NATIONAL GEOSPATIAL-INTELLIGENCE AGENCY

NGA.SP.0009.02_1.0.1_SIFTENT
2019-08-02

National System for Geospatial Intelligence (NSG) and United States MASINT System (USMS) Sensor Integration Framework (SIF) Standards Profile (SP) Technical View 1 - Enterprise

(2019-08-02)

Version 1.0.1

NATIONAL CENTER FOR GEOSPATIAL INTELLIGENCE STANDARDS

Change Log

Version	Approval Date	POC	Change Description
1.0.0	12/14/17	C. Heazel	GWS/GWG Review Draft
1.0.1	08/02/19	C, Heazel	Updated references to Ontology and UML, clarified conformance classes.

Forward

Sensing systems come in many shapes and sizes. From complex space-based telescopes which measure the background radiation of the Universe, to disposable stick-on thermometers. Likewise the degree of access to sensing systems varies widely. From direct connections to the high-speed Internet to hanging off the end of a low-speed, low quality, intermittent communications link. Yet, it is highly desirable that any authorized user should have access to any sensor and the data that it produces, from anywhere, and at any time. Clearly there is no single suite of technology which can do that. Likewise there is no single set of standards which can support that goal. This is the problem that the Sensor Integration Framework Standards Profile (SIF-SP) attempts to solve.

The SIF-SP establishes an architecture framework to decompose sensing systems into their constituent parts, and identify standards suitable for each of those parts. This framework is defined at two levels. The Reference View (RV) provides an abstract architecture framework. This level is agnostic to any specific technology. It captures the essence of what a sensor system needs to do regardless of how it is implemented and what domain it targets. Technical Views (TV) apply the Reference View architecture framework to a specific technology environment. TVs not only provide specific instruction on how to implement the SIF using a specific technology, they also specify how that implementation maps back into the Reference View. By tracing every Technical View back to the Reference View, the ability to achieve interoperability across technology environments is greatly enhanced.

This specification is the Technical View #1 of SIF-SP. It describes the implementation of the SIF architecture within the constraints of the broader Internet-based Enterprise environment.

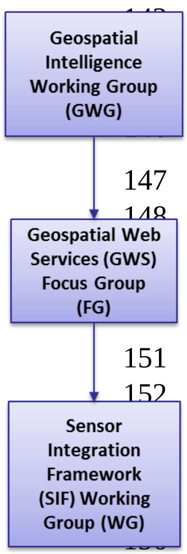
Table of Contents

1 Introduction

1.1 Background

The purpose of this document is to provide guidance required for sensor data producers and consumers to implement a sensor information enterprise that meets operational requirements, achieves United States (U.S.) Department of Defense (DoD) and Intelligence Community (IC) Chief Information Officer (CIO) goals, and conforms to applicable policy. Additionally, this profile shall define conditions, specifically those applicable to defense computing environments limited by functional mission areas. This profile, while originating from the National Systems for Geo-Spatial Intelligence (NSG) and U.S. MASINT System (USMS) communities, is designed to accommodate the broadest range of sensor information use cases possible. Sensor information implementers can expect this document to 1) identify a collection of necessary standards; 2) constrain those standards to an adequate level of detail; 3) extend those standards as needed; 4) provide overall guidance to employ those standards together.

1.2 Scope



This Sensor Integration Framework Standards Profile (SIF-SP) is produced by the Sensor Integration Framework Working Group (SIFWG) of the Geospatial Web Services (GWS) Focus Group (FG) of the Geospatial-Intelligence Standards Working Group (GWG). The GWG serves as a U.S. Department of Defense (DoD), Intelligence Community (IC), Federal, and Civil community-based forum to advocate for IT standards and standardization activities related to GEOINT. The GWG performs two major roles:

1) As a Technical Working Group (TWG) of the DoD and IC CIO Joint Enterprise Standards Committee (JESC); and

2) As a coordinating body for the GEOINT community to address all aspects of GEOINT standards.

The SIF-SP describes an architecture and standards framework for the integration of sensors and sensor systems across all deployment environments. As such, the scope of the SIF-SP is overarching among the community and reaches across multiple areas of interest horizontally rather than vertically. It provides a framework which is applicable to all sensors, regardless of the intelligence discipline. Since almost all sensors have a spatial-temporal component, the GWG was chosen as the most appropriate authority to manage this work.

The purpose of the SIF-SP is to define a framework for the integration of standards-based capabilities. This profile is built around an architecture which is representative of sensing systems and the systems that use them. Standards are then mapped onto that Architecture providing the specifications needed for implementation.

The SIF architecture is documented in two levels:

- The Reference View presents an architecture which is independent of any implementing technology, the concepts presented apply to any implementation environment.

- Technical Views present architectures within the constraints of specific technology implementations. As there are multiple environments where sensing systems are deployed, so also there are many Technical Views. Each Technical View is scoped to the technology constraints of a specific implementation environment.

This Technical View defines how the Reference View should be implemented by systems operating in the Enterprise Context.

2 Context

This Technical View defines how the SIF-SP Reference View should be implemented in the Enterprise Context. The Enterprise Context is based on the U.S. Department of Defense and Intelligence Community (DoD/IC) implementation of the Internet and World Wide Web. Three global networks have been deployed; one unclassified, one Secret, and one Top Secret. These platforms are extended by the DoD/IC communities with common services to support the Defense and Intelligence mission.

Characteristics of the enterprise environment include:

- Security. Multiple security regimes supported by globally accessible services
- Connectivity. Internet technology which is always on
- Bandwidth. Effectively unlimited
- Segmented. Global Reach
- Reliability. Low signal to noise ratios. Error detection and correction resolves most errors.
- Mobility. Environment accommodates mobile nodes, making mobility invisible to the users.

As the least constrained and most connected context of all SIF-SP Technical Views, this is the level where sensor integration comes together.

In selecting whether to adopt the enterprise technical view or a different technical view, users must assess the suitability of each for the systems to be employed. Individual parameters such as bandwidth availability will inform but not determine technical view selection and must be considered in conjunction with other parameters. For example, the bandwidth sufficient for a single sensor's feed may not be sufficient for multiple feeds or when overall network load is considered. While some studies indicate that bandwidths at or above 256 kbps can support enterprise volumes and bandwidths below 16 kbps cannot, both must be caveated based on network loading, reliability, and operational employment considerations. The ultimate selection of this technical view is a program/system decision.

All other technical views include discussions on how sensor data and information from the originating environment are integrated with the enterprise. These discussions also identify requirements for capabilities in each environment to transform data and provide a bridge for information exchanges between environments.

The SIF-SP Enterprise Technical View is the fusion of technologies from two communities. The DoD and IC communities contribute the Distributed Data Framework (DDF), Globally Unique Identifiers, and the Security Infrastructure. The Open Geospatial Consortium (OGC) contributes the Sensor Web Enablement (SWE) service and data standards as well as the OGC Web Services (OWS) platform they are built upon. Those technologies are summarized in the following sections.

2.1 Identifiers

Applicable Specifications:

- ITU-T Rec. X.667
- IETF RFC 4122
- IC.ID (GUIDE ID)
- NSG Recommended Practice for Universally Unique Identifiers

DoD and IC resources are required to have a unique identifier which is independent of hosting system and persistent for the foreseeable future. Resources are frequently moved or copied from one host to another. The identity of each resource must be preserved regardless of where or when it is accessed. Furthermore, DoD and IC resources are often classified as National Records. Under U.S. law they must be preserved for a set number of years or, in some cases, indefinitely. Therefore, compliance with the DoD and IC standards for identifiers is a requirement for all SIF resources.

2.2 Security Infrastructure

Applicable Specifications:

- ARH.XML (Access Rights and Handling)
- ISM.XML (Information Security Markings)
- ITU-T Rec. X.509 (PKI)
- NTK.XML (Need to Know)
- UIAS (Security Attributes)
- LDAP (Attribute Authority)
- XACML (Security Policy Language)

The security infrastructure for the DoD and IC Enterprise consists of Attributed Based Access Control (ABAC) controls enabled by Public Key Infrastructure (PKI) Identification and Authentication (I&A) services and standard security markings on each resource.

Unfortunately, this infrastructure is not universal. Legacy systems and policies are not always aligned with current capabilities. As a result, the enterprise is segmented into security "domains." Systems within a domain are accredited to interoperate with each other but not with systems outside of the domain. For example, systems on a secret domain cannot interoperate with those on an unclassified domain. Information operations between domains are enabled by Cross-Domain Guards. Cross-Domain Guards apply release rules, and in some cases human review, to determine if a file meets the criteria for release to the target domain.

2.3 Distributed Data Framework

Applicable Specifications:

- Distributed Data Framework documentation page at http://www.codice.org/ddf/documentation.htm

The Distributed Data Framework (DDF) is a free and open-source common data layer that abstracts services and business logic from the underlying data structures to enable rapid integration of new data sources. It forms the common software base for Distributed Common Ground System (DCGS) family of systems. DCGS in turn forms the backbone of the Defense Intelligence Information Enterprise (DI2E). DI2E is the DoD enterprise for information sharing. The DDF architecture is illustrated in Figure 1. A detailed description of the DDF is provided in Annex D.

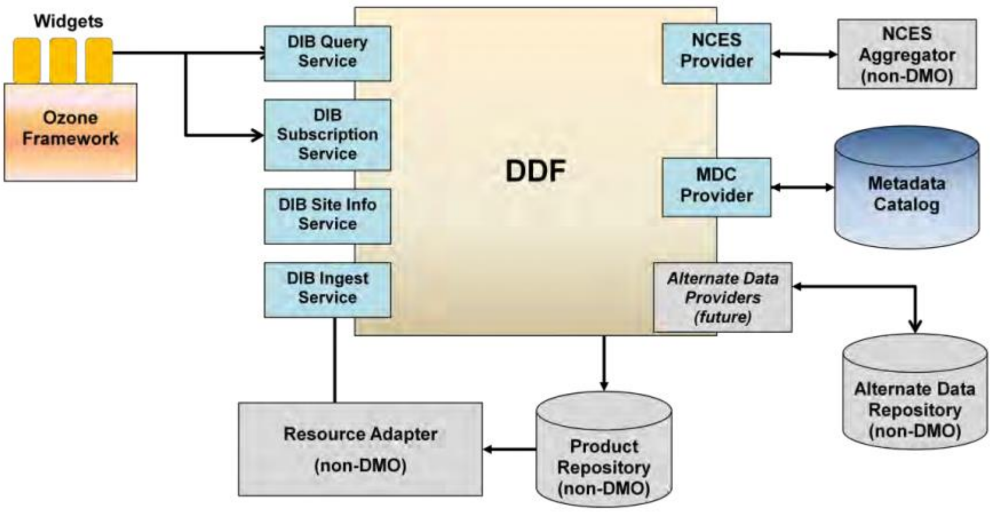


Figure 1 : DDF Architecture

The primary purpose of the DDF is to provide a federated discovery capability for all DCGS resources. This Catalog Framework is illustrated in Figure 2.

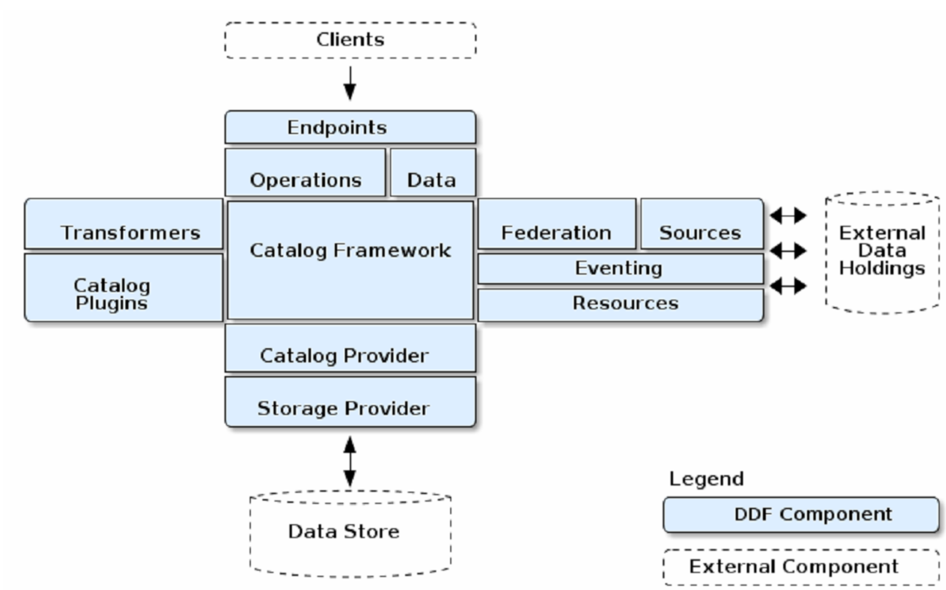


Figure 2 : DDF Catalog Architecture

2.3.1 Metadata

In the DDF, resources are the data products, files, reports, or documents of interest to users of the system. Metadata is information about those resources, organized into a schema to make search possible. The Catalog stores this metadata and allows access to it. Metacards are single instances of metadata, representing a single resource, in the Catalog. Metacards follow one of several schemas to ensure reliable, accurate, and complete metadata. Essentially, metacards function as containers of metadata.

2.3.2 Search

DDF provides the capability to search the Catalog for metadata. These capabilities differ both in the protocol used and in the selection criteria supported. The two primary protocols are OGC Catalog Service for the Web (CSW) and Content Discovery and Retrieval (CDR). CSW is an industry standard developed by the Open Geospatial Consortium. DDF implements CSW 2.0 with a few DDF specific enhancements. CDR is a version of OpenSearch which has been profiled for use by the IC and DoD. Both of these profiles support multiple filtering languages. Most important is their support for the OGC Filter Encoding language and the OGC Common Query Language. These languages support the spatial and temporal selection constructs necessary to work in a Common Operational Picture (COP) environment.

2.3.3 Ingest

Ingest is the process of bringing data products, metadata, or both into the catalog to enable search, sharing, and discovery. Ingested files are transformed into a neutral format that can be searched against as well as migrated to other formats and systems. There are multiple ways to submit data for ingest into the DDF. In all cases, the task of converting the submitted data into a metacard lies within the DDF itself.

2.3.4 Content

The Catalog Framework can interface with Storage Providers to store resources in a specific storage format, e.g., file system, relational database, XML database. Unless configured otherwise, resources are stored in a default file system within the DDF.

Storage providers act as a proxy between the Catalog Framework and the mechanism storing the content. Storage providers expose the storage mechanism to the Catalog Framework. Storage plugins provide pluggable functionality that can be executed either immediately before or immediately after content has been stored or updated.

Storage providers provide the capability to the Catalog Framework to create, read, update, and delete content in the content repository.

2.3.5 Federation

Federation is the ability of the DDF to query other data sources, including other DDFs. By default, the DDF is able to federate using OpenSearch and CSW protocols. The minimum configuration necessary to configure those federations is to supply a query address.

2.4 SWE Common

Applicable Specifications:

- SWE Common 2.0

The primary focus of the SWE Common Data Model is to define and package sensor-related data in a self-describing and semantically enabled way. The main objective is to achieve interoperability, first at the syntactic level, and later at the semantic level (by using ontologies and probably semantic mediation) so that sensor data can be better understood by machines, processed automatically in complex workflows and easily shared between nodes.

Most SWE Common data components are descended from the AbstractDataComponent class illustrated in Figure 3. Subclasses of AbstractDataComponent specify the syntax of the data represented. The “definition” element is an identifier or reference to the definition of an instance of that class. It serves to define the semantics of a SWE object. In the case of SIF objects, the definition should identify an element in the SIF Ontology.

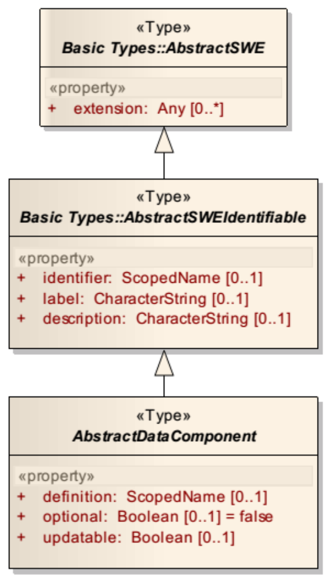


Figure 3 : SWE Common Root Classes

A detailed description of SWE Common is provided in Annex E.

2.5 OWS Service Metadata

Applicable Specifications:

- OWS Common 2.0

The OGC Web Service (OWS) Service Metadata is a set of metadata which provides a description of a specific service instance. It is typically retrieved through the GetCapabilities() operation prior to using the service. A GetCapabilities() request only returns metadata for the service which processed the request.

Service Metadata is a collection of metadata documents (Figure 4). Clients can retrieve the full set or select individual documents for download. There are four documents which all OGC Services are required to provide:

- **Service Identification:** The ServiceIdentification metadata document contains basic metadata about this specific server. Including the service type and profile, access constraints, and fees.

- **Service Provider:** The ServiceProvider metadata document contains metadata about the organization operating this server. This includes points of contact, web pages, e-mail addresses.
- **Operations Metadata:** The OperationsMetadata metadata document contains metadata about the operations provided by this service and implemented by this server, including the URLs for operation requests.
- **Contents:** The Contents metadata document describes the data offered by the service.

The Service Metadata can be extended to address the needs of a specific service type. Two extensions relevant to the SIF are the Sensor Offering metadata and the Observation Offering metadata. Sensor Offerings are provided through the Service Metadata provided by the Sensor Planning Service. Observation Offerings are provided through the Service Metadata provided by the Sensor Observation Service.

An additional metadata document which is relevant to the SIF is the Filter Capabilities. The FilterCapabilties metadata document contains information about the resource selection filters supported by the services query language. The Sensor Observation Service supports Filter Capabilities metadata as an optional component of the Service Metadata.

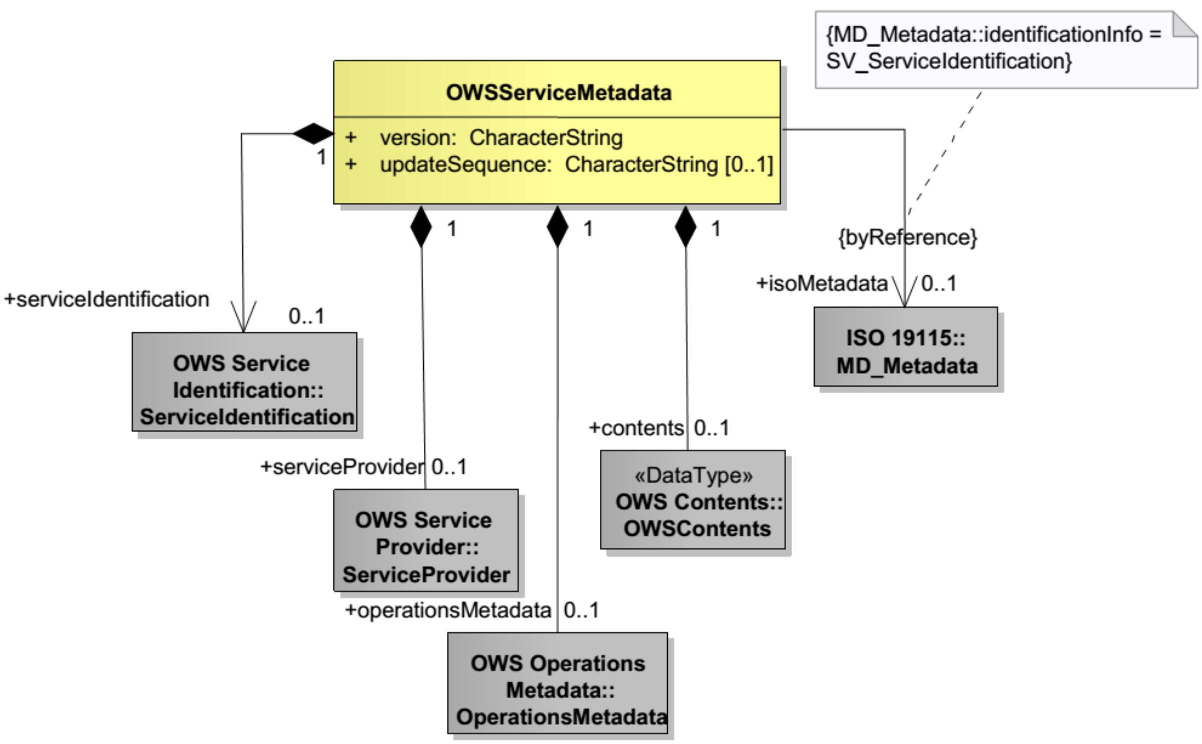


Figure 4 : Service Metadata

2.6 Sensor Observation Service

Applicable Specifications:

- OGC Sensor Observation Service Version 2.0

The OGC Sensor Observation Service (SOS) Implementation Standard defines a web service interface for discovery and retrieval of observations and sensor system information. Observations may be from in-situ sensors (e.g., water monitoring devices) or dynamic sensors (e.g., imagers on Earth-observation satellites). The Sensor Observation Service standard defines a core capability and three optional extensions (see Figure 5).

