service Elements section. For GetObservation, the value "GetObservation" **shall** be used.

7.6.1.3 PROPERTY_NAME

The **optional** PROPERTY_NAME parameter lists the parameters to be returned by this GetObservation request. The value of the PROPERTY_NAME parameter is a comma-separated list of one or more valid observable parameter names.

If omitted, the SCS should return all of the PROPERTY_NAME values offered by the SCS.

7.6.1.4 SENSOR_ID

The **optional** SENSOR_ID parameter lists the unique ID(s) of the sensor(s) to be queried. The SENSOR_ID can be any unique string.

7.6.1.5 BBOX

The **optional** BBOX parameter allows a Client to request a set of sensors within a particular Bounding Box. The value of the BBOX parameter in a GetObservation request is a list of comma-separated numbers of the form "`minx,miny,maxx,maxy`". BBOX should be used in when specific SENSOR_ID's are not known or desired.

7.6.1.6 SRS

The **optional** SRS parameter states which Spatial Reference System applies to the values in the BBOX parameter.

7.6.1.7 TIME

The use of Time values is specified in Annex B.

7.6.1.8 FORMAT

The **required** FORMAT parameter states the desired format of the response to an operation. Supported values for a GetObservation request on a SCS instance are listed in the Capabilities XML.

7.6.1.9 EXCEPTIONS

The **optional** EXCEPTIONS parameter states the manner in which errors are to be reported to the client.

7.6.2 GetObservtion Response

The response to a valid GetObservation request **shall** be *Observed Values*, *ValueArray, Measurement Array* or *Measurement Collection* from a single sensor (or several sensors), referencing *Observable Types* and units of measure.

An invalid GetObservation request **shall** yield an error output in the requested Exceptions format (or a network protocol error response in extreme cases).

In an HTTP environment, the MIME type of the returned value's Content-type entity header **shall** match the format of the return value.

7.7 GetObservation Request Examples

In the following examples, the client is knows which parameters (or Property Types) are supported by each specific sensor in the SCS. Using the HTTP GET form, request examples would be as follows

1 This would get measurements for all sensors that have the Property Name "Nox" for the specific region and time period.

http://dev.polexis.com:8080/scs/scs?request=GetObservation&BBOX=-74.179,40.565,-73.757,40.864&srs=EPSG:4326&PropertyName=Nox&Time=2002-01-29T12:00:00Z/2002-01-29T16:00:00Z/1

2 This would get three sensor parameters for a specific sensor for a period of 3 hours in 1 hour intervals.

http://water.usgs.gov/usa/nwis/scs?VERSION=0.0.1&REQUEST=GetObservation&PropertyName&GageHeight,Discharge,WaterTemperature&SENSORID= USGS_01362230&TIME=2001-12-21T01:00-05:00/2001-12-21T04:00-05:00/01:00&EXCEPTIONS=OGC_XML

3 This would get all of the water discharge measurement for sensors in a specific region for a specific period of time.

http://water.usgs.gov/usa/scs?request=GetObservation&BBOX=-74.179,40.565,-73.757,40.864&srs=EPSG:4326&PropertyName=Disharge&Time=2002-01-28T12:00:00Z/2002-01-29T16:00:00Z/1

