OGC® DOCUMENT: 21-049

**External identifier of this OGC® document:** http://www.opengis.net/doc/is/poi/1.0

Open Geospatial Consortium

# OGC (POINTS OF INTEREST)

## STANDARD
## Implementation

**DRAFT**

**Version:** 1.0
**Submission Date:** 2024-02-08
**Approval Date:** 2029-03-30
**Publication Date:** 2024-02-08
**Editor:** Charles Heazel, Matthew Brian, John Purss

**Notice for Drafts:** This document is not an OGC Standard. This document is distributed for review and comment. This document is subject to change without notice and may not be referred to as an OGC Standard.
Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

**License Agreement**

Use of this document is subject to the license agreement at https://www.ogc.org/license

Suggested additions, changes and comments on this document are welcome and encouraged. Such suggestions may be submitted using the online change request form on OGC web site: http://ogc.standardstracker.org/

**Copyright notice**

Copyright © 2024 Open Geospatial Consortium
To obtain additional rights of use, visit https://www.ogc.org/legal

**Note**

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

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

# CONTENTS

I. ABSTRACT ..... vii

II. KEYWORDS ..... vii

III. PREFACE ..... viii

IV. SECURITY CONSIDERATIONS ..... ix

V. SUBMITTING ORGANIZATIONS ..... x

1. SCOPE ..... 2

2. CONFORMANCE ..... 4
   - 2.1. Implementation Specifications ..... 4
   - 2.2. Implementations ..... 4
   - 2.3. Conformance Classes ..... 4

3. NORMATIVE REFERENCES ..... 6

4. TERMS AND DEFINITIONS ..... 8

5. CONVENTIONS ..... 12
   - 5.1. Identifiers ..... 12
   - 5.2. UML Notation ..... 12

6. POI CORE REQUIREMENTS ..... 17
   - 6.1. Feature Model ..... 17
   - 6.2. Geometry ..... 19
   - 6.3. POI ISO Extensions ..... 21
   - 6.4. POI Class Model ..... 27
   - 6.5. POI Payload ..... 31
   - 6.6. POI Data Dictionary ..... 34

ANNEX A (INFORMATIVE) ABSTRACT TEST SUITE (NORMATIVE) ..... 41
   - A.1. Conformance Class Core ..... 41

ANNEX B (INFORMATIVE) ISO DATA DICTIONARY ..... 59
   - B.1. General Feature Model ..... 59
   - B.2. Geometry ..... 61
   - B.3. Citation and responsible party information ..... 63

B.4. Constraint information ..... 66
B.5. Identification information ..... 69
B.6. Name types ..... 71
B.7. Primitive types ..... 74

ANNEX C (INFORMATIVE) REVISION HISTORY ..... 77

BIBLIOGRAPHY ..... 79

# LIST OF TABLES

Table 1 ..... 34
Table 2 ..... 34
Table 3 ..... 35
Table 4 ..... 36
Table 5 ..... 36
Table 6 ..... 36
Table 7 ..... 37
Table 8 ..... 37
Table 9 ..... 38
Table 10 ..... 38
Table 11 ..... 38
Table 12 ..... 39
Table B.1 — Any Feature Class ..... 59
Table B.2 — Feature Type Class ..... 60
Table B.3 — GM_Object Class ..... 61
Table B.4 — GM_Point Class ..... 62
Table B.5 — GM_LineString Class ..... 62
Table B.6 — GM_Polygon Class ..... 63
Table B.7 — CI_Contact Class ..... 63
Table B.8 — CI_Individual Class ..... 64
Table B.9 — CI_Organisation Class ..... 65
Table B.10 — CI_Party Class ..... 65
Table B.11 — CI_Responsibility Class ..... 66
Table B.12 ..... 67
Table B.13 ..... 67
Table B.14 ..... 68
Table B.15 ..... 69
Table B.16 — MD_KeywordClass Class ..... 69
Table B.17 — MD_Keywords Class ..... 70

