Open Geospatial Consortium Inc.

Date: 2010-04-02

Reference number of this document: OGC 09-163r2

Version: 0.3.0

Category: OGC® Discussion Paper

Editor(s): Frédéric Houbie, Fabian Skivée, Simon Jirka

OGC® Catalogue Services Specification 2.0
Extension Package for ebRIM Application Profile :

SensorML

Warning

This document is not an OGC Standard. This document is an OGC Discussion Paper and is therefore not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, an OGC Discussion Paper should not be referenced as required or mandatory technology in procurements.

Document type : OGC® Application Profile
Document subtype : Discussion Paper
Document stage : Draft
Document language : English

Contents

Page

Figures

Page

Tables

Page

Preface

This document describes the mapping of description of sensors using SensorML specification 1.0 [OGC 07-000] to an ebRIM structure within an OGC™ Catalogue 2.0.2 (Corrigendum 2 Release) [OGC 07-006r1] implementing the CSW-ebRIM Registry Service – part 1: ebRIM profile of CSW [OGC 07-110r4].

In addition this document contains the definition of a SensorML profile for Discovery which defines a minimum set of metadata to be provided within SensorML documents as well as the structure this data shall possess. This profile is based on the OGC OWS-6 SensorML Profile for Discovery Engineering Report [OGC 09-033].

It defines the way sensors metadata are organized and implemented in the Catalogue for discovery, retrieval and management.

i. Document terms and definitions

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

ii. Submitting organizations

- **ERDAS**

iii. Document contributor contact points

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

Name	Organization	Email
Frédéric Houbie (Editor)	ERDAS	frederic.houbie<at>erdas.com
Fabian Skivée (Editor)	ERDAS	fabian.skivee<at>erdas.com
Alexandre Robin	SPOT	alexandre.robin<at>spotimage.fr
Simon Jirka	University of Münster – IfGI	jirka<at>uni-muenster.de
Arne Bröring	52° North	broering<at>52north.org
Daniel Nüst	University of Münster – IfGI	daniel.nuest<at>uni-muenster.de

iv. Revision history

Please see the Annex H for a complete Document Change History.

Date	Release	Editor	Sections modified	Description
18/09/2009	0.0.1	F. Houbie, F. Skivée	Initial version	

9/11/2009	0.0.2	S. Jirka	Added Discovery profile	
10/02/2010	0.0.3	F. Houbie	Update slot table and align EO Profile	

v. Changes to the OGC Abstract Specification

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

vi. Foreword

This document has been created under the impulsion of European Space Agency in the scope of the Heterogeneous Missions Accessibility project. Furthermore it is based on work performed within the EU funded projects OSIRIS[1] and GENESIS[2]. It includes also results developed under the Sensor Web Enablement thread during the OGC Web Services Phase 6

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium Inc. shall not be held responsible for identifying any or all such patent rights. Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

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

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

Introduction

This Discussion Paper describes the mapping of sensor metadata encoded following SensorML specification 1.0 [OGC 07-000] and a SensorML profile for discovery to an ebRIM structure within an OGC Catalogue 2.0.2 (Corrigendum 2 Release) [OGC 07-006r1] implementing the CSW-ebRIM Registry Service – part 1: ebRIM profile of CSW [OGC 07-110r4].

Using the mapping described within this document it becomes possible to make sensors as well as SWE services discoverable through an OGC Catalogue and thus achieving a better integration of sensors and sensor data into spatial data infrasturctures.

Due to the fact that the SensorML data model specifies a majority of its elements as optional and as it allows expressing the same information in several, differently structured ways, this document contains also a SensorML profile for discovery. This approach ensures that a minimum set of metadata is provided for every sensor in a common structure so that on the one hand automatic harvesting becomes possible and that on the other hand all necessary metadata is present. Based on the defined minimum set of metadata the mapping described in this document allows to search sensors based on spatial (area for which sensor data is needed), temporal (point in time for which sensor data is needed) and thematic criteria (phenomena for which sensor data is needed).

1 Scope

This OGC™ document specifies the SensorML Extension Package for ebRIM (ISO/TS 15000-3) Application Profile of CSW 2.0. It relies on a SensorML profile for discovery which defines a minimum set of metadata including its structure.

It enables CSW-ebRIM catalogues to handle a variety of sensor metadata as well as SWE services.

This proposed application profile document describes the interfaces, bindings and encodings required to discover, search and present metadata from sensor and SWE catalogues. The profile presents a minimum specification for catalogue interoperability within the SWE context.

2 Normative references

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

[ebRIM30] *OASIS ebXML Registry Information Model Version 3.0*

[ISO 19115:2003] *Geographic information : Metadata*

[OGC 04-095] *OGC™ Filter Encoding Implementation Specification, version 1.1.0, 2005/05/03*

[OGC 05-008] *OGC™ Web Services Common Specification Corrigendum, version 1.0.0, 2005/11/22*

[OGC 07-110r4] OGC™ *CSW-ebRIM Registry Service - Part 1: ebRIM profile of CSW, version 1.0.1, 2009/02/05.*

[OGC 07-144r4] OGC™ *CSW-ebRIM Registry Service – Part 2 : Basic extension package, version 1.0.1, 2009/02/05*

[OGC 08-103r2] OGC™ *CSW-ebRIM Registry Service – Part 3: Abstract Test Suite (1.0.1), version 1.0.1, 2009/02/05*

[OGC 06-080r4] *OGC™ GML 3.1.1 Application Schema for Earth Observation Products, version 0.9.3, 2008/07/21*

[OGC 07-006r1] *OGC™ Catalogue Services Specification, version 2.0.2 (Corrigendum 2 Release), 2007/02/23*

[OGC 07-092] *OGC™ Definition identifier URNs in OGC namespace, version 1.3.0, 2006/01/15.*

[OGC 07-000] *OGC™ Sensor Model Language (SensorML) Standard*, Version 1.0

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

[OGC 09-033] *OGC™ OWS-6 SensorML Profile for Discovery Public Engineering Report.*

[OGC] *Sensor Observable Registry Discussion Paper.*

In addition to this document, this specification includes several normative XML Schema Document files as specified in Annex B.

3 Terms and definitions

For the purposes of this specification, the definitions specified in Clause 4 of the [OGC 05-008] OGC™ OWS Common Implementation Specification shall apply. In addition, the following terms and definitions apply.

4.1
application profile
see the term “profile” in this list

4.2
client
software component that can invoke an **operation** from a **server**

4.3
data level
stratum within a set of layered levels in which data is recorded that conforms to definitions of types found at the application model level [ISO 19101]

4.4
dataset series (dataset collection[3])
collection of datasets sharing the same product specification [ISO 19113, ISO 19114, ISO 19115]. In the earth observation context, a collection typically corresponds to datasets (i.e. products) derived from data acquired by a single sensor onboard a satellite or series of satellites.

4.5
datastrip
a satellite acquisition

4.6
geographic dataset
dataset with a spatial aspect [ISO 19115]

4.7
geographic information
information concerning phenomena implicitly or explicitly associated with a location relative to the Earth [ISO 19128 draft]

4.8
georesource
geographic information of a specific type (e.g. geographic dataset, geographic application, geographic service)

4.9
identifier
linguistically independent sequence of characters capable of uniquely and permanently identifying that with which it is associated.

4.10
interface
named set of operations that characterize the behavior of an entity [ISO 19119]

[3] Due to historical reasons we will mainly use the term 'dataset collection' in this document, although the term 'dataset series' is used in the ISO/TC211 Terminology Maintenance Group.

4.11
metadata entity
set of metadata elements describing the same aspect of data

NOTE 1 May contain one or more metadata entities

NOTE 2 Equivalent to a class in UML terminology [ISO 19115]

4.12
metadata schema
conceptual schema describing metadata

NOTE ISO 19115 describes a standard for a metadata schema. [ISO 19101]

4.13
metadata section
subset of metadata that defines a collection of related metadata entities and elements [ISO 19115]

4.14
operation
specification of a transformation or query that an object may be called to execute [ISO 19119]

4.15
parameter
variable whose name and value are included in an operation **request** or **response** [ISO 19119]

4.16
profile
set of one or more base standards and – where applicable – the identification of chosen clauses, classes, subsets, options and parameters of those base standards that are necessary for accomplishing a particular function [ISO 19101, ISO 19106]

NOTE The terms “profile” and “application profile” are used interchangeably in this document

4.17
qualified name
name that is prefixed with its naming context

4.18
request
invocation of an **operation** by a **client**

4.19
response
result of an **operation,** returned from a **server** to a **client**

4.20
schema
formal description of a model [ISO 19101, ISO 19103, ISO 19109, ISO 19118]

4.21
server
service instance
a particular instance of a **service** [ISO 19128]

4.22
service
distinct part of the functionality that is provided by an entity through interfaces [ISO 19119]

4.23
service interface
shared boundary between an automated system or human being and another automated system or human being [ISO 19101]

4.24
service metadata
metadata describing the **operations** and **geographic information** available at a **server** [ISO 19128]

4.25
state
(of an object) persistent data object reflecting the internal values of all the member attributes or measurable descriptions of an object at a given time

NOTE State is usually associated to an object by its identity and to a time by a timestamp. [ISO 19132]

4.26
transfer protocol
common set of rules for defining interactions between distributed systems [ISO 19118]

4.27
version
version of an Implementation Specification (document) and XML Schemas to which the requested operation conforms

NOTE An OWS Implementation Specification version may specify XML Schemas against which an XML encoded operation request or response should be validated.

4.28
SensorML profile
a restriction of the SensorML schema

4.29
sensor discovery
the process of searching for sensors or SWE services that encapsulate them

4 Conventions

4.1 Abbreviated terms

Some more frequently used abbreviated terms:

API	Application Program Interface
ATM	Atmospheric Type (Namespace)
BPEL	Business Process Execution Language
COTS	Commercial Off The Shelf
CQL	Common Query Language
CRS	Coordinate Reference System
CSW	Catalogue Service for Web
DC	Dublin Core
DCMI	Dublin Core Metadata Initiative
DCP	Distributed Computing Platform
EBRIM	ebXML Registry Information Model
EO	Earth Observation
EOP	Basic Earth Observation Product Type (Namespace)
EU	European Union
GENESIS	GENeric European Single Information Space for environment
GML	Geography Markup Language
HMA	Heterogeneous Missions Accessibility
HTTP	HyperText Transfer Protocol
ISO	International Organization for Standardization
OGC	Open Geospatial Consortium
OPT	Optical Type (Namespace)
OSIRIS	Open architecture for Smart and Interoperable networks in Risk management based on In-situ Sensors
OWS	OGC Open Web Services
O&M	Observations and Measurements
PHR	Pleiades High Resolution Type (Namespace)
SAR	Synthetic Aperture Radar Type (Namespace)
SAS	Sensor Alert Service
SensorML	Sensor Model Language

SOAP	Simple Object Access Protocol
SOS	Sensor Observation Service
SPS	Sensor Planning Service
SQL	Structured Query Language
SWE	Sensor Web Enablement
TML	Transducer Markup Language
UCUM	Unified Code for Units of Measure
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UTF-8	Unicode Transformation Format-8
WSDL	Web Service Definition Language
W3C	World Wide Web Consortium
WNS	Web Notification Service
XML	eXtensible Markup Language

4.2 UML notation

Some of the diagrams in this document are presented using the Unified Modeling Language (UML) static structure diagram. The UML notations used in this document are described in Figure 1, below.