7.8 GetObservation Response Example

The following example illustrates the response document for example 1 above.

```
<?xml version="1.0" encoding="UTF-8"?>
<ows:MeasurementCollection xmlns:ows="http://www.opengis.net/ows" xmlns:gml="http://www.opengis.net/gml">
    <gml:name>New York State Ambient Air Monitoring System</gml:name>
    <gml:boundedBy>
        <gml:null>unknown</gml:null>
    </gml:boundedBy>
    <ows:measurementMember>
        <ows:MeasurementCollection>
            <gml:boundedBy>
                <gml:null>unknown</gml:null>
            </gml:boundedBy>
            <ows:measuredAt>
                <ows:Station gml:id="709310">
                    <gml:name>P.S. 59</gml:name>
                    <gml:location>
                        <gml:Point>
                            <gml:coordinates>-73.96708,40.75982</gml:coordinates>
                        </gml:Point>
                    </gml:location>
                    <ows:code>709310</ows:code>
                    <ows:url>http://www.dec.state.ny.us/website/dar/bts/airmon/709310site.htm</ows:url>
```

```
            </ows:Station>
        </ows:measuredAt>
        <ows:measurementMember>
            <ows:MeasurementArray observable="Nox">
                <gml:boundedBy>
                    <gml:null>unknown</gml:null>
                </gml:boundedBy>
                <ows:measurementMembers>
                    <ows:Measurement>
                        <gml:tInstant>
                            <gml:tPosition>2002-01-29T12:00:00Z</gml:tPosition>
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
                    </ows:Measurement>
                    <ows:Measurement>
                        <gml:tInstant>
                            <gml:tPosition>2002-01-29T13:00:00Z</gml:tPosition>
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
                    </ows:Measurement>
                    <ows:Measurement>
                        <gml:tInstant>
                            <gml:tPosition>2002-01-29T14:00:00Z</gml:tPosition>
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
                    </ows:Measurement>
                    <ows:Measurement>
                        <gml:tInstant>
                            <gml:tPosition>2002-01-29T15:00:00Z</gml:tPosition>
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
                    </ows:Measurement>
                    <ows:Measurement>
                        <gml:tInstant>
                            <gml:tPosition>2002-01-29T16:00:00Z</gml:tPosition>
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
                    </ows:Measurement>
                </ows:measurementMembers>
                <ows:unitsOfMeasure>PPM</ows:unitsOfMeasure>
            </ows:MeasurementArray>
        </ows:measurementMember>
    </ows:MeasurementCollection>
</ows:measurementMember>
<ows:measurementMember>
```

```
<ows:MeasurementCollection>
    <gml:boundedBy>
        <gml:null>unknown</gml:null>
    </gml:boundedBy>
    <ows:measuredAt>
        <ows:Station gml:id="709407">
            <gml:name>IS 52</gml:name>
            <gml:location>
                <gml:Point>
                    <gml:coordinates>-73.90216,40.81539</gml:coordinates>
                </gml:Point>
            </gml:location>
            <ows:code>709407</ows:code>
            <ows:url>http://www.dec.state.ny.us/website/dar/bts/airmon/709407site.htm</ows:url>
        </ows:Station>
    </ows:measuredAt>
    <ows:measurementMember>
        <ows:MeasurementArray observable="Nox">
            <gml:boundedBy>
                <gml:null>unknown</gml:null>
            </gml:boundedBy>
            <ows:measurementMembers>
                <ows:Measurement>
                    <gml:tInstant>
                        <gml:tPosition>2002-01-29T12:00:00Z</gml:tPosition>
                    </gml:tInstant>
                    <ows:resultOf>
                        <gml:Quantity>
                            <gml:amount>ND</gml:amount>
                        </gml:Quantity>
                    </ows:resultOf>
                </ows:Measurement>
                <ows:Measurement>
                    <gml:tInstant>
                        <gml:tPosition>2002-01-29T13:00:00Z</gml:tPosition>
                    </gml:tInstant>
                    <ows:resultOf>
                        <gml:Quantity>
                            <gml:amount>ND</gml:amount>
                        </gml:Quantity>
                    </ows:resultOf>
                </ows:Measurement>
                <ows:Measurement>
                    <gml:tInstant>
                        <gml:tPosition>2002-01-29T14:00:00Z</gml:tPosition>
                    </gml:tInstant>
                    <ows:resultOf>
                        <gml:Quantity>
                            <gml:amount>ND</gml:amount>
                        </gml:Quantity>
                    </ows:resultOf>
                </ows:Measurement>
                <ows:Measurement>
                    <gml:tInstant>
                        <gml:tPosition>2002-01-29T15:00:00Z</gml:tPosition>
                    </gml:tInstant>
                    <ows:resultOf>
                        <gml:Quantity>
                            <gml:amount>ND</gml:amount>
                        </gml:Quantity>
                    </ows:resultOf>
                </ows:Measurement>
                <ows:Measurement>
                    <gml:tInstant>
                        <gml:tPosition>2002-01-29T16:00:00Z</gml:tPosition>
```

```
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
                    </ows:Measurement>
                </ows:measurementMembers>
                <ows:unitsOfMeasure>PPM</ows:unitsOfMeasure>
            </ows:MeasurementArray>
        </ows:measurementMember>
    </ows:MeasurementCollection>
</ows:measurementMember>
<ows:measurementMember>
    <ows:MeasurementCollection>
        <gml:boundedBy>
            <gml:null>unknown</gml:null>
        </gml:boundedBy>
        <ows:measuredAt>
            <ows:Station gml:id="709609">
                <gml:name>College Point Po</gml:name>
                <gml:location>
                    <gml:Point>
                        <gml:coordinates>-73.85161,40.78768</gml:coordinates>
                    </gml:Point>
                </gml:location>
                <ows:code>709609</ows:code>
                <ows:url>http://www.dec.state.ny.us/website/dar/bts/airmon/709609site.htm</ows:url>
            </ows:Station>
        </ows:measuredAt>
        <ows:measurementMember>
            <ows:MeasurementArray observable="Nox">
                <gml:boundedBy>
                    <gml:null>unknown</gml:null>
                </gml:boundedBy>
                <ows:measurementMembers>
                    <ows:Measurement>
                        <gml:tInstant>
                            <gml:tPosition>2002-01-29T12:00:00Z</gml:tPosition>
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
                    </ows:Measurement>
                    <ows:Measurement>
                        <gml:tInstant>
                            <gml:tPosition>2002-01-29T13:00:00Z</gml:tPosition>
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
                    </ows:Measurement>
                    <ows:Measurement>
                        <gml:tInstant>
                            <gml:tPosition>2002-01-29T14:00:00Z</gml:tPosition>
                        </gml:tInstant>
                        <ows:resultOf>
                            <gml:Quantity>
                                <gml:amount>ND</gml:amount>
                            </gml:Quantity>
                        </ows:resultOf>
```

```
                        </ows:Measurement>
                        <ows:Measurement>
                            <gml:tInstant>
                                <gml:tPosition>2002-01-29T15:00:00Z</gml:tPosition>
                            </gml:tInstant>
                            <ows:resultOf>
                                <gml:Quantity>
                                    <gml:amount>ND</gml:amount>
                                </gml:Quantity>
                            </ows:resultOf>
                        </ows:Measurement>
                        <ows:Measurement>
                            <gml:tInstant>
                                <gml:tPosition>2002-01-29T16:00:00Z</gml:tPosition>
                            </gml:tInstant>
                            <ows:resultOf>
                                <gml:Quantity>
                                    <gml:amount>ND</gml:amount>
                                </gml:Quantity>
                            </ows:resultOf>
                        </ows:Measurement>
                    </ows:measurementMembers>
                    <ows:unitsOfMeasure>PPM</ows:unitsOfMeasure>
                </ows:MeasurementArray>
            </ows:measurementMember>
        </ows:MeasurementCollection>
    </ows:measurementMember>
</ows:MeasurementCollection>
```