Table B.18 — Generic Name Class ..... 71
Table B.19 — Local Name Class ..... 71
Table B.20 — Member Name Class ..... 72
Table B.21 — Namespace Class ..... 72
Table B.22 — Scoped Name Class ..... 73
Table B.23 — Type Name Class ..... 73
Table B.24 — Date Class ..... 74
Table B.25 — DateTime Class ..... 74
Table B.26 — Time Class ..... 75
Table C.1 ..... 77

# LIST OF FIGURES

Figure 1 — UML notation (see ISO TS 19103, Geographic information - Conceptual schema language). ..... 13
Figure 2 — Example UML diagram demonstrating the UML notation and coloring scheme used throughout the POI Standard. ..... 15
Figure 3 — Feature Model ..... 18
Figure 4 — Geometry Model ..... 20
Figure 5 — POI UML Model - ISO Extensions ..... 22
Figure 6 — POI UML Model - Core ..... 27
Figure 7 — POI UML Model - Payload ..... 32

# LIST OF RECOMMENDATIONS

REQUIREMENTS CLASS 1: CORE REQUIREMENTS CLASS ..... 17
REQUIREMENT 1: REQUIREMENT — GENERAL FEATURE MODEL ..... 19
REQUIREMENT 2: REQUIREMENT — GEOMETRY ..... 21
REQUIREMENT 3: REQUIREMENT — ABSTRACT FEATURE ..... 22
REQUIREMENT 4: REQUIREMENT — ABSTRACT FEATURE DESCRIPTION ..... 23
REQUIREMENT 5: REQUIREMENT — ABSTRACT FEATURE FEATURE ID ..... 23
REQUIREMENT 6: REQUIREMENT — ABSTRACT FEATURE IDENTIFIER ..... 23
REQUIREMENT 7: REQUIREMENT — ABSTRACT FEATURE NAME ..... 24
REQUIREMENT 8: REQUIREMENT — FEATURE WITH LIFESPAN ..... 24
REQUIREMENT 9: REQUIREMENT — FEATURE WITH LIFESPAN CREATION DATE ..... 26

REQUIREMENT 10: REQUIREMENT — FEATURE WITH LIFESPAN TERMINATION DATE ..... 26
REQUIREMENT 11: REQUIREMENT — FEATURE WITH LIFESPAN VALID FROM ..... 26
REQUIREMENT 12: REQUIREMENT — FEATURE WITH LIFESPAN VALID TO ..... 27
REQUIREMENT 13: REQUIREMENT — ABSTRACT POI ..... 28
REQUIREMENT 14: REQUIREMENT — POI CONTACT INFORMATION ..... 28
REQUIREMENT 15: REQUIREMENT — POI FEATURE OF INTEREST ..... 29
REQUIREMENT 16: REQUIREMENT — POI METADATA ..... 29
REQUIREMENT 17: REQUIREMENT — POI KEYWORDS ..... 29
REQUIREMENT 18: REQUIREMENT — POI RIGHTS ..... 29
REQUIREMENT 19: REQUIREMENT — POI SYMBOLOGY ..... 29
REQUIREMENT 20: REQUIREMENT — POI PAYLOAD ..... 30
REQUIREMENT 21: REQUIREMENT — LINK CLASS ..... 30
REQUIREMENT 22: REQUIREMENT — POI CLASS ..... 30
REQUIREMENT 23: REQUIREMENT FEATURE OF INTEREST ..... 30
REQUIREMENT 24: REQUIREMENT — POI-PAYLOAD ..... 33
REQUIREMENT 25: REQUIREMENT — POI PAYLOAD FEATURE OF INTEREST ..... 33
REQUIREMENT 26: REQUIREMENT — POI PAYLOAD HAS DEFINITION ..... 33
REQUIREMENT 27: REQUIREMENT — POI PAYLOAD USES SCHEMA ..... 33
CONFORMANCE CLASS A.1 ..... 41

# I ABSTRACT

The OGC Points of Interest (POI) conceptual model is an open data model for representing information about POI. It is defined through a Unified Modeling Language (UML) object model. This UML model extends the ISO Technical Committee 211 (TC211) conceptual model standards for spatial and temporal data. Building on the ISO foundation assures that the features described in the POI Models share the same spatial-temporal universe as described by related standards (e.g., CityGML).