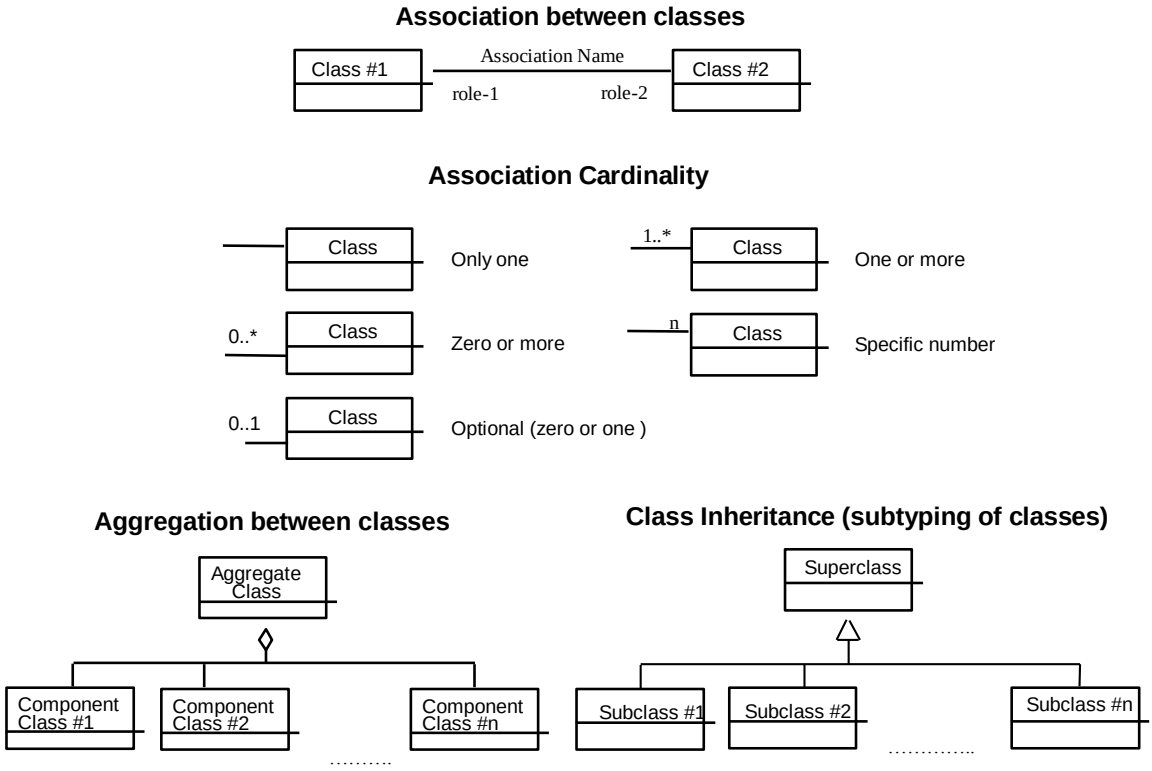


Figure 1 – UML notations

In these UML class diagrams, the class boxes with a light background are the primary classes being shown in this diagram, often the classes from one UML package. The class boxes with a gray background are other classes used by these primary classes, usually classes from other packages.

4.3 Used parts of other documents

This document uses significant parts of OGC™ GML Application Schema for EO Products document [OGC 06-080r4]. To reduce the need to refer to that document, this document copies some of those parts.

5 System context

This section focuses on the purpose, scope and policies of catalogue services that comply with the given profile. It documents special requirements and describes the context of use.

5.1 Application domain

The catalogue services proposed in this specification are intended to support the retrieval of EO products metadata possibly in two steps.

Step 1: identification of collections of interest through a catalogue of collections metadata.

Step 2: identification of EO products of interest within one or several collections through a catalogue of EO products metadata.

This specification covers step 2. For the retrieval of EO products metadata, the defined interface should allow to implement both generic clients not aware of sensor specific metadata and more specialized clients aware of sensors specific metadata.

5.2 Essential use cases

The use cases for this specification are the ebRIM profile of CSW use cases applied to Earth Observation. See [OGC 07-110r4].

6 Catalogue Infrastructure Overview

Catalogues are intended to store metadata describing resources published by providers and allow clients to find these resources. These resources metadata are organized in Catalogues according to specific data models, based on the registry information model.

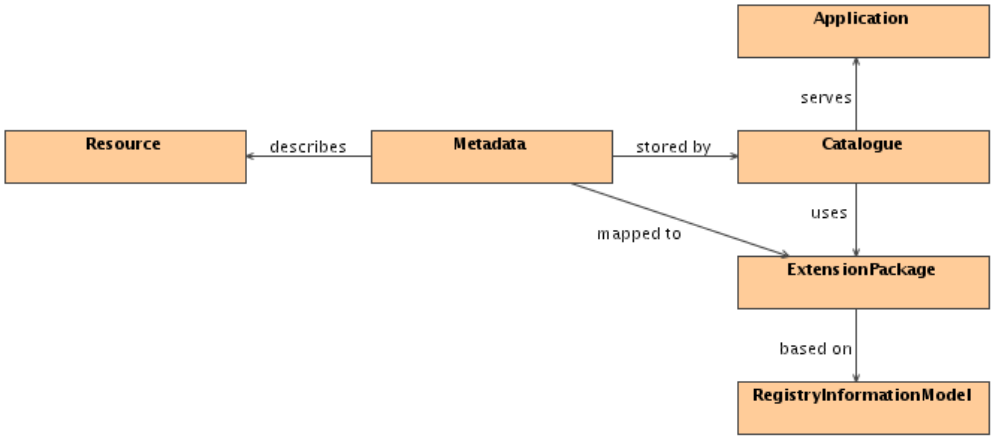


Figure 2 – General Catalogue Infrastructure

This schema clearly describes the following:

- *Resources* are described using *Metadata*,
- *Metadata* are stored in *Catalogues*, according to a *ExtensionPackage* defining accurately the mapping of such type of resources (*ExtensionPackage* is resource-specific),
- *ExtensionPackage* is based on a generic model, called the *RegistryInformationModel* (aka *RIM*). The *RegistryInformationModel* is common to all resources within a catalogue,
- *Applications* use *Catalogues* to discover resources through their metadata.

In the present context:

- *Resources* are *sensors*,
- *Metadata* (describing *sensors*) are typically encoded into sensorML documents in conformance with [OGC 07-000],
- *Metadata* are stored in *Catalogues* according to the *sensorML ExtensionPackage*,
- The sensorML *ExtensionPackage* is based on the *ebRIM* (a *RegistryInformationModel* used in an *Application Profile* of CSW 2.0 compliant catalogues),

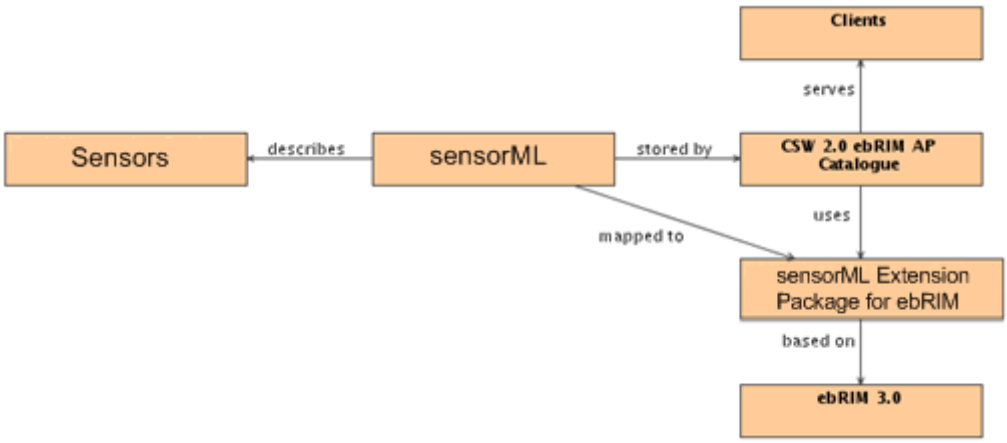


Figure 3 – Sensors Oriented Catalogue Infrastructure

Specifications define:

- The way to interact with Catalogues through operations on the service (in this case a Web Service) to publish and query data,
- The way to use a registry information model (in this case ebRIM) to allow mapping of resources in Catalogues,
- The way to map each kind of metadata to the RIM. A specific kind of resource (e.g., an sensor description) shall be modeled using the same set of ebRIM objects in all Catalogues. Indeed, discover queries are strongly dependant on the chosen mapping. Even if this strong dependence can be a drawback for client implementation, flexibility can be enhanced and complexity reduced from the client's point of view using mechanisms like stored queries (see section 16 of [OGC 07-110r4]). A single and common (standardized) mapping ensures interoperability between Catalogue implementations.

This document defines the extension package for Earth Observation Products.

7 Package overview

7.1 Purpose

This section defines the ebRIM model that represents the sensors. The objective is not to store all the SensorML information but only the attributes useful for discoverying sensors.

Due to the variety of sensors with business-specific properties and structures, it is difficult to define a generic model to store all types of sensors in the same catalog. On the other hand, a generic client must be able to interact with any kind of sensor catalog and to present the common attributes.

To encompass this difficulty, this extension package defines a generic model able to store any kind of sensors and allows extending this generic model into business-oriented profiles for domains like meteorology, earth observation, etc.

The generic ebRIM model for SensorML is depicted in Figure 4.

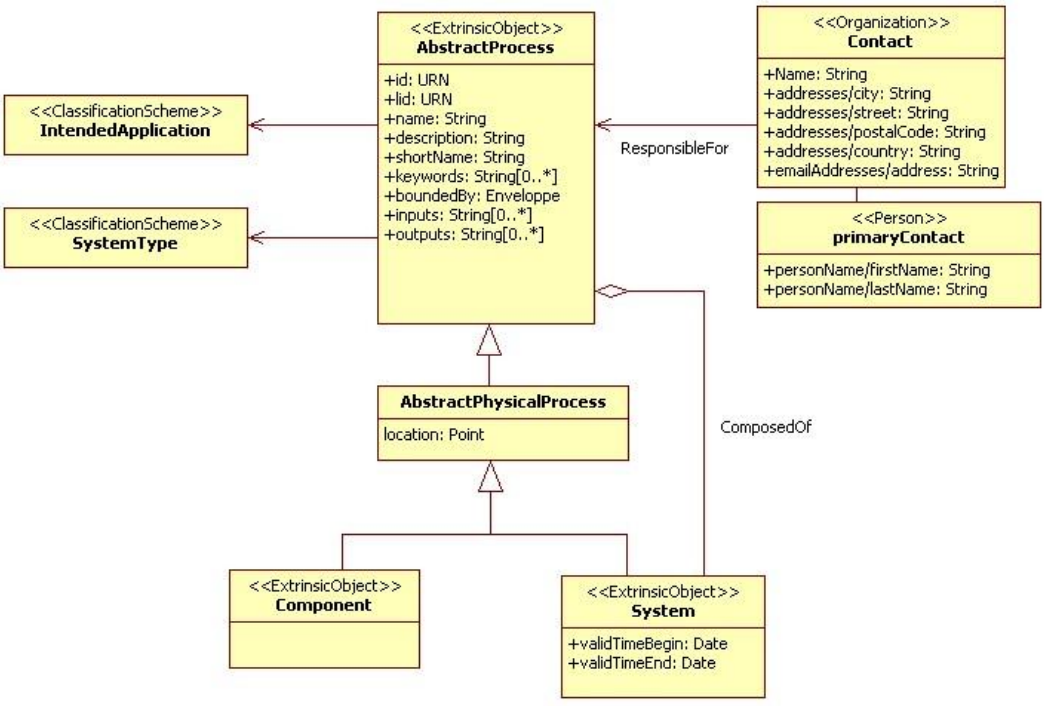


Figure 4 – Generic SensorML ebRIM model

The model contains two main objects “Component” and “System”. For the clarity of the reading, the common properties are stored in two abstract parents “AbstractProcess” and “AbstractPhysicalProcess”.

In a first iteration, only the physical processes are taken into account. In a future iteration, the non-physical process will be analysed and will probably be modeled as an "AbstractNonPhysicalProcess" object that inherits from "AbstractProcess".

A system is composed of a series of AbstractProcess i.e. "Component or System". The contact information of the SensorML is associated to an AbstractProcess.

In the SensorML, two classifications schemes are informatively proposed:

- IntendedApplication: the application for which the sensor could be used.
- SystemType: the sensor type like "Wave Measuring", "Satelitte" …

In the ebRIM model, the AbstractProcess can be classified regarding these two ClassificationSchemes.

In SensorML, some information like Inputs and Outputs contains the unit of measure in which the value is expressed. In a catalog context, allowing to express value in different units is not appropriate since it imposes units management on the client side and increase the computational time to process a query (due to unit mapping). To avoid storing measurement units in the catalog, each slot defines a unique unit of measure (respecting International System of Units (SI)).

The Figure 5 shows how a Wave Rider sensor is represented in the sensor generic ebRIM model.