Annex A (normative)
XML Schemas for Sensor Collection Service

A.1 SCS GetCapabilities Request Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.opengis.net/namespaces/scs"
xmlns="http://www.w3.org/2001/XMLSchema" xmlns:scs="http://www.opengis.net/namespaces/scs"
elementFormDefault="qualified">
    <!-- ============================================================
     Root element
     ============================================================ -->
    <element name="GetCapabilities" type="scs:GetCapabilitiesType"/>
    <!-- ============================================================
     Types
     ============================================================ -->
    <complexType name="GetCapabilitiesType">
        <attribute name="version" type="string" use="required"/>
    </complexType>
</schema>
```

A.2 SCS GetCapabilities Response Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.1 (http://www.xmlspy.com) by Rob Atkinson (Social Change Online) -->
<!-- this updated version based on use of an abstract bindingTemplate concept -->
<xs:schema targetNamespace="http://www.opengis.net/scs" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:scs="http://www.opengis.net/scs" xmlns:ogc="http://www.opengis.net/ogc"
xmlns:registry="urn:oasis:names:tc:ebxml-regrep:rim:xsd:2.0" xmlns:ows="http://www.opengis.net/ows"
elementFormDefault="qualified">
    <xs:import namespace="http://www.opengis.net/ogc" schemaLocation="../wfs/FilterCapability.xsd"/>
    <xs:import namespace="http://www.opengis.net/ogc" schemaLocation="../wfs/FilterRequest.xsd"/>
    <xs:import namespace="http://www.opengis.net/ows" schemaLocation="../ows/OGC_Capabilities.xsd"/>
    <!-- include registryModel  our common mechanism for making objects persistent and supporting discovery -->
    <xs:import namespace="urn:oasis:names:tc:ebxml-regrep:rim:xsd:2.0" schemaLocation="../ows/rim.xsd"/>
    <xs:element name="WFS_Capabilities" type="scs:WFS_CapabilitiesType"/>
    <xs:complexType name="WFS_CapabilitiesType">
        <xs:annotation>
            <xs:documentation>This is a template for service capabilities documents. Each Service type could
define a specific capabilities  this as required.</xs:documentation>
        </xs:annotation>
        <xs:complexContent>
            <xs:restriction base="ows:CapabilitiesType">
                <xs:sequence>
                    <xs:element ref="ows:ServiceType" maxOccurs="unbounded"/>
                    <xs:element ref="scs:ObservationContents" maxOccurs="unbounded"/>
                    <xs:element ref="ows:ContentTypeList" minOccurs="0"/>
                    <xs:element ref="ows:ContentInstanceList" minOccurs="0"/>
                    <xs:element name="TaxonomyScheme" type="ows:ClassificationSchemeType"
minOccurs="0" maxOccurs="unbounded"/>
                    <!-- TODO - these should be refactored to be classificationNodes of RIM -->
                </xs:sequence>
                <xs:attribute name="version" type="xs:string"/>
                <xs:attribute name="service" type="xs:string"/>
            </xs:restriction>
        </xs:complexContent>
    </xs:complexType>
    <!-- A FeatureType as exposed by a WFS instance is a recipe for invocation by the getFeature() operation -->
```

```
    <xs:element name="ObservationContents" type="scs:ObservationContentsType"
substitutionGroup="ows:ServiceOffer">
        <xs:annotation>
            <xs:documentation>Recursively included FeatureContents objects refine the parent according to the
rules of "bindingTemplate theory"</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:complexType name="ObservationContentsType">
        <xs:complexContent>
            <xs:extension base="ows:SpatiallyBoundedServiceOfferingType">
                <xs:sequence>
                    <xs:element name="ObservableType" type="xs:anyURI" minOccurs="0"
maxOccurs="unbounded">
                        <xs:annotation>
                            <xs:documentation>Actually a classification node?</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element ref="scs:ObservationContents" minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <!-- these things look just like WFS filters at this stage - we probably want to specialise them further to "fix"
particular shapes of filters or replace with a "well known query" model -->
    <xs:element name="FilterDomain" type="scs:FilterDomainType">
        <xs:annotation>
            <xs:documentation>This is a complex object whose semantics and structure must be interprested by
the client. This needs more work, but basically the idea is that a Filter may be represented as a QueryByExample
and the parameters that may be varied specified "out of band". How best to reference an element within the filter is
the question - perhaps Xpath would make sense here? </xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:complexType name="FilterDomainType">
        <xs:annotation>
            <xs:documentation>This type procvides the pattern for WFS instances to restrict the allowalble filters.
Instantiate an object of this type in the domain of the FILTER message part using the "body" choice of a domain
restriction.</xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element name="filterParameter" type="scs:FilterParameterType" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element ref="ogc:Filter" minOccurs="0">
                <xs:annotation>
                    <xs:documentation>This filter is a filter template that restricts the pattern of the filter, and
provides placeholders for the parameters named in this object. If absent, a simple pattern of properties ANDed is
assumed.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="FilterParameterType">
        <xs:annotation>
            <xs:documentation>Binds a paramterDomain to a named object within the Filter
QueryByExample</xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element name="filterNode" type="xs:string">
                <xs:annotation>
                    <xs:documentation>References the node within the Filter object that the following values may
be bound to. Could be a name or an Xpath expression ?</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="values" type="ows:parameterDomainType">
                <xs:annotation>
```

```
                    <xs:documentation>This filter is a filter template that restricts the pattern of the filter, and
provides placeholders for the parameters named in this object. If absent, a simple pattern of properties ANDed is