The aim of developing the OGC POI conceptual model is to reach a common definition of the basic entities, attributes, and relations of "points of interest." In the broadest terms, a point of interest is a location about which information of general interest is available. A POI can be as simple as a set of coordinates and an identifier, or more complex such as a three-dimensional model of a building with names in various languages, information about open and closed hours, and a civic address.

# II KEYWORDS

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, API, openapi, html

# III PREFACE

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

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

# IV SECURITY CONSIDERATIONS

The POI Conceptual model defines a POI as a type of Feature. By building on the same Feature Model as other OGC Feature models, POI implementations inherit the security controls and vulnerabilities of their associated Feature Dataset. They are a Feature like any other.

This document is a Standard for a Conceptual Model. Implementations of this Standad (Implementation Specifications) are free to add additional details and content necessary to enable implementation-specific security controls. In the event that anything in this Standard prevents implementation of needed controls, implementors are requested to notify the POI Standards Working Group and help devise a solution.

# V SUBMITTING ORGANIZATIONS

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

- Digital Flancers
- Google
- HeazelTech
- Pangaea Innovations
- PEREY Research Consulting
- US Army Geospatial Center

1

# SCOPE

# 1 SCOPE

This document describes a conceptual model for representing information about points of interest (POI).

In the broadest terms, a “point of interest” is a location about which information of general interest is available. A POI can be as simple as a set of coordinates and an identifier, or more complex such as a three-dimensional model of a building with names in various languages, information about open and closed hours, and a civic address.

POI data has many uses including navigation systems, mapping, geocaching, location-based social networking games, and augmented reality browsers.

POI data has traditionally been exchanged in proprietary formats by various transport mechanisms. This specification defines a flexible, lightweight, extensible POI data model. This will enable content publishers to effectively describe and efficiently serve and exchange POI data.

To achieve these goals, this document describes a generic data model that may be instantiated in a variety of serializations, including XML, JSON and RDF. The data model is designed to be extended with POI information specific to the geospatial data it represents.

2

# CONFORMANCE

# 2 CONFORMANCE

This standard defines a Conceptual Model which is independent of any encoding or formatting techniques. The Standardization Target for this standard is technology-specific POI Implementation Specifications.

## 2.1. Implementation Specifications

Implementation Specifications define how a Conceptual Model should be implemented using a specific technology. Conformant Implementation Specifications provide evidence that they are an accurate representation of the Conceptual Model. This evidence should include implementations of the abstract tests specified in Annex A (normative) of this document.

Since this standard is agnostic to the implementing technologies, the specific techniques to be used for conformance testing cannot be specified. Implementation Specifications need to provide evidence of conformance which is appropriate for the implementing technologies. This evidence should be provided as an annex to the Implementation Specification document.

## 2.2. Implementations

POI implementations will typically be a simplified representation of a more complex dataset. Implementors may want to extend the POI model to include properties specific to that dataset. These extensions are accomplished using the POI Payload mechanism described in POI Payload. Since the POI Payload has its own definition of syntax and semantics, conformance with the POI Standard cannot ensure payload conformance.

## 2.3. Conformance Classes

This standard identifies one “Core” conformance class. This conformance class defines the conformance criteria for the requirements defined in one “Core” requirements class. The tests this conformance class are documented in Annex A. These tests are organized by Requirements Class. So an implementation of the Core conformance class must pass all tests specified in Annex A for the Core Requirements Class.

The POI Conceptual Model is defined by the POI UML model. This Standard is a representation of that UML model in document form. In the case of a discrepancy between the UML model and this document, the UML model takes precedence.