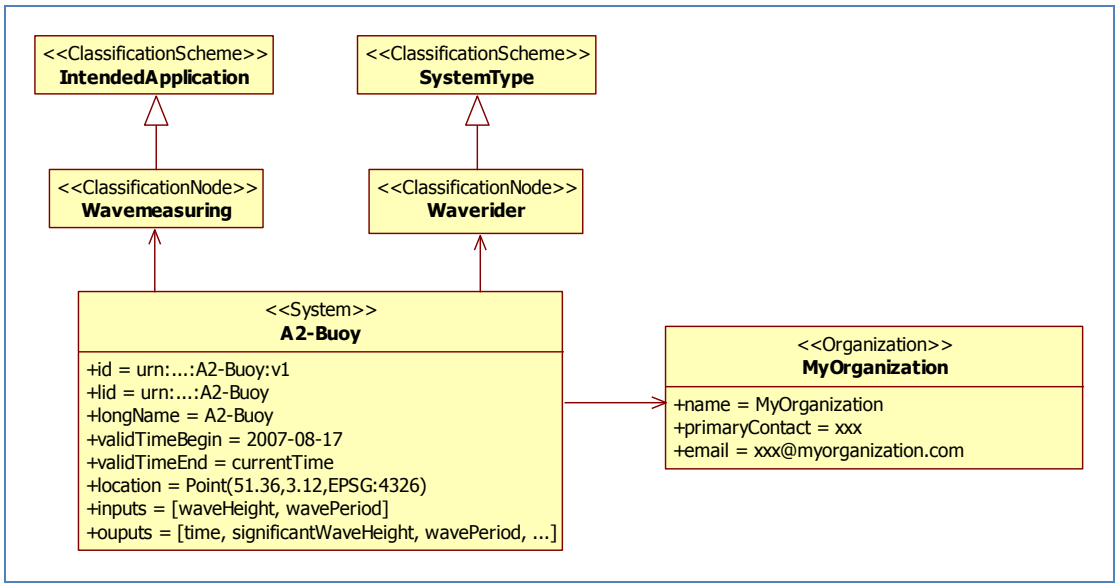


Figure 5 – Wave rider sensor sample

7.2 Dependencies

This Extension Package depends on:

- [OGC07-110r4] CSW-ebRIM Registry Service - Part 1: ebRIM profile of CSW. Version: 1.0.1 (Corrigendum 1), 05 February 2009.
- [OGC07-144r4] CSW-ebRIM Registry Service – Part 2: Basic extension package. Version: 1.0.1 (Corrigendum 1), 05 February 2009.
- [OASIS regrep-rim-3.0-cd-02] ebXML Registry Information Model Version 3.0. Committee Draft Specification 02, 15 March 2005.

The following canonical ebRIM 3.0 classification schemes are required by this package:

- ObjectType Classification Scheme
- AssociationType Classification Scheme
- DataType Classification Scheme

8 Classification schemes

8.1 Intended application

The Inteded Application classification scheme defines the applications for which a sensor could be of interest. A specific System or Component instance shall be classified using one or several node(s) in this scheme. The properties of the scheme are summarized in the table below.

Table 1 – Classification scheme: IntendedApplication

Property	Value
Identifier	urn:ogc:def: classificationScheme:OGC-CSW-ebRIM-Sensor::IntendedApplication
Name	Intended application taxonomy
Description	Defines a taxonomy that may be used to classify sensors according to their purpose.
Node type	urn:oasis:names:tc:ebxml-regrep:NodeType:UniqueCode
Internal nodes	true

8.2 System types

The System Types classification scheme defines the types of the sensors. A specific System or Component instance shall be classified using one or several node(s) in this scheme. The properties of the scheme are summarized in the table below.

Table 2 – Classification scheme: SystemTypes

Property	Value
Identifier	urn:ogc:def: classificationScheme:OGC-CSW-ebRIM-Sensor::SystemTypes
Name	System types taxonomy
Description	Defines a taxonomy that may be used to classify sensors according to their types.
Node type	urn:oasis:names:tc:ebxml-regrep:NodeType:UniqueCode
Internal nodes	true

9 Classification nodes

9.1 Object types

9.1.1 System

The SensorML specification (OGC07-000) defines a System as *"System is a physical equivalent of a ProcessChain. A System may include several physical and non-physical processes that all act to provide a certain set of System outputs, based on the System inputs and parameters"*.

A System item denotes an extrinsic object that describes a System as defined in OGC07-000. A System inherits all the parameters from the abstract objects AbstractProcess and AbstractPhysicalProcess defined in Section 7.1. The properties of the classification node are summarized in the table below.

Table 3 – Object type: System

Property	Value
Identifier	urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System
Name	System
Description	Describes a System.
Parent	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject:ExtrinsicObject
Code	System

9.1.2 Component

The SensorML specification (OGC07-000) defines a Component as *"Any physical process can be modeled as a Component in SensorML if it either cannot be subdivided into smaller subprocesses or if one chooses to treat it as a single indivisible process. A Component can be considered as a real-world equivalent of a ProcessModel. A Component can participate as part of a ProcessChain or System"*.

A Component item denotes an extrinsic object that describes a Component as defined in OGC07-000. A Component inherits all the parameters from the abstract objects AbstractProcess and AbstractPhysicalProcess defined in Section 7.1. The properties of the classification node are summarized in the table below.

Table 4 – Object type: Component

Property	Value
Identifier	urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component
Name	Component
Description	Describes a Component.
Parent	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject:ExtrinsicObject
Code	Component

9.2 Association types

9.2.1 ComposedOf

The "ComposedOf" association relates a System with a sub-system that could be a System item or a Component item. For an association of this type, the source and

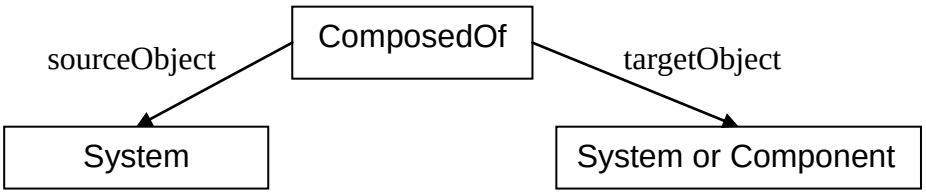


Figure 6 – Object type constraints for the 'ComposedOf' association

target objects shall be of the types indicated in Figure 6.

Example: The "ComposedOf" association.

```
<rim:Association
  id="urn:uuid:ba8348a0-93b2-11dc-893e-0002a5d5c51b"
  sourceObject="urn:uuid:aa779e20-93b2-11dc-a6e1-0002a5d5c51b"
  targetObject="urn:uuid:7f2a1b80-93b2-11dc-bee7-0002a5d5c51b"
  associationType="urn:ogc:def:associationType:OGC-CSW-ebRIM-
Sensor::ComposedOf" />

<wrs:ExtrinsicObject
  id="urn:uuid:aa779e20-93b2-11dc-a6e1-0002a5d5c51b"
  objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-
Sensor::System" />

<wrs:ExtrinsicObject
  id="urn:uuid:7f2a1b80-93b2-11dc-bee7-0002a5d5c51b"
```

```
  objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-
Sensor::Component" />
```

The properties of the classification node are summarized in the table below.

Table 5 – Association type: ComposedOf

Property	Value
Identifier	urn:ogc:def:associationType:OGC-CSW-ebRIM-Sensor::ComposedOf
Name	ComposentOf
Description	Links a System to its sub-systems (System or Component).
Parent	urn:oasis:names:tc:ebxml-regrep:classificationScheme:AssociationType
Code	ComposedOf

9.2.2 AccessibleThrough

The "AccessibleThrough" association relates a System or Component with a Service via which it can be accessed. For an association of this type, the source and target objects shall be of the types indicated in Figure 6.

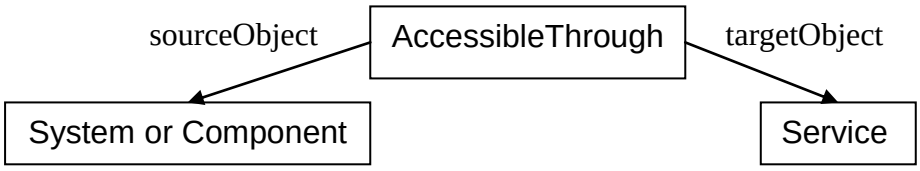


Figure 7 – Object type constraints for the 'ComposedOf' association

Example: The "AccessibleThrough" association.

```
<rim:Association
  id="urn:uuid:ba8348a0-93b2-11dc-893e-0002a5d5c51b"
  sourceObject="urn:uuid:aa779e20-93b2-11dc-a6e1-0002a5d5c51f"
  targetObject="urn:uuid:7f2a1b80-93b2-11dc-bee7-0002a5d5c51g"
  associationType="urn:ogc:def:associationType:OGC-CSW-ebRIM-
Sensor::AccessibleThrough" />

<wrs:ExtrinsicObject
  id="urn:uuid:aa779e20-93b2-11dc-a6e1-0002a5d5c51f"
  objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-
Sensor::System" />

<rim:Service id=" urn:uuid:7f2a1b80-93b2-11dc-bee7-
0002a5d5c51g " />
```

The properties of the classification node are summarized in the table below.

Table 6 – Association type: AccessibleThrough

Property	Value
Identifier	urn:ogc:def:associationType:OGC-CSW-ebRIM-Sensor::AccessibleThrough
Name	AccessibleThrough
Description	Links a System or Component to the providing Service.
Parent	urn:oasis:names:tc:ebxml-regrep:classificationScheme:AssociationType
Code	ComposedOf

10 Slots

The SensorML package includes the slots defined in the following tables. The value "–" appearing in the tables below indicates the absence of a more specific constraint.

Table 7 – Slot: ShortName

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ShortName
Definition	The short name of the sensor
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:String
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 8 – Slot: Keywords

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Keywords
Definition	The list of keywords of the sensor
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:String
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 9 – Slot: ObservedBoundingBox

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::BoundedBy
Definition	Defines the area that is observed by the sensor. In case of in-situ sensors this bounding box contains only one point.
Source	–
Slot type	urn:ogc:def:dataType:ISO-19107:2003:GM_Envelope
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 10 – Slot: Inputs

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Inputs
Definition	The list of inputs of the sensor
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:String
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 11 – Slot: Outputs

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Outputs
Definition	The list of the outputs of the sensor expressed in the International System of Units (SI)
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:String
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 12 – Slot: Location

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Location
Definition	The location of the sensor
Source	–
Slot type	urn:ogc:def:dataType:ISO-19107:2003:GM_Point
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 13 – Slot: ValidTimeBegin

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ValidTimeBegin
Definition	The date when the sensor information starts to be valid
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:DateTime
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 14 – Slot: ValidTimeEnd

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ValidTimeEnd
Definition	The date when the sensor information ends to be valid
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:DateTime
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 15 – Slot: ServiceSpecificSensorID

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ServiceSpecificSensorID
Definition	The id of a sensor within the providing service
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:String
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

11 Predefined queries

This section defines the Adhoc query generic to all kind of sensors.

11.1 findSensors

Invoking the `findSensors` query returns a listing of all sensors matching the provided parameters.

The *systemType* parameter may be included to request a subset of sensors classified using the System Types classification scheme (see Clause 8.2).

The *name* parameter may be included to request a subset of sensors which match the provided name, shortName or longName.

The *location* parameter may be included to request a subset of the sensors that lie within the provided envelope.

The *observedBBox* parameter may be included to request a subset of the sensors that provide observation data for the provided bounding box.

The *intendedApplication* parameter may be included to request a subset of sensors classified using the IntendedApplication classification scheme.

The *inputs* parameter may be included to request a subset of sensors that contain at least one of the provided inputs.

The *outputs* parameter may be included to request a subset of sensors that contain at least one of the provided outputs.

The *keywords* parameter may be included to request a subset of sensors that containsat least one of the provided keywords.

The *validity* parameter may be included to request a subset of sensors valid at the provided date.

Table 16 – Predefined query: findSensors

Identifier	urn:ogc:def:query:OGC-CSW-ebRIM-Sensor::findSensors
Name	findSensors
Parameters	*systemType [String[]]* (optional) One or several id(s) of system type from the SystemTypes classification scheme. *name [String]* (optional) The name of the sensor *location [Envelope]* (optional) An envelope in which the sensor must be located *intendedApplication [String[]]* (optional) One or several id(s) of application from the IntendedApplication classification scheme. *inputs [String[]]* (optional) One or several input(s) of the sensor *outputs (String[])* (optional) One or several output(s) of the sensor *keywords (String[])* (optional) One or several keyword(s) of the sensor *validity [Date]* (optional) The date when the Sensor must be valid.
Response	A <csw:GetRecordsResponse> element containing <rim:RegistryPackage> items that match the provided parameters.