assumed.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

A.3 SCS GetObservation Request Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
    <xs:complexType name="BoundingBoxType">
        <xs:sequence>
            <xs:element name="coord" type="coordType" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="SRS" type="xs:string" use="required"/>
    </xs:complexType>
    <xs:element name="Format">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:enumeration value="GML3"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:element>
    <xs:element name="GetObservation">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="BoundingBox" type="BoundingBoxType"/>
                <xs:element ref="PropertyName"/>
                <xs:element ref="SensorID"/>
                <xs:element ref="Time"/>
                <xs:element name="Output" type="OutputType"/>
            </xs:sequence>
            <xs:attribute name="version" type="xs:string" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:complexType name="OutputType">
        <xs:sequence>
            <xs:element ref="Format"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="PropertyName" type="xs:string"/>
    <xs:element name="SensorID" type="xs:string"/>
    <xs:element name="Time" type="xs:string"/>
    <xs:element name="X" type="xs:decimal"/>
    <xs:element name="Y" type="xs:decimal"/>
    <xs:complexType name="coordType">
        <xs:sequence>
            <xs:element ref="X"/>
            <xs:element ref="Y"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```

A.4 SCS GetObservation Response Schema

In this schema we derive XML types for measurements, which map values to location, timing and sensor. The schema imports the Value declarations.

MeasurementBase Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.opengis.net/ows" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:gml="http://www.opengis.net/gml" xmlns:ows="http://www.opengis.net/ows"
xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified" attributeFormDefault="unqualified"
version="0.91">
  <xs:annotation>
    <xs:documentation>
measurementBase.xsd

A GML conformant schema
for Measurements
i.e. Time and Space located Features with properties containing Measurements made by Sensors

derived from gml3 Observations
restricts valueOf properties to be resultOf properties containing a Measurement
adds Measurement collections

SJDC  2002-01-29
</xs:documentation>
  </xs:annotation>
  <!-- ============================================================================ -->
  <!-- bring in other schemas -->
  <xs:import namespace="http://www.opengis.net/gml" schemaLocation="./observationAndValue.xsd"/>
  <!-- ============================================================================ -->
  <!-- global elements -->
  <xs:element name="Measurement" type="ows:MeasurementType" substitutionGroup="gml:_Observation"/>
  <!-- -->
  <xs:element name="MeasurementCollection" type="ows:MeasurementCollectionType"
substitutionGroup="ows:Measurement"/>
  <xs:element name="MeasurementArray" type="ows:MeasurementArrayType"
substitutionGroup="ows:MeasurementCollection"/>
  <!-- components for MeasurementCollections -->
  <xs:element name="measurementMember" type="ows:MeasurementMemberType"
substitutionGroup="gml:featureMember"/>
  <xs:element name="measurementMembers" type="ows:MeasurementMembersType"
substitutionGroup="gml:featureMember"/>
  <!-- -->
  <xs:element name="Station" type="ows:LocatedFeatureType" substitutionGroup="gml:_Feature"/>
  <!-- Properties of Measurements -->
  <xs:element name="measuredAt" type="gml:LocationPropertyType" substitutionGroup="gml:location"/>
  <xs:element name="resultOf" type="gml:ValuePropertyType" substitutionGroup="gml:valueOf"/>
  <xs:element name="errorIn" type="ows:ErrorPropertyType" substitutionGroup="gml:valueOf"/>
  <xs:element name="measuredBy" type="ows:RelatedFeatureType" substitutionGroup="gml:_property">
    <xs:annotation>
      <xs:documentation>This element contains or points to a Sensor description in SensorML.
      A restricted type definition will be constructed when SensorML is available. </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="relatedFeature" type="ows:RelatedFeatureType" substitutionGroup="gml:_property"/>
  <xs:element name="relatedMeasurement" type="ows:RelatedMeasurementType"
substitutionGroup="ows:relatedFeature"/>
  <xs:element name="relatedStation" type="ows:RelatedStationType" substitutionGroup="ows:relatedFeature"/>
  <!-- ============================================================================ -->
  <!-- Measurement types -->
  <xs:complexType name="MeasurementBaseType">
    <xs:annotation>
      <xs:documentation>restrict
      location to be measuredAt
```

```
    valueOf to be resultOf</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
   <xs:restriction base="gml:AbstractObservationType">
    <xs:sequence>
     <xs:element ref="gml:_metaDataProperty" minOccurs="0"/>
     <xs:element ref="gml:description" minOccurs="0"/>
     <xs:element ref="gml:name" minOccurs="0"/>
     <xs:element ref="gml:boundedBy" minOccurs="0"/>
     <xs:group ref="gml:dynamicProperties"/>
     <xs:element ref="ows:measuredAt" minOccurs="0"/>
     <xs:element ref="ows:resultOf"/>
    </xs:sequence>
   </xs:restriction>
  </xs:complexContent>
 </xs:complexType>
 <xs:complexType name="MeasurementType">
  <xs:annotation>
   <xs:documentation>  add errrorIn, measuredBy and  relatedFeature
       errorIn gives an estimate of the error of the result.
       measuredBy points to the sensor or party or software that generated the measurement
       relatedFeature is a pointer to another arbitrary feature
</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
   <xs:extension base="ows:MeasurementBaseType">
    <xs:sequence>
     <xs:element ref="ows:errorIn" minOccurs="0"/>
     <xs:element ref="ows:measuredBy" minOccurs="0"/>
     <xs:element ref="ows:relatedFeature" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>
 <!-- ========================================================================== -->
 <!-- Measurement Collection types -->
 <xs:complexType name="MeasurementCollectionBaseType">
  <xs:annotation>
   <xs:documentation>location is called "measuredAt"</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
   <xs:restriction base="gml:AbstractObservationType">
    <xs:sequence>