3

# NORMATIVE REFERENCES

# 3 NORMATIVE REFERENCES

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.


ISO: ISO 19101-1:2014, *Geographic information — Reference model — Part 1: Fundamentals*. International Organization for Standardization, Geneva (2014). https://www.iso.org/standard/59164.html.

ISO: ISO 19103, *Geographic information — Conceptual schema language*. International Organization for Standardization, Geneva https://www.iso.org/standard/56734.html.

ISO: ISO 19107:2003, *Geographic information — Spatial schema*. International Organization for Standardization, Geneva (2003). https://www.iso.org/standard/26012.html.

ISO: ISO 19109:2015, *Geographic information — Rules for application schema*. International Organization for Standardization, Geneva (2015). https://www.iso.org/standard/59193.html.

ISO: ISO 19115-1:2014, *Geographic information — Metadata — Part 1: Fundamentals*. International Organization for Standardization, Geneva (2014). https://www.iso.org/standard/53798.html.

ISO: ISO 19507:2012, ISO (2012).

Arliss Whiteside Jim Greenwood: OGC 06-121r9, *OGC Web Service Common Implementation Specification*. Open Geospatial Consortium (2010).

Policy SWG: OGC 08-131r3, *The Specification Model — Standard for Modular specifications*. Open Geospatial Consortium (2009). https://portal.ogc.org/files/?artifact_id=34762&version=2.

Open Geospatial Consortium. *OGC Definitions Register*. https://www.opengis.net/def/glossary

Object Management Group (OMG), *Unified Modeling Language (UML)*, Version 2.5.1, December 2017, https://www.omg.org/spec/UML/2.5.1

4

# TERMS AND DEFINITIONS

# 4 TERMS AND DEFINITIONS

This document uses the terms defined in OGC Policy Directive 49, which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word "shall" (not "must") is the verb form used to indicate a requirement to be strictly followed to conform to this document and OGC documents do not use the equivalent phrases in the ISO/IEC Directives, Part 2.

This document also uses terms defined in the OGC Standard for Modular specifications (OGC 08-131r3), also known as the 'ModSpec'. The definitions of terms such as standard, specification, requirement, and conformance test are provided in the ModSpec.

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

## 4.1. class::

description of a set of *objects* that share the same *attributes*, *operations*, methods, relationships, and semantics

**Note 1 to entry:** A *class* may use a set of interfaces to specify collections of *operations* it provides to its environment. The term was first used in this way in the general theory of object-oriented programming, and later adopted for use in this same sense in UML.

[**SOURCE:** ISO 19103, Clause 4.27, modified – Note 1 to entry has been added from ISO 19117:2012, 4.2]

## 4.2. concept::

unit of knowledge created by a unique combination of characteristics

**Note 1 to entry:** *Concepts* (Clause 4.2) are not necessarily bound to particular languages. They are, however, influenced by the social or cultural background which often leads to different categorizations.

[**SOURCE:** ISO 1087-1, Clause 3.2.1]

## 4.3. conceptual model::

model that defines *concepts* (Clause 4.2) of a universe of discourse

[**SOURCE:** ISO 19101-1:2014, Clause 4.1.5]

## 4.4. conformance class::

a class of conformance tests. A conformant implementation must pass all the tests in the class.

[**SOURCE:** OGC Definitions Register]

## 4.5. feature::

abstraction of real-world phenomena

**Note 1 to entry:** A *feature* (Clause 4.5) may occur as a type or an instance. In this document, *feature* (Clause 4.5) instance is meant unless otherwise specified.

[**SOURCE:** ISO 19101-1:2014, Clause 4.1.11, modified – Note 1 to entry has been added from ISO 19156, 4.6]

## 4.6. feature type::

*class* (Clause 4.1) of *features* (Clause 4.5) having common characteristics

[**SOURCE:** ISO 19156:2011, Clause 4.7]

## 4.7. implementation specification::

guidance for software engineers that is so specific that any two independent software implementations of the specification can “plug and play” for each other.