Limited integration of the SOS with the DDF/DIB has been demonstrated at ENTERPRISE CHALLENGE 2016 and 2017. The SOS published sensor description metacards to the DDF/DIB. Once those sensor metacards were retrieved, users connected directly to the corresponding SOS to receive observations from selected sensors.

An operational implementation of the SOS should incorporate the SOS core, Transactional, and Result Handling extensions.

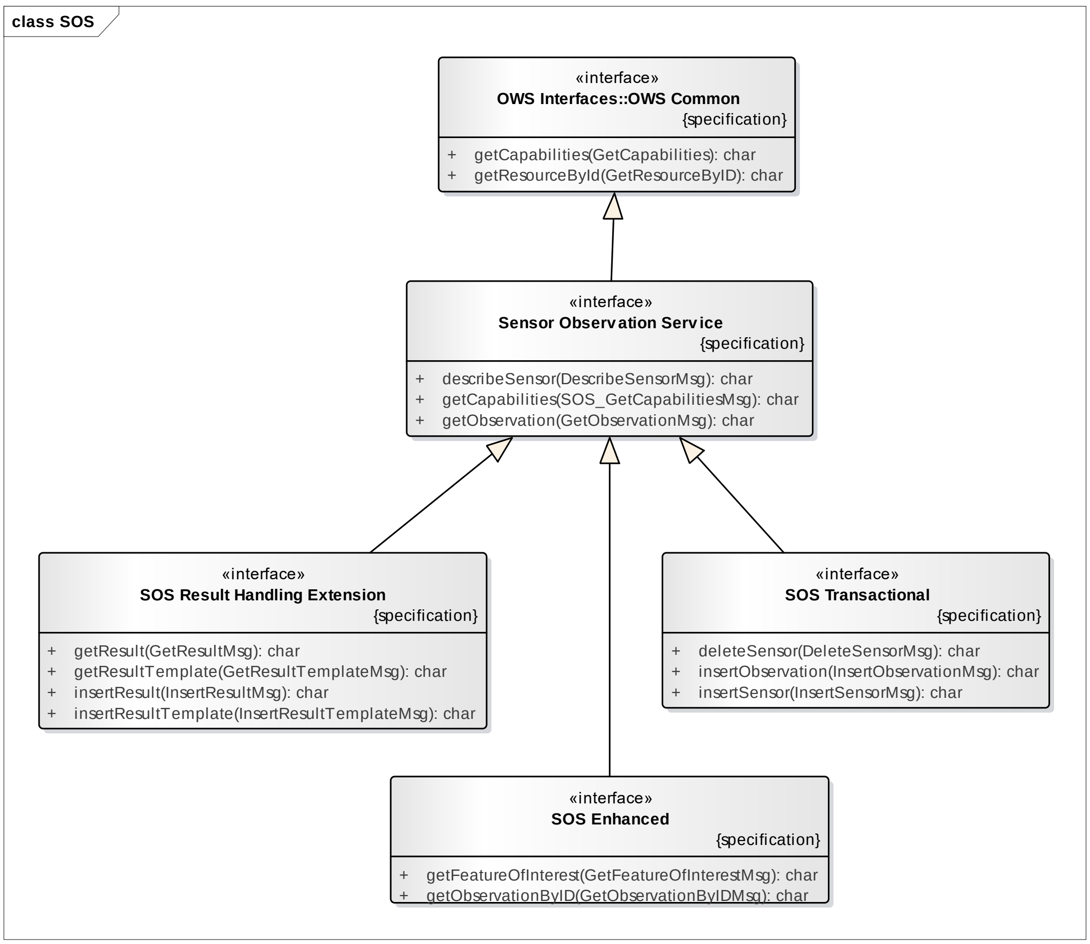


Figure 5 : Sensor Observation Service

2.7 Sensor Planning Service

Applicable Specifications:

- OGC Sensor Planning Service Version 2.0

The OGC Sensor Planning Service (SPS) Implementation Standard defines an interface to task sensors or models. Using SPS, sensors can be reprogrammed or calibrated, sensor missions can be started or changed, simulation models executed and controlled. The feasibility of a tasking request can be checked and alternatives may be provided. The OGC SPS Earth Observation Satellite Tasking Extension supports the programming process of Earth Observation (EO) sensor systems used by many satellite data providers.

3 Conformance

The SIF-SP Reference View provides an architecture framework which is agnostic to the implementing technology. This SIF Technical View extends that architecture within the technical constraints of the DoD and IC enterprise networking environment as supported by the DCGS Family of Services and industry standard Web Services. Conformance with this Technical View is defined by two conformance classes.

- **Sensor Web Enablement (SWE).** The first conformance class defines conformance with the OGC Sensor Web Enablement (SWE) body of standards. Conformance to this class is required of any Enterprise node which proposes to expose sensors and sensor data in accordance with the SIF-SP.
- **Distributed Data Framework (DDF).** The second conformance class defines conformance with the technology infrastructure as defined by the Distributed Data Framework (DDF). This is required of any implementation which will be deployed as a node on the DCGS Enterprise.

Details on the requirements and corresponding abstract tests are provided in Annex A

4 Related Specifications

4.1 Normative Specifications

- ASPRS, LAS Specification Version 1.4-R13, 15 July 2013
- CIPA DC-008-2016, Exchangeable image file format for digital still cameras: Exif Version 2.3.1, July 2016
- IETF RFC 4122, A Universally Unique IDentifier (UUID) URN Namespace, July 2005
- ISO/IEC 15444-1:2016, Information technology -- JPEG 2000 image coding system: Core coding system, October 2016
- ISO/IEC 15444-2:2004, Information technology -- JPEG 2000 image coding system: Extensions, May 2004
- ISO/IEC 15948:2004, Information technology -- Computer graphics and image processing -- Portable Network Graphics (PNG): Functional specification, March 2004
- ITU-T Rec. X.667, Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: Generation and registration of Universally Unique Identifiers (UUIDs) and their use as ASN.1 object identifier components, September 2004
- ITU-T T.808, JPEG 2000 Interactive Protocol (Part 9 – JPIP), January 2005
- MIL-STD-2500C, National Imagery Transmission Format (Version 2.1), 01 May 2006

- MIL-STD-2500C, National Imagery Transmission Format (Version 2.1) Change Notice (CN) 1, 01 February 2017
- MISP-2019.1, Motion Imagery Standards Profile (MISP), November 2018
- NSG.RP.0001, National System for Geospatial Intelligence (NSG) Recommended Practice for Universally Unique Identifiers, 3 January 2013
- NGA.SP.0009.01_1.0.1_SIFR, National System for Geospatial Intelligence (NSG) Sensor Integration Framework Standards Profile (SIF-SP) Reference View, 2 August 2019
- ODNI IC.ID.V1, Intelligence Community Identifier (GUIDE ID) v1, 10 April 2013
- OGC 07-036, OGC Geography Markup Language v3.2 (also published as ISO 19136:2007, Geographic information — Geography Markup Language), 27 August 2007
- OGC 10-004, OGC Observations and Measurements v2.0 (also published as ISO/DIS 19156:2010, Geographic information — Observations and Measurements), 17 September 2013
- OGC 12-000, OGC® SensorML: Model and XML Encoding Standard v2.0, 4 February 2014
- OGC 12-006, OGC® Sensor Observation Service Interface Standard v2.0, 16 April 2012
- OGC 09-000, OGC® Sensor Planning Service Implementation Standard v2.0, 28 March 2011
- OGC 08-094, OGC® SWE Common Data Model Encoding Standard v2.0, 4 January 2011
- OGC 06-121, OGC® Web Services Common Implementation Specification v2.0, 7 April 2010
- OGC 09-001, OpenGIS® SWE Service Model Implementation Standard v2.0, 21 March 2011
- OSGeo, GeoTIFF Format Specification, Version 1.8.2, Revision 1.0, 10 November 1995
- SIF-SP Ontology, https://github.com/ngageoint/Sensor_Integration_Framework
- SIF-SP UML Model, https://github.com/ngageoint/Sensor_Integration_Framework

4.2 Informative Specifications

- CMSTT, Department of Defense Discovery Metadata Specification (DDMS) 2.0, 17 July 2008
- DDF Documentation is at http://codice.org/ddf/documentation.html
- IETF RFC 4511, Lightweight Directory Access Protocol (LDAP): The Protocol, June 2006
- ITU-T Rec X.509, Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks, 14 October 2016
- OASIS, eXtensible Access Control Markup Language (XACML) Version 3.0, 22 January 2013
- ODNI, IC/DoD Content Discovery & Retrieval Specification Framework v 2.0, 10 April 2013
- ODNI, IC/DoD REST Interface Encoding Specification for CDR Search v 3.0, 3 October 2012
- ODNI, IC/DoD SOAP Interface Encoding Specification for CDR Search v 3.0, 3 October 2012
- ODNI, IC/DoD REST Interface Encoding Specification for CDR Brokered Search v 2.0, 6 September 2013

- ODNI, IC/DoD SOAP Interface Encoding Specification for CDR Brokered Search v 2.0, 6 September 2013
- ODNI ARH.XML.V3, XML Data Encoding Specification for Access Rights and Handling v3, 6 September 2013
- ODNI IC.ISM.V13, Information Security Marking Metadata v13, 9 May 2014
- ODNI NTK.XML.V10, XML Data Encoding Specification for Need To Know Metadata v10, 6 September 2013
- ODNI UIAS, IC Enterprise Attribute Exchange Between IC Services Unified Identity Attribute Set v 3.1, 9 May 2014
- OGC 07-006, OpenGIS® Catalogue Services Specification v 2.0, 23 February 2007
- OGC 09-026, OpenGIS® Filter Encoding v2.0 (also published as ISO/DIS 19143:2010, Geographic information — Filter Encoding), 22 November 2010
- OpenSearch.org, OpenSearch v1.1, retrieved from http://www.opensearch.org on 20 September 2017

5 Terms and Definitions

The SIF-SP Terms and Definitions can be found in **Error! Reference source not found.**.

6 Abbreviations

The SIF-SP list of abbreviations can be found in **Error! Reference source not found.**.

7 Information Viewpoint

The information viewpoint describes information elements that are exchanged and processed as described in the Computational viewpoint (Section 8).

7.1 Descriptions

Descriptions are information or metadata that describe a resource, an observable, and a performer or process.

Description	**Definition**
Resource	any addressable unit of information or service [IETF RFC 2396] EXAMPLES Examples include files, images, documents, programs, and query results. NOTE The means used for addressing a resource is a URI (Uniform Resource Identifier) reference
Observable	A parameter or a characteristic of a phenomenon subject to observation. Synonym for determinand.[O&M] A physical property of a phenomenon that can be observed and measured (e.g. temperature, gravitational force, position, chemical concentration, orientation, number-of-individuals, physical switch status, etc.), or a characteristic of one or more feature types, the value for which will be estimated by application of some procedure in an observation. It is thus a physical stimulus that can be sensed by a detector or created by an actuator.

Performer	Any entity - human, automated, or any aggregation of human and/or automated - that performs an activity and provides a capability.
Process	An operation that takes one or more inputs and, based on a set of parameters and a methodology, generates one or more outputs.

Table 1 : Described Resources

7.1.1 Resource Description

Applicable Specifications:

- DDMS 2.0
- SensorML 2.0 to DDMS 2.0 Data Mapping
- Observations and Measurements 2.0 to DDMS 2.0 Data Mapping
- SOS Service Metadata 2.0 to DDMS 2.0 Data Mapping Version 1.0

A Resource Description is a standard metadata record which can be used to describe any resource. As such, it is limited to general descriptive concepts such as contact information, resource type, area of coverage, etc. The standard governing Resource Descriptions in the DoD/IC Enterprise is the Defense Discovery Metadata Standard (DDMS). DDMS 2.0 is the most commonly used version of the DDMS standard although version 5.0 is gaining acceptance. This version of the SIF-SP will only address DDMS 2.0.

Queries against DDMS are the first step in finding DoD enterprise resources. It has some elements which are specific to a single resource type, but on the whole it is independent of the resource type.

SIF implementations are expected to generate DDMS metadata records as follows:

	Generate DDMS	In accordance with
For Sensors, processes, and systems	From SensorML documents	SensorML 2.0 to DDMS 2.0 Data Mapping Version 1.0
For Observations	From O&M Observations	Observations and Measurements 2.0 to DDMS 2.0 Data Mapping Version 1.0
For Observables	From Offerings elements extracted from the Service Metadata	SOS Service Metadata 2.0 to DDMS 2.0 Data Mapping Version 1.0

Table 2 : DDMS Generation Rules

7.1.2 Observable Description

Applicable Specifications:

- Sensor Observation Service 2.0

The SOS Service Metadata includes a set of Observation Offering elements. An Observation Offering describes a set of Observations that may be or have been generated through a Process. An Observation Offering is scoped by four properties:

- Who/What generates Observations and how (e.g. a sensor or processor)
- Which Observable Properties can be measured, (e.g. color, temperature, humidity)

- Which Features can be observed, (e.g. the color of a car, the temperature of an engine, the humidity of the air in a room)
- Where (spatial-temporal boundaries) the observations take place.

Initially this information will describe the planned observation events (mission). As Observations are collected, the Observation Offering is updated to reflect the new information. The rules for populating an Observation Offering are as follows:

An Observable Description shall:

- Have a unique identifier (Section 2.1)
- Identify the system and process used to generate observations
- Identify the types of target which may be collected against
- Identify the property types which may be collected against each target type
- Identify the spatial-temporal extent over which collections may take place.

The mapping from the Reference View to Observation Offerings is provided in Table 3.

RV		TV1		
Observable Description		Observation Offering		
Element	#	Element	#	Comments
Identifier	1..1	identifier	0..1	Unique identifier for this resource. See Section 0.
		name	0..n	
Process	1..1	procedure	1..1	SWE combines the process and performer into a single component. Therefore, only one element is required.
performer	1..1			
		procedureDescription Format	0..n	Element is specific to this Technical View.
phenomenology	1..n	observableProperty	0..n	A list of the Observable Properties that have or may be observed by Observations described by this Offering.
target	0..n	relatedFeature	0..n	A list of the Features of Interest that have been observed by Observations described by this Offering.
targetType	1..n	featureOfInterestType	0..n	A list of the types of Features of Interest that may be observed by Observations described by this Offering.
footprint	1..1	observedArea	0..1	These three elements specify an envelope in space and time which encompasses all of the Observations described by this offering.
phenomenonTime	1..1	phenomenonTime	0..1	
resultTime	1..1	resultTime	0..1	
observations	0..n	NA		Accessible via GetObservation() using values from the Observation Offering.
		responseFormat	0..n	Element is specific to this Technical View. It specifies the formats available for reporting Observations.
schema	0..n	observationType	0..n	See Section 2.4 for more information.
encoding	0..n			

Table 3 : Offering Description to Observation Offering Mapping

As Observations are collected, the Observable Description shall be updated as follows:

- The FeatureOfInterest of the Observation (the Target) shall be added to the relatedFeature list.
- The spatial-temporal extent shall be updated to include the new Observation.

There are a few limitations to the use of Observation Offerings.

- Discovery and Accessibility of Observation Offerings. They can only be discovered and accessed by browsing the SOS Service Metadata (Section 2.5).
- Discovery of Related Resources. The ability to browse for related resources is limited. The "procedure" element provides direct access to descriptions of the Processes and Performers used but not the Observation Descriptions themselves.
- Accessibility of Observation Descriptions. Observation descriptions are not directly accessible, they must be discovered by querying the SOS by one or more of these parameters: "procedure" element, the spatial/temporal extent, and one value from each of the "relatedFeature" and "observableProperty" elements. See Sections 8.2.2.2, and 8.2.2.3 for more information.

7.1.3 Performer and Activity Description

Applicable Specifications:

- SensorML 2.0

7.1.3.1 Activity (Process) Descriptions

At its most fundamental level, a sensor is a performer which executes an activity or process. It receives input, processes that input, and generates output. The processing is defined by an algorithm and controlled through parameters. This same definition applies to services. For this reason, the OGC SWE standards classify both sensors and services as Processes.

OGC SWE standards describe processes using the SensorML 2.0 standard. Processes can be simple or complex. SensorML supports both simple processes and aggregate (complex) processes. The UML model for SensorML processes is provided in Figure 6. SensorML elements are mapped against the Reference View Process Description in Table 4.

RV		TV1		
Activity Description		SensorML Abstract Process		
Element	**#**	**Element**	**#**	**Comments**
input	1..n	inputs	0..n	See section 7.1.3.3 for more details
output	1..n	outputs	0..n	See section 7.1.3.3 for more details
method	0..1	method	0..1	Note that Aggregate Processes do not have a method. In most cases the connections will prove adequate to model the workflow. This issue will be raised with the OGC.
parameters	0..n	parameters	0..n	See section 7.1.3.3 for more details

RV		TV1		
Activity Description		**SensorML Abstract Process**		
Element	**#**	**Element**	**#**	**Comments**
executedBy	0..n	NA	NA	Processes are associated with a Performer through inheritance. The properties of the Process become properties of the Performer.
processingInformation	1..1	multiple	0..n	Any additional information which is not an input or parameter.
hasProcessStep	0..n	Components and connections	0..n	If a process contains process steps, then it is an Aggregate Process.

Table 4 : SensorML Mapping to Activity Description

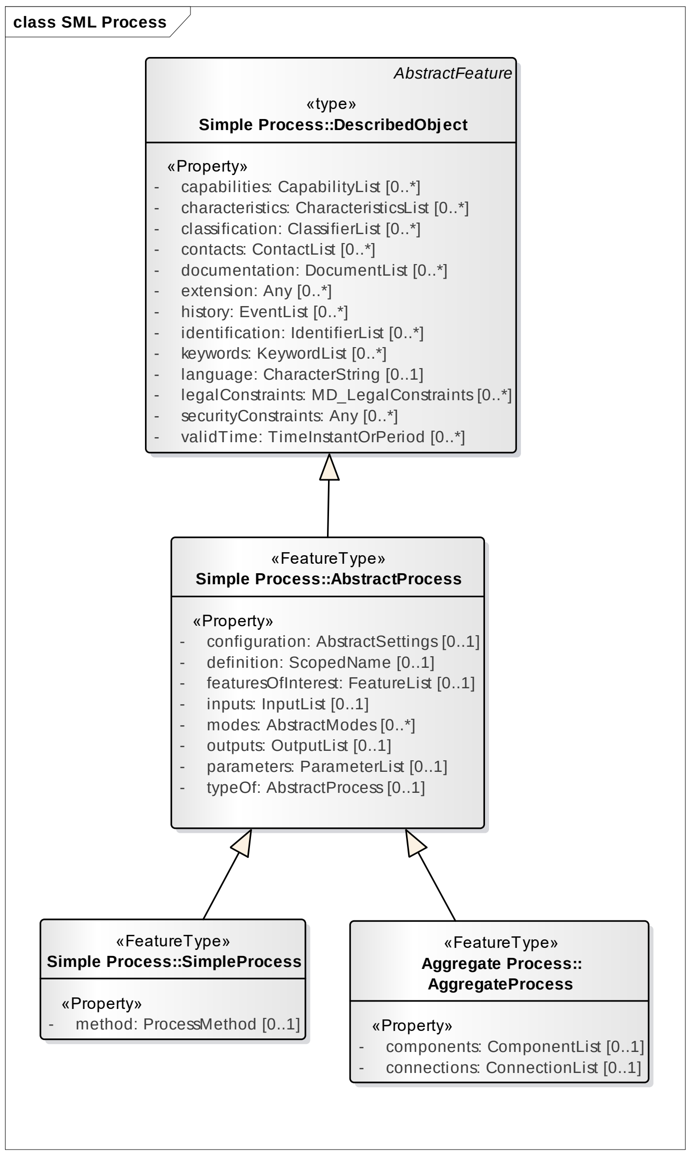


Figure 6 : SensorML Processes

7.1.3.2 Performer Description

Applicable Specifications:

- SensorML 2.0

A description of a process is incomplete without a description of the physical entity which implements that process. The SIF-SP Reference View associates a Performer with a Process. OGC SWE establishes this association by treating the Performer as a subclass of the Process. A description of a Process may also be a description of the Performer who executes the process. In this way a single Process/Performer pair is a single identifiable entity. The OGC SWE approach is illustrated in Figure 7. A mapping from the Reference View Performer Description into Physical Components and Physical Systems is provided in Table 5.