     <xs:element ref="gml:_metaDataProperty" minOccurs="0"/>
     <xs:element ref="gml:description" minOccurs="0"/>
     <xs:element ref="gml:name" minOccurs="0"/>
     <xs:element ref="gml:boundedBy"/>
     <xs:group ref="gml:dynamicProperties"/>
     <xs:element ref="ows:measuredAt" minOccurs="0"/>
    </xs:sequence>
   </xs:restriction>
  </xs:complexContent>
 </xs:complexType>
 <xs:complexType name="MeasurementCollectionType">
  <xs:annotation>
   <xs:documentation>add member, measuredBy, relatedFeature</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
   <xs:extension base="ows:MeasurementCollectionBaseType">
    <xs:sequence>
     <xs:element ref="ows:measurementMember" minOccurs="0" maxOccurs="unbounded"/>
     <xs:element ref="ows:measuredBy" minOccurs="0"/>
     <xs:element ref="ows:relatedFeature" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>
 <xs:complexType name="MeasurementArrayBaseType">
  <xs:annotation>
```

```
      <xs:documentation>All members must be of the same type</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:restriction base="ows:MeasurementCollectionType">
        <xs:sequence>
          <xs:element ref="gml:_metaDataProperty" minOccurs="0"/>
          <xs:element ref="gml:description" minOccurs="0"/>
          <xs:element ref="gml:name" minOccurs="0"/>
          <xs:element ref="gml:boundedBy"/>
          <xs:group ref="gml:dynamicProperties"/>
          <xs:element ref="ows:measuredAt" minOccurs="0"/>
          <xs:element ref="ows:measurementMembers" minOccurs="0"/>
          <xs:element ref="ows:measuredBy" minOccurs="0"/>
          <xs:element ref="ows:relatedFeature" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="MeasurementArrayType">
    <xs:annotation>
      <xs:documentation>
      Adds the referenceSystem - may be used if the array members are scalar Values
      Select only one from dictionary, categoryFrame, uom and frame
Add optional "members" or "observable" attribute for weak typing</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="ows:MeasurementArrayBaseType">
        <xs:sequence>
          <xs:group ref="gml:referenceSystem"/>
        </xs:sequence>
        <xs:attribute name="members" type="xs:Name" use="optional"/>
        <xs:attribute name="observable" type="xs:anyURI" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <!-- components of Measurement Collections -->
  <xs:complexType name="MeasurementMemberType">
    <xs:complexContent>
      <xs:restriction base="gml:AbstractAggregationType">
        <xs:sequence>
          <xs:element ref="ows:Measurement" minOccurs="0"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="MeasurementMembersType">
    <xs:complexContent>
      <xs:restriction base="gml:AbstractArrayAggregationType">
        <xs:sequence>
          <xs:element ref="ows:Measurement" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <!-- ========================================================================== -->
  <xs:complexType name="ErrorPropertyType">
    <xs:annotation>
      <xs:documentation>add a "type" parameter</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="gml:ValuePropertyType">
        <xs:attribute name="errorType" type="xs:string"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <!-- ========================================================================== -->
  <!-- Related Features -->
  <xs:complexType name="RelatedFeatureType">
    <xs:annotation>
```

```
    <xs:documentation>xlink:role compulsory in this context?</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element ref="gml:_Feature" minOccurs="0"/>
  </xs:sequence>
  <xs:attributeGroup ref="gml:AssociationAttributeGroup"/>
</xs:complexType>
<!-- -->
<xs:complexType name="RelatedMeasurementType">
  <xs:complexContent>
    <xs:restriction base="ows:RelatedFeatureType">
      <xs:sequence>
        <xs:element ref="ows:Measurement" minOccurs="0"/>
      </xs:sequence>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
<!-- -->
<xs:complexType name="RelatedStationType">
  <xs:complexContent>
    <xs:restriction base="ows:RelatedFeatureType">
      <xs:sequence>
        <xs:element ref="ows:Station" minOccurs="0"/>
      </xs:sequence>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
<!-- ========================================================================= -->
<!-- utility elements -->
<xs:element name="localDictionary" type="ows:DictionaryPropertyType" substitutionGroup="gml:_metaDataProperty"/>
<xs:element name="SkipReference" type="ows:SkipReferenceType" substitutionGroup="gml:_MetaData"/>
<xs:element name="Definition" type="ows:DefinitionType" substitutionGroup="gml:_MetaData"/>
<!--Located Feature-->
<xs:complexType name="LocatedFeatureType">
  <xs:annotation>
    <xs:documentation>
      A generic located object e.g. on which measurements may be made
    </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:extension base="gml:AbstractFeatureType">
      <xs:sequence>
        <xs:element ref="gml:location"/>
        <xs:element name="descriptionRef" type="xs:anyURI" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<!-- ========================================================================= -->
<!-- Metadata -->
<xs:complexType name="DictionaryPropertyType">
  <xs:annotation>
    <xs:documentation> Use this to provide links to external dictionaries</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
    <xs:restriction base="gml:AbstractMetaDataPropertyType">
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="ows:SkipReference"/>
        <xs:element ref="ows:Definition"/>
      </xs:choice>
      <xs:attribute ref="gml:about" use="optional"/>
    </xs:restriction>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="DefinitionType">
  <xs:annotation>
    <xs:documentation> Use this to provide links to external dictionaries</xs:documentation>
  </xs:annotation>
  <xs:simpleContent>
```

```
    <xs:extension base="xs:string">
      <xs:attribute name="gml.id" type="xs:ID" use="required"/>
    </xs:extension>
  </xs:simpleContent>
 </xs:complexType>
 <xs:complexType name="SkipReferenceType">
  <xs:annotation>
   <xs:documentation> Use this to provide links to values in external dictionaries</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
   <xs:extension base="gml:AbstractMetaDataType">
    <xs:attribute ref="xlink:href"/>
    <xs:attribute name="gml.id" type="xs:ID" use="required"/>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>
</xs:schema>
```