[**SOURCE:** OGC Definitions Register]

## 4.8. requirements class::

a class of requirements, comprising a logical grouping of normative statements that shall be satisfied as a group in conformant implementations. May have dependencies on other *requirements classes* (Clause 4.8) , but there should be no circular dependencies else the classes must always be satisfied together so are functionally one class.

[**SOURCE:** OGC Definitions Register]

## 4.9. standardization target::

*standardization target* (Clause 4.9)

[**SOURCE:** OGC Definitions Register]

5

# CONVENTIONS

# 5 CONVENTIONS

## 5.1. Identifiers

The normative provisions in this document are denoted by the URI

http://www.opengis.net/spec/POI/1.0

All requirements and conformance tests that appear in this document are denoted by partial URIs relative to this base.

## 5.2. UML Notation

The POI Conceptual Model (CM) Standard is defined in a Unified Modeling Language (UML) model. It is presented in this document through diagrams using the UML static structure diagram. The UML notations used in this standard are described in the diagram in Figure 1.

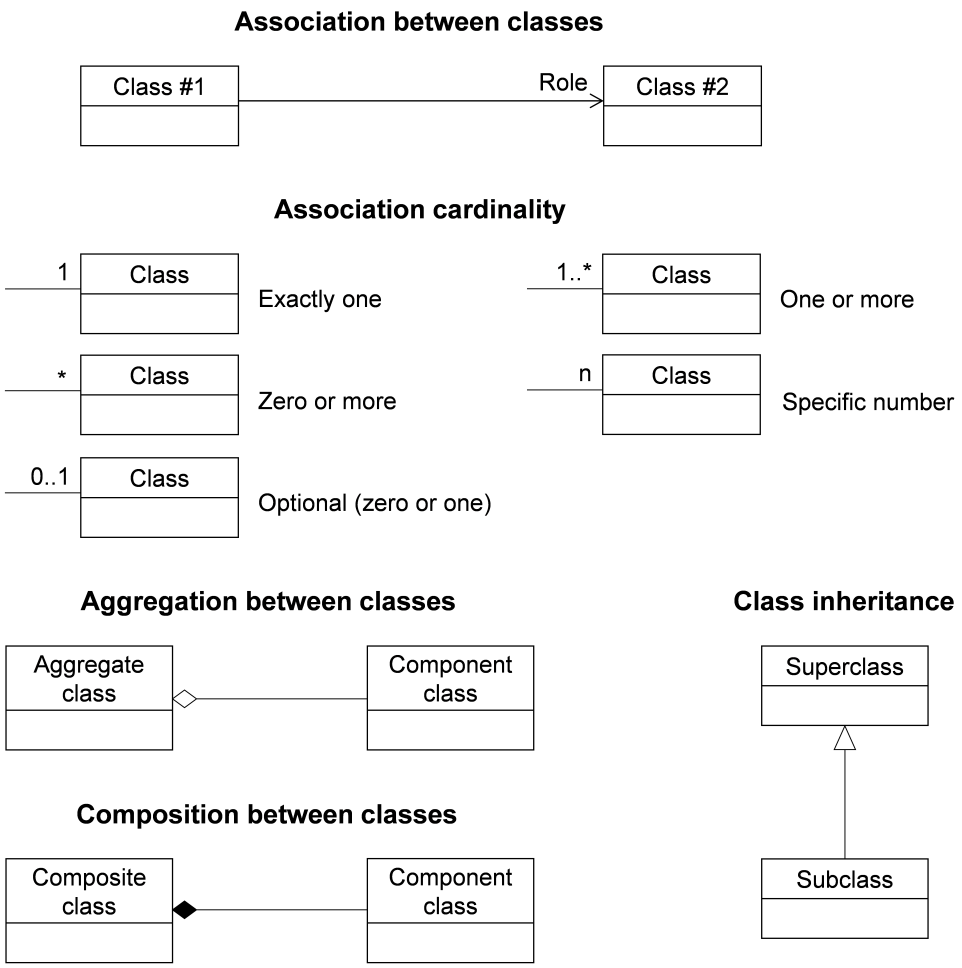



**Figure 1** — UML notation (see ISO TS 19103, Geographic information - Conceptual schema language).

All associations between model elements in the POI Conceptual Model are uni-directional. Thus, associations in the model are navigable in only one direction. The direction of navigation is depicted by an arrowhead. In general, the context an element takes within the association is indicated by its role. The role is displayed near the target of the association. If the graphical representation is ambiguous though, the position of the role has to be drawn to the element the association points to.