RV		TV1		
Performer Description		**SensorML Physical Component**		
Element	**#**	**Element**	**#**	**Comments**
identifier	1..1	gml:id	1..1	The SensorML XML schema defines this element as a Non-Colonized Name. As such, it cannot be a URI. SIF implementations shall populate this element with the 32-character representation of the UUID.
		identifier	0..1	Unique identifier for this resource. See Section 0.
definition	0..1	definition	0..1	Defines what this physical component is through a URI which resolves to an element in the SIF-SP ontology
observables	0..n	outputs	0..n	See Section 7.1.3.5
		featuresOfInterest	0..n	
pointOfContact	0..n	contacts	0..n	This is a list of contacts formatted in accordance with CI_ResponsibleParty from ISO 19115/NMF
IndividualName	0..1	individualName	0..1	Child elements of CI_ResponsibleParty. One of these two or positionName is required
OrganizationName	0..1	organisationName	0..1	
Phone	0..1	CI_Telephone/ voice	0..n	Descendent elements of CI_Contact
ElectronicMailAddress	0..1	CI_Address/ electronicMailAddres s	0..n	
properties	0..n	Multiple		Sensor properties map into a number of SensorML elements. A discussion of SensorML properties is provided in Section 7.1.3.3.
commands	0..n	None		No current mapping into SensorML
state	0..n	none		No current mapping into SensorML
Property States				
Observable States				
Command States				

RV		TV1		
Performer Description		**SensorML Physical Component**		
Element	**#**	**Element**	**#**	**Comments**
hasComponent	0..n	attachedTo	0..1	SensorML supports navigation from the lowest level component up to the sensor system. This is opposite the direction used in the Reference View.
executes	0..n	NA	1..1	Performer is associated with Process though inheritance.

Table 5 : SensorML Mapping to Performer Description

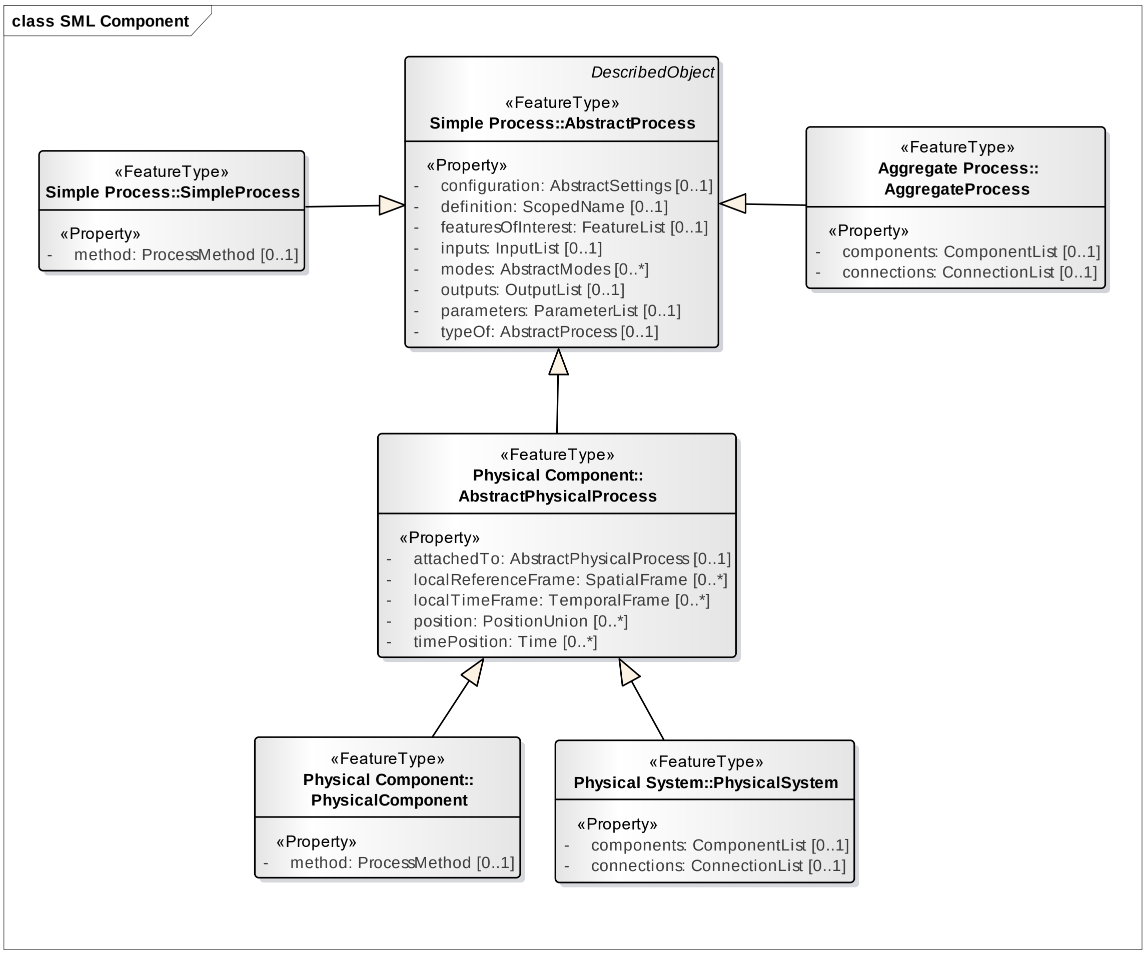


Figure 7 : SensorML Performers

7.1.3.3 Performer Properties

The Reference View provides a taxonomy of Performer Properties. The elements of this taxonomy are as follows:

Category	Description
Capabilities	Properties that further clarify or quantify the output of the process (e.g. dynamic range, sensitivity, threshold, etc.). These can assist in the discovery of processes that meet particular requirements.

Category	Description
Characteristics	Useful properties of this process that do not further qualify the output values (e.g. component dimensions, battery life, operational limits, etc.).
Configuration	Value settings that further constrain the properties of the base process.
Documentation	Additional external online documentation of relevance to this process (e.g. user's guides, product manuals, specification sheets, images, technical papers, etc.)
Identification	Identifiers useful for discovery of the process (e.g. short name, mission id, wing id, serial number, etc.)
Identifier	Often, a special identifier is assigned to an object by the maintaining authority with the intention that it is used in references to the object. For such cases, the codeSpace shall be provided. That identifier is usually unique either globally or within an application domain. gml:identifier is a pre-defined property for such identifiers.
Position	Provides positional information relating the component's spatial reference frame to an external spatial reference frame. Positional information can be given by location, by full body state, by a time-tagged trajectory, or by a measuring or computational process.
Mode	A collection of parameters that can be set at once through the selection of a particular predefined mode.
Name	The gml:name property provides a label or identifier for the object, commonly a descriptive name. An object may have several names, typically assigned by different authorities. gml:name uses the gml:CodeType content model. The authority for a name is indicated by the value of its (optional) codeSpace attribute. The name may or may not be unique, as determined by the rules of the organization responsible for the codeSpace. In common usage there will be one name per authority, so a processing application may select the name from its preferred codeSpace.
Parameters	The list of data components (and their properties and semantics) that the process will accept as parameters; In the standard linear equation y=mx+b; x is the input, m and b are the parameters, and y is the output.

Table 6 : Reference View Performer Property Taxonomy

OGC SWE and this Enterprise View do not define specific Performer Properties. Rather, the SIF-SP Ontology provides definitions for an expanding set of known Performer Properties and associates each with one of the members of the taxonomy.

7.1.3.4 Inputs, outputs, and parameters:

Processes typically receive input, then run that input through an algorithm (methodology) which is constrained by parameters, and then output the results.

Some processes, such as detectors, receive physical stimuli as input and generate digital values as output. For example, the temperature of the atmosphere is an Observable Property of a Feature of Interest (atmosphere). Before it is measured, the temperature is simply a property of the atmosphere that can be defined and measured. After measurement by a detector, the temperature may be represented as a Quantity with units of measure, a value, and an indication of our degree of confidence in the measurement.

Other processes receive digital values as both input and output. LIDAR, for example, must go through a complex processing workflow to produce useable output. Usable, however, is in the eye of the beholder.

So the LIDAR workflow has well defined points where intermediate products are made available. A user may be able to request Level 0, 1, 2, or 3 LIDAR based on the process steps they want executed on the raw data.

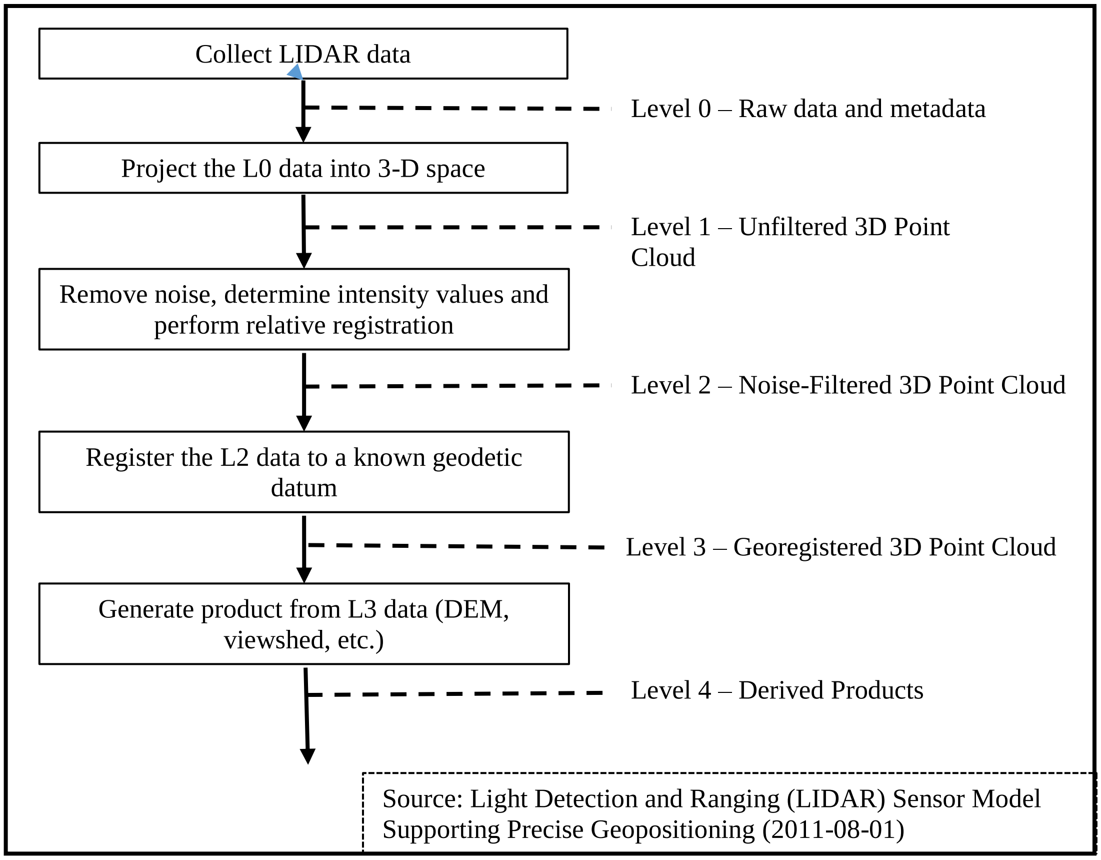


Figure 8 : LIDAR Processing Workflow

In a complex workflow, inputs, outputs, and parameters cease to be discrete entities. Outputs from one process become inputs to another. Outputs from another process may become parameters governing the processing of a third process. There are even recursive processes where the output of the process loops around to serve as input for the next iteration.

Due to this complexity, SensorML uses a single data construct, “Data Component or Observable”, to model inputs, outputs, and parameters. This is illustrated in Figure 9.

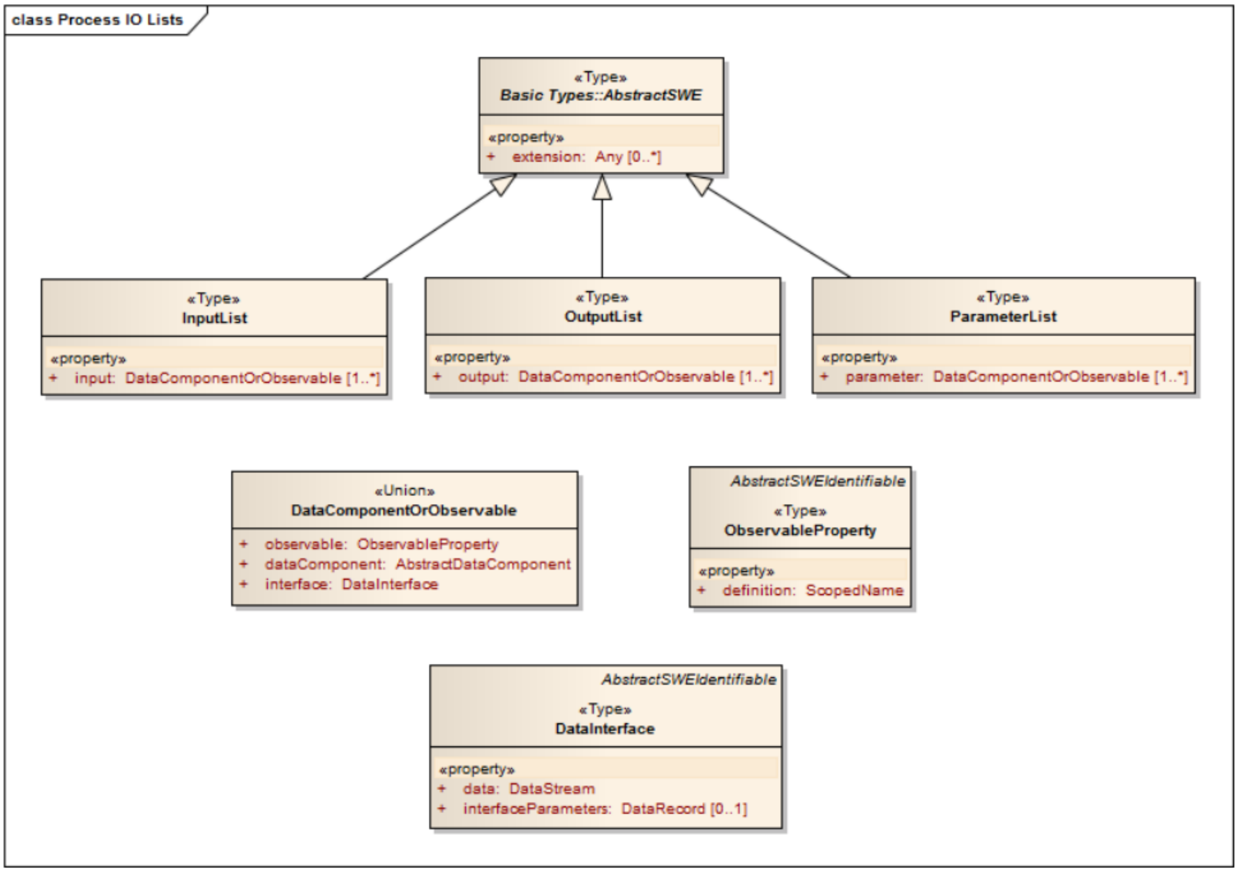


Figure 9 : Inputs, Outputs, and Parameters

The Data Component of Observable entity is a Union of three elements; an ObservableProperty, an AbstractDataComponent, and a DataInterface. As a Union, one of these elements must be populated, but only one may be populated at a time.

7.1.3.4.1 Abstract Data Component

Abstract Data Component is the parent class for SWE Common data. Inputs, outputs, and parameters based on Abstract Data Component are data values formatted in accordance with the OGC SWE Common standard. SWE Common data formats are described in Section Section 2.4

7.1.3.4.2 Observable Property

An ObservableProperty is a physical property of a phenomenon (Feature of Interest) that can be observed and measured (e.g. temperature, gravitational force, position, chemical concentration, orientation, number-of-individuals, physical switch status, etc.), or a characteristic of one or more feature types, the value for which will be estimated by application of some procedure in an observation.

An ObservableProperty is metadata. It describes a property but it cannot convey a measured value for that property.

ObservableProperty is encoded using SWE Common. Its elements are described in Table 7. Note that this data structure has little practical value without an accompanying ontology.

Element	#	Description
identifier	0..1	ObservableProperties are discretely managed and accessible resources. Therefore, the identifier property is required for SIF implementations. See Section 2.1.
label	0..1	Not required
description	0..1	Not required
definition	1..1	A reference to the concept in the SIF-SP Ontology which describes this property.

Table 7 : Observable Properties

7.1.3.4.3 DataInterface

Input, output, and parameters are not always discrete entities. The DataInterface entity provides a description of measured values which are provided by a service endpoint such as a live Motion Imagery feed.

A DataInterface is metadata. It describes the service endpoint where the measured values are available as well as the information necessary to use that endpoint. The measured values themselves are delivered by the service endpoint.

Element	#	Description
identifier	0..1	Data Interfaces are discretely managed and accessible resources. Therefore, the identifier property is required for SIF implementations. See Section 2.1.
label	0..1	Not required
description	0..1	Not required
data	1..1	A SWE Common Data Stream describing the service endpoint.
interfaceParameters	0..1	Additional parameters to use with the service endpoint.

Table 8 : Data Interfaces

7.1.3.5 SensorML and Observation Offerings

The SIF-SP Reference View describes a navigable link between the Performer Description and the Observables which that Performer can deliver. SWE does not provide a similar association between Physical Processes and Observation Offerings. Therefore, a client who wishes to identify the offerings supported by a sensor system must first access the SensorML document, then access and browse the Observation Offerings. SensorML and Observation Offering elements correlate as described in Table 9.

SensorML	#	ObservationOffering	#	Comments
identifier	1..1	procedure	1..1	1 to 1 correlation between identifiers
outputs	1..n	observableProperty	0..n	The ObservationOffering provides a URI to a GFI_PropertyType. This should be the same as or mapable to the SensorML entity.
featuresOfInterest	0..n	realtedFeature	0..n	FeaturesOfInterest and relatedFeature correlate. FeatureOfInterestType can be derived from the feature types represented in the FeaturesOfInterest.
		featureOfInterestType	0..n	
position	0..n	observedArea	0..1	Geometry filters should correlate
tiimePosition	0..n	PhenomenonTime	0..1	Temporal filters should correlate

Table 9 : Correlating SensorML and ObservationOfferings

7.1.3.6 SensorML Ontology Mapping

SensorML is implemented using SWE Common. Therefore, the use of SensorML within the SIF context is best explained by mapping SensorML elements into their corresponding concepts in the SIF-SP Ontology. SensorML is a family of four different document types. All four types are an extension of SensorML Abstract Process. A mapping of SensorML Abstract Process into the SIF-SP Ontology is provided in Table 10.

SensorML Abstract Process		SIF-SP Ontology / Performer Description	
Element	**Description**	**Concept**	**Comments**
metaDataProperty	Supporting metadata. No format specified	none	Element has no standardized semantics
Description	String	none	Optional element not required for SIF implementations
descriptionReference	Reference to a Description which is not a part of the SensorML document	none	Optional element not required for SIF implementations
Identifier	ScopedName	Identifier	
name	String	Name	
boundedBy	A bounding geometry	Envelope	A GeographicPolygon and optional TemporalLocation
location	Location expressed as geometry, name, or keyword.	PositionProperty	An unconstrained GeographicShape.
language	String	none	
Keywords	Each element is a scope and set of keywords.	none	
Identification	Each element is a set of terms	ComponentIdentification	Any element within the ComponentIdentification scope is valid.
Classification	Each element is a set of terms useful for discovery. Examples include; sensorType, processType, and intendedApplication.	none	
validTime	Time during which this document is valid	Envelope	The temporal component of the envelope.
securityConstraints		SecurityProperties	
legalConstraints		none	
Characteristics	Each element is a list of AbstractDataComponent	ComponentCharacteristics	Any element within the ComponentCharacteristics scope is valid with the

SensorML Abstract Process		SIF-SP Ontology / Performer Description	
Element	**Description**	**Concept**	**Comments**
			exception of RelationshipProperty
Capabilities	Each element is a list of AbstractDataComponent	ComponentCapabilities	Any element within the ComponentCapabilities scope is valid
Contacts	Each element is a list of CI_ResponsibleParty	ResourcesProperty	Optional element not required for SIF implementations
Documentation	Each element is a list of CI_OnlineResource	ResourcesProperty	Optional element not required for SIF implementations
History	Each element is a list of Events. Events might, specify calibration or maintenance history of a sensor, changes to an algorithm or parameter within a computational process, or deployment and maintenance events.	none	Optional element not required for SIF implementations
definition	Pointer into an ontology	ComponentType	Any element within the ComponentType scope is valid.
typeOf	AbstractProcess	none	
Configuration	AbstractSettings	ComponentConfiguration	Any element within the ComponentConfiguration scope is valid.
featuresOfInterest	Each element is a list of GFI_Feature	ComponentObservables	
Inputs	(list) Name AND (AbstractDataComponent OR ObservableProperty OR DataInterface)	DataInputsProperty	
Outputs	(list) Name AND (AbstractDataComponent OR ObservableProperty OR DataInterface)	DataOutputsProperty	
Parameters	(list) Name AND (AbstractDataComponent OR ObservableProperty OR DataInterface)	ComponentParameters	Any element within the ComponentParameters scope is valid.
modes	AbstractModes	none	

Table 10 : SensorML Ontology Mapping – Abstract Process

SensorML Physical Components and Physical Systems are extensions to the Abstract Process mapped in Table 10. These components share many common elements. These elements are defined in the Abstract Physical Process metadata. The Abstract Physical Process is mapped to the SIF-SP Ontology in Table 11.