ValueBase Schema

In this schema we derive XML types for five generic Value types and several generic Value collections. Elements declared in this schema can be used directly in data instances, or may be used as components of Measurements.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.opengis.net/gml" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:gml="http://www.opengis.net/gml" xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="0.99">
 <xs:annotation>
  <xs:documentation>
valueBase.xsd

a GML schema for Observed Values in which the
the content model of the base scalar value types are developed by
    • choosing the appropriate type element for the "numerator"
    • adding a reference-system or scale, and an "observable" parameter

SJDC  2002-01-30

</xs:documentation>
 </xs:annotation>
 <!-- ================================================================== -->
 <xs:include schemaLocation="../../base/gmlBase.xsd"/>
 <!-- ================================================================== -->
 <!-- global elements -->
 <!--   <xs:element name="_Value" abstract="true" substitutionGroup="gml:_GML">
this will work only if _GML is an anyType -->
 <xs:element name="_Value" abstract="true">
  <xs:annotation>
   <xs:documentation>
    head Measurement element is abstract -
    this means that potentially either simpleContent or complexContent elements can substitute
   </xs:documentation>
  </xs:annotation>
 </xs:element>
 <!-- Scalar measurements -->
 <xs:element name="_ScalarValue" type="gml:ScalarValueType" abstract="true" substitutionGroup="gml:_Value"/>
 <xs:element name="Category" type="gml:CategoryType" substitutionGroup="gml:_ScalarValue"/>
 <xs:element name="OrderedCategory" type="gml:OrderedCategoryType" substitutionGroup="gml:_ScalarValue"/>
 <xs:element name="Quantity" type="gml:QuantityType" substitutionGroup="gml:_ScalarValue"/>
 <xs:element name="Position" type="gml:PositionType" substitutionGroup="gml:_ScalarValue"/>
 <xs:element name="Count" type="gml:CountType" substitutionGroup="gml:Quantity"/>
 <!-- Compound measurements -->
 <xs:element name="ValueCollection" type="gml:ValueCollectionType" substitutionGroup="gml:_Value"/>
 <xs:element name="CompositeValue" type="gml:CompositeValueType" substitutionGroup="gml:ValueCollection"/>
 <xs:element name="ValueArray" type="gml:ValueArrayType" substitutionGroup="gml:ValueCollection">
  <xs:annotation><xs:documentation>Use ValueArray for arrays of CompositeValues and other ValueCollections.
```

```
_ValueLists are more efficient for arrays of Scalar Values</xs:documentation></xs:annotation>
  </xs:element>
  <xs:element name="_ValueList" abstract="true" substitutionGroup="gml:_Value">
    <xs:annotation>
      <xs:documentation>Compact encoding of arrays of scalar values</xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="CategoryList" type="gml:CategoryListType" substitutionGroup="gml:_ValueList"/>
  <xs:element name="OrderedCategoryList" type="gml:OrderedCategoryListType" substitutionGroup="gml:_ValueList"/>
  <xs:element name="QuantityList" type="gml:QuantityListType" substitutionGroup="gml:_ValueList"/>
  <xs:element name="PositionList" type="gml:PositionListType" substitutionGroup="gml:_ValueList"/>
  <xs:element name="CountList" type="gml:CountListType" substitutionGroup="gml:_ValueList"/>
  <!-- other global elements -->
  <xs:element name="valueMember" type="gml:ValuePropertyType" substitutionGroup="gml:_property"/>
  <xs:element name="valueComponent" type="gml:ValueComponentType" substitutionGroup="gml:valueMember"/>
  <xs:element name="valueArrayMembers" type="gml:ValueArrayMembersType" substitutionGroup="gml:valueMember"/>
  <xs:element name="valueMetaData" type="gml:ValueMetaDataPropertyType"
substitutionGroup="gml:_metaDataProperty"/>
  <!-- -->
  <!-- <xs:element name="name" type="xs:string"/> -->
  <xs:element name="amount" type="xs:double"/>
  <xs:element name="iAmount" type="xs:integer"/>
  <xs:element name="ordinate" type="xs:double"/>
  <xs:element name="null" type="gml:NullType"/>
  <xs:group name="nameOrNull">
    <xs:choice>
      <xs:element ref="gml:name"/>
      <xs:element ref="gml:null"/>
    </xs:choice>
  </xs:group>
  <xs:group name="amountOrNull">
    <xs:choice>
      <xs:element ref="gml:amount"/>
      <xs:element ref="gml:null"/>
    </xs:choice>
  </xs:group>
  <xs:group name="iAmountOrNull">
    <xs:choice>
      <xs:element ref="gml:iAmount"/>
      <xs:element ref="gml:null"/>
    </xs:choice>
  </xs:group>
  <xs:group name="ordinateOrNull">
    <xs:choice>
      <xs:element ref="gml:ordinate"/>
      <xs:element ref="gml:null"/>
    </xs:choice>
  </xs:group>
  <!-- -->
  <xs:element name="dictionary" type="xs:anyURI"/>
  <xs:element name="categoryFrame" type="xs:anyURI"/>
  <xs:element name="unitsOfMeasure" type="xs:anyURI"/>
  <xs:element name="frame" type="xs:anyURI"/>
  <xs:group name="referenceSystem">
    <xs:choice minOccurs="0">
      <xs:element ref="gml:dictionary"/>
      <xs:element ref="gml:categoryFrame"/>
      <xs:element ref="gml:unitsOfMeasure"/>
      <xs:element ref="gml:frame"/>
    </xs:choice>
  </xs:group>
  <!-- types -->
  <xs:complexType name="AbstractValueType" abstract="true">
    <xs:annotation>
      <xs:documentation>top of hierarchy is abstract
      observable binds the value to its observableType
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
```