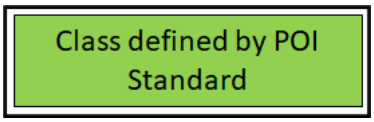
Aggregations are a form of association where the Component Class is treated as an attribute of the Aggregate Class. However, the Component Class is not an integral part of the Aggregate Class. A Component Class can be aggregated by more than one Aggregate Class.

Compositions are a form of association where the Component Class is treated as an attribute of the Composit Class. Component Classes are an integral part of the Composit Class and cannot be sharred by multiple Composit Classes. No Compositions are used in this Standard.

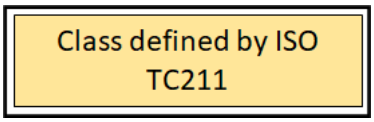
The following stereotypes are used in this model:

- «*Abstract*» a classes that doesn't include a complete implementation. Therefore, abstract classes can't be directly instantiated; they have to be specialized (inherited).
- «*DataType*» defines a set of properties that lack identity. A data type is a classifier with no operations, whose primary purpose is to hold information.
- «*FeatureType*» represents features that are similar and exhibit common characteristics. Features are abstractions of real-world phenomena and have an identity.
- «*Metaclass*» (Optional) a profile class and packageable element which may be extended through one or more stereotypes, which defines how an existing metaclass may be extended as part of a profile.
- «*Property*» denotes attributes and association roles. This stereotype does not add further semantics to the conceptual model but is required to be able to add tagged values to the attributes and association roles that are relevant for the encoding.
- «*Type*» denotes classes that are not directly instantiable, but are used as an abstract collection of operation, attribute and relation signatures. The stereotype is used in the POI Conceptual Model only for classes that are imported from the ISO standards 19103, 19107, 19109, and 19115.

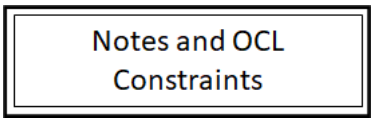
To enhance the readability of the POI UML diagrams, classes are depicted in different colors. The following coloring scheme is applied:




Classes painted in green belong to the POI Requirements Class.




Classes painted in tan are defined in the ISO standards 19107, 19109, or 19115. Class names are preceded by the UML package name in which the class is defined.




The color white is used for notes and Object Constraint Language (OCL) constraints that are provided in the UML diagrams.

The example UML diagram in Figure 2 demonstrates the UML notation and coloring scheme used throughout this standard. The generalization, link, and instance associations are also illustrated.