12 SensorML profile for discovery

This section explains a SensorML profile for sensor and SWE service discovery. In order to allow a reliable mapping between SensorML documents and an ebRIM data model it is necessary to ensure that a minimum set of sensor metadata is provided in a common way. This is achieved through this profile.

This profile is based on a SensorML example that formally describes a home weather station which is able to measure ambient temperature, wind speed, wind direction and rain fall (in order to reduce the complexity of this document only the thermometer, wind sensors and the rain gauge are described). The presented SensorML instance relies strongly on the example of a weather station published by ROBIN (2006)[4].

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

The profile is intended to be primarily applied to SensorML documents returned by the DescribeSensor operation of a Sensor Observation Service (or by operations that are provided by other types of SWE services). It is assumed that the request of a single sensor description returns the entire model of a system to which it affiliates. Since the entire system model incorporates the associated sensor descriptions, this approach enables harvesting engines to access the information of the aggregating system as well as its associated components.

To define a formal SensorML profile the Schematron (JELIFFE 2005)[5] language is applied. This language is highly qualified for this task because it allows the formulation of separate rules which restrict an existing schema. The original XML schema does not have to be modified. Instead the profile can be applied to the schema.

The weather station used in the example is modeled as a `System` which allows a very accurate description. It can be considered as a platform for several detectors. These detector components are aggregated by the station. In the present example the `System` logically incorporates the following sensors:

---

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

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

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

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

12.1 System and Component Description

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

12.1.1 gml:description

In addition to the SensorML specific sections a `gml:description` shall be provided for each `System` and `Component`. Listing 12-1 shows an example for this.

Listing 12-1: Example of a gml:description

```
<gml:description>Weather station located on the roof of the Insititute for
          Geoinformatics of the University Münster,
Germany.</gml:description>
```

12.1.1.1 Schematron Rules

The following rules ensure that a `gml:description` element is present within a `System` or `Component`.

Listing 12-2: Schematron rules for ensuring that a gml:description is provided

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

12.1.2 Keywords

The `keywords` section provides a list of keywords which may be used by a user to search for a sensor. This profile requires that at least the observed phenomenon shall be mentioned here. Listing 12-3 shows an excerpt of this section within the SensorML

document of the exemplary home weather station. The `keywords` section shall be provided by any `System` and `Component`.

Listing 12-3: Example of a keywords section

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

12.1.2.1 Schematron Rules

In order to make sure that a list of keywords describing the sensor is provided the Schematron rule shown in Listing 12-4 has been defined.

Listing 12-4: Schematron rules for ensuring that a KeywordList is provided

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

12.1.3 Identification

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

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

In addition for a component the following element shall be present:

- One identifier shall carry the definition "urn:ogc:def:identifier:OGC:1.0:parentSystemUniqueID". The value of its contained "Term" element uniquely identifies the System to which the Component belongs.

Listing 12-5 shows an excerpt of this section within the SensorML document of the temperature sensor (Component) of the exemplary home weather station (System).

Listing 12-5: Example of an identification section

```
<identification>
  <IdentifierList>
    <identifier name="uniqueID">
      <Term definition="urn:ogc:def:identifier:OGC:1.0:uniqueID">
        <value>urn:ogc:object:feature:Sensor:IFGI:thermometer123</value>
      </Term>
    </identifier>
    <identifier name="longName">
      <Term definition="urn:ogc:def:identifier:OGC:1.0:longName">
        <value>OSIRIS Thermometer at weather station 123</value>
      </Term>
    </identifier>
    <identifier name="shortName">
      <Term definition="urn:ogc:def:identifier:OGC:1.0:shortName">
        <value>OSIRIS Thermometer 123</value>
      </Term>
    </identifier>
    <identifier name="parentSystemUniqueID">
      <Term
definition="urn:ogc:def:identifier:OGC:1.0:parentSystemUniqueID">
        <value>OSIRIS Weather Station 123</value>
      </Term>
    </identifier>
  </IdentifierList>
</identification>
```

12.1.3.1 Schematron Rules

The rule defined in Listing Listing 12-6 restricts the possible structure of the identification section. Each listed identifier shall contain the definition attribute. The value of this attribute serves as a link to the semantic description of the identifier.

Listing 12-6: Schematron rule for ensuring that every identifier contains a definition

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

To be able to identify a System or Component uniquely the rule in Listing 12-7 is defined. One identifier of the identification section shall declare a definition attribute with the value urn:ogc:def:identifier:OGC:1.0:uniqueID. Then the value of the identifier's Term element uniquely identifies the instance.

Listing 12-7: Schematron rule for ensuring that a uniqueID is provided

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

Furthermore each sensor shall possess a long and a short name (e.g. weather station 123). This is ensured by the rule shown in Listing 12-8.

Listing 12-8: Schematron rules for ensuring that a longName and a shortName are provided

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

Finally, within every `Component` one identifier referring to the uniqueID of the parent system shall be present:

Listing 12-9: Schematron rule for ensuring that the parentSystemUniqueID is provided for every Component

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

12.1.4 Classification

The classification section gives further information regarding the type of the `System` or `Component`. This information shall include the intended application and the sensor type (see Listing 12-10).

Listing 12-10: Example of a classification section

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

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

12.1.4.1 Schematron Rules

The rule defined in Listing 12-11 restricts the possible structure of the `classification` section. Each listed classifier shall contain the `definition` attribute. The value of this attribute serves as a link to the semantic description of the identifier.

Listing 12-11: Schematron rule for ensuring that every classification element contains a definition

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

Besides the previous rule every `System` and `Component` description shall fulfill the Schematron rule specified in Listing 12-12. It defines that a `System` or `Component` element shall contain at least one classifier with the definition "`urn:ogc:def:classifier:OGC:1.0:sensorType`". The value of its contained `Term` element states the type of the sensor. In addition `System` or `Component` element shall contain at least one classifier with the definition "`urn:ogc:def:classifier:OGC:1.0:intendedApplication`".

Listing 12-12: Schematron rules for ensuring that for every System or Component the sensor type as well as its intended application is described

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

12.1.5 ValidTime

The `validTime` section provides information about the time period in which a sensor description is valid. Each `System` and `Component` shall contain this information. As the SensorML schema defines this can be either a `gml:TimePeriod` or a `gml:TimeInstant`. Within this profile this is restricted to a `gml:TimePeriod` (see Listing 12-13).

Listing 12-13: Example of a validTime section

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

12.1.5.1 Schematron Rules

Each `System` and `Component` description shall contain information about the time it is valid. As the SensorML schema defines this can be either a `gml:TimePeriod` or a `gml:TimeInstant`. In this case a restriction is made that a `gml:TimePeriod` is required.

Listing 12-14: Schematron rules for ensuring that for every System and Component a validTime element is provided

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

12.1.6 Capabilities

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

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

An example of this section is presented in Listing 12-15.

Listing 12-15: Example of a capabilities section

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

12.1.6.1 Schematron Rules

Listing 12-16 restricts the structure of the `capabilities` section. To ease the usage of the `capabilities` element it is defined that a `swe:DataRecord` containing a number of `swe:field` elements shall be used to specify the capabilities of a `System` or `Component`. The child element of each `swe:Field` element shall contain a `definition` attribute so that a client is able to look up the semantics of the capability. The next rule (Listing 12-17) defines that if the child-element of the

swe:Field is a swe:Quantity it shall contain the swe:uom element which specifies the code attribute.

Listing 12-16: Schematron rule for restricting the structure of the capabilities section

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

Listing 12-17: Schematron rule for ensuring that the unit of measurement is provided if in the capabilities section a swe:Quantity is used

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

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

Listing 12-18: Schematron rules for ensuring that an observedBBOX is provided

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

12.1.7 Contact

The following contact section references the organization or person which is responsible for the sensor. This information shall be provided for every System and Component. In general this is the provider of the sensor (see Listing 12-19).

Listing 12-19: Example of a contact section

```
<contact>
  <ResponsibleParty gml:id="WWU_IfGI_thermometer_contact">
    <individualName>Simon Jirka</individualName>
    <organizationName>Institute for Geoinformatics - Westfälische Wilhelms-
         Universität Münster - Sensor Web and Simulation
Lab</organizationName>
```

```
    <contactInfo>
      <address>
        <deliveryPoint>Weseler Straße 253</deliveryPoint>
        <city>Münster</city>
        <postalCode>48151</postalCode>
        <country>Germany</country>
        <electronicMailAddress>swsl-ifgi@listserv.uni-
                               muenster.de</electronicMailAddress>
      </address>
    </contactInfo>
  </ResponsibleParty>
</contact>
```

12.1.7.1 Schematron Rules

A "contact" element shall be present for each System which contains at least the name of the organization operating the sensor.

Listing 12-20: Schematron rules for ensuring that for every System and Component a contact element is provided which contains at least a organisationName element

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

12.1.8 Position

The position section specifies the sensor's position in space. It shall be given for every System and Component. Listing 12-21 presents an exemplary definition of a three dimensional position.

Listing 12-21: Example of a position section

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

```
                                <swe:uom code="m"/>
                                <swe:value>297.0</swe:value>
                            </swe:Quantity>
                        </swe:coordinate>
                    </swe:Vector>
                </swe:location>
            </swe:Position>
        </position>
```

12.1.8.1 Schematron Rules

A `System` or `Component` description shall define a position in space. Listing 12-22 - Listing 12-24 specify the encoding of this position and restrict the SensorML schema which offers several options to do this. Listing 12-22 specifies the rule that a `position` element shall contain a `swe:Position` and it redefines the "optional" `referenceFrame` attribute of a `swe:Position` as "required". The `swe:Position` element shall incorporate a `swe:location` containing at least two `swe:Vector/swe:coordinate/swe:Quantity` elements to express the position coordinates (Listing 12-23). Listing 12-24 specifies three conditions. First, it is required that this `swe:Quantity`, which is used to declare the coordinate value, owns an `axisID` attribute to link to the axis to which it refers. Second, a `swe:value` element is utilized to specify the coordinate value. And third the `swe:uom` element uses a `code` attribute to define the unit of the coordinate value.

Listing 12-22: Schematron rule for ensuring that a Position (including a referenceFrame attribute) is provided

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

Listing 12-23: Schematron rule for defining the description of the position

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

Listing 12-24: Schematron rule for defining the format of coordinates

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

12.1.9 Interfaces

For every `System` and `Component` an interfaces section shall be provided. This section contains the information through which SWE services a sensor is accessible. Thus, this is an essential information element for allowing SWE service discovery.

For each SWE service through which a sensor is available an according `InterfaceDefintion` on the `serviceLayer` shall be provided. Such an `InterfaceDefintion` consists of a data record containing the following fields:

- `urn:ogc:def:interface:OGC:1.0:ServiceURL`: The URL of the SWE service that encapsulates the sensor.
- `urn:ogc:def:interface:OGC:1.0:ServiceType`: The type of the SWE service (i.e. SOS, SPS, SAS, SES) that encapsulates the sensor.
- `urn:ogc:def:interface:OGC:1.0:ServiceSpecificSensorID`: The ID of the sensor within the SWE service instance.

Listing 12-25 shows an example for such a description of interfaces:

Listing 12-25: Example of an interface description

```
<interfaces>
  <InterfaceList>
    <interface name="SOS1">
      <InterfaceDefinition>
        <serviceLayer>
          <swe:DataRecord

definition="urn:ogc:def:interface:OGC:1.0:SWEServiceInterface">
            <swe:field name="urn:ogc:def:interface:OGC:1.0:ServiceURL">
              <swe:Text>
                <swe:value>http://v-swe.uni-
                              muenster.de:8080/WeatherSOS/sos</swe:value>
              </swe:Text>
            </swe:field>
            <swe:field name="urn:ogc:def:interface:OGC:1.0:ServiceType">
              <swe:Text>
                <swe:value>SOS</swe:value>
              </swe:Text>
            </swe:field>
            <swe:field

name="urn:ogc:def:interface:OGC:1.0:ServiceSpecificSensorID">
              <swe:Text>
                <swe:value>sensor02</swe:value>
              </swe:Text>
            </swe:field>
          </swe:DataRecord>
        </serviceLayer>
      </InterfaceDefinition>
    </interface>
  </InterfaceList>
</interfaces>
```

12.1.9.1 Schematron Rules

For each System or Component at least one link to a SWE service shall be provided (Listing 12-26). Each SWE service interface definition shall contain the URL or the SWE service, the service type and the ID of the sensor within the SWE service instance.