```
    <xs:element ref="gml:_metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="observable" type="xs:anyURI" use="optional"/>
</xs:complexType>
<!-- scalar measurement types -->
<xs:complexType name="ScalarValueType" abstract="true">
  <xs:complexContent>
    <xs:extension base="gml:AbstractValueType"/>
  </xs:complexContent>
</xs:complexType>
<!-- Category -->
<xs:complexType name="CategoryType">
  <xs:complexContent>
    <xs:extension base="gml:AbstractValueType">
      <xs:sequence>
        <xs:group ref="gml:nameOrNull"/>
        <xs:element ref="gml:dictionary" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="CategoryListType">
  <xs:complexContent>
    <xs:extension base="gml:AbstractValueType">
      <xs:sequence>
        <xs:group ref="gml:nameOrNull" maxOccurs="unbounded"/>
        <xs:element ref="gml:dictionary" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<!-- Ordered Category -->
<xs:complexType name="OrderedCategoryType">
  <xs:complexContent>
    <xs:extension base="gml:AbstractValueType">
      <xs:sequence>
        <xs:group ref="gml:nameOrNull"/>
        <xs:element ref="gml:categoryFrame" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="OrderedCategoryListType">
  <xs:complexContent>
    <xs:extension base="gml:AbstractValueType">
      <xs:sequence>
        <xs:group ref="gml:nameOrNull" maxOccurs="unbounded"/>
        <xs:element ref="gml:categoryFrame" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<!-- Quantity -->
<xs:complexType name="QuantityType">
  <xs:complexContent>
    <xs:extension base="gml:AbstractValueType">
      <xs:sequence>
        <xs:group ref="gml:amountOrNull"/>
        <xs:element ref="gml:unitsOfMeasure" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
<xs:complexType name="QuantityListType">
  <xs:complexContent>
    <xs:extension base="gml:AbstractValueType">
      <xs:sequence>
        <xs:group ref="gml:amountOrNull" maxOccurs="unbounded"/>
        <xs:element ref="gml:unitsOfMeasure" minOccurs="0"/>
```

```
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
 </xs:complexType>
 <!-- Count -->
 <xs:complexType name="CountType">
  <xs:complexContent>
   <xs:restriction base="gml:QuantityType">
    <xs:sequence>
     <xs:element ref="gml:_metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
     <xs:group ref="gml:iAmountOrNull"/>
     <xs:element ref="gml:unitsOfMeasure" fixed="identity" minOccurs="0" maxOccurs="0"/>
    </xs:sequence>
   </xs:restriction>
  </xs:complexContent>
 </xs:complexType>
 <xs:complexType name="CountListType">
  <xs:complexContent>
   <xs:restriction base="gml:QuantityType">
    <xs:sequence>
     <xs:element ref="gml:_metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
     <xs:group ref="gml:iAmountOrNull" maxOccurs="unbounded"/>
     <xs:element ref="gml:unitsOfMeasure" fixed="identity" minOccurs="0" maxOccurs="0"/>
    </xs:sequence>
   </xs:restriction>
  </xs:complexContent>
 </xs:complexType>
 <!-- Position -->
 <xs:complexType name="PositionType">
  <xs:complexContent>
   <xs:extension base="gml:AbstractValueType">
    <xs:sequence>
     <xs:group ref="gml:ordinateOrNull"/>
     <xs:element ref="gml:frame" minOccurs="0"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>
 <xs:complexType name="PositionListType">
  <xs:complexContent>
   <xs:extension base="gml:AbstractValueType">
    <xs:sequence>
     <xs:group ref="gml:ordinateOrNull" maxOccurs="unbounded"/>
     <xs:element ref="gml:frame" minOccurs="0"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>
 <!-- -->
 <xs:complexType name="ValueCollectionType">
  <xs:annotation>
   <xs:documentation>ValueCollection is built up from other Values </xs:documentation>
  </xs:annotation>
  <xs:complexContent>
   <xs:extension base="gml:AbstractValueType">
    <xs:sequence>
     <xs:element ref="gml:valueMember" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
 </xs:complexType>
 <xs:complexType name="CompositeValueType">
  <xs:annotation>
   <xs:documentation>A compositeValue is conceptually a single Value but represented by several components
</xs:documentation>
  </xs:annotation>
  <xs:complexContent>
   <xs:restriction base="gml:ValueCollectionType">
    <xs:sequence>
```