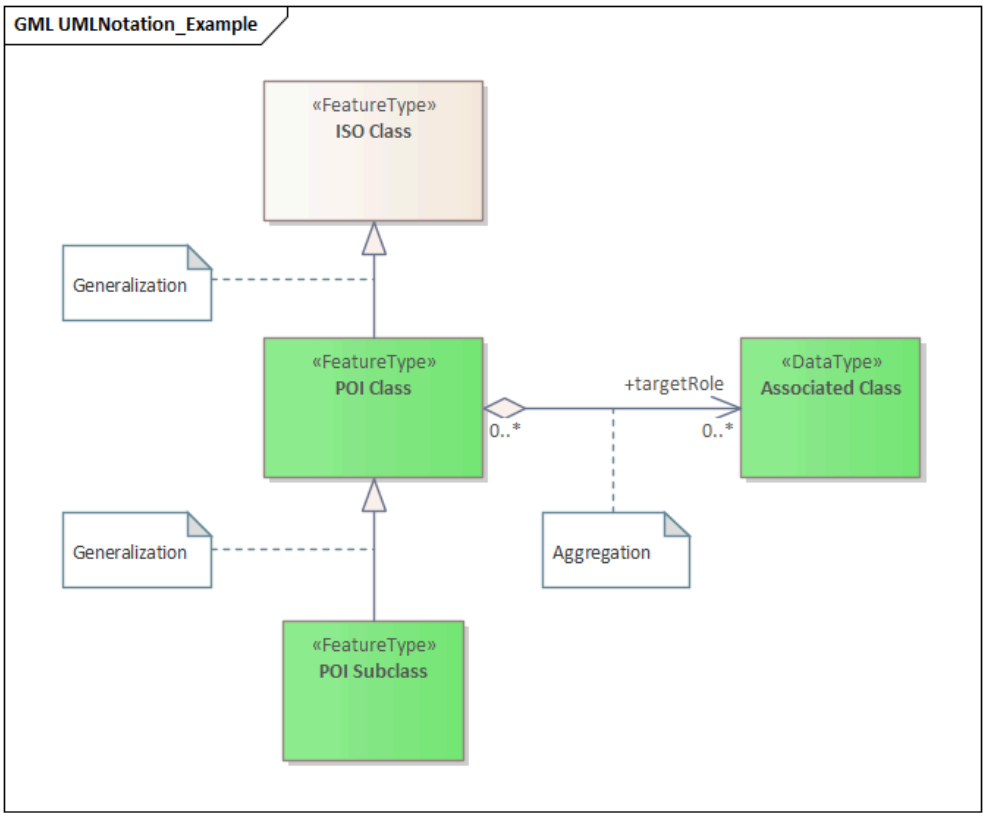



**Figure 2** — Example UML diagram demonstrating the UML notation and coloring scheme used throughout the POI Standard.

6

# POI CORE REQUIREMENTS

# 6 POI CORE REQUIREMENTS

| REQUIREMENTS CLASS 1: CORE REQUIREMENTS CLASS | |
|---|---|
| **OBLIGATION** | requirement |
| **DESCRIPTION** | Requirements Class<br>• urn:iso:ts:iso:19103<br>• urn:iso:ts:iso:19107<br>• urn:iso:ts:iso:19108<br>• urn:iso:ts:iso:19109 |

## 6.1. Feature Model

A Point of Interest (POI) is a Feature. Therefore, it is important to understand what a POI inherits from the OGC Feature model.

The OGC Feature Model is defined in ISO 19109:2015. A UML model showing applicable portions of the OGC Feature Model is provided in Figure 3.

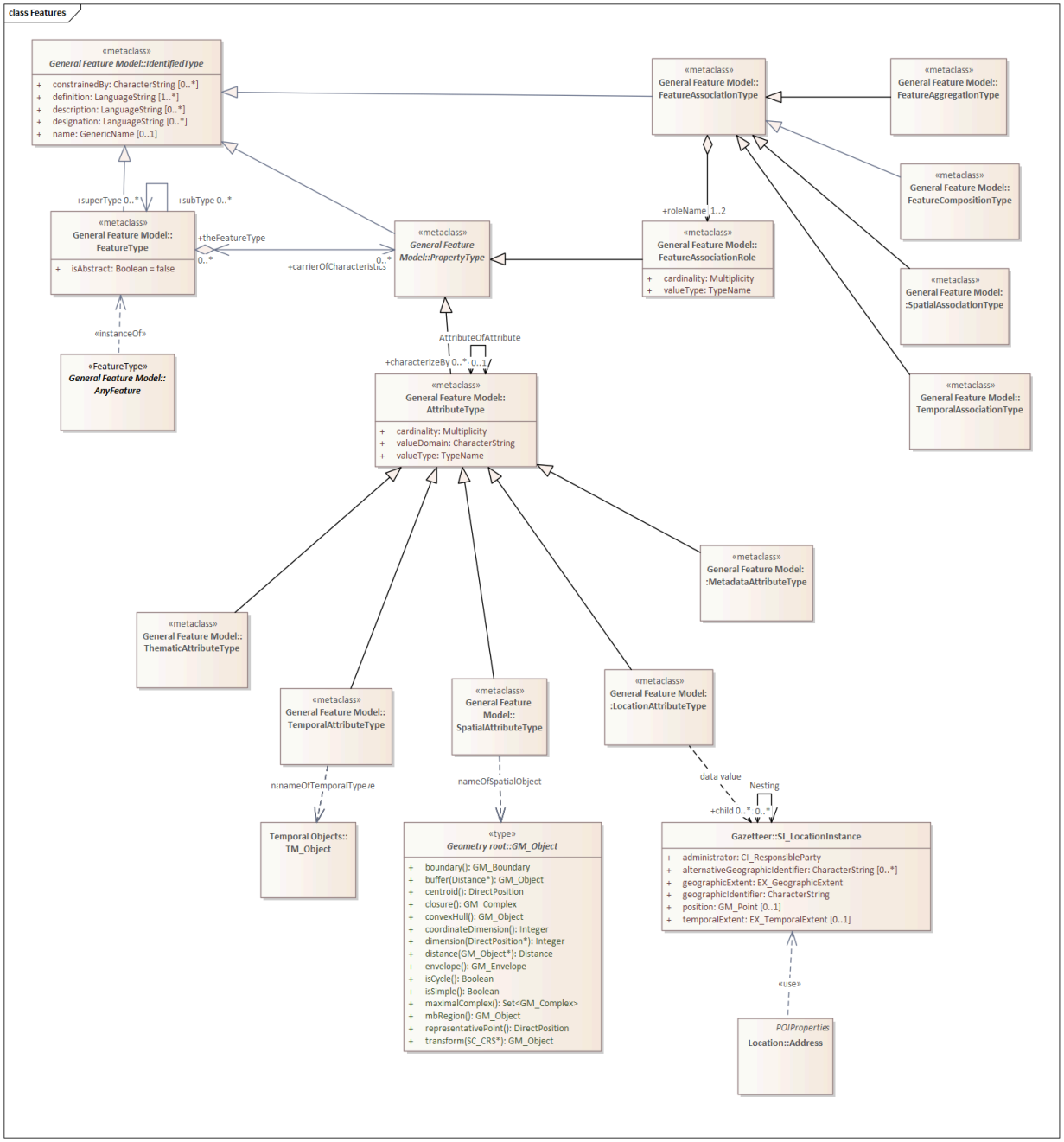



**Figure 3** — Feature Model

The most relevant classes defined by this model are described below:

**FeatureType:** This class describes how a feature class shall be constructed in an Application Schema. In accordance with the conformance clause of the standard, instances of this class are instantiated as feature classes in an Application Schema

**AnyFeature:** The class AnyFeature is an instance of the «metaclass» FeatureType. It represents the set of all classes which are feature types.

In an implementation, this abstract class shall be substituted by a concrete class representing a feature type from an application schema associated with a domain of discourse.

| REQUIREMENT 1: REQUIREMENT — GENERAL FEATURE MODEL | |
|---|---|
| **LABEL** | /req/core/generalfeaturemodel |
| **STATEMENT** | An encoding of the POI Conceptual Model SHALL be compliant with the General Feature Model defined in ISO 19109. |

## 6.2. Geometry

The OGC Geometry model is defined in ISO 19107:2003. While there is a new version of this standard, it has not been widely implemented. Therefore, the 2003 version has been used in this Standard.

The OGC Geometry Model can represent very complex geometries. Much more complex than are needed for a POI. Therefore, POI geometries are restricted to Points, lines, and Polygons. Figure 4 provides a UML model of the classes from ISO 19107 which are applicable to POIs.