Listing 12-26: Schematron rule for ensuring that at least one SWE service interface definition is provided

```
<rule context="//sml:System/sml:interfaces">
     <assert
test="count(swe:InterfaceList/swe:interface/swe:InterfaceDefinition/swe:serv
iceLayer/swe:DataRecord[@definition =
'urn:ogc:def:interface:OGC:1.0:SWEServiceInterface']) > 0" >Error: at least
one interface definition for a SWE service has to be provided.</assert>
</rule>
<rule context="//sml:Component/sml:interfaces">
     <assert
test="count(swe:InterfaceList/swe:interface/swe:InterfaceDefinition/swe:serv
iceLayer/swe:DataRecord[@definition =
'urn:ogc:def:interface:OGC:1.0:SWEServiceInterface']) > 0" >Error: at least
one interface description for a SWE service has to be provided.</assert>
</rule>
```

Listing 12-27: Schematron rules for ensuring that for a SWE service interface link all necessary information is provided

```
<rule
context="//sml:interfaces/swe:InterfaceList/swe:interface/swe:InterfaceDefin
ition/swe:serviceLayer/">
     <assert test="count(swe:DataRecord/swe:field [@name =
'urn:ogc:def:interface:OGC:1.0:ServiceURL']) = 1" >Error: The URL of the SWE
service has to be provided.</assert>
</rule>
<rule
context="//sml:interfaces/swe:InterfaceList/swe:interface/swe:InterfaceDefin
ition/swe:serviceLayer/">
     <assert test="count(swe:DataRecord/swe:field [@name =
'urn:ogc:def:interface:OGC:1.0:ServiceType']) = 1" >Error: The type of the
SWE service has to be provided.</assert>
</rule>
<rule
context="//sml:interfaces/swe:InterfaceList/swe:interface/swe:InterfaceDefin
ition/swe:serviceLayer/">
     <assert test="count(swe:DataRecord/swe:field [@name =
'urn:ogc:def:interface:OGC:1.0: ServiceSpecificSensorID']) = 1" >Error: The
sensor ID within the SWE service has to be provided.</assert>
</rule>
```

12.1.10 Inputs

The `inputs` section (Listing 12-28) is mandatory for every `System` and `Component`. It lists up the phenomena observed by the sensor. Therefore every `input` utilizes the `swe:ObservableProperty` and its `definition` attribute which takes a URN to point to a dictionary entry. This dictionary entry defines the measured phenomenon in detail and declares its semantics.

Listing 12-28: Example of a inputs section

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

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

12.1.10.1 Schematron Rules

The rules of Listing 12-32 define that the `inputs` section shall be present for a `System` or a `Component`.

Listing 12-29: Schematron rules for ensuring that an inputs section is provided

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

The structure of the `inputs` section is restricted by the rule of Listing 12-33. It is specified that the child elements of each `input` element shall possess a `definition` attribute. The value of this attribute can be considered as a link to the semantics of the `input`.

Listing 12-30: Schematron rule for ensuring that the inputs section contains a definition element

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

12.1.11 Outputs

Within the `outputs` section (Listing 12-31) which is mandatory for every `System` and `Component` the signals recorded by the sensor are defined. The sub-element of each output references the captured phenomenon and specifies the unit of measure in which the measured values are expressed.

Listing 12-31: Example of an outputs section

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

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

12.1.11.1 Schematron Rules

The rules of Listing 12-32 define that the `outputs` section shall be present for a `System` or a `Component`.

Listing 12-32: Schematron rules for ensuring that an outputs section is provided

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

The structure of the `output` section is restricted by the rule of Listing 12-33. It is specified that the child elements of each `output` element shall possess a `definition` attribute. The value of this attribute can be considered as a link to the semantics of the `output`.

Listing 12-33: Schematron rule for ensuring that outputs section contains a definition element

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

12.1.12 Components

The distinct sensors of the station are separately modeled as `Components` which are associated with the whole `System`. The `components` section of the SensorML description lists up all these sensors. The description of a component can be either defined inline within the `System` document or it can be referenced using the `xlink:href` attribute. In the example below (Listing 12-34) both alternatives are shown.

Listing 12-34: Example of a components section

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

12.1.12.1 Schematron Rules

The `components` section of a system description lists the descriptions of the associated sensors. The `component` element shall contain either the attribute `xlink:href` to specify an external reference to the sensor description or a `Component` as a child element which describes the sensor inline. This condition is formulated in Listing 12-35.

Listing 12-35: Schematron rule for ensuring that descriptions for the Components of a System are provided

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

12.2 Issues

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

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

- Inheritance of metadata items: In the current version the SensorML Discovery Profile contains a full set for systems as well as components. In order to reduce redundancies within the metadata inheritance mechanisms need to be supported in the future. This means that components inherit metadata items from the systems they belong to, so that common metadata do not need to be explicitly provided for every component.

- Dictionaries for identifiers: For example, the phenomena measured by a sensor are identified within a SensorML document by URIs. In order to ensure a consistent use of these phenomenon identifiers and to make the definitions that are assigned to these URIs accessible, it is necessary to provide a phenomenon dictionary or registry. An important functionality would be an operation providing access to the phenomenon definitions (e.g. resolving the URIs) but also for exploiting semantic relationships (e.g. finding equivalent or similar definitions). A first approach for such a Dictionary is described in the OGC Sensor Observable Registry Discussion Paper [OGC 09-112].

13 SensorML metadata mapping

The following tables define the use of predefined attributes and slots, in order to map information coming from the SensorML metadata file to these additional *ExtrinsicObject.*

In the SensorML model, a System is composed of Components or Systems. For each SensorML System, a new System Extrinsic Object must be created. For each SensorML Component, a new Component Extrinsic Object must be created. If a System has a ComponentsList, an association "ComposedOf" must be created between the System and its sub-systems (of type Component or System).

The section 13.1 defines the mapping of a SensorML System object to a System Extrinsic Object. The section 13.2 defines the mapping of a SensorML Component object to a Component Extrinsic Object.

If an SensorML Contact object is defined, a new Organization Registry Object must be created and associated to the corresponding System or Component with an "ResponsibleFor" association (as adviced in Section 5.4 of [ebRIM30]) . The section 13.3 defines the mapping between a SensorML Contact and a Organization Registry Object.

If a SWE service interface is defined, a new Service Registry Object must be created and associated to the corresponding System or Component with an "AccessibleThrough" association. The section 13.4 defines the mapping between a SWE interface and the corresponding Service Registry Object and Association.

13.1 System Extrinsic Object mapping

The Table 17 defines the mapping for a SensorML metadata file containing a System object.

Table 17 – System ExtrinsicObject Correspondence

SensorML Metadata Xpath	System ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
"urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System" (fixed value)	/wrs:ExtrinsicObject/@objectType	/	Yes	1
/sml:System/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition=urn:ogc:def:identifier:OGC:1.0:uniqueID/sml:value	/wrs:ExtrinsicObject/@lid[6]	String	Yes	1
/sml:System/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition=urn:ogc:def:identifier:OGC:1.0:longName]/sml:value	/wrs:ExtrinsicObject[@objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System"]/rim:Name/rim:LocalizedString/@value	String	Yes	1
/sml:System/gml:description	/wrs:ExtrinsicObject[@objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System"]/rim:Description/rim:LocalizedString/@value	String	Yes	0..1
/sml:System/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition=urn:ogc:def:identifier:OGC:1.0:	/wrs:ExtrinsicObject[@objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-	String	Yes	1

[6] Each RegistryObject instance MUST have a lid (Logical Id) attribute. The lid is used to refer to a logical RegistryObject in a version independent manner. All versions of a RegistryObject MUST have the same value for the lid attribute. Note that this is in contrast with the id attribute that MUST be unique for each version of the same logical RegistryObject. The lid attribute MAY be specified by the submitter when creating the original version of a RegistryObject. If the submitter assigns the lid attribute, she must guarantee that it is a globally unique URN. A registry MUST honor a valid submitter-supplied LID. If the submitter does not specify a LID then the registry MUST assign a LID and the value of the LID attribute MUST be identical to the value of the id attribute of the first (originally created) version of the logical RegistryObject ([ebRIM30], section 2.5.6).

SensorML Metadata Xpath	System ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
shortName]/sml:value	Sensor::ShortName”]/rim:ValueList/rim:Value[1]			
/sml:System/sml:keywords/sml:KeywordList/sml:keyword	/wrs:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Keywords”] /rim:ValueList/rim:Value[*]	String	Yes	1..N
/sml:System/sml:capabilities/swe:DataRecord/swe:field[@name="observedBBOX"]/swe:Envelope	/wrs:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ObservedBoundingBox”] /wrs:ValueList/wrs:AnyValue[1]	GM_Envelope	Yes	1
/sml:System/sml:inputs/sml:InputList/sml:input/swe:ObservableProperty/@definition	/wrs:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Inputs”] /rim:ValueList/rim:Value[*]	String	Yes	1..N
/sml:System/sml:outputs/sml:OutputList/sml:output/ swe:Quantity/@definition	/wrs:ExtrinsicObject[@objectType=” urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Outputs”] /rim:ValueList/rim:Value[*]	String	Yes	1..N
/sml:System/sml:position/swe:Position/swe:location	/wrs:ExtrinsicObject[@objectType=” urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Location”] /wrs:ValueList/wrs:AnyValue[1]	GM_Point	Yes	1
/sml:System/sml:validTime/gml:TimePeriod/gml:beginPosition	/wrs:ExtrinsicObject[@objectType=” urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ValidTimeBegin”] /rim:ValueList/rim:Value[1]	Date	Yes	1
/sml:System/sml:validTime/gml:TimePeriod/gml:endPosition	/wrs:ExtrinsicObject[@objectType=” urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=”	Date	Yes	1

SensorML Metadata Xpath	System ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ValidTimeEnd”] /rim:ValueList/rim:Value[1]			
/sml:System/sml:classification/sml:ClassifierList/sml:classifier/sml:Term[@definition=”urn:ogc:def:property:OGC:1.0:application”]	Add a Classification to the Corresponding ClassificationNode of the “IntendedApplication” ClassificationScheme. (see section 13.5)	Classification	Yes	1..N
/sml:System/sml:classification/sml:ClassifierList/sml:classifier/sml:Term[@definition=”urn:ogc:def:property:OGC:1.0:sensorType”]	Add a Classification to the Corresponding ClassificationNode of the “SystemType” ClassificationScheme. (see section 13.6)	Classification	Yes	1

For each components present in the System’s ComponentList, an association “ComposedOf” to the Component is created if the Component exists in the Catalog.

13.2 Component Extrinsic Object mapping

The Table 18 defines the mapping for a SensorML metadata file containing a Component object.

Table 18 – Component ExtrinsicObject Correspondence

SensorML Metadata Xpath	Component ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
"urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component" (fixed value)	/wrs:ExtrinsicObject/@objectType	/	Yes	1
/sml:Component/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition=urn:ogc:def:identifier:OGC:1.0: uniqueID/sml:value	/wrs:ExtrinsicObject/@lid[7]	String	Yes	1
/sml:Component/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition=urn:ogc:def:identifier:OGC:1.0:longName']/sml:value	/wrs:ExtrinsicObject[@objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component"]/rim:Name/rim:LocalizedString/@value	String	Yes	1
/sml:Component/gml:description	/wrs:ExtrinsicObject[@objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor:: Component"]/rim:Description/ rim:LocalizedString/@value	String	Yes	0..1
/sml:Component/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition=urn:ogc:def:identifier:OGC:1.0:shortName']/sml:value	/wrs:ExtrinsicObject[@objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ShortName"]/rim:ValueList/rim:Value[1]	String	Yes	1
/sml:Component/sml:keywords/sml:KeywordList/sml:keyword	/wrs:ExtrinsicObject[@objectType=" urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Keywords"]	String	Yes	1..N

[7] The same guidlines as for Systems apply here, see Table 16 - System ExtrinsicObject Correspondence.