SensorML 2.0 Abstract Physical Process		SIF-SP Ontology	
Element	**Description**	**Concept**	**Process Description**
AbstractProcess	See Table 10		
AbstractPhysicalProcess			
attachedTo	The attachedTo property provides a reference from the attached process to the process to which it is attached.	AttachedTo	
localReferenceFrame	A definition of direct orthogonal (i.e. Cartesian) reference frames that are assumed to be attached to the physical component where they are described	none	
localTimeFrame	Temporal reference frames can include a particular calendar, a particular time of day reference frame, or a frame attached to an event of interest.	none	
Position	The location, position, or dynamic state of an object. (includes the external special reference system)	ComponentLocation	Any element within the ComponentLocation scope is valid
timePosition	Date and time object was at the position.	TemporalPositionProperty	

Table 11 : SensorML Ontology Mapping – Abstract Physical Process

SensorML Physical Component and Physical System are extensions to the Abstract Physical Process mapped in Table 11. These components are mapped to the SIF-SP Ontology in Table 12.

SensorML 2.0 Components and Systems		SIF-SP Ontology	
Element	**Description**	**Concept**	**Process Description**
AbstractPhysicalProcess	See Table 11		
PhysicalComponent	Any processing device which provides a processing function with well-defined inputs and outputs, if there is no intent to further divide the device description into component sub-processes, and if knowledge of its physical location is useful.		

SensorML 2.0 Components and Systems		SIF-SP Ontology	
Element	Description	Concept	Process Description
method	The Method element provides a description of the methodology used by the process to execute and generate output based on the input and parameter values.	PerformerComponent	Any element within the PerformerComponent scope is valid
PhysicalSystem	An aggregate model of a group or array of process components, which can include detectors, actuators, or sub-systems.		
components	A list of components. A component is any concept derived from AbstractProcess	RelationshipProperty	
connections	A list of tuple consisting of the identifiers for the source and destination components.		

Table 12 : SensorML Ontology Mapping – Systems and Components

7.2 Observables, Observations, and Measures

Observables, Observations, and Measures represent three levels of abstraction for the results of sensing and processing activities. Each level builds on the information captured by the previous levels. As a result, the boundaries between these three concepts are rather porous. Different technologies may draw the lines a little differently. But in all cases implementations of these concepts should form a coherent whole.

Term	Definition	
Observable	A parameter or a characteristic of a phenomenon subject to observation. Synonym for determinand.[O&M] A physical property of a phenomenon that can be observed and measured (e.g.) temperature, gravitational force, position, chemical concentration, orientation, number-of-individuals, physical switch status, etc.), or a characteristic of one or more feature types, the value for which will be estimated by application of some procedure in an observation. It is thus a physical stimulus that can be sensed by a detector or created by an actuator.	Example
Observation	Act of observing a property or phenomenon [ISO/DIS 19156, definition 4.10] Note: The goal of an observation may be to measure, estimate or otherwise determine the value of a property	Example
Measurement	An observation whose result is a measure [O&M]	Example

Table 13 : Observables, Observations, and Measures

7.2.1 Observables

Applicable Specifications:

- Observations and Measurements 2.0
- Sensor Observation Service 2.0

Observables report on the *types* of information that may result from a sensing procedure. In the Enterprise Technical View, they are represented by the Observation class from the Observations and Measurements standard (Figure 10). However, the O&M Observation is not an ideal fit for the Observable concept. The Sensor Observation Service addresses this through the concept of a Template (see SOM Table 36). A Template associates an Observation Offering with a decimated Observation. This decimated Observation is to serve as a template for creation of populated Observations as measurements are performed. The constraints that SOS places on the Observation Template are:

"The observation that is provided by the client in the ResultTemplate shall have null as value of om:phenomenonTime, om:resultTime and om:result. For the first two properties, the nilReason shall be set to the value 'template.' The procedure, featureOfInterest, and observedProperty of the observation template shall not be empty. Other observation properties can be set by the client." [1]

This use of O&M Observations aligns well with the SIF-SP Reference View concept. The mapping between the Reference View Observables and O&M Observations is provided in Table 14.

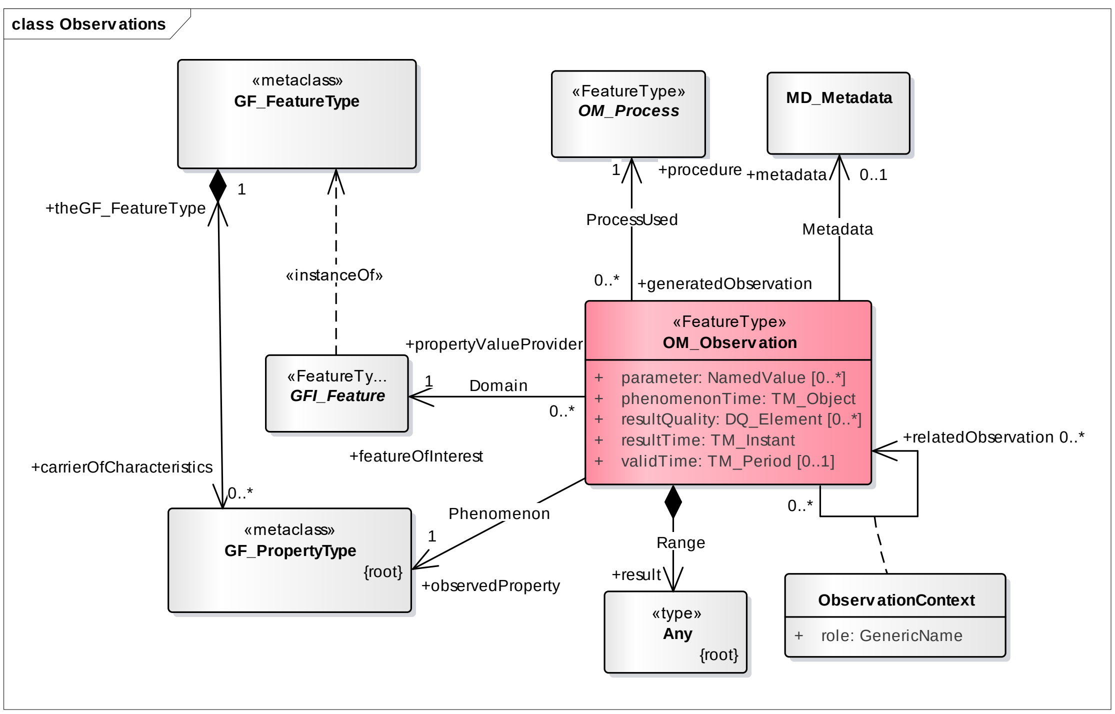


Figure 10 : SWE Observations

[1] Sensor Observation Service Version 2.0 Requirement 77

SIF Observable		O&M Observation (Observable)		
Element	**#**	**Element**	**#**	**Comments**
identifier	1..1	identifier	0..1	See Section 0
NA		name	0..n	
Position	1..1	boundedBy	0..1	These parameters identify the spatial extent which can be covered by this offering.
		location	0..1	
NA		Type	0..1	The data type used for the results.
NA		parameter	0..n	Parameters are usually captured in the Process metadata. They may be replicated in the Observation if impractical to keep the Process metadata current or if the parameter is specific to this observation.
NA		phenomenonTime	1..1	Time period over which measurements have been taken. Value shall be null[2] if no measurements have been taken.
NA		resultTime	1..1	Time period over which result values have been generated. Value shall be null[3] if no measurements have been taken.
NA		validTime	0..1	Do not use after this date/time. Element should not be included if no measurements have been taken.
NA		resultQuality	0..n	Do not populate
		Associations		
NA		featureOfInterest	1..1	The Reference View concept of Phenomenon associates mutually dependent Observed Properties with a Feature of Interest type to form a single concept (ex. wind speed and direction). For an Observable, FeatureOfInterest should reference an abstract Feature, representing a Feature Type but not an instance.
reportsOn		observedProperty	1..1	
producedThrough	1..1	procedure	1..1	Reference to the Process or Physical Component which performed the measurement activity.
NA		result	1..1	Value shall be null[4].
reports	0..n	relatedObservation	0..n	Used to associate Observations with the Observable.

Table 14 : SIF to SWE Observable Mapping

7.2.2 Observations

Applicable Specifications:

- Observations and Measurements 2.0

[2] If a null value is used, then set the nullReason attribute to "template"
[3] If a null value is used, then set the nullReason attribute to "template"
[4] If a null value is used, then set the nullReason attribute to "template"

- Sensor Observation Service 2.0

Note that an Observation can only report on one Property. However, an Observation may be composed of other Observations. This will allow one Observation to aggregate other Observations.

A sensor executes a process to measure or otherwise determine (observe) the value of a property. That value is then reported out for further processing and exploitation. The observed value alone cannot satisfy this purpose. Users also need metadata describing when, where, and how the observed value was collected. SWE addresses this need through the Observation class as defined in the Observations and Measurements standard.

O&M Observations fulfill two roles: 1) they provide metadata which describe the context of the measuring activity, and 2) they serve as nodes associating the resources relevant to the collection.

Observations are the instantiation of an Observable. It is only fitting that an Observation and Observable share the same data structure, and where appropriate the same content. Table 15 documents how an O&M Observation is populated as a SIF Observation.

SIF Observation		**O&M Observations**		
Element	**#**	**Element**	**#**	**Comments**
identifier	1..1	identifier	0..1	Generate a new identifier per Section 2.1
NA		name	0..n	
Position	1..1	boundedBy	0..1	The spatial extent or location of the measured value.
		location	0..1	
NA		Type	0..1	The data type used for the results. See Section 7.2.3
NA		parameter	0..n	Parameters are usually captured in the Process metadata. They may be replicated in the Observation if is impractical to keep the Process metadata current or if the parameter is specific to this observation.
phenomenonTime	1..1	phenomenonTime	1..1	The date/time this measurement was taken
resultTime	1..1	resultTime	1..1	The date/time this Observation was first generated
validTime	0.1	validTime	0..1	Do not use after this date/time value.
resultQuality	0..1	resultQuality	0..n	A description of the uncertainty in the measured value.
		Association		
target	1..1	featureOfInterest	1..1	A reference to the instantiated feature whose property was measured.
reportsOn	1..1	observedProperty	1..1	A reference to the property being measured.
producedThrough	1..1	procedure	1..1	See Table 14
Values	0..1	result	1..1	Reference to measured values. See Section **Error! Reference source not found.**
reports	0..n	relatedObservation	0..n	Use to decompose a complex Observation into its constituent parts.

Table 15 : SIF to SWE Observation Mapping

7.2.3 Measures

The primary purpose of a sensor system is to provide a digital representation of a real-world phenomenon. Just as there are many different types of phenomenon, there must be many ways of representing phenomenon digitally. The SIF-SP Reference View provides the taxonomy of digital representations (Measures) illustrated in Figure 11.

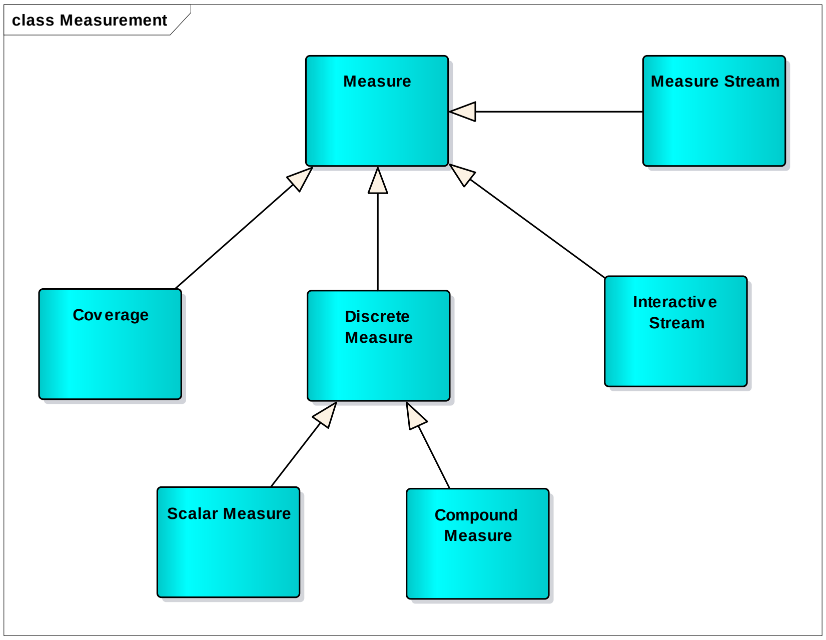


Figure 11: Measurement Taxonomy

Types of Measurement differ in both the logical structure of the information as well as how that information is encoded. Therefore, this Enterprise Technical View identifies standards for each Measurement type that include rules for both the schema and encoding of the measurements. A mapping of Reference View Measurements to Enterprise View specifications is provided in Table 16.

Measurement	Specification	Comments
Measurement Stream	MISP	A single MPEG essence stream
Multiplexed Stream	MISP	A multiplexed stream consists of one MPEG essence stream with at least one MPEG metadata stream.
Scalar Measurement	SWE Common AND SIF-SP Ontology	SWE Common provides reusable types. The SIF-SP Ontology provides semantics for those types.
Compound Measurement	SWE Common AND SIF-SP Ontology	SWE Common provides reusable types. The SIF-SP Ontology provides semantics for those types.
Interactive Stream	JPIP	Only useful for JPEG 2000 compressed imagery.
Coverage	Multiple	See Table 17

Table 16: Measures Standards and Specifications

Coverages provide a particular issue since there are many different formats which may be used. Table 17 provides a list of coverage formats, indicates which should be used, and why.

Format	Name	Discussion
JP2 / JPX	JPEG 2000	Required for JPIP access
Exif	Exchangeable image file format	Industry standard for commercial and consumer grade cameras. Rich metadata support.
PNG	Portable Network Graphics	Can be used if the Exif metadata chunk is populated. Else do not use since there is not sufficient metadata support.
NITF	National Imagery Transmission Format	NSG standard for Earth Observation imagery
GeoTIFF	Georeferenced Tagged Image File Format	Preference is to use the NSG version since it has a more rigorous metadata model.
LAS	LASer File Format	For point clouds

Table 17 : Coverage File Formats

Measures by themselves can be difficult to use. Therefore, the standards cited should also address how the association between the Measures and metadata describing the Measure is to be maintained.

7.2.3.1 Observations and Measurements

Observations do not carry schema or encoding information about their results. Instead, SWE sub-classes the Observation class based on the result data type. This allows the association between the Measures and their metadata to be maintained through the sub-class relationship. The measured value and its metadata are combined into a single data component.

The Observation "type" element is used to specify which sub-class of Observation is represented in that instance. Valid subclasses are defined at http://www.opengis.net/def/observationType/OGC-OM/2.0/. A mapping of these subclasses to Reference View measures is provided in Table 18. These values are current as of August 2017.

Reference View		Observation Subclass		
Measure	**Comments**	**SubClass**	**Data Type**	**Domain**
Scalar Measure		Count Observation	integer	CS Primitive
Scalar Measure		Truth Observation	Boolean	CS Primitive
Scalar Measure	Double precision floating point with a unit of measure identifier.	Measurement	MeasureType	GML
Scalar Measure	Xlink to the category entry	Category Observation	ReferenceType	GML
Scalar Measure	Date/time instance or period.	Temporal Observation	AbstractTimeObject	GML
Compound Measure	Points, lines, polygons, etc.	Geometry Observation	AbstractGeometry	GML
Compound Measure		Complex Observation	DataRecord or Vector	SWE

Reference View		Observation Subclass		
Measure	**Comments**	**SubClass**	**Data Type**	**Domain**
Compound Measure, Interactive Stream, Measure Stream, Multiplexed Stream	Streaming Measures are only available through the DataStream type.	SWE Array Observation	DataArray, Matrix, or DataStream	SWE
Scalar Measure		SWE Scalar Observation	AbstractSimpleComponent	SWE
Coverage	Due to their size and complexity, these values are rarely packaged with the Observation	Discrete Coverage Observation	URI	Remote
Coverage		Point Coverage Observation	URI	Remote
Coverage		Time Series Observation	URI	Remote

Table 18 : Observation Result Types

The Data Types column in Table 18 identify the structure or schema to be used for the measured values under that Observation sub-class. These data types are defined in different domains. These domains are:

- Computer Science primitives or Geography Markup Language (GML). Data types which are computer science primitives or GML types are encoded within the Observation.
- SWE Common data types capture the schema and encoding information within the Observation, but the data itself may reside elsewhere.
- Remote values are always defined and stored elsewhere. They are only referenced from the Observation.

Values that are encoded within the Observation have their schema defined by the data type and the encoding defined by the encoding of the Observation. SWE data types describe the schema and encoding of the values in the Observation but the values themselves may be stored elsewhere.

The SWE DataStream type is a special case. When used with an Observation, a DataStream adds metadata to describe a remote set of measured values which is accessible through a streaming delivery. DataStream identifies the schema and encoding used, the number of instances (if applicable), and provides a link to where the values can be accessed.

7.2.3.2 Raw Results

In some cases, the volume and velocity of the measured values is too great to conveniently access through an Observation. It is sufficient to have the service endpoint to the data, and to feed it directly to the client. However, it is still necessary to maintain an association between the metadata and the measured values. It is also necessary to know the schema and encoding used for the measured values. The Result Template addresses these requirements.

A Result Template is created by the measurement provider. It defines an association between the Offering, Observation, Schema, and Encoding. The Result Template and role played by each element is described in Table 19.

Property	Description	#	Notes/Comments
identifier		0..1	See Section 2.1
offering	Pointer to an ObservationOffering for which results will be requested in subsequent GetResultcalls.	1..1	A pointer to an existing offering
observationTemplate	A partially populated Observation that can be used to create a complete Observation once result values are available.	1..1	A partially populated Observation. This Observation will be used to update the offering to include the new measured values.
resultStructure	The structure of the results which may be requested in subsequent GetResult calls. Constrained to AbstractDataComponents	1..1	The schema used by the measured values
resultEncoding	The encoding of the results which may be requested in subsequent GetResult calls. Constrained to AbstractEncodings.	1..1	The encoding used by the measured values

Table 19 : Result Template

The observation template that is provided in the Result Template shall have null as value of phenomenonTime, resultTime and result. The procedure, featureOfInterest, and observedProperty elements of the observation template shall not be empty.

7.3 Spatial-Temporal

Applicable Specifications:

- GML 3.2.1

The spatial-temporal concepts used in this Technical View are specified through the ISO TC211 body of standards and their XML encoding in GML 3.2.1. JSON and JSON-LD encodings are under development.

OGC 07-036 (GML 3.2.1)		SIF-SP Ontology	
Element	**Description**	**Concept**	**Comments**
gml:Arc	An Arc is an arc string with only one arc unit, i.e. three control points including the start and end point.	GeographicArc	
gml:AbstractCurve Segment	Implemented per the example in section E.2.4.7 of OGC 07-036.	GeographicEllipse	
gml:Polygon	A special surface that is defined by a single surface patch. The boundary of this patch is coplanar and the polygon uses planar interpolation in its interior.	GeographicPolygon	
gml:LineString	A special curve that consists of a single segment with linear interpolation. It is defined by two or more coordinate	GeographicPolyline	

OGC 07-036 (GML 3.2.1)		SIF-SP Ontology	
Element	**Description**	**Concept**	**Comments**
	tuples, with linear interpolation between them.		
gml:Point	Defined by a single coordinate tuple	GeographicPosition	
		TemporalLocation	A union (choice) of a TemporalInstance or a TemporalPeriod.
gml:TimeInstant	A zero-dimensional geometric primitive that represents an identifiable position in time	TemporalInstance	
gml:TimePeriod	A one-dimensional geometric primitive that represents an identifiable extent in time.	TemporalPeriod	

Table 20 : Concepts for Space and Time

8 Computational Viewpoint

8.1 Messaging

Applicable Specifications:

- OWS Common 2.0

Messaging in the Enterprise is almost exclusively implemented using Direct Messaging over HTTP (see Table 21). Four different approaches are allowed. The first three follow the traditional Service Oriented Architecture Request-Response pattern. The fourth is something different.