```
        <xs:element ref="gml:_metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="gml:valueComponent" minOccurs="0" maxOccurs="unbounded"/>
       </xs:sequence>
       <xs:attribute name="observable" type="xs:anyURI" use="optional"/>
     </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ValueArrayBaseType" abstract="true">
    <xs:annotation>
      <xs:documentation>ValueBaseArray restricts ValueCollection to contain an array of Values of homogeneous type
First, remove the valueComponent</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:restriction base="gml:ValueCollectionType">
        <xs:sequence>
          <xs:element ref="gml:_metaDataProperty" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="observable" type="xs:anyURI" use="optional"/>
      </xs:restriction>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ValueArrayType" abstract="true">
    <xs:annotation>
      <xs:documentation>
      Add content - either an array of members, or a valueList
      Add the referenceSystem element - may be used if the array members are scalar Values
      Add the members attribute - "types" the array by indicating what its member elements are - also very useful if the
arrayMembers element uses a href to point to values in an external file</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="gml:ValueArrayBaseType">
        <xs:sequence>
          <xs:element ref="gml:valueArrayMembers"/>
          <xs:group ref="gml:referenceSystem"/>
        </xs:sequence>
        <xs:attribute name="members" type="xs:Name" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <!-- compositing -->
  <xs:complexType name="ValuePropertyType">
    <xs:sequence>
      <xs:element ref="gml:_Value"/>
    </xs:sequence>
    <xs:attributeGroup ref="gml:AssociationAttributeGroup"/>
  </xs:complexType>
  <xs:complexType name="ValueComponentType">
    <xs:annotation>
      <xs:documentation>add an "axis" attribute for additional typing of the relationship</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="gml:ValuePropertyType">
        <xs:attribute name="axis" type="xs:string" use="optional"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="ValueArrayMembersType">
    <xs:sequence>
      <xs:element ref="gml:_Value" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="gml:AssociationAttributeGroup"/>
  </xs:complexType>
  <!-- -->
  <xs:complexType name="ValueMetaDataPropertyType">
    <xs:annotation>
      <xs:documentation> Use this to provide a gloss on the measurement(s)</xs:documentation>
    </xs:annotation>
    <xs:complexContent>
      <xs:extension base="gml:AbstractMetaDataPropertyType">
```

```
      <xs:sequence>
        <xs:element name="ValueDescription" type="xs:string" minOccurs="0"/>
      </xs:sequence>
      <xs:attribute ref="gml:about" use="optional"/>
    </xs:extension>
  </xs:complexContent>
 </xs:complexType>
</xs:schema>
```

Annex B
(normative)

Formatting Dates and Times

B.1 Overview

This annex specifies the encoding of moments and periods in time. These allow OGC Sensor Collection Service to support temporal data descriptions and requests. This specification is based on [ISO 8601:1988(E)]; it **limits** ISO 8601 in the following ways:

- Time markers must be fully expressed out to at least the whole second
- Instances in time are represented as one time marker
- Temporal ranges are represented as two (and only two) time markers separated by "/"
- Multiple time instances are separated by commas

B.2 Time Format Details

B.2.1 Basic Syntax

The basic time format uses ISO 8601:1988(E) "extended" format: up to 14 digits specifying century, year, month, day, hour, minute, seconds, and seconds, and optionally a decimal point followed by zero or more digits for fractional seconds, with non-numeric characters to separate each piece:

ccyy-mm-ddThh:mm:ss.sssZ

ISO 8601 states that the precision may be reduced by omitting least-significant digits, for requesting convenience, the SCS TIME parameter requires full expression of the time marker. ISO 8601:1988(E) prefers a decimal comma before fractional seconds but allows a decimal period as in this document.

All times **should** be expressed in Coordinated Universal Time (UTC) as indicated by the suffix Z (for "zulu"); this suffix is **required** when UTC applies if the hours field appears in the time string. When a local time applies, a numeric time zone suffix as defined by clause 5.3.3.1 of [ISO 8601:1988(E)] **may** be used. The absence of any suffix at all means local time in an undefined zone, which **must not** be used in the global network of sensors enabled by the SCS specification.

Request Examples

EXAMPLE 1: Request observations for a single instant in time

2002-03-27T11:03:55Z

EXAMPLE 2: Request observations for the first of each month, March-June

2002-03-01T00:00:00Z,2002-04-01T00:00:00Z,2002-05-01T00:00:00Z

EXAMPLE 3: Request observations for a given time period, March

2002-03-01T00:00:00Z/2002-03-31T11:59:59Z

Bibliography

[OGCWFS] Vretanos, *Open GIS Web Feature Server Specification*, v. 0.0.13, March 10, 2001

[OGCWCS] Evans, *Web Coverage Service IPR*

[OWSSML] Botts, *Sensor Model Language (SensorML) for In-situ and Remote Sensors, IPR,* OGC 02-026.

[OWSOM] Cox, *Observations and Measurements IPR,* OGC 02-027.