SensorML Metadata Xpath	Component ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
	/rim:ValueList/rim:Value[*]			
/sml:Component/sml:capabilities/swe:DataRecord/swe:field[@name="observedBBOX"]/swe:Envelope	/wrs:ExtrinsicObject[@objectType=" urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ObservedBoundingBox"] /wrs:ValueList/wrs:AnyValue[1]	GM_Envelope	Yes	1
/sml:Component/sml:inputs/sml:InputList/sml:input/swe:ObservableProperty/@definition	/wrs:ExtrinsicObject[@objectType=" urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Inputs"] /rim:ValueList/rim:Value[*]	String	Yes	1..N
/sml:Component/sml:outputs/sml:OutputList/sml:output/swe:Quantity/@definition	/wrs:ExtrinsicObject[@objectType=" urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Outputs"] /rim:ValueList/rim:Value[*]	String	Yes	1..N
/sml:Component/sml:position/swe:Position/swe:location	/wrs:ExtrinsicObject[@objectType=" urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::Component"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Location"] /wrs:ValueList/wrs:AnyValue[1]	GM_Point	Yes	1
/sml:Component/sml:validTime/gml:TimePeriod/gml:beginPosition	/wrs:ExtrinsicObject[@objectType=" urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor:: Component"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ValidTimeBegin"] /rim:ValueList/rim:Value[1]	Date	Yes	1
/sml: Component /sml:validTime/gml:TimePeriod/gml:endPosition	/wrs:ExtrinsicObject[@objectType=" urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor:: Component"]/rim:Slot[@name=" urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ValidTimeEnd"] /rim:ValueList/rim:Value[1]	Date	Yes	1
/sml:Component/sml:classification/sml:ClassifierList/sml:cla	Add a Classification to the Corresponding ClassificationNode of the	Classification	Yes	1

SensorML Metadata Xpath	Component ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
ssifier/sml:Term[@definition=”urn:ogc:def:property:OGC:1.0:application”]	“IntendedApplication” ClassificationScheme. (see section 13.5)			
/sml:Component/sml:classification/sml:ClassifierList/sml:classifier/sml:Term[@definition=”urn:ogc:def:property:OGC:1.0:sensorType”]	Add a Classification to the Corresponding ClassificationNode of the “SystemType” ClassificationScheme. (see section 13.6)	Classification	Yes	1

13.3 Organization Registry Object Mapping

The Table 19 defines the mapping for a SensorML Contact object to an Organization Registry Object.

Table 19 – Organization Registry Object Correspondence

SensorML Metadata Xpath	Organization RegistryObject Attribute	(Slot) Type	Queryable	Cardinality
/sml:contact/sml:ResponsibleParty/sml:organizationName	/rim:Organization/rim:Name/rim:LocalizedString/@value[8]	String	Yes	1
/sml:contact/sml:ResponsibleParty/sml:individualName	/rim:Person/rim:PersonName[9]	String	Yes	0..1
/sml:contact/sml:ResponsibleParty/sml:contactInfo/sml:address/sml:city	/rim:Organization/rim:addresses/rim:Address/rim:city/@value	String	Yes	0..1
/sml:contact/sml:ResponsibleParty/sml:contactInfo/sml:address/sml:deliveryPoint	/rim:Organization/ rim:Address/rim:street/@value	String	Yes	0..1
/sml:contact/sml:ResponsibleParty/sml:contactInfo/sml:addre	/rim:Organization/ rim:Address/rim:postalCode/@value	String	Yes	0..1

[8] The primaryContact attribute of the Organization contains the uuid of the Person instance based on sml:individualName.

[9] In ebRIM, the Person object contains a PersonName which has three attributes: firstName, lastName and middleName. The sml:individualName should be split into these attributes.

SensorML Metadata Xpath	Organization RegistryObject Attribute	(Slot) Type	Queryable	Cardinality
ss/sml:postalCode				
/sml:contact/sml:ResponsibleParty/sml:contactInfo/sml:address/sml:country	/rim:Organizationrim:Address/rim:country/@value	String	Yes	0..1
/sml:contact/sml:ResponsibleParty/sml:contactInfo/sml:address/sml:electronicMailAddress	/rim:Organization/rim:EmailAddress/@address	String	Yes	0..1

13.4 Service Registry Object Mapping

The Table 19 defines the mapping for a SWE InterfaceDefinition object to a Service Registry Object.

Table 20 – Service Registry Object Correspondence

SensorML Metadata Xpath	Service RegistryObject Attribute	(Slot) Type	Queryable	Cardinality
/swe:InterfaceList/swe:interface/swe:InterfaceDefinition/swe:serviceLayer/swe:DataRecord/swe:field[@name=”urn:ogc:def:interface:OGC:1.0:ServiceURL”]	/rim:Service/rim:ServiceBinding/@accessURI	String	Yes	1
/swe:InterfaceList/swe:interface/swe:InterfaceDefinition/swe:serviceLayer/swe:DataRecord/swe:field[@name=”urn:ogc:def:interface:OGC:1.0:ServiceSpecificSensorID”]	/rim:Association[@associationType=” urn:ogc:def:associationType:OGC-CSW-ebRIM-Sensor::AccessibleThrough”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::ServiceSpecificSensorID”]/rim:ValueList/rim:Value[1]	String	Yes	1

13.5 IntendedApplication ClassificationScheme mapping

In a SensorML metadata, a ClassifierList could be associated to a System or a Component. For each Classifier Term of type “urn:ogc:def:classifier:OGC:application” (it means, where the “definition” attribute is equal to “urn:ogc:def:classifier:OGC:application”), the

System (or Component) must be classified to the corresponding ClassificationNode of the “IntendedApplication” Classification Scheme. If the ClassificationNode with the corresponding value does not exist, a new one is created in the “IntededApplication” Classification Scheme.

13.6 SystemType ClassificationScheme mapping

In a SensorML metadata, a ClassifierList could be associated to a System or a Component. For each Classifier Term of type “urn:ogc:def:property:OGC:sensorType” (it means, where the “definition” attribute is equal to “urn:ogc:def:property:OGC:sensorType”), the System (or Component) must be classified to the corresponding ClassificationNode of the “SystemType” Classification Scheme. If the ClassificationNode with the corresponding value does not exist, a new one is created in the “SystemType” Classification Scheme.

13.7 Service ClassificationScheme mapping

In a SensorML metadata the interfaces section can contain a list of interfaces to access inputs and outputs of a System or Component. For each SWE InterfaceDefinition of type “urn:ogc:def:interface:OGC:1.0:SWEServiceInterface” (it means, where the “definition” attribute is equal to “urn:ogc:def:interface:OGC:1.0:SWEServiceInterface”), the created Service must be classified based on the SWE field of the type “urn:ogc:def:interface:OGC:1.0:ServiceType” (it means, where the “name” attribute is equal to “urn:ogc:def:interface:OGC:1.0: ServiceType”) to the corresponding ClassificationNode of the “ISO 19119 services taxonomy” Classification Scheme as defined in [OGC 07-144r4]. If the ClassificationNode with the corresponding value does not exist, a new one is created.

13.8 Grouping the Complete Mapping Schema into a Registry Package

To allow easier retrieving of the full representation of an Sensor, the objects defines earlier shall be grouped into a Registry Package. In the ebRIM specification [ebRIM30], RegistryPackage instances allow for grouping of logically related RegistryObject. A query on the Registry Package element shall return all the objects that represent a full Sensor.

Annex A: DOCUMENT CHANGE HISTORY (INFORMATIVE)

A.1 Changes with respect to the 0.1.1 Version

Annex B: EARTH OBSERVATION SENSOR PROFILE

The Earth Observation domain comes with business specific objects like « platform » « instrument », « detector »… and attributes like « along track pointing range », « nadir swath width », « spectral range ». This section defines the Earth Observation sensor model that extends the generic sensor model.

B.1 Package overview

B.1.1 Purpose

The Figure 7 shows the Earth Observation sensor ebRIM model.

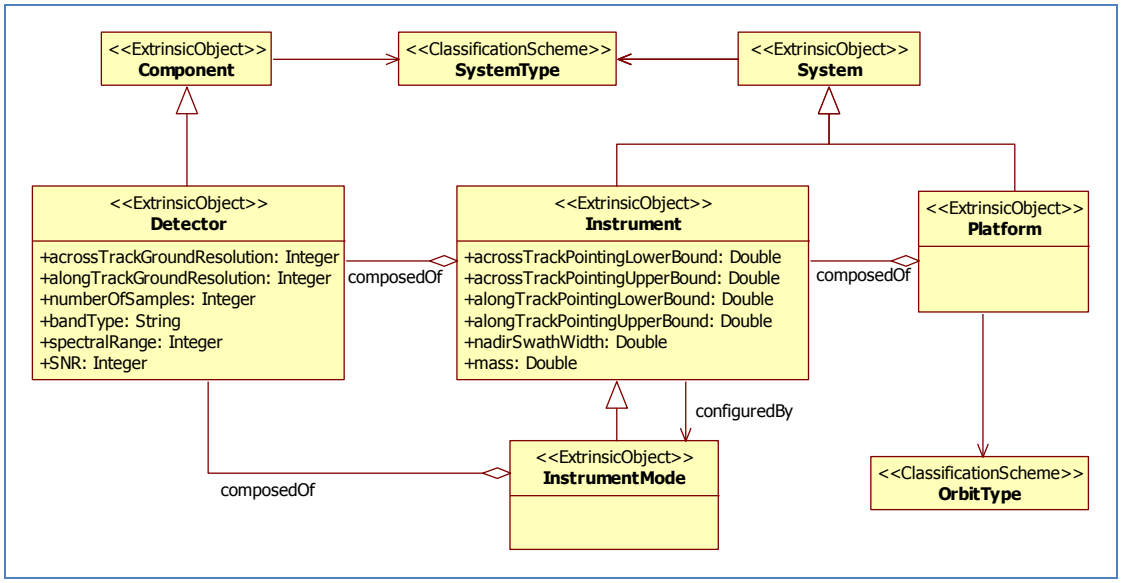


Figure 7 – Earth Observation sensor ebRIM model[10]

This model defines the following objects :

- **Platform** : The "Platform" object contains technical characteristics of an EO Platform that carries one or more several remote sensing instruments. The following characteristics are of interest:
  - Platform UID, short name, long name and description
  - Platform and orbit type (LEO, GEO, etc…)
  - Operator name and contact information
  - List of carried instruments

[10] For the understanding of the diagram, the conputal name of the object is used like « Detector », « Instrument », « Platform »… In the ebRIM model, all the ExtrinsicObject objet types are either System or Component and are classified according to their types using the SystemType Classification Scheme.

- **Instrument**: The "Instrument" object contains technical characteristics of an EO instrument and all of its detectors (Some of these characteristics marked with * can be overridden by specific modes):
    - Instrument UID, short name, long name and description
    - Association to parent platform
    - Instrument type and applications
    - Across-track and along-track pointing ranges*
    - Swath width at nadir*
    - List of possible modes
    - For each detector

- **Detector :**

- **InstrumentMode** : The "Instrument Mode" object describes technical characteristics of one specific mode of a remote sensing instrument. It contains information such as (some characteristics can be inherited from the parent instrument if not mode specific. Such parameters are marked with *):
    - Instrument mode UID, short name, long name and description
    - Association to parent instrument
    - Across-track and along-track field of views*
    - Swath width at nadir*
    - List of detectors used in this mode

In this model, we only extract information usefull for searching object. The goal is not to store all the information in the ebRIM model.

The orbit type information is stored as a Classification Scheme. The Figure 8 shows the Orbit Type Classification Scheme.

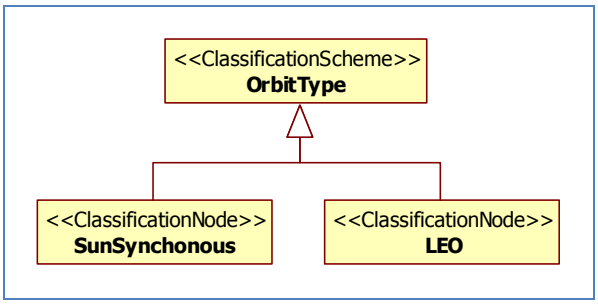


Figure 8 – Orbit Type Classification Scheme

This Earth Observation sensor model profile also extends the IntendedApplication and SystemType Classification Scheme.