Capability	Operation	Discussion
Messaging		
Direct Messaging		The SOS currently supports the request-response transaction model running over HTTP.
Post Message	HTTP GET/POST	
Deliver Message	HTTP GET/POST	
	Web Sockets	WebSocket support has recently been added for streaming delivery.
Publish-Subscribe Messaging		OGC has defined a Publish-Subscribe standard but it has not yet been integrated with SOS.
Publish	NA	
Subscribe	NA	
Notify	NA	
Message Oriented Middleware	NA	Not supported
Post Message	NA	

Capability	Operation	Discussion
Route Message	NA	

Table 21 : Enterprise Messaging

8.1.1 Key-Value Pair

The Key-Value Pair (KVP) approach to messaging encodes the request parameters in the query string of the request URL. This is the simplest form for a request message but suffers from the limited size and expressiveness of the URL syntax.

The response message from a KVP request is either the response as defined by the implementing standard or an exception report if the service is unable to respond correctly.

8.1.2 XML

The XML approach to messaging encodes the request parameters in an XML document. This document is then sent to the service using an HTTP POST request. The implementing standard defines the format of the XML document through an XML Schema. The XML approach allows the request parameters to be more complex than those possible with the KVP approach.

The response message from a XML request is either the response as defined by the implementing standard or an exception report if the service is unable to respond correctly.

8.1.3 SOAP

The SOAP approach to messaging extends the XML approach by packaging the XML document in a SOAP envelope. This approach is typically used when the advanced capabilities provided by the SOAP headers (such as SAML security) are required.

The response to a SOAP request is a SOAP response message which contains a Body element as specified in the implementing standard. If an error is detected, the Body element is a Fault element containing an exception report.

8.1.4 REST

The KVP, XML, and SOAP messaging approaches are all based on the Service-Oriented Architecture (SOA) architectural pattern. The technology used within the Enterprise Technical View also includes Representational State Transfer (REST) messaging. REST differs from SOA in that it is a Resource Oriented Architecture (ROA). An ROA deals with manipulating resources across a networked environment. Services are only present as a secondary effect of manipulating resources.

REST messages are limited[5] to the HTTP suite of GET, PUT, POST, DELETE, and HEAD. REST resource operations are mapped against the HTTP messages in Table 22. While REST has gained

[5] Technically REST is not required to run over HTTP. However, no non-HTTP implementations have been identified. So from a practical perspective, REST = HTTP

popularity across the Enterprise, it is not universal. The issues related to support for all of the capabilities of a SOA within a REST architecture are still being worked out.

Resource Operation	Description	HTTP operation
createResource	Create a new resource (and the corresponding unique identifier)	PUT
getResourceRepresentation	Retrieve the representation of the resource	GET
deleteResource	Delete the resource (optionally including linked resources)	DELETE (referred resource only) POST (can be used if the delete is including linked resources)
modifyResource	Modify the resource	POST
getMetaInformation	Obtain meta information about the resource	HEAD

Table 22 : REST and HTTP[6]

8.2 Discovery

Catalogs typically fall into two categories, service catalogs and content catalogs. Service catalogs advertise things that compute. In the SIF-SP Reference View, these are the entities described in the Computational Viewpoint. Content catalogs advertise data and information. In the SIF-SP Reference view, these are the entities described in the Information Viewpoint. These two catalogs differ greatly in terms of metadata, query parameters, and use of the advertised resources. Therefore, where appropriate this section distinguishes between activities based on their resource type:

- Process – Activities in support of the discovery of Activities
- Performer – Activities in support of the discovery of Performers
- Content – Activities in support of the discovery of Information

8.2.1 DDF

Applicable Specifications:

- Distributed Data Framework documentation page at http://www.codice.org/ddf/documentation.htm
- OGC Catalog Services for the Web
- Content Discovery and Retrieval
- OpenSearch
- OGC Filter Encoding

[6] Roberto Lucchi and Michel Millot, Resource Oriented Architecture and REST, retrieved from http://inspire.ec.europa.eu/reports/ImplementingRules/network/Resource_orientated_architecture_and_REST.pdf on September 19, 2017

The Distributed Data Framework (DDF) is a free and open-source common data layer that abstracts services and business logic from the underlying data structures to enable rapid integration of new data sources. It forms the common software base for Distributed Common Ground System (DCGS) family of systems. DCGS in turn forms the backbone of the Defense Intelligence Information Enterprise (DI2E). DI2E is the DoD enterprise for information sharing.

The primary purpose of the DDF is to serve as a distributed catalog for discovery of DCGS resources. The Discovery Activities supported by the DDF are described in Table 23.

Discovery Activity	Resource	Service	Discussion
Browse	Process, Performer, and Content	REST	HTTP GET
Describe	Process, Performer, and Content	None	The DDF only hosts metacards. It does not support the complex information required to provide Describe services. This capability can be provided through the SOS (see Section 8.2.2)
Register	Process, Performer, and Content	REST	HTTP POST
		CS-W	csw:Insert operation csw:update operation
		Ingest	The DDF is capable of extracting metadata from some data types and using that metadata to generate a metadata card. This ingest capability is described in Section 8.2.1.2.
Submit Query	Process, Performer, and Content	REST	HTTP GET
		CometD	Supports the OGC Common Query Language
		CS-W	Supports the OGC Common Query Language
		CDR	Supports the OGC Filter Encoding query language
Remove	Process, Performer, and Content	REST	HTTP DELETE
		CS-W	csw:Delete operation

Table 23 : DDF Discovery Activities

In the DDF, resources are the data products, files, reports, or documents of interest to users of the system. Metadata is information about those resources, organized into a schema to make search possible. The DDF Catalog stores this metadata and allows access to it. Metacards are single instances of metadata that represent a single resource in the Catalog. Metacards follow one of several schemas to ensure reliable, accurate, and complete metadata. Essentially, metacards function as containers of metadata.

8.2.1.1 Discovery Endpoints

Activities:

- Browse (Process, Performer, Content)
- SubmitQuery (Process, Performer, Content)
- Remove (Process, Performer, Content)

DDF provides the capability to search the Catalog for metadata. There are a number of different types of searches that can be performed on the Catalog, and these searches are accessed using one of several interfaces. This section provides a very high-level overview of introductory concepts of searching with DDF. These concepts are expanded upon in Annex D.

8.2.1.1.1 REST

DDF supports simple CRUD operations using HTTP operations.

Operation	HTTP Request	Details	Example URL
create	HTTP POST	HTTP request body contains the input to be ingested.	http://localhost:8181/services/catalog?transform=<input transformer> <input transformer> is the name of the transformer to use when parsing metadata (optional).
update	HTTP PUT	The ID of the Metacard to be updated is appended to the end of the URL. The updated metadata is contained in the HTTP body.	http://localhost:8181/services/catalog/<metacardId>?transform=<input transformer> <metacardId> is the Metacard.ID of the metacard to be updated and <input transformer> is the name of the transformer to use when parsing an override metadata attribute (optional).
delete	HTTP DELETE	The ID of the Metacard to be deleted is appended to the end of the URL.	http://localhost:8181/services/catalog/<metacardId> <metacardId> is the Metacard.ID of the metacard to be deleted.
read	HTTP GET	The ID of the Metacard to be retrieved is appended to the end of the URL. By default, the response body will include the XML representation of the Metacard.	http://localhost:8181/services/catalog/<metacardId> <metacardId> is the Metacard.ID of the metacard to be retrieved.
federated read	HTTP GET	The SOURCE ID of a federated source is appended to the URL before the ID of the Metacard to be retrieved is appended to the end.	http://localhost:8181/services/catalog/sources/<sourceId>/<metacardId> <sourceId> is the FEDERATED SOURCE ID and <metacardId> is the Metacard.ID of the Metacard to be retrieved.
sources	HTTP GET	Retrieves information about federated sources, including sourceId, availability, contentTypes, and version.	http://localhost:8181/services/catalog/sources/

Table 24 : The DDF Catalog REST Endpoints

8.2.1.1.2 CometD:

The CometD endpoint enables asychronous search capabilities. The CometD protocol is used to execute searches, retrieve results, and receive notifications. CometD is a low latency WebSocket and HTTP based event and message routing bus. In default mode it implements the publish-subscribe messaging pattern. The DDF implementation supports the OGC Common Query Language.

See https://docs.cometd.org/current/reference

8.2.1.1.3 OGC Catalog Service for the Web (2.0.2)

Catalogue services support the ability to publish and search collections of descriptive information (metadata) for data, services, and related information objects. Catalog metadata represents resource characteristics that can be queried and presented for evaluation and further processing.

The OpenGIS® Catalogue Services Specification defines the interfaces between a catalog service and its clients. These interfaces are defined through both an abstract and three implementation-specific models. One of these implementation-specific models is for use over HTTP protocols. This model is commonly referred to as Catalog Service for the Web. The CSW operations relevant to this Technical View are described in Table 25.

Operation	Purpose
GetCapabilities	Request for a description of service capabilities
DescribeRecord	This request allows a user to discover elements of the information model supported by the catalogue.
GetDomain	Requests the actual values of some specified request parameter or other data element.
GetRecords	The principal means of searching the catalogue. Conveys a query expression to the catalog service for processing. Also includes parameters to govern the size and format of the result set.
Insert	Submits one or more records to the catalogue.
Update	Update statements may replace an entire record or only update part of a record
Delete	Deletes one or more catalogue items that satisfy some set of conditions.
Harvest	Requests that the catalogue attempt to harvest a resource from some network location identified by the source URL

Table 25: CS-W Operations

The CSW standard is designed to be extensible. The DDF takes advantage of that option by adding an additional operation. This operation is a publish-subscribe variant of the GetRecords operation. This operation allows a client to submit a query once and then receive notifications whenever new data matches that query.

The Catalog Services Specification also defines the Common Query Language. The Common Query Language is based on SQL with extensions to support spatial elements and operations.

8.2.1.1.4 OpenSearch and CDR

The Intelligence Community/Department of Defense (IC/DoD) Content Discovery and Retrieval (CDR) Specification Framework [CDR-SF] serves as the primary mechanism to expose content collections for discovery and retrieval. The CDF-SF consists of five components. Those components are described in Table 26.

Component	Description
Brokered Search	Supports the distribution of search requests to applicable/relevant sources and aggregate the results returned from different sources.
Delivery	Supports the delivery of a specified resource payload directly to a consumer specified location.
Query Management	Manages Saved Searches and may initiate search requests based on Saved Searches.

Component	Description
Retrieval	Supports the retrieval of a specified resource from a content collection and returning that content to the requestor.
Search	Supports queries against metadata records and returns the results to the client. CDR Search is based on OpenSearch.

Table 26 : CDR Components

8.2.1.2 Ingest

Activities: Register (Process, Performer, Content)

Ingest is the process of bringing data products, metadata, or both into the catalog to enable search, sharing, and discovery. Ingested files are transformed into a neutral format that can be searched against as well as migrated to other formats and systems.

Upon ingest, a transformer will read the metadata from the ingested file and populate the fields of a metacard. Exactly how this is accomplished depends on the origin of the data, but most fields are imported directly.

8.2.1.2.1 Ingest Command (Command Console)

The DDF console application has a command-line option for ingesting files. The syntax for the ingest command is ingest -t <transformer type> <file path> relative to the installation path. For example, the following command will to ingest and extract metadata from the XML document sample.xml:

ingest -t xml examples/metacards/xml ./sample.xml

8.2.1.2.2 Directory Monitor

The Catalog application contains a Directory Monitor feature that allows files placed in the monitored directory to be ingested automatically. Simply place the desired files in the monitored directory and it will be ingested automatically. If, for any reason, the files cannot be ingested, they will be moved to an automatically created sub-folder named .errors. Optionally, ingested files can be automatically moved to a sub-folder called .ingested.

8.2.1.2.3 External Methods

Several third-party tools, such as cURL.exe and the Chrome Advanced Rest Client, can be used to send files and other types of data to DDF for ingest.

8.2.1.2.4 Advanced (more records, automated ingest)

The DDF provides endpoints for both REST and SOAP services, allowing integration with other data systems and the ability to further automate ingesting data into the catalog.

8.2.2 Sensor Observation Service

Activities:

- Browse (Process, Performer, Content)
- Describe (Process, Performer, Content)
- Register (Process, Performer, Content)
- Remove (Process, Performer)

- SubmitQuery (Content)

Applicable Specifications:

- OGC Sensor Observation Service Version 2.0

The OpenGIS Sensor Observation Service (SOS) Implementation Standard defines a web service interface for requesting, filtering, and retrieving observations and sensor system information. Observations may be from in-situ sensors (e.g., water monitoring devices) or dynamic sensors (e.g., imagers on Earth-observation satellites).

Discovery Activity	Resource	Operation	Discussion
Browse	Process, Performer, and Content	getCapabilities()	The contents section of the Service Metadata document includes descriptions of the resources accessible through this SOS. This is a document so you cannot query against it. The most important element, from a discovery perspective, is the Observation Offering.
Describe	Process and Performer	describeSensor()	Procedure parameter identifies the sensor Procedure Description Format parameter defines how the returned data should be formatted Valid Time parameter supports filtering for sensor properties within a specified time box.
Describe	Content	getObservationById()	
Register	Process, Performer and Content	insertSensor()	Registers a sensor or procedure with the associated offering
Submit Query	Content	getObservation()	Discovers and returns observations based on: Feature(s) of interest Observed properties Observation Offering(s) Procedure(s) – sensors Spatial filter Temporal filter
Remove	Process and Performer	deleteSensor()	Remove a Process or Performer

Table 27 : Sensor Observation Service Discovery Activities

8.2.2.1 Get Capabilities

Activities: Browse (Process, Performer, Content)

The getCapabilities() operation returns the Service Metadata documents (see Section 2.5 and 7.1.2). This is the comprehensive descriptive metadata for the SOS and all of the resources it supports. The SOS Service Metadata is not static. This document is updated with every new sensor or offering. As such, retrieving the Service Metadata document is often the first step in SOS discovery.

Service Metadata for the Sensor Observation Service is organized into five sections:

- Service Identification

- Service Provider
- Operations Metadata
- Contents
- Filter Capabilities

Clients have the option of retrieving any combination of these sections. However, the Contents section is most relevant for discovery. This section includes the Observation Offering (Section 7.1.2) which allows clients to browse for Processes, Performers, and Observables.

8.2.2.2 Describe Sensor

Activities: Describe (Process, Performer)

The describeSensor() operation enables querying of metadata about the sensors and sensor systems available through an SOS server.

This operation is performed by sending a describeSensor request message to the service. This request includes an identifier for the sensor and the format to be used for the returned data. By default, the current sensor description is returned. If supported by the service, a client may request descriptions the valid Processes and Procedures at a certain point in time or within a time period. The format of the sensor description is specified by the "procedureDescriptionFormat" parameter. The preferred (and default) format is SensorML 2.0.

Parameter	Description	#	Mapping
Procedure	Pointer to the procedure/sensor of which the description shall be returned	1..1	This parameter is required by the Reference View. SOS type is a URI. Preferred values are a URN encoded UUID or a GUIDE ID.
procedureDescriptionFormat	identifier of the requested procedure/sensor description format	1..1	This parameter is specific to the Enterprise View. Values for this parameter should be: http://www.opengis.net/sensorML/1.0.1 or http://www.opengis.net/sensorML/2.0.0
validTime	Time instant or time period for which the then valid sensor description shall be retrieved.	0..n	Not currently supported in the Reference View.

Table 28 : Describe Sensor Parameters

8.2.2.3 Get Observation

Activities: Submit Query (Content)

The GetObservation() operation is designed to query an SOS to retrieve observation data structured according to the O&M specification. An O&M Observation may, but is not required to, include the measurements associated with that observation. In the case where the measurement is included, the GetObservation() operation combines the Describe and Delivery capabilities. In the case where the measurement is omitted, then getObservation() functions as a Describe service. In this case the return will be zero or more O&M Observations as described in Section 7.2.2.

The parameters for the geoObservation() operation mapped into the Reference View in Table 29.

Parameter	Description	#	Mapping
featureOfInterest	Pointer to a feature of interest for which observations are requested.	0..n	Resource Filter: Identifies the target or target type
observedProperty	Pointer to an observedProperty for which observations are requested.	0..n	Resource Filter: identifies the phenomenology
Offering	Pointer to an ObservationOffering advertised in the Service Metadata document for which observations are requested.	0..n	Resource Filter: Identifies the Observable Description
Procedure	Pointer to a procedure for which observations are requested. It defines a filter for the procedure property of the observations.	0..n	Resource Filter: Identifies the Process/Performer Description
responseFormat	Identifier of desired responseFormat for the requested observations. The supported responseFormats are listed in the ObservationOffering.	0..1	This parameter is specific to the Enterprise View. Valid values for this parameter shall include http://www.opengis.net/om/2.0
spatialFilter	Specifies a filter which applies to a spatial property of an observation. This property is defined in the valueReference element of the SpatialOperator.	0..1	Resource Filter: spatial components
temporalFilter	Specifies a filter for a time property of requested observations. This property is defined in the valueReference element of the TemporalOperator.	0..n	Resource Filter: temporal components

Table 29 : Get Observation Parameters

8.2.2.4 Delete Sensor

Activities: Remove (Process, Performer)

The DeleteSensor() operation removes a process (sensor) from access through the SOS. It has the side effect of also removing access to all offerings and observations associated with that sensor.

Parameter	Description	#	Mapping
procedure	Pointer to the procedure/sensor that shall be deleted.	1..1	Resource Identifier

Table 30 : Delete Sensor Parameters

8.2.2.5 Insert Sensor

Activities: Register (Process, Performer, Content)

The InsertSensor() operation establishes metadata to describe a Process, Performer, and Observable within a SOS. The parameters used by the insertSensor() operation are described in Table 31. The "procedureDescription" element contains a SensorML document which is used to create the Process and Performer Descriptions (Sections 7.1.3 and 7.1.3.2). A pointer to that document is then used with the other parameters to create an Observable Description as described in Table 31. A side effect of this operation is an update to the Service Metadata (Sections 2.5 and 7.1.2).

Insert Sensor Parameters			Observable Description
Parameter	**Description**	**#**	**Mapping**
procedureDescriptionFormat	identifier of the format in which the procedure/sensor description is given in	1..1	NA – scope is local to the Enterprise View. Valid formats are; SensorML 1.0 SensorML 2.0
procedureDescription	a description of the procedure	1..1	Process Description or Performer Description
observableProperty	Pointer to a property that can be observed by the procedure, not a property that has already been observed.	1..n	Phenomenology
relatedFeature	feature that is directly or indirectly observed/observable by the procedure; can be any feature which the requestor thinks the procedure can make valuable observations for	0..n	target
metadata	Additional information required for inserting the sensor at a specific service.	0..n	Note: a single metadata element is composed of one or more featureOfInterestType and one or more observationType elements.
featureOfInterestType	identifier of feature of interest type associated with observation produced by the sensor	1..n	targetType
observationType	identifier of observation type (with unique result type) which is produced by the sensor	1..n	Schema and Encoding

Table 31 : Insert Sensor Parameters

8.2.2.6 Get Observation by Id

Activities: Describe (Content)

The GetObservationByID() operation allows the client to retrieve an observation by passing a Pointer to that observation. The returned value is an O&M Observation which contains metadata which describes that Observation.

Parameter	Description	#	
observation	Pointer to the observation which shall be returned.	1..n	This parameter is required by the Reference View.

Table 32 : Get Observation by ID Parameters

8.3 Delivery

Activities: Discrete Delivery, Interactive Delivery, Streaming Delivery, Delivery/Register

Applicable Specifications:

- OGC Sensor Observation Service Version 2.0

Delivery Services within the SIF Enterprise are provided by the Sensor Observation Service (SOS). The OpenGIS Sensor Observation Service (SOS) Implementation Standard defines a web service interface for requesting, filtering, and retrieving observations and sensor system information. Observations may be from in-situ sensors (e.g., water monitoring devices) or dynamic sensors (e.g., imagers on Earth-observation satellites).

The SIF-SP Reference view distinguishes between Discrete Delivery, Interactive Delivery, and Streaming Delivery. The SOS does not make that distinction at the Observation level. Rather, that is a property of the Measurement (see Section 7.2.3).

Discovery Activity	Operation	Discussion
Discrete Delivery	getObservation()	
Discrete Delivery	getObservationById()	
Discrete Delivery, Interactive Delivery, Streaming Delivery	getResult()	
Discrete Delivery, Interactive Delivery, Streaming Delivery	getResultTemplate()	
Delivery/Register	insertResult()	
Delivery/Register	insertResultTemplate()	
Delivery/Register	insertObservation()	

Table 33 : Sensor Observation Service Delivery Activities

8.3.1 Get Observation

Activities: Discrete Delivery

The GetObservation() operation is designed to query an SOS to retrieve observation data structured according to the O&M specification. An O&M Observation may, but is not required to, include the measurements associated with that observation. Parameters for the getObservation() operation are described in Table 34. This table also provides a mapping between the parameter and the Reference View concepts. Observations are discussed in Section 7.2.2. Measurements are discussed in Section 7.2.3.