The Figure 9 shows the new Classification Nodes added to the Intended Application Classification Scheme.

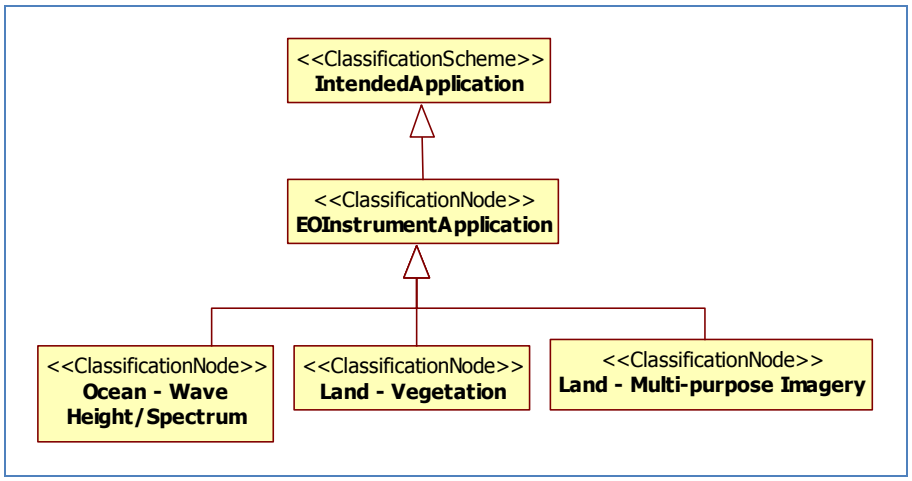


Figure 9 – Earth Observation Intended Application Classification Scheme

The Figure 10 shows the Classification Nodes added to the SystemType Classification Scheme.

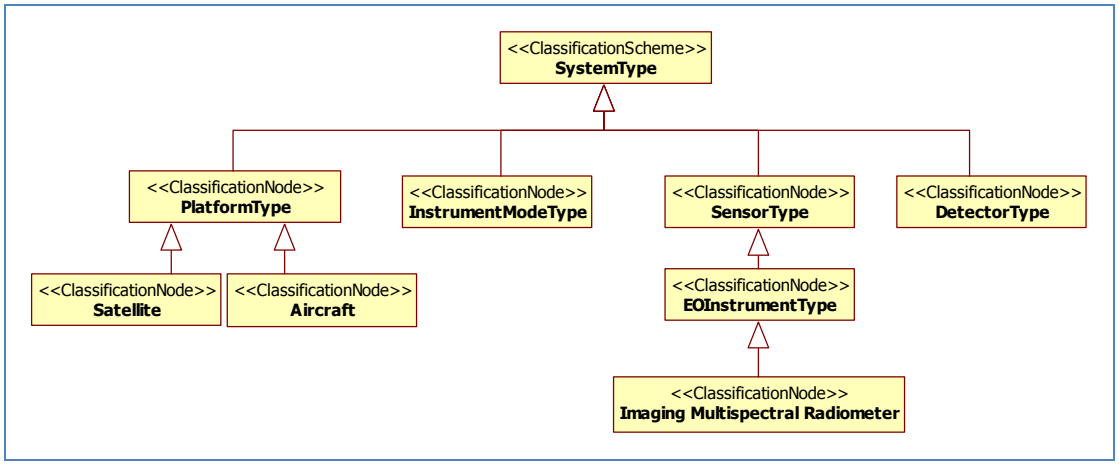


Figure 10 – Earth Observation System Type Classification Scheme

The Figure 11 shows how the Spot5 satellite is represented in the Earth Observation sensor ebRIM model.

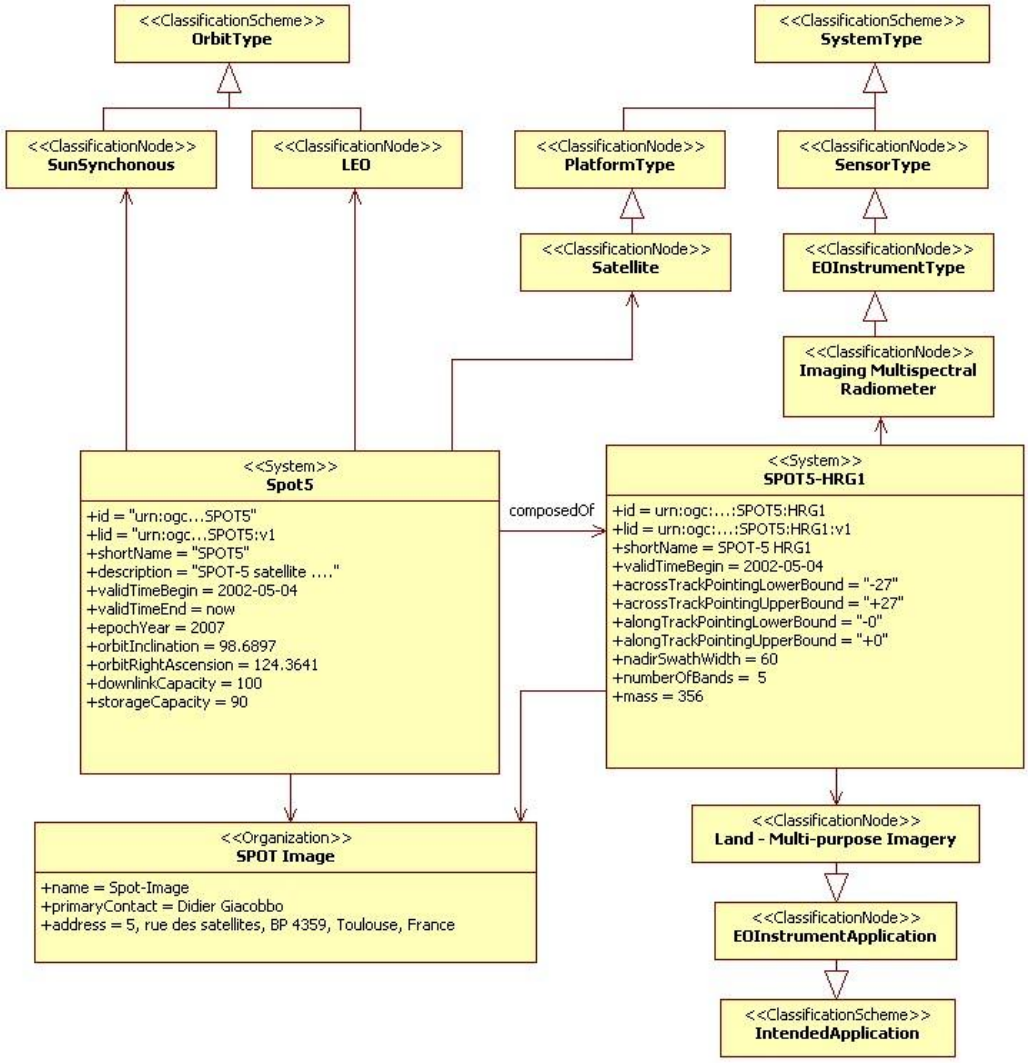


Figure 11 – Spot 5 satellite sample in the Earth Observation sensor ebRIM model

B.1.2 Dependencies

[TO BE DEFINED]

B.2 ClassificationSchemes

B.2.1 Intended Application

This profile defines a list of ClassificationNode added to the Intended Application classification scheme. The Committee on Earth Observation Satellites (CEOS) defines an "instrument application" dictionary notably used by the ESA. This dictionary defines a list of applications in which the instrument could be used. To group all the

applications related to the Earth Observation domain, a Classification Node "EOInstrumentApplication" is added to the "IntendedApplication" classification scheme. Table 21 defines the classification nodes added to the "EOInstrumentApplication" classification node.

Table 21 – Classification nodes: EOInstrumentApplication

Code	Description
Atmosphere - Aerosols	Atmosphere – Aerosols http://www.eohandbook.com/eohb05/ceos/aerosols_pop.html
Atmosphere - Humidity Fields	Atmosphere - Atmospheric Humidity Fields http://www.eohandbook.com/eohb05/ceos/atmos_humid_pop.html
Atmosphere - Temperature Fields	Atmosphere - Atmospheric Temperature Fields http://www.eohandbook.com/eohb05/ceos/atmos_temp_pop.html
Atmosphere - Winds	Atmosphere - Atmospheric Winds http://www.eohandbook.com/eohb05/ceos/atmos_winds_pop.html
Atmosphere - Cloud Particles	Atmosphere - Cloud Particle Properties/Profile http://www.eohandbook.com/eohb05/ceos/cloud_particle_pop.html
Atmosphere - Cloud Type	Atmosphere - Cloud Type/Cloud Amount http://www.eohandbook.com/eohb05/ceos/cloud_type_pop.html
Atmosphere - Cloud Top Temp	Atmosphere - Cloud Top Temperature http://www.eohandbook.com/eohb05/ceos/cloud_type_pop.html
Atmosphere – Water Precipitation	Atmosphere - Liquid Water/Precipitation Rate http://www.eohandbook.com/eohb05/ceos/liquid_water_pop.html
Atmosphere – Ozone	Atmosphere – Ozone http://www.eohandbook.com/eohb05/ceos/ozone_pop.html
Atmosphere - Radiation Budget	Atmosphere - Radiation Budget http://www.eohandbook.com/eohb05/ceos/rad_budget_pop.html

Atmosphere - Trace Gases	Atmosphere - Trace Gases http://www.eohandbook.com/eohb05/ceos/trace_gases_pop.html
Land – Albedo Reflectance	Land - Albedo/Reflectance http://www.eohandbook.com/eohb05/ceos/albedo_pop.html
Land - Topography	Land - Topography http://www.eohandbook.com/eohb05/ceos/land_topog_pop.html
Land - Soil Moisture	Land - Soil Moisture http://www.eohandbook.com/eohb05/ceos/soil_moisture_pop.html
Land - Vegetation	Land - Vegetation http://www.eohandbook.com/eohb05/ceos/vegetation_pop.html
Land - Surface Temperature	Land - Surface Temperature http://www.eohandbook.com/eohb05/ceos/surface_temp_land_pop.html
Land - Multi-purpose Imagery	Land - Multi-purpose Imagery http://www.eohandbook.com/eohb05/ceos/multi_purpose_land_pop.html
Ocean - Color Biology	Ocean - Color/Biology http://www.eohandbook.com/eohb05/ceos/ocean_colour_pop.html
Ocean - Topography Currents	Ocean - Topography/Currents http://www.eohandbook.com/eohb05/ceos/ocean_topog_pop.html
Ocean - Surface Winds	Ocean - Surface Winds http://www.eohandbook.com/eohb05/ceos/ocean_winds_pop.html
Ocean - Surface Temperature	Ocean - Surface Temperature http://www.eohandbook.com/eohb05/ceos/ocean_temp_pop.html
Ocean - Wave Height	Ocean - Wave Height/Spectrum

Spectrum	http://www.eohandbook.com/eohb05/ceos/wave_height_pop.html
Ocean - Multi-purpose Imagery	Ocean - Multi-purpose Imagery http://www.eohandbook.com/eohb05/ceos/multi_purpose_ocean_pop.html
Ice - Topography	Snow/Ice - Ice Sheet Topography http://www.eohandbook.com/eohb05/ceos/ice_sheet_topog_pop.html
Snow - Cover	Snow/Ice - Snow Cover/Edge/Depth http://www.eohandbook.com/eohb05/ceos/snow_cover_pop.html
Ice - Cover	Snow/Ice - Sea Ice Cover/Edge/Thickness http://www.eohandbook.com/eohb05/ceos/sea_ice_cover_pop.html
Gravity - Magnetic Geodynamic	Gravity - Magnetic/Geodynamic Measurements http://www.eohandbook.com/eohb05/ceos/gravity_pop.html

B.2.2 SystemType

This profile defines a list of ClassificationNode added to the Sensor Type classification scheme. As shown in Figure 10, a first general level of system type is defined with “PlatformType”, “SensorType”, “DetectorType” and “InstrumentModeType”.

Table 22 – Classification nodes: System Type

Code	Description
PlatformType	Platform system type
SensorType	Sensor sytem type
DetectorType	Detector system type
InstrumentModeType	Instrument mode system type

The Committee on Earth Observation Satellites (CEOS) defines a "Platform Type" dictionary notably used by the ESA. This dictionary defines a list of list of platform type. This dictionary could be extended with other platform type.