Parameter	Description	#	Mapping
featureOfInterest	Pointer to a feature of interest for which observations are requested.	0..n	Resource Filter: Identifies the target or target type
observedProperty	Pointer to an observedProperty for which observations are requested.	0..n	Resource Filter: identifies the phenomenology
Offering	Pointer to an ObservationOffering advertised in the Service Metadata document for which observations are requested.	0..n	Resource Filter: Identifies the Observable Description
Procedure	Pointer to a procedure for which observations are requested. It defines a filter for the procedure property of the observations.	0..n	Resource Filter: Identifies the Process/Performer which generated this Observation
responseFormat	Identifier of desired responseFormat for the requested observations. The supported responseFormats are listed in the ObservationOffering.	0..1	This parameter is specific to the Enterprise View. Valid values for this parameter shall include http://www.opengis.net/om/2.0
spatialFilter	Specifies a filter which applies to a spatial property of an observation. This property is defined in the valueReference element of the SpatialOperator.	0..1	Resource Filter: spatial components
temporalFilter	Specifies a filter for a time property of requested observations. This property is defined in the valueReference element of the TemporalOperator.	0..n	Resource Filter: temporal components

Table 34 : Get Observation Parameters

8.3.2 Get Observation by Id

Activities: Discrete Delivery

The GetObservationByID() operation allows the client to retrieve an observation by passing a Pointer to that observation. The returned values are identical to those of GetObservation().

Parameter	Description	#	
observation	Pointer to the observation which shall be returned.	1..n	This parameter is required by the Reference View.

Table 35 : Get Observation by ID Parameters

8.3.3 Get Result

Activities: Discrete Delivery, Interactive Delivery, Streaming Delivery

The GetResult() and GetResultTemplate() operations allows sensor measurements to be delivered independently from the metadata which describes those measurements. GetResultTemplate() delivers the metadata. GetResult() delivers the data.

GetResult() has filtering capabilities which are similar to those of GetObservation(). These parameters are described and mapped in Table 36.

GetResult() is valuable in that is places no constraints on the structure or encoding of the measured values. As a result, this is the preferred way to deliver streaming and interactive content. However, discrete content can be delivered this way as well.

Since the structure and encoding of the content is not specified, clients must have a way to acquire this information. The getResultTemplate() operation (Section 8.3.4) fulfils that role.

Parameter	Description	#	Mapping
featureOfInterest	Pointer to a feature of interest for which results are requested.	0..n	Resource Filter: Identifies the target or target type
observedProperty	Pointer to the observedProperty for which results are requested.	1..1	Resource Filter: identifies the phenomenology
Offering	Pointer to the ObservationOffering advertised in the Service Metadata document for which results are requested.	1..1	Resource Filter: Identifies the Observable Description
spatialFilter	Specifies a filter which applies to a spatial property of an observation. This property is defined in the valueReference element of the SpatialOperator.	0..1	Resource Filter: spatial components
temporalFilter	Specifies a filter for a time property of requested observations. This property is defined in the valueReference element of the TemporalOperator.	0..n	Resource Filter: temporal components

Table 36 : GetResult Parameters

8.3.4 Get Result Template

Activities: Discrete Delivery, Interactive Delivery, Streaming Delivery

Note: Technically GetResultTemplate() is not a Delivery action. However, it works in conjunction with GetResult(). Since GetResult() would not be usable otherwise, GetResultTemplate() has been included.

The GetResult() and GetResultTemplate() operations allows sensor measurements to be delivered independently from the metadata which describes those measurements. GetResultTemplate() delivers the metadata. GetResultTemplate() delivers the metadata.

GetResultTemplate() allows a user to select sensor measurements based on an offering and one observed property supported by that offering. It returns the structure (data type or schema) and encoding for those measurements. These values, when combined with the offering and observed property) form the result template. This template forms a complete description of the data that can be received through the corresponding getResult() operation.

Parameter	Description	#	Mapping
observedProperty	Pointer to an observedProperty for which observations are requested.	1..1	Resource Filter: identifies the phenomenology
Offering	Pointer to an ObservationOffering advertised in the Service Metadata document for which observations are requested.	1..1	Resource Filter: Identifies the Observable Description

Table 37 : GetResultTemplate Parameters

8.3.5 Insert Result

Activities: Delivery/Register

The InsertResult() operation posts sensor measurements to SOS for retrieval by the GetResults() operation. This operation is useful when most of the metadata contained in a set of operations is the same. The metadata is delivered separately. Delivery of the results can then be optimized for the unique characteristics of the measurements.

Parameter	Description	#	Mapping
resultValues	The results of observations which shall be inserted.	1..1	
template	Pointer to the template defining the structure and encoding of the results.	1..1	

Table 38 : Insert Result Parameters

8.3.6 Insert Result Template

Activities: Delivery/Register

The InsertResultTemplate() operation associates a template for new Observations with an existing Observation Offering. The template will be used to create Observations when measured results are posted through the “insertResults()” operation.

Parameter	Description	#	Mapping
ProposedTemplate	Specifies the observation metadata and also information about the structure and encoding of the result, but no result value	1..1	

Table 39 : Insert Result Template Parameters

Parameter	Description	#	Mapping
Offering	Pointer to an ObservationOffering advertised in the Capabilities document for which observations are requested.	1..1	Observable Description – Note that this must already exist. The Observable (Observation Template) is being associated with this Observable Description.
observationTemplate	An O&M Observation template that is used to form complete observations when result values that are inserted later on. See Section 7.2.1.	1..1	Observation being added.
resultStructure	The structure of the results which may be requested in subsequent GetResultcalls.	1..1	Observation Description – schema element
resultEncoding	The encoding of the results which may be requested in subsequent GetResultcalls.	1..1	Observation Description – schema element

Table 40 : Result Template

8.3.7 Insert Observation

Activities: Delivery/Register

The InsertObservation() operation posts sensor measurements to the SOS for retrieval by the GetObservation() operation. The observation must be associated with an existing Offering.

Parameter	Description	#	Mapping
observation	Observation to insert	1..n	
Offering	Pointer to an ObservationOffering to which the observation(s) shall be added.	1..n	

Table 41 : Insert Observation Parameters

8.4 Command

Commands are used to start, modify, and stop the execution of processes. It is not clear that there is a valid scenario where a sensor owner will allow a remote user to command their sensor. Until a valid use case is identified, it would be premature to address commands in the SIF-SP.

8.5 Sensing

The conversion of physical phenomena into digital output is outside the scope of this document. Neither address specifically how the sensor converts a physical phenomenon into a digital measurement.

8.6 Information Assurance

Applicable Specifications:

- ARH.XML (Access Rights and Handling)
- ISM.XML (Information Security Markings)
- ITU-T Rec. X.509 (PKI)
- NTK.XML (Need to Know)
- UIAS (Security Attributes)
- LDAP (Attribute Authority)
- XACML (Security Policy Language)

The security infrastructure for the DoD and IC Enterprise consists of Attributed Based Access Control (ABAC) controls enabled by Public Key Infrastructure (PKI) Identification and Authentication (I&A) services and standard security markings on each resource.

8.6.1 PKI Infrastructure

The DoD and IC Public Key Infrastructure (PKI) provides a means of performing Identification and Authentication using public key exchange protocols. Public Key protocols allocate two keys to each entity. One is the public key. This key can be freely shared. The second is the private key. This is the secret key and must never be shared. The keys are generated such that anything encrypted with the private key can only be decrypted with the public key. Likewise, anything encrypted with the public key can only be decrypted with the private key.

A typical exchange is conducted as follows:

1) Bob asserts his identity to Jane through a message encrypted with Bob's private key
2) Jane receives the message
3) Jane retrieves a trusted public key for Bob from her Certificate Authority
4) Jane successfully decrypts the message from Bob using Bobs' public key

Since only Bob should have the private key, the message must have come from Bob.

If this exchange is performed in both directions, then both Jane and Bob have a high degree of confidence that the other party is who they claim to be.

8.6.2 Attribute Based Access Control

Attribute Based Access Controls (ABAC) restrict the privileges of a Person or Non-Person Entity based on a set of attributes assigned to that entity. The criteria required to access a resource are defined in a security policy. That policy can restrict access based on the requesting entities' identity, their security attributes, the targeted resource identity, the security markings for that resource, and any combination of the above.

The primary services which make up an ABAC infrastructure are:

- Attribute Authority: This is the register which holds the security attributes. There is one entry for each entity. Access to an entry requires that the requestor have the authenticated identity for that entity. This service is typically implemented using an LDAP directory service.
- Policy Decision Point (PDP): This is the service which processes the rules of the security policy in response to an access request. The results can be to either grant of refuse the access request.
- Policy Enforcement Point (PEP): This is the service which governs whether the entity can access the resource or not. In response to an access request from an entity, it gathers the necessary information and submits a request to the PDP. Whether or not it allows the access is determined by the response from the PDP.

8.6.3 Security Markings

Most security policies require security attributes for both the requesting entity and for the resource being accessed. The attributes for resources are usually included as a part of the resource metadata. These are the security markings. The standards for resource security markings are coordinated with but different from the CAPCO registered markings for documents. The difference is due to their intended use. The CAPCO register was established for human readable markings. They are not suitable for machine processing, like that required from a PDP. The DoD/IC security markings standards are designed for inclusion in Information Technology resources and processing by DoD/IC PDP services.

8.6.4 Cross-Domain Guards

Unfortunately, the ABAC infrastructure is not universal. Differences in security policy, levels of technical maturity, and levels of acceptable risk all serve to divide the DoD/IC environment into security domains. For example, systems on a secret domain cannot interoperate directly with those on an unclassified domain. Cross-Domain Guards facilitate the flow of information between these domains. These services apply release rules, and in some cases human review, to determine if a file meets the criteria for release to the target domain.

9 Conformance Description

Guidance: The conformance description clause describes what it means to conform to the standard. This clause expands upon the content of the introductory conformance clause found near the beginning of the standardization document.

This clause defines conformance classes and/or conformance levels, and describes the anticipated use cases that may have been introduced in the introductory conformance clause.

It may also be used to define conformance relationships as they may pertain to other standardization documents.

Conformance for the SIF Enterprise Technical View is defined by two conformance classes.

- **Sensor Web Enablement (SWE).** The first conformance class defines conformance with the OGC Sensor Web Enablement (SWE) body of standards. Conformance to this class is required of any Enterprise node which proposes to expose sensors and sensor data in accordance with the SIF-SP.
- **Distributed Data Framework (DDF).** The second conformance class defines conformance with the technology infrastructure as defined by the Distributed Data Framework (DDF). This is required of any implementation which will be deployed as a node on the DCGS Enterprise.

Details on the requirements and corresponding abstract tests are provided in Annex A .

Annex A Abstract Test Suite

Error! Reference source not found. provides a Requirement Trace Matrix which maps SIF-SP TV-3 r equirements to the abstract tests which validate compliance with each requirement. The tests are identified by their Annex A paragraph number.

Requirements
Requirement 1: A Component that claims to be conformant with the SIF-SP TV-1 SWE Conformance Class shall demonstrate compliance with the Core Requirements Class of the SOS standard. Tests: A.1.1
Requirement 2: A Component that claims to be conformant with the SIF-SP TV-1 SWE Conformance Class shall demonstrate compliance with the Requirements Classes from the Transactional Extension of the SOS standard. Tests: A.1.2
Requirement 3: A Component that claims to be conformant with the SIF-SP TV-1 SWE Conformance Class should demonstrate compliance with the Requirements Classes from the Results Handling Extension of the SOS standard. Tests: A.1.3
Requirement 4: A Component that claims to be conformant with the SIF-SP TV-1 SWE Conformance Class shall demonstrate compliance with the Requirements Classes from the Spatial Filtering Profile of the SOS standard.. Tests: A.1.4
Requirement 5: A Component that claims to be conformant with the SIF-SP TV-1 SWE Conformance Class shall demonstrate the ability to generate valid and complete DDMS 2.0 documents from the sensor metadata (SensorML) maintained by the SOS. Tests: A.1.5
Requirement 6: A Component that claims to be conformant with the SIF-SP TV-1 SWE Conformance Class shall demonstrate the ability to generate valid and complete DDMS 2.0 documents from the Observations (OM_Observation) maintained by the SOS.. Tests: A.1.6
Requirement 7: A Component that claims to be conformant with the SIF-SP TV-1 SWE Conformance Class shall demonstrate conformance to the client-side requirements of one or more of the DDF publication interfaces. Tests: A.1.7
Requirement 8: A Component that delivers Interactive Steaming measures shall use one or more of the Interactive Stream protocols identified in Table 16. Test: A.1.8
Requirement 9: A Component that delivers Coverage measures shall use one or more of the coverage formats identified in Table 17. Test: A.1.9
Requirement 10: A Component that delivers Measurement Streams shall use one or more of the Measurement Stream protocols identified in Table 16. Test: A.1.10

Requirements
Requirement 11: A Component that claims to be conformant with the SIF-SP TV-1 DDF Conformance Class shall demonstrate conformance to the requirements of one or more of the DDF discovery interfaces.   Test: A.2.1
Requirement 12: A Component that claims to be conformant with the SIF-SP TV-1 DDF Conformance Class shall demonstrate conformance to the server-side requirements of one or more of the DDF publication interfaces.   Test: A.2.2

A.1. SIF-SP TV-1 SWE Conformance Class Module

A.1.1 SOS Core Requirements

a) Test Purpose: Verify that the service implements the core requirements from the OGC Sensor Observation Service standard.
b) Test Method: See Section 14.1 of the Sensor Observation Service standard
c) References: Sensor Observation Service standard cited in the Normative Specifications section.
d) Test Type: Capability

A.1.2 SOS Transaction Requirements

a) Test Purpose: Verify that the service implements the requirements from the Transactional Extension to the OGC Sensor Observation Service standard.
b) Test Method: See Section 14.3 of the Sensor Observation Service standard
c) References: Sensor Observation Service standard cited in the Normative Specifications section.
d) Test Type: Capability

A.1.3 SOS Results Handling Requirements

a) Test Purpose: Verify that the service implements the requirements from the Results Handling Extension to the OGC Sensor Observation Service standard
b) Test Method: See Section 14.4 of the Sensor Observation Service standard
c) References: See Section 14.4 of the Sensor Observation Service standard
d) Test Type: Capability

A.1.4 SOS Spatial Filtering

a) Test Purpose: Verify that the service implements the requirements from the Spatial Filtering Profile of the OGC Sensor Observation Service standard.
b) Test Method: See Section 14.5 of the Sensor Observation Service standard
c) References: See Section 14.4 of the Sensor Observation Service standard
d) Test Type: Capability

A.1.5 DDMS Generation for Sensors

a) Test Purpose: Verify that the service can generate valid and complete DDMS 2.0 documents from SensorML metadata.
b) Test Method: Select a collection of SensorML documents from the service. For each document:

a. Generate a DDMS 2.0 document
b. Validate the DDMS 2.0 document against the DDMS 2.0 XML Schema and schematron rules
c. Validate that all information that could be populated in the DDMS document has been populated.

c) References:
   a. OGC SensorML Model and XML Encoding Standard version 2.0
   b. Department of Defense Discovery Metadata Specification 2.0
d) Test Type: Capability

A.1.6 DDMS Generation for Observations

a) Test Purpose: Verify that the service can generate valid and complete DDMS 2.0 documents from O&M Observations.
b) Test Method: Select a collection of O&M Observations from the service. For each Observation:
   a. Generate a DDMS 2.0 document
   b. Validate the DDMS 2.0 document against the DDMS 2.0 XML Schema and schematron rules
   c. Validate that all information that could be populated in the DDMS document has been populated.
c) References:
   a. OGC Observations and Measurements XML Implementation standard version 2.0
   b. Department of Defense Discovery Metadata Specification 2.0
d) Test Type: Capability

A.1.7 DDF Publication Client Side

a) Test Purpose: Test Purpose: Verify that a component that implements the SOS can successfully publish DDMS 2.0 documents to the DDF.
b) Test Method: Publish the DDMS documents generated through tests A.1.5 and A.1.6 to an instance of the DDF. Query the DDF using a standard client and verify that the DDMS documents can be discovered.
c) References: DDF documentation at http://www.codice.org/ddf/
d) Test Type: Capability

A.1.8 Interactive Streaming

a) Test Purpose: Verify that Interactive Streaming measures comply with the protocol standards identified in Table 16.
b) Test Method: For each Component which delivers Interactive Streaming measures, validate sample measures against the compliance criteria for the specified protocols.
c) References: See citation for JPIP in the Normative Specifications section.
d) Test Type: Capability

A.1.9 Coverages

a) Test Purpose: Verify that coverage measures comply with the standards identified in Table 17.
b) Test Method: For each Component which delivers coverage measures, validate sample measures against the compliance criteria for the specified format standards.
c) References: See citations for JPEG 2000, Exif, PNG, NITF, GeoTIFF and LAS in the Normative Specifications section

d) Test Type: Capability

A.1.10 Measurement Streams

a) Test Purpose: Verify that Measurement Streams comply with the standards identified in Table 16.
b) Test Method: For each Component which delivers Measurement Streams, validate sample measures against the compliance criteria for the specified protocol standards.
c) References: See citations for MISP in the Normative Specifications section
d) Test Type: Capability

A.2. SIF-SP TV-1 DDF Conformance Class Module

A.2.1 DDF Discovery

a) Test Purpose: Verify that service supports one or more of the DDF discovery interfaces.
b) Test Method: Publish the DDMS documents generated through tests A.1.5 and A.1.6 to the service. Query the service using a standard DDF client and verify that the DDMS documents can be discovered.
c) References: DDF documentation at http://www.codice.org/ddf/
d) Test Type: Capability

A.2.2 DDF Publication

a) Test Purpose: Verify that service supports one or more of the DDF publication interfaces.
b) Test Method: Publish the DDMS documents generated through tests A.1.5 and A.1.6 to the service. Query the service using a standard DDF client and verify that the DDMS documents can be discovered.
c) References: DDF documentation at http://www.codice.org/ddf/
d) Test Type: Capability

Annex B Terms and Definitions

For the purpose of this document, the following terms and definitions apply:

Abstract Test Case

A generalized test for a particular requirement. [ISO 19105]

Abstract Test Method

A method for testing an implementation that is independent of any particular test procedure. [ISO 19105]

Abstract Test Module

A set of related abstract test cases. Abstract test modules may be nested in a hierarchical way. [ISO 19105]

Abstract Test Suite (ATS)

A set of abstract test modules and associated abstract test cases that collectively specify all the requirements to be satisfied for conformance. [digest from ISO 19105]

Actuator <SensorML 2.0>

A type of transducer that converts a signal to some real-world action or phenomenon.

Aggregate Process <SensorML 2.0>

Composite process consisting of interconnected sub-processes, which can in turn be Simple Processes or themselves Aggregate Processes. An aggregate process can include possible data sources. A description of an aggregate process should explicitly define connections that link input and output signals of sub-processes together. Since it is a process itself, an aggregate process also has its own inputs, outputs and parameters.

Capability Test

A test designed to determine whether an IUT conforms to a particular characteristic of a standard as described in the test purpose. [ISO 19105]

Client <OWS Common 2.0>

software component that can invoke an operation from a server

Common Operational Picture (COP)

A single identical display of relevant information shared by more than one command that facilitates collaborative planning and assists all echelons to achieve situational awareness (JP3-0)

Compliance

Adherence to policy, directives, instructions, guidance, etc. Often used to define or mean the same as conformance. E.g. an implementation exhibits conformance when it complies with the conformance requirements of the applicable information standards.

Conformance

The fulfilment of specified requirements. [ISO 19105]

Conformance Class

Conformance classes may be used to group, define, and label different kinds of conformance requirements pertinent to implementation of the standard. [digest from ISO 19105]

Conformance Level

A conformance level is a special kind of conformance class in which the conformance requirements of a higher level contain all the requirements of the lower levels. [digest from ISO 19105]

Data Component <SensorML 2.0>

Element of sensor data definition corresponding to an atomic or aggregate data type

Note: A data component is a part of the overall dataset definition. The dataset structure can then be seen as a hierarchical tree of data components.

Detector <SensorML 2.0>

Atomic part of a composite Measurement System defining sampling and response characteristic of a simple detection device. A detector has only one input and one output, both being scalar quantities. More complex Sensors, such as a frame camera, which are composed of multiple detectors, can be described as a detector group or array using a System or Sensor model.

Determinand <SensorML 2.0>

A Parameter or a characteristic of a phenomenon subject to observation. Synonym for observable. [O&M]

Emitter <SIF-SP>

The component of an active sensor which emits a signal which is collected by the detector after interacting with a target.