Table 23 – Classification nodes: Platform Type

Code	Description
Satellite	Satellite platform type
Aircraft	Aircraft platform type
Helicopter	Helicopter platform type

The Committee on Earth Observation Satellites (CEOS) defines an "Instrument Type" dictionary notably used by the ESA. This dictionary defines a list of list of instrument type used in the Earth Observation domain. To group all the instrument type related to the Earth Observation domain, a Classification Node "EOInstrumentType" is added below the "SensorType" node. The SensorType could be extended with other sensor type than the Earth Observation instrument types.

Table 24 – Classification nodes: EO Instrument Type

Code	Description
Imaging Multispectral Radiometer	Instrument measuring radiation intensity in multiple narrow, precisely calibrated spectral channels. http://www.eohandbook.com/eohb05/ceos/part3_1_pop6.html
Stereo Imaging Radiometer	Instrument measuring radiation intensity from multiple view angles to reconstruct surface topography.
Imaging Microwave Radar	Instrument transmitting at frequencies of around 1 to 10GHz and measuring the backscattered signals to generate microwave images of Earth's surface at high spatial resolutions (between 10m and 100m). http://www.eohandbook.com/eohb05/ceos/part3_1_pop8.html
Lidar	Ranging instrument measuring the radiation that is returned either from particles in the atmosphere or from the Earth's surface when illuminated by a laser source. http://www.eohandbook.com/eohb05/ceos/part3_1_pop9.html
Radar Altimeters	Non-imaging instrument which use the ranging capability of radar to measure the surface topographic profile parallel to the platform

	track. They provide precise measurements of the platform's altitude by measuring the time interval between the transmission and reception of very short electromagnetic pulses. http://www.eohandbook.com/eohb05/ceos/part3_1_pop12.html

B.2.3 OrbitType

The Committee on Earth Observation Satellites (CEOS) coordinates civil space-borne observations of the Earth. This organization defines an Orbit Types dictionary notably used by the ESA. The OrbitTypes Classification Scheme represents this dictionary. Each platform could be classified according to one or several nodes of this scheme. The properties of the scheme are summarized in the table below.

Table 25 – Classification scheme: Orbit types

Property	Value
Identifier	urn:ogc:def: classificationScheme:OGC-CSW-ebRIM-Sensor::OrbitTypes
Name	Orbit types taxonomy
Description	Defines a taxonomy that may be used to classify platform according to their Orbit Types.
Node type	urn:oasis:names:tc:ebxml-regrep:NodeType:UniqueCode
Internal nodes	true

Table 26 – Classification nodes: Orbit types

Code	Description
Sun-Synchronous	Sun-Synchronous orbit type
Geosynchronous	Geosynchronous orbit type
Geostationary	Geostationary orbit type
Low_Earth	Low Earth orbit type

B.3 ClassificationNode

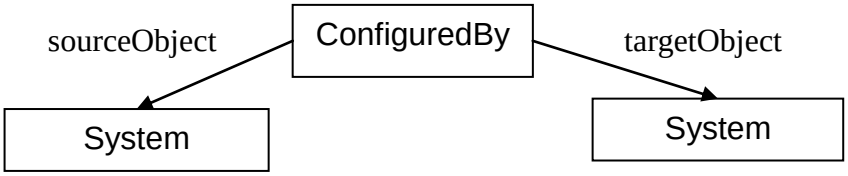
B.3.1 Object types

No new objet types are added by this profile

B.3.2 AssociationTypes

ConfiguredBy

The "ConfiguredBy" association relates an Instrument with an InstrumenConfig item. For an association of this type, the source and target objects shall be of the types



indicated in Figure 6.

Figure 12 – Object type constraints for the 'ConfiguredBy' association

Example: The "ConfiguredBy" association.

```
<rim:Association
  id="urn:uuid:ba8348a0-93b2-11dc-893e-0002a5d5c51b"
  sourceObject="urn:uuid:aa779e20-93b2-11dc-a6e1-0002a5d5c51b"
  targetObject="urn:uuid:7f2a1b80-93b2-11dc-bee7-0002a5d5c51b"
  associationType="urn:ogc:def:associationType:OGC-CSW-ebRIM-
Sensor::ConfiguredBy" />

<wrs:ExtrinsicObject
  id="urn:uuid:aa779e20-93b2-11dc-a6e1-0002a5d5c51b"
  objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-
Sensor::System" />

<wrs:ExtrinsicObject
  id="urn:uuid:7f2a1b80-93b2-11dc-bee7-0002a5d5c51b"
  objectType="urn:ogc:def:objectType:OGC-CSW-ebRIM-
Sensor::System" />
```

The properties of the classification node are summarized in the table below.

Table 27 – Association type: ConfiguredBy

Property	Value
Identifier	urn:ogc:def:associationType:OGC-CSW-ebRIM-Sensor::ConfiguredBy
Name	ConfiguredBy
Description	Links an Instrument (System) to its InstrumentMode (System).
Parent	urn:oasis:names:tc:ebxml-regrep:classificationScheme:AssociationType
Code	ConfiguredBy

B.4 Slots

This profile adds the slots defined in the following tables. The value "–" appearing in the tables below indicates the absence of a more specific constraint.

Table 28 – Slot: AcrossTrackPointingLowerBound

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::AcrossTrackPointingLowerBound
Definition	The across track pointing lower bound in degrees
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Double
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 29 – Slot: AcrossTrackPointingUpperBound

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::AcrossTrackPointingUpperBound
Definition	The across track pointing upper bound in degrees
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Double
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 30 – Slot: AlongTrackPointingLowerBound

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::AlongTrackPointingLowerBound
Definition	The along track pointing lower bound in degrees
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Double
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 31 – Slot: AlongTrackPointingUpperBound

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::AlongTrackPointingUpperBound
Definition	The along track pointing upper bound in degrees
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Double
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 32 – Slot: NadirSwathWidth

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::NadirSwathWidth
Definition	The nadir swath width in kilometers
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Double
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 33 – Slot: Mass

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::Mass
Definition	The mass of the instrument in kilograms
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Double
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 34 – Slot: AcrossTrackGroundResolution

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::AcrossTrackGroundResolution
Definition	The across track ground resolution in meters
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Integer
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 35 – Slot: AlongTrackGroundResolution

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::AlongTrackGroundResolution
Definition	The along track ground resolution in meters
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Integer
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 36 – Slot: NumberOfSamples

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::NumberOfSamples
Definition	The number of samples
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Integer
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 37 – Slot: BandType

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::BandType
Definition	The band type
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Integer
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

Table 38 – Slot: SNR

Name	urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::SNR
Definition	The Signal-to-Noise Ratio in decibel
Source	–
Slot type	urn:oasis:names:tc:ebxml-regrep:DataType:Integer
Parent object type	urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject

B.5 Predefined queries

[TO BE DEFINED]

B.6 Metadata extraction rules

B.6.1 Platform mapping

Table 39 – Platform Correspondence

SensorML Metadata Xpath	System ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
“urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System” (fixed value)	/rim:ExtrinsicObject/@objectType	/	Yes	1
/sml:System/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition=" urn:ogc:def:property:CEOS:eop:PlatformID"]/sml:value	/rim:ExtrinsicObject/@id	URI	Yes	1
/sml:System/sml:classification/sml:ClassifierList/sml:classifier/sml:Term[@definition=” urn:ogc:def:property:OGC:platformType”]	Add a Classification to the Corresponding ClassificationNode of the “SystemType” ClassificationScheme.	Classification	Yes	0..N
/sml:System/sml:classification/sml:ClassifierList/sml:classifier/sml:Term[@definition=” urn:ogc:def:property:OGC:orbitType”]	Add a Classification to the Corresponding ClassificationNode of the “OrbitType” ClassificationScheme.	Classification	Yes	0..N

B.6.2 Instrument mapping

Table 40 – Instrument Correspondence

SensorML Metadata Xpath	System ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
“urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System” (fixed value)	/rim:ExtrinsicObject/@objectType	/	Yes	1
/sml:System/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition="urn:ogc:def:property:CEOS:eop:InstrumentID"]/sml:value	/rim:ExtrinsicObject/@id	URI	Yes	1

SensorML Metadata Xpath	System ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:QuantityRange[@definition=”urn:ogc:def:property:CEOS:eop:AcrossTrackPointingRange”]/swe:value	rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor:: AcrossTrackPointingLowerBound”]/rim:ValueList/rim:Value[1]	Double	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:QuantityRange[@definition=”urn:ogc:def:property:CEOS:eop:AcrossTrackPointingRange”]/swe:value	rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor:: AcrossTrackPointingUpperBound”]/rim:ValueList/rim:Value[1]	Double	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:QuantityRange[@definition=”urn:ogc:def:property:CEOS:eop: AlongTrackPointingRange”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor:: AlongTrackPointingLowerBound”]/rim:ValueList/rim:Value[1]	Double	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:QuantityRange[@definition=”urn:ogc:def:property:CEOS:eop: AlongTrackPointingRange”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor:: AlongTrackPointingUpperBound”]/rim:ValueList/rim:Value[1]	Double	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:QuantityRange[@definition=”urn:ogc:def:property:CEOS:eop: NadirSwathWidth”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor:: NadirSwathWidth”]/rim:ValueList/rim:Value[1]	Double	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:QuantityRange[@definition=”urn:ogc:def:property:OGC:mass”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor:: Mass”]/rim:ValueList/rim:Value[1]	Double	Yes	0..1
/sml:System/sml:classification/sml:ClassifierList/sml:classifier/sml:Term[@definition=”	Add a Classification to the Corresponding ClassificationNode of the “SystemType” ClassificationScheme			

SensorML Metadata Xpath	System ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
urn:ogc:def:property:OGC:sensorType”]				

B.6.3 Instrument Mode mapping

Table 41 – InstrumentMode Correspondence

SensorML Metadata Xpath	System ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
“urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System” (fixed value)	/rim:ExtrinsicObject/@objectType	/	Yes	1
/sml:System/sml:identification/sml:IdentifierList/sml:identifier/sml:Term[@definition=" urn:ogc:def:property:CEOS:eop:InstrumentMode"]/sml:value	/rim:ExtrinsicObject/@id	URI	Yes	1
	Add a Classification to the “InstrumentModeType” ClassificationNode of the “SystemType” ClassificationScheme			

B.6.4 Detector mapping

Table 42 – Detector Correspondence

SensorML Metadata Xpath	Component ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
“urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System” (fixed value)	/rim:ExtrinsicObject/@objectType	/	Yes	1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:Quantity[@definition=”urn:ogc:def:property:CEOS:eop:AcrossTrackGroundResolution”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::	Integer	Yes	0..1

SensorML Metadata Xpath	Component ExtrinsicObject Attribute	(Slot) Type	Queryable	Cardinality
	AcrossTrackGroundResolution”]/rim:ValueList/rim:Value[1]			
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:Quantity [@definition=”urn:ogc:def:property:CEOS:eop:AlongTrackGroundResolution”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor:: AlongTrackGroundResolution”]/rim:ValueList/rim:Value[1]	Integer	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:Count[@definition=”urn:ogc:def:property:CEOS:opt:NumberOfSamples”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::NumberOfSamples”]/rim:ValueList/rim:Value[1]	Integer	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:Category[@definition=”urn:ogc:def:property:CEOS:eop:BandType”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::BandType”]/rim:ValueList/rim:Value[1]	String	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe: QuantityRange[@definition=”urn:ogc:def:property:CEOS:opt:SpectralRange”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::SpectralRange”]/rim:ValueList/rim:Value[1]	Integer	Yes	0..1
/sml:System/sml:characteristics/swe:DataRecord/swe:field/swe:Quantity[@definition=”urn:ogc:def:property:CEOS:eop:SNR”]/swe:value	/rim:ExtrinsicObject[@objectType=”urn:ogc:def:objectType:OGC-CSW-ebRIM-Sensor::System”]/rim:Slot[@name=” urn:ogc:def:slot:OGC-CSW-ebRIM-Sensor::SNR”]/rim:ValueList/rim:Value[1]	Integer	Yes	0..1
	Add a Classification to the “DetectorType” ClassificationNode of the “SystemType” ClassificationScheme			