Event <SWE Service Model>

anything that happens or is contemplated as happening at an instant or over an interval of time [OGC 09-032]

Event object <SWE Service Model>

object that represents, encodes, or records an event, generally for the purpose of computer processing [see OGC 09-032]

Executable Test Suite (ETS)

A set of executable test cases. [ISO 19105]

An executable test suite (ETS) is an instantiation of an ATS, in which all implementation-dependent parameters are assigned specific values. An executable test case is derived from an abstract test case and is in a form that allows it to be run on the IUT. Executable test cases result from the instantiation of specific values for parameters in abstract test cases. Executable test cases may be unique to each IUT. [digest from ISO 19105]

Feature <SensorML 2.0>

Abstraction of real-world phenomena [ISO 19101:2002, definition 4.11]

Note: A feature may occur as a type or an instance. Feature type or feature instance should be used when only one is meant.

Implementation Conformance Statement (ICS)

A statement made by the supplier of an implementation or system claimed to conform to a given standard (or set of standards/specifications), asserting which capabilities have been conformingly implemented. [digest from ISO 19105]

Implementation Under Test (IUT)

The realization of a specification that is the focus of test. [digest from ISO 19105]

Interface <OWS Common 2.0>

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

Location <SensorML 2.0>

A point or extent in space relative to a coordinate system. For point-based systems, this is typical expressed as a set of n-dimensional coordinates within the coordinate system. For bodies, this is typically expressed by relating the translation of the origin of an object's local coordinate system with respect to the origin of an external reference coordinate system.

Measurand <SensorML 2.0>

Physical parameter or a characteristic of a phenomenon subject to a measurement, whose value is described using a Measure (ISO 19103). Subset of determinand or observable. [O&M]

Measure (noun) <SensorML 2.0>

Value described using a numeric amount with a scale or using a scalar reference system [ISO/TS 19103]. When used as a noun, measure is a synonym for physical quantity

Measurement (noun) <SensorML 2.0>

An observation whose result is a measure [O&M]

Measurement (verb) <SensorML 2.0>

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

Multiplexed Data Stream <SensorML 2.0>

A data stream that consists of disparate but well-defined data packets within the same stream.

Notification <SWE Service Model>

synonym: message

container for event objects [see OGC 09-032]

Observable, Observable Property (noun) <SensorML 2.0>

A parameter or a characteristic of a phenomenon subject to observation. Synonym for determinand.[O&M]

A physical property of a phenomenon that can be observed and measured (e.g. temperature, gravitational force, position, chemical concentration, orientation, number-of-individuals, physical switch status, etc.), or a characteristic of one or more feature types, the value for which will be estimated by application of some procedure in an observation. It is thus a physical stimulus that can be sensed by a detector or created by an actuator.

Observation <SensorML 2.0>

Act of observing a property or phenomenon [ISO/DIS 19156, definition 4.10]

Note: The goal of an observation may be to measure, estimate or otherwise determine the value of a property

Observation Offering <SOS 2.0>

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

Observation Result <O&M 2.0>

estimate of the valueof a property determined through a known procedure [ISO/DIS 19156:2010]

Observed Property <SOS 2.0>

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

Observed Value <SensorML 2.0>

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

Operation <OWS Common 2.0>

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

Orientation <SensorML 2.0>

The rotational relationship of an object relative to an external coordinate system. Typically expressed by relating the rotation of an object’s local coordinate axes relative to those axes of an external reference coordinate system.

Parameter <OWS Common 2.0>

variable whose name and value are included in an operation request or response

Performer <DoDAF 2.0>

Any entity - human, automated, or any aggregation of human and/or automated - that performs an activity and provides a capability.

Phenomenon <SensorML 2.0>

A physical state that can be observed and its properties measured.

Physical System <SensorML 2.0>

An aggregate model of a group or array of process components, which can include detectors, actuators, or sub-systems. A Physical System relates an Aggregate Process to the real world and therefore provides additional definitions regarding relative positions of its components and communication interfaces.

Platform <OWS Common 2.0>

the underlying infrastructure in a distributed system (Adapted from ISO 19119)

NOTE A platform describes the hardware and software components used in a distributed system. To achieve interoperability, an infrastructure that allows the components of a distributed system to interoperate is needed. This infrastructure, which may be provided by a Distributed Computing Platform (DCP), allows objects to interoperate across computer networks, hardware platforms, operating systems and programming languages. (Adapted from Subclause 10.1 of ISO 19119)

Position <SensorML 2.0>

The location and orientation of an object relative to an external coordinate system. For body-based systems (in lieu of point-based systems) is typically expressed by relating the object's local coordinate system to an external reference coordinate system. This definition is in contrast to some definitions (e.g. ISO 19107) which equate position to location.

Procedure <SOS 2.0>

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

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

Process <SensorML 2.0>

An operation that takes one or more inputs, and based on a set of parameters, and a methodology generates one or more outputs.

Pro forma

Latin for the term "form".

Property <SensorML 2.0>

Facet or attribute of an object referenced by a name [ISO/DIS 19143:2010]

Example : Abby's car has the color red, where "color" is a property of the car instance, and "red" is the value of that property.

Reference Frame <SensorML 2.0>

A coordinate system by which the position (location and orientation) of an object can be referenced.

Reference Implementation (RI)

A conformant, trusted, or well-known exemplar implementation of one or more standards used to support standards conformance and interoperability testing. In some instances, the RI is suitable for reuse by developers in their own instantiations of the standardized function or service.

Request <OWS Common 2.0>

invocation of an operation by a client

Response <OWS Common 2.0>

result of an operation, returned from a server to a client

Resource <OWS Common 2.0>

any addressable unit of information or service [IETF RFC 2396]

EXAMPLES Examples include files, images, documents, programs, and query results.

NOTE The means used for addressing a resource is a URI (Uniform Resource Identifier) reference

Result <SensorML 2.0>

An estimate of the value of some property generated by a known procedure. [O&M]

Sensor <SensorML 2.0>

An entity capable of observing a phenomenon and returning an observed value. Type of observation procedure that provides the estimated value of an observed property at its output.

Note: A sensor uses a combination of physical, chemical or biological means in order to estimate the underlying observed property. At the end of the measuring chain electronic devices often produce signals to be processed.

Sensor System <SOS 2.0>

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

Service Metadata <OWS Common 2.0>

metadata describing the operations and geographic information available at a server [ISO 19128 draft]

Standards Conformance Testing

Testing performed to determine the extent to which a system or subsystem adheres to or implements a standard. It involves testing the capabilities of an implementation against both the conformance requirements in the relevant standard(s) and the statement of the implementation's capabilities. [NSGM 3202]

Subscription <SWE Service Model>

represents the relationship between consumer and producer, including any content or channel filter, along with any relevant policy and context information (compare with OASIS WS-BaseNotification)

System Under Test (SUT)

The computer hardware, software and communication network required to support an IUT. [ISO 19105]

Target <SIF-SP>

Synonymous with Feature of Interest.

Value <SensorML 2.0>

A member of the value-space of a datatype. A value may use one of a variety of scales including nominal, ordinal, ratio and interval, spatial and temporal. Primitive datatypes may be combined to form aggregate datatypes with aggregate values, including vectors, tensors and images. [ISO11404]

Annex C Abbreviations

In this document the following abbreviations and acronyms are used or introduced:

ABAC	Attribute Based Access Control
AOI	Area of Interest
ATS	Abstract Test Suite
CAPCO	Controlled Access Program Coordinating Office
CDR	Content Discovery and Retrieval
CDR SF	CDR Specification Framework
CIO	Chief Information Officer
CMSTT	Community Metadata Standards Tiger Team
COP	Common Operational Picture
CRUD	Create, Read, Update, Delete
CSW	Catalogue Service for the Web
DCGS	Distributed Common Ground Station
DDF	Distributed Data Framework
DI2E	Defense Integrated Intelligence Enterprise
DIB	DCGS Integrated Backbone
DoD	Department of Defense
EO	Earth Observation
ETS	Executable Test Suite
FMV	Full Motion Video
HTTP	Hypertext transfer protocol
GUIDE	Globally Unique Identifier for Everything
GWG	Geospatial-Intelligence Standards Working Group
GWS	Geospatial Web Service
I&A	Identification and Authentication
IC	Intelligence Community
ICS	Implementation Conformance Statement
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force

IP	Internet Protocol
ISO	International Organization for Standardization
IUT	Implementation Under Test
ITU-T	Telecommunication Standardization Sector of the International Telecommunications Union
JESC	Joint Enterprise Standards Committee
JSON	JavaScript Object Notation
KVP	Key Value Pair
LDAP	Lightweight Directory Access Protocol
MDC	Metadata Catalogue
MDF	Metadata Framework
MISP	Motion Imagery Standards Profile
NGA	National Geospatial-Intelligence Agency
NSG	National System for Geospatial-Intelligence
O&M	Observations and Measurements
OASIS	Organization for the Advancement of Structured Information Standards
ODNI	Office of the Director of National Intelligence
OGC	Open Geospatial Consortium
OWF	Ozone Widget Framework
OWS	OGC Web Services
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PKI	Public Key Infrastructure
REST	Representational State Transfer
RI	Reference Implementation
SensorML	Sensor Model Language
SIF	Sensor Integration Framework
SIF-SP	Sensor Integration Framework Standards Profile
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol

SOS	Sensor Observation Service
SPS	Sensor Planning Service
SUT	System Under Test
SWE	Sensor Web Enablement
UML	Unified Modeling Language
USMS	US MASINT System
XML	eXtended Markup Language

Annex D Distributed Data Framework (DDF)

Applicable Specifications:

- Distributed Data Framework documentation page at http://www.codice.org/ddf/documentation.htm

The Distributed Data Framework (DDF) is a free and open-source common data layer that abstracts services and business logic from the underlying data structures to enable rapid integration of new data sources. It forms the common software base for Distributed Common Ground System (DCGS) family of systems. DCGS in turn forms the backbone of the Defense Intelligence Information Enterprise (DI2E). DI2E is the DoD enterprise for information sharing.

D.1. Metadata

In the DDF, resources are the data products, files, reports, or documents of interest to users of the system. Metadata is information about those resources, organized into a schema to make search possible. The Catalog stores this metadata and allows access to it. Metacards are single instances of metadata, representing a single resource, in the Catalog. Metacards follow one of several schemas to ensure reliable, accurate, and complete metadata. Essentially, Metacards function as containers of metadata.

D.2. Search

DDF provides the capability to search the Catalog for metadata. There are a number of different types of searches that can be performed on the Catalog, and these searches are accessed using one of several interfaces. This section provides a very high level overview of introductory concepts of searching with DDF. These concepts are expanded upon in later sections.

There are four basic types of metadata search. Additionally, any of the types can be combined to create a compound search.

D.2.1 Text Search

A text search is used when searching for textual information. It searches all textual metadata fields by default, although it is possible to refine searches to a text search on a single attribute. It is similar to a Google search over the metadata contained in the Catalog. Text searches may use wildcards, logical operators, and approximate matches.

D.2.2 Spatial Search

A spatial search is used for Area of Interest (AOI) searches. Polygon and point radius searches are supported.

D.2.3 Temporal Search

A temporal search finds information from a specific time range. Two types of temporal searches are supported: relative and absolute. Relative searches contain an offset from the current time, while absolute searches contain a start and an end timestamp. Temporal searches can use the created or modified date attributes.

D.2.4 Datatype Search

A datatype search is used to search for metadata based on the datatype of the resource. Wildcards (*) can be used in both the datatype and version fields. Metadata that matches any of the datatypes (and associated versions if specified) will be returned. If a version is not specified, then all metadata records for the specified datatype(s) regardless of version will be returned.

D.3. Ingest

Ingest is the process of bringing data products, metadata, or both into the catalog to enable search, sharing, and discovery. Ingested files are transformed into a neutral format that can be searched against as well as migrated to other formats and systems. See Section 8.2.1.2 for the various methods of ingesting data.

Upon ingest, a transformer will read the metadata from the ingested file and populate the fields of a metacard. Exactly how this is accomplished depends on the origin of the data, but most fields (except id) are imported directly.

D.4. Content

The Catalog Framework can interface with Storage Providers to provide storage of resources to specific types of storage, e.g., file system, relational database, XML database. A default file system implementation is provided by default.

Storage providers act as a proxy between the Catalog Framework and the mechanism storing the content. Storage providers expose the storage mechanism to the Catalog Framework. Storage plugins provide pluggable functionality that can be executed either immediately before or immediately after content has been stored or updated.

Storage providers provide the capability to the Catalog Framework to create, read, update, and delete content in the content repository.

D.5. Catalog Framework

The Catalog Framework wires all the Catalog components together.

It is responsible for routing Catalog requests and responses to the appropriate source, destination, federated system, etc.

Endpoints send Catalog requests to the Catalog Framework. The Catalog Framework then invokes Catalog Plugins, Transformers, and Resource Components as needed before sending requests to the intended destination, such as one or more Sources.

The Catalog Framework decouples clients from service implementations and provides integration points for Catalog Plugins and convenience methods for Endpoint developers.

D.6. Federation

Federation is the ability of the DDF to query other data sources, including other DDFs. By default, the DDF is able to federate using OpenSearch and CSW protocols. The minimum configuration necessary to configure those federations is to supply a query address.

Federation enables constructing dynamic networks of data sources that can be queried individually, or aggregated into specific configuration to enable a wider range of accessibility for data and data products.

Federation provides the capability to extend the DDF enterprise to include Remote Sources, which may include other instances of DDF. The Catalog handles all aspects of federated queries as they are sent to the Catalog Provider and Remote Sources, as they are processed, and as the query results are returned. Queries can be scoped to include only the local Catalog Provider (and any Connected Sources), only specific Federated Sources, or the entire enterprise (which includes all local and Remote Sources). If the query is supposed to be federated, the Catalog Framework passes the query to a Federation Strategy, which is responsible for querying each federated source that is specified. The Catalog Framework is also responsible for receiving the query results from each federated source and returning them to the client in the order specified by the particular federation strategy used. After the federation strategy handles the results, the Catalog returns them to the client through the Endpoint. Query results returned from a federated query are a list of metacards. The source ID in each metacard identifies the Source from which the metacard originated.

D.7. Events and Subscriptions

DDF can be configured to receive metacards whenever metadata is created, updated, or deleted in any federated sources. Creations, updates, and deletions are collectively called Events, and the process of registering to receive them is called Subscription.

The behavior of these subscriptions is consistent, but the method of configuring them is specific to the Endpoint used.

D.8. Registry

The Registry Application serves as an index of registry nodes and their information, including service bindings, configurations and supplemental details.

Each registry has the capability to serve as an index of information about a network of registries which, in turn, can be used to connect across a network of DDFs and other data sources. Registries communicate with each other through the CSW endpoint and each registry node is converted into a registry metacard to be stored in the catalog. When a registry is subscribed to or published from, it sends the details of one or more nodes to another registry.

D.8.1 Identity Node

The Registry is initially comprised of a single registry node, referred to as the identity, which represents the registry’s primary configuration.

D.8.2 Subscription

Subscribing to a registry is the act of retrieving its information, specifically its identity information and any other registries it knows about. By default, subscriptions are configured to check for updates every 30 seconds.

D.8.3 Publication

Publishing is the act of sending a registry's information to another registry. Once publication has occurred, any updates to the local registry will be pushed out to the registries that have been published to.

D.9. Endpoints

Endpoints expose the Catalog Framework to clients using protocols and formats that the clients understand.

Endpoint interface formats encompass a variety of protocols, including (but not limited to):

- SOAP Web services
- RESTful services
- JMS
- JSON
- OpenSearch

The endpoint may transform a client request into a compatible Catalog format and then transform the response into a compatible client format. Endpoints may use Transformers to perform these transformations. This allows an endpoint to interact with Source(s) that have different interfaces. For example, an OpenSearch Endpoint can send a query to the Catalog Framework, which could then query a federated source that has no OpenSearch interface.

Endpoints are meant to be the only client-accessible components in the Catalog.

The following endpoints are available in a standard installation of DDF:

1. Application Upload Endpoint: Uploads new/upgraded applications to the system.
2. Catalog REST Endpoint: Allows clients to perform CRUD operations on the Catalog using REST, a simple architectural style that performs communication using HTTP. Implements REST specification.
3. CometD Endpoint: Enables asynchronous search capabilities. Implements CometD.
4. CSW Endpoint: Enables a client to search collections of descriptive information (metadata) about geospatial data and services. Implements Catalogue Services for Web (CSW) standard, XML-RPC, ISO 19115/ISO191119.
5. FTP Endpoint: Provides a method for ingesting files directly into the DDF Catalog using the FTP protocol. Implements FTP.
6. IdP Endpoint: Enables configuration of DDF as an IdP server. Implements SAML 2.0.
7. KML Endpoint: Allows a user to generate a view-based KML Query Results Network Link. This network link can be opened with Google Earth, establishing a dynamic connection between Google Earth and DDF. Implements Keyhole Markup Language.
8. Metrics Endpoint: Used by the Metrics Collection Application to report on system metrics.
9. Endpoint: Provides an endpoint that a client accesses to send query parameters and receive search results. Implements JAX-RS, [CDR IPT BrokeredSearch], CDR IPT OpenSearch, CDR REST Brokered Search 1.1, CDR REST Search v3.0, and OpenSearch.

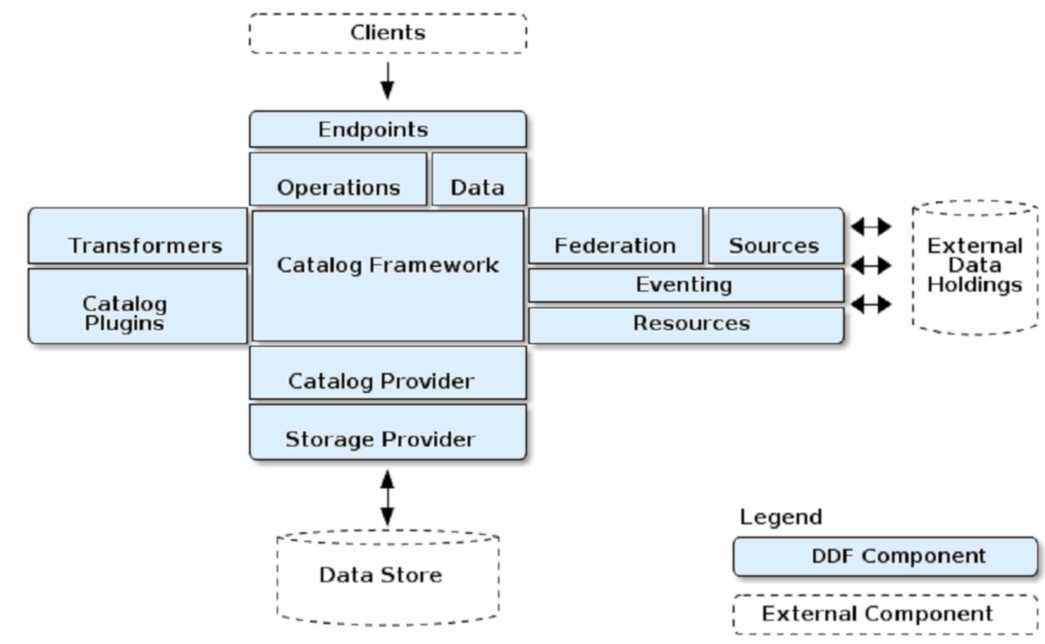


Figure 12 : DDF Catalog Architecture

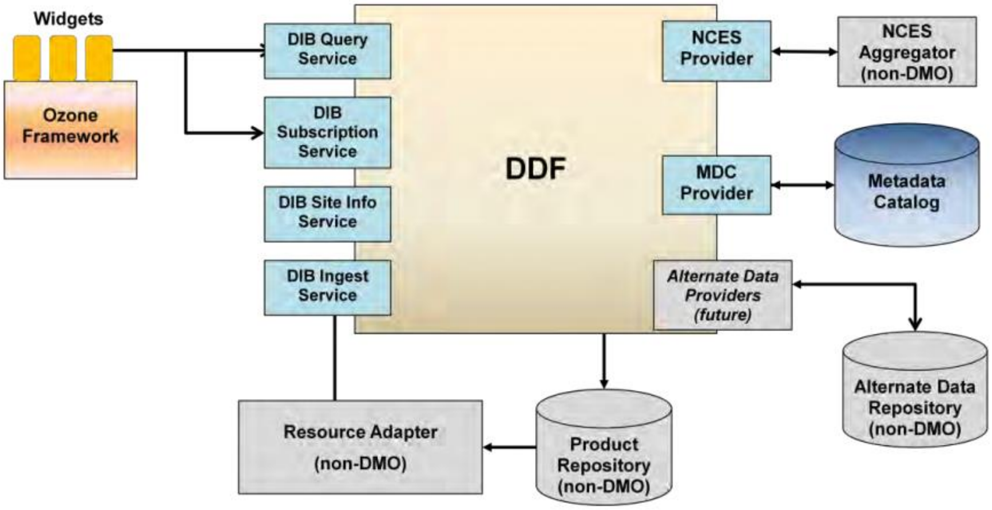
D.10. DIB 4.0

The current version of the DIB is version 4.0 released on 27 March 2012. This release introduced several major improvements including:

From the DIB:

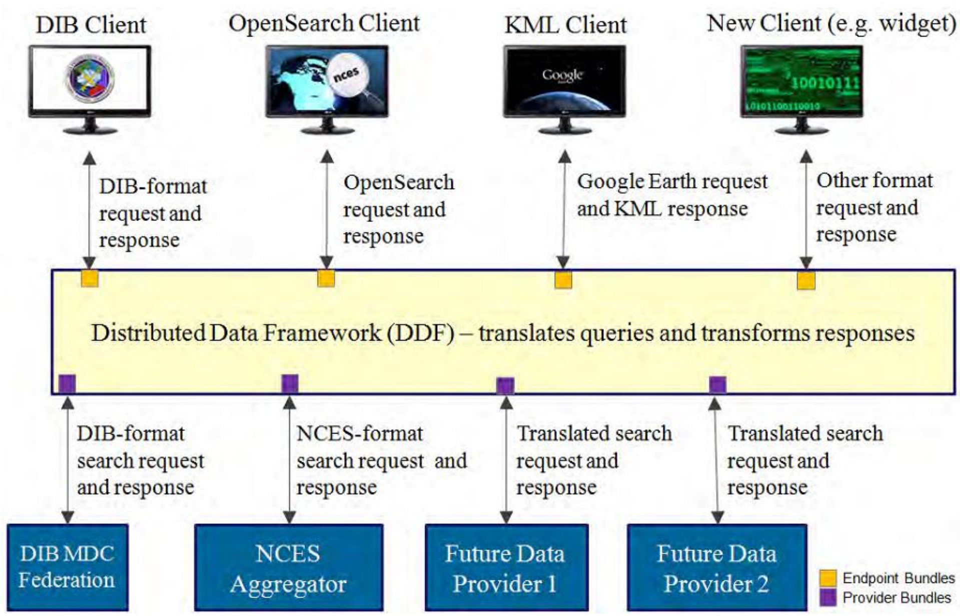
1. Componentizes development and delivery to simplify the integration of new web service components and data sources.
2. Separates DIB capabilities (e.g., portal, service registry, metadata catalog) to facilitate integration of individual components, as needed. Provides both cost avoidance (no need to maintain expensive, single-solution, interface adapters), and data exposure to a larger set of applications, analytics, and user interfaces (e.g., widgets, portals).
3. Decreases dependence on specific software products such as JBoss, Oracle WebLogic, and Oracle Database.
4. Maximizes value of agile development by moving "away from high-risk waterfall product release processes towards lower risk, incremental feature releases".
5. Incorporates new Department of Defense/Intelligence Community (DoD/IC) Content Discovery & Retrieval (CD&R) specifications, broadening the scope of DIB's existing exposure, search, discovery and retrieval capabilities by embracing the community's standards.
6. Provides the option to improve the performance of DIB node interactions in low-bandwidth environments through the use of Efficient XML Interchange (EXI).
7. Continues support for Attribute-Based Access Control (ABAC) security capability, using either the security services Reference Implementation (RI) from DIB v2.0, or any third-party RI conformant security services implementation.

8. Introduces the Distributed Data Framework (DDF) as a means to evolve the legacy DIB Metadata Framework (MDF) to abstract services and business logic from underlying data structures and expose heterogeneous data sources (e.g., NCES).



From the DDF:

1. Provides a flexible integration framework with Advanced Programming Interfaces (APIs) to facilitate customization of queries and results (e.g., sorting preferences, KML conversion), while maintaining interoperability. The APIs provide a defined and extensible set of interfaces to support quick and easy integration with a variety of data repositories and/or applications.
2. Exists as Government Open Source Software (GOSS) based on a Free and Open Source Software (FOSS) core.
3. Replaces the legacy DIB portal with an Ozone Widget Framework (OWF) interface to leverage on-going community investment in widget development.
4. Re-hosts existing web service interfaces from the legacy Metadata Framework (MDF) that serve as the basis for interoperability and backward-compatibility, while deprecating MDF code.
5. Provides a data abstraction layer, enabling integrators to decouple user interfaces (portals, widgets, etc.) from the underlying data repositories, thereby breaking application „stove-pipes" and facilitating migration to, and maintenance of, a service oriented architecture (SOA).
6. Provides a standard means of interfacing to not only the Metadata Catalog (MDC), the operationally-proven community standard for federated data-sharing that continues to exist as a key DIB feature, but also to a wide range of non-MDC based sources of command and control (C2), Intelligence Community (IC), unstructured, and "big data".


DIB Client
OpenSearch Client
KML Client
New Client (e.g. widget)
DIB-format request and response
OpenSearch request and response
Google Earth request and KML response
Other format request and response
Distributed Data Framework (DDF) – translates queries and transforms responses
DIB-format search request and response
NCES-format search request and response
Translated search request and response
Translated search request and response
DIB MDC Federation
NCES Aggregator
Future Data Provider 1
Future Data Provider 2
Endpoint Bundles
Provider Bundles

Annex E Sensor Web Enablement (SWE) Common

Applicable Specifications:

- SWE Common 2.0

The primary focus of the SWE Common Data Model is to define and package sensor-related data in a self-describing and semantically enabled way. The main objective is to achieve interoperability, first at the syntactic level, and later at the semantic level (by using ontologies and probably semantic mediation) so that sensor data can be better understood by machines, processed automatically in complex workflows and easily shared between nodes.

E.1. Data Components

All SWE Common data components are descended from the AbstractDataComponent class illustrated in Figure 3. This foundation provides a common approach to identify and define the semantics of a SWE data component.

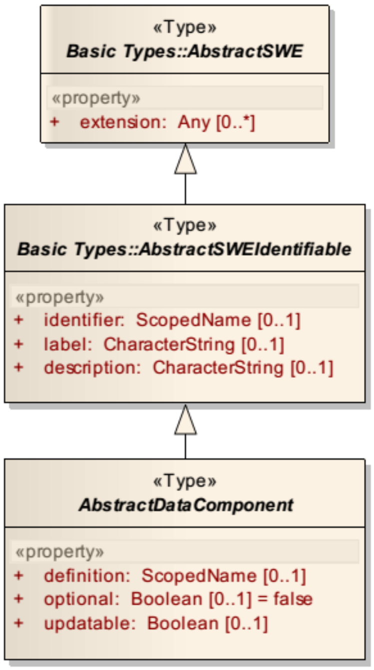


Figure 13 : SWE Common Root Classes

The elements that make up the AbstractDataComponent, and by extension all SWE data components are:

- Identifier: The globally unique identifier for this component. This identifier is mandatory for any SIF data component which may be independently managed or accessed. See Section 2.1 for the applicable specifications.
- Label: a short descriptive name for the component
- Description: a narrative description of the component
- Definition: Identifies the semantics of the data component. This element is a scoped name that references an entry in the SIF-SP ontology.

- Optional: Indicates if the component value can be omitted in the data stream.
- Updatable: Indicates if the component value is fixed or can be updated.

SWE Common specializes AbstractDataComponent into fourteen data types. The syntax of every SWE data component is defined in terms of these fourteen types. For purposes of organization, these types are organized into four categories.

E.1.1 Simple Types

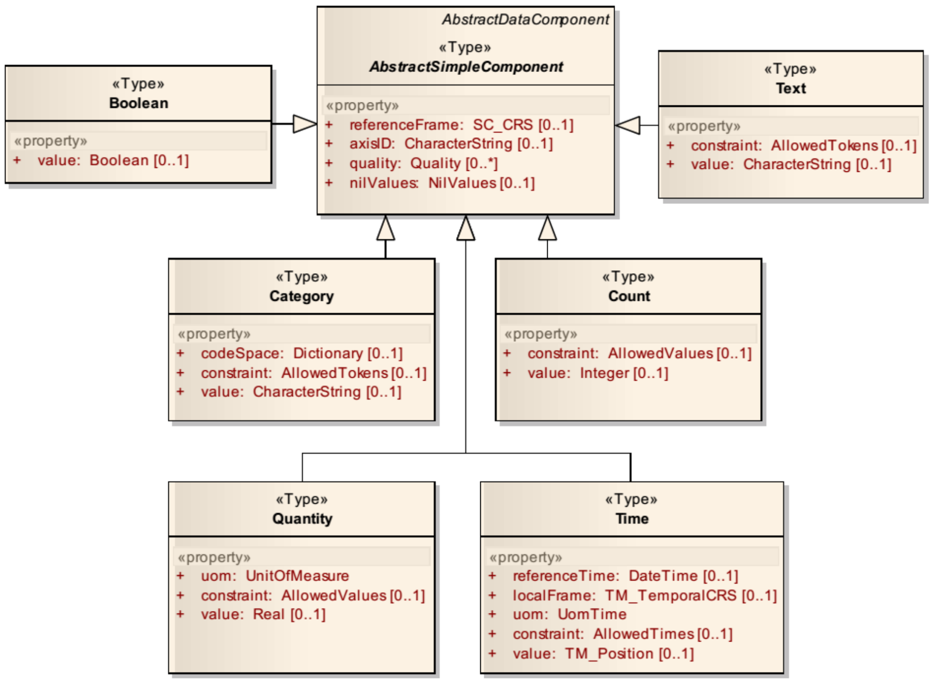


Figure 14 : Simple Data Components

The SWE Common Simple data types are:

- **Boolean:** A Boolean representation of a property can take only two values that should be “true/false” or “yes/no”.
- **Text:** A textual representation is useful for providing human readable data, expressed in natural language, as well as various alphanumeric tokens that cannot be assigned to well-defined categories.
- **Category:** A categorical representation is a type of discrete representation of a property that only allows picking a value from a well-defined list of possibilities (i.e. categories).
- **Count:** Discrete countable properties are represented through a numerical integer representation. They do not require a unit since the unit is always the unit of count.
- **Quantity:** A quantity is used for continuous values and is represented by a decimal (often floating point) number associated to a scale or unit of measure. The unit specification is mandatory even for quantities such as ratios that have no physical unit.

- **Time:** Represents a value with a date-time representation that is projected along the axis of a temporal reference frame.

E.1.2 Range Types

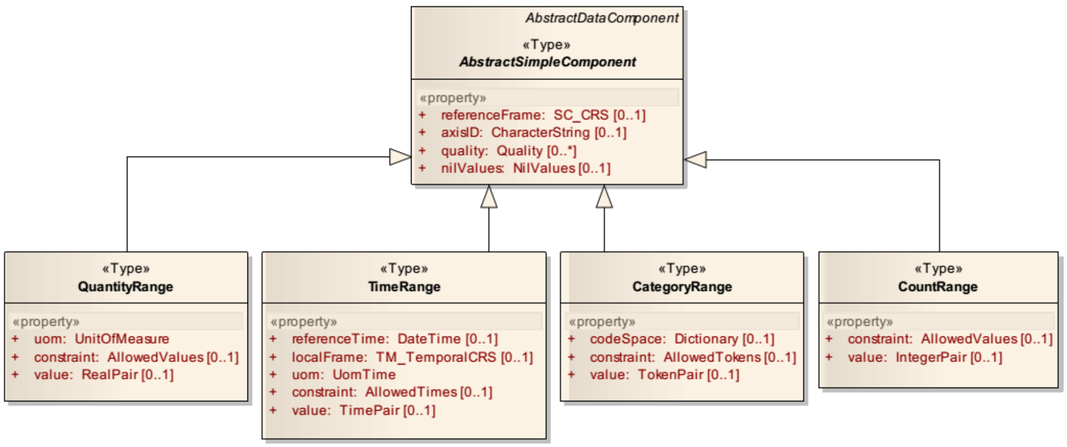


Figure 15 : Range Data Components

The SWE Common Range data types are:

- **CategoryRange:** express a value extent using the categorical representation of a property.
- **CountRange:** express a value extent using the count representation of a property.
- **QuantityRange:** express a value extent using the quantity representation of a property.
- **TimeRange:** express a value extent using the time representation of a property.

E.1.3 Record Types

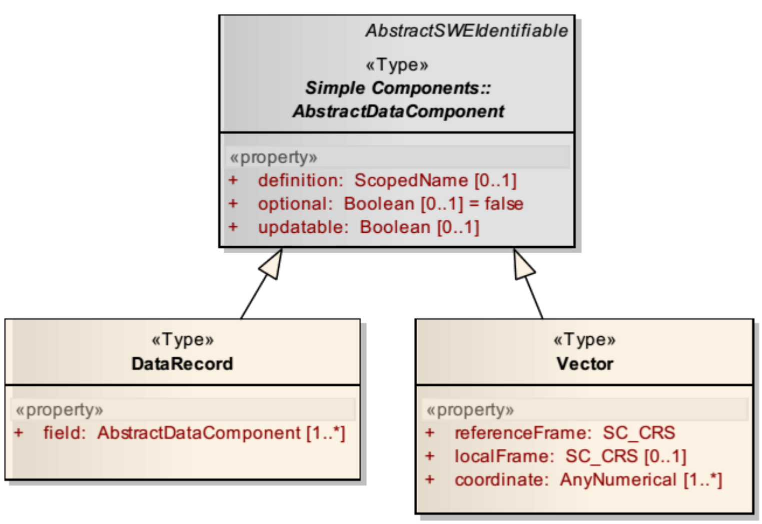


Figure 16 : Record Data Components

The SWE Record data types are:

- **DataRecord:** a record is a composite data type composed of one to many fields, each of which having its own name and type definition. Thus it defines some logical collection of components of any type that are grouped for a given purpose.
- **Vector:** used to express multi-dimensional quantities with respect to a well-defined referenced frame (usually a spatial or spatio-temporal reference frame).

DataRecord presents a conundrum. It is defined as a collection of one to many fields. A field is an instance of AbstractDataComponent. How then do we distinguish one field from another? SWE Common provides us with two elements to address this problem.

1. Field name: It is not clear from the UML diagram but each field in the data record has an associated "name" attribute. This name should be used to identify each field in the record. These names are not guaranteed to be globally unique so they can only serve to identify fields within the context of the record in which they appear.
2. Field definition: Every "field" element in a Data record is descended from the AbstractDataComponent class. Therefore, each field contains a "definition" element. These elements are used to reference an entry in the SIF-SP ontology. That entry provides a semantic definition of the referencing field.

The result is that every field in a record contains a name and a definition for that field.

E.1.4Block Types

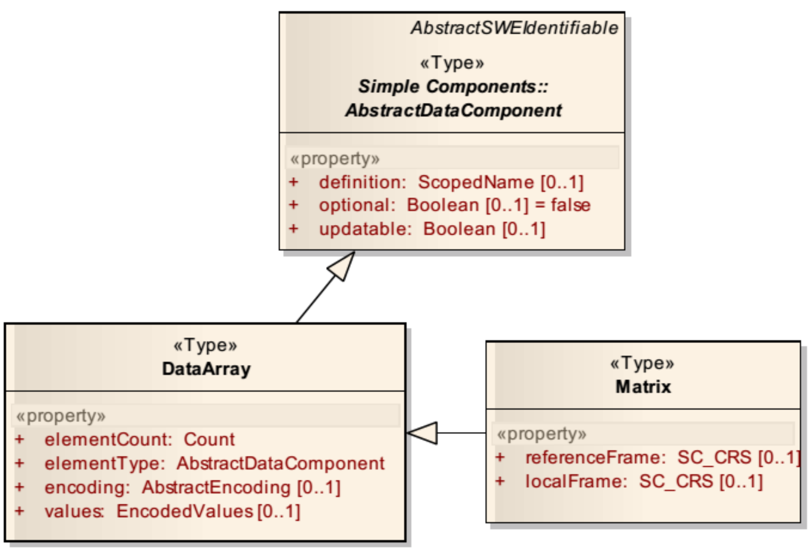


Figure 17 : Block Data Components

The SWE Block data types are:

- DataArray: a collection of elements of the same type (as opposed to a record where each field can have a different type), with a defined size.
- Matrix: A form of DataArray which includes a reference frame within which the matrix elements are expressed and a local frame of interest.

E.2. DataStream

A "DataStream" is similar to a "DataArray" class but it does not derive from AbstractDataComponent. As a result, it cannot be used as a child of other aggregate components and does not possess a sematic definition (no definition element).

Data Streams should be used as the metadata object containing the information essential to process a data of stream. In this context, a stream of data is a sequence of entities, each with the same structure. An important feature is that a stream can be open ended (i.e. the number of elements is not known in advance) and is thus designed to support real time streaming of data.

Data Streams also support out-of-band delivery of the values. The values element can be an encoding of the values, or it can reference another location where those values reside. In this way it allows separation of the metadata describing the data from the data values themselves.

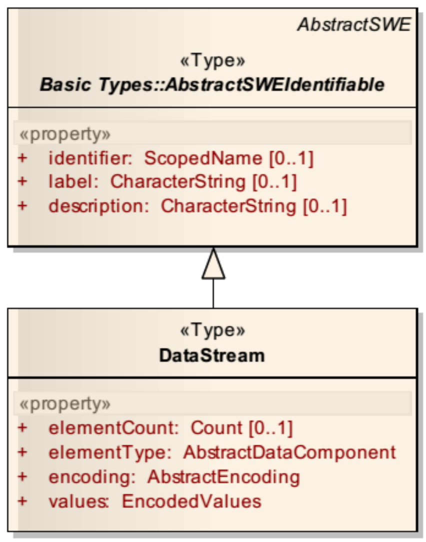


Figure 18 : Data Streams

The elements that make up the Data Stream are:

- Identifier: The globally unique identifier for this component. This identifier is mandatory for any SIF data component which may be independently managed or accessed. See Section 0 for the applicable specification.
- Label: a short descriptive name for the component
- Description: a narrative description of the component
- ElementCount: used to indicate the number of elements in the stream if it is known. For example, the frame count of a FMV clip would be known. The frame count for a FMV stream would be unknown.
- ElementType: Defines the structure of each element in the stream. These elements do not include any values.
- Encoding: specifies how the elements are encoded

- Values: provides the encoded values provided through the stream. This element can be a reference to another location such as a FMV feed.

E.3. Encodings

A key concept of the SWE Common Data Model is the ability to separate data values from the description of the data structure, semantics and representation. This allows verbose metadata to be used in order to robustly define the content and meaning of a dataset while still being able to package the data values in very efficient manners. It also allows the data values to be passed separate (out of band) from the metadata which describes those values.

Data encoding methods define how the data is packed as blocks that can efficiently be transferred or stored using various protocols and formats. Different methods allow encoding the data as XML, text (CSV like), binary and even compressed or encrypted formats in a way that is agnostic to a particular structure. This allows any of the encodings methods to be selected and used based on a particular requirement, such as performance, re-use of tools, alignment with existing standards and so on.

Annex F Implementation Conformance Statement (ICS)

An ICS is a statement made by the supplier of an implementation or system claimed to conform to a given standard (or set of standards/specifications), asserting which capabilities have been conformingly implemented. An ICS provides a uniform means for the implementer to declare the mandatory, conditional, and optional provisions of the standard that were actually implemented.

The following ICS may be used by the supplier or sponsor of an implementation as a framework to document the standards conforming capabilities of the implementation of this standard

SIF-SP TV-1 - Implementation Conformance Statement (ICS)

B=Baseline KML P=Profile Obligation I=Implemented P/F=Pass/Fail

M=Mandatory O=Optional C=Conditional

Implementation Under Test: **Conformance Level (1, 2 or 3):**
Test Point: **Profile Identifier:**
Date of Initial ICS Completion: **Test Sponsor:**
Date of Test Completion: **Test Organization:**

SWE Conformance Level – component can share Sensors and Sensor data in accordance with SIF-SP standards.	Component conforms to the Core Requirements Class of the SOS standard	M			
	Component conforms to the Transaction Extension of the SOS standard.	M			
	Component conforms to the Results Handling Extension of the SOS standard	M			
	Component conforms to the Spatial Filtering Profile of the SOS standard.	M			
	Component can generate valid SensorML documents	M			
	Component can generate valid SOS Observation Offerings	M			
	Component can generate valid O&M Observation documents	M			
	Component can generate valid DDMS 2.0 documents for described sensors	M			
	Component can generate valid DDMS 2.0 documents for Observations	M			
	Component delivers Interactive Streaming measures using SIF-SP identified standards	M			
	Component delivers Coverage measures using SIF-SP identified standards	M			
	Component delivers Measurement Stream measures using SIF-SP identified standards	M			
DDF Conformance Class – component can participate as a member of the DCGS federation.	Component implements one or more of the DDF discovery interfaces	M			
	Component implements server side publication interfaces defined by the DDF.	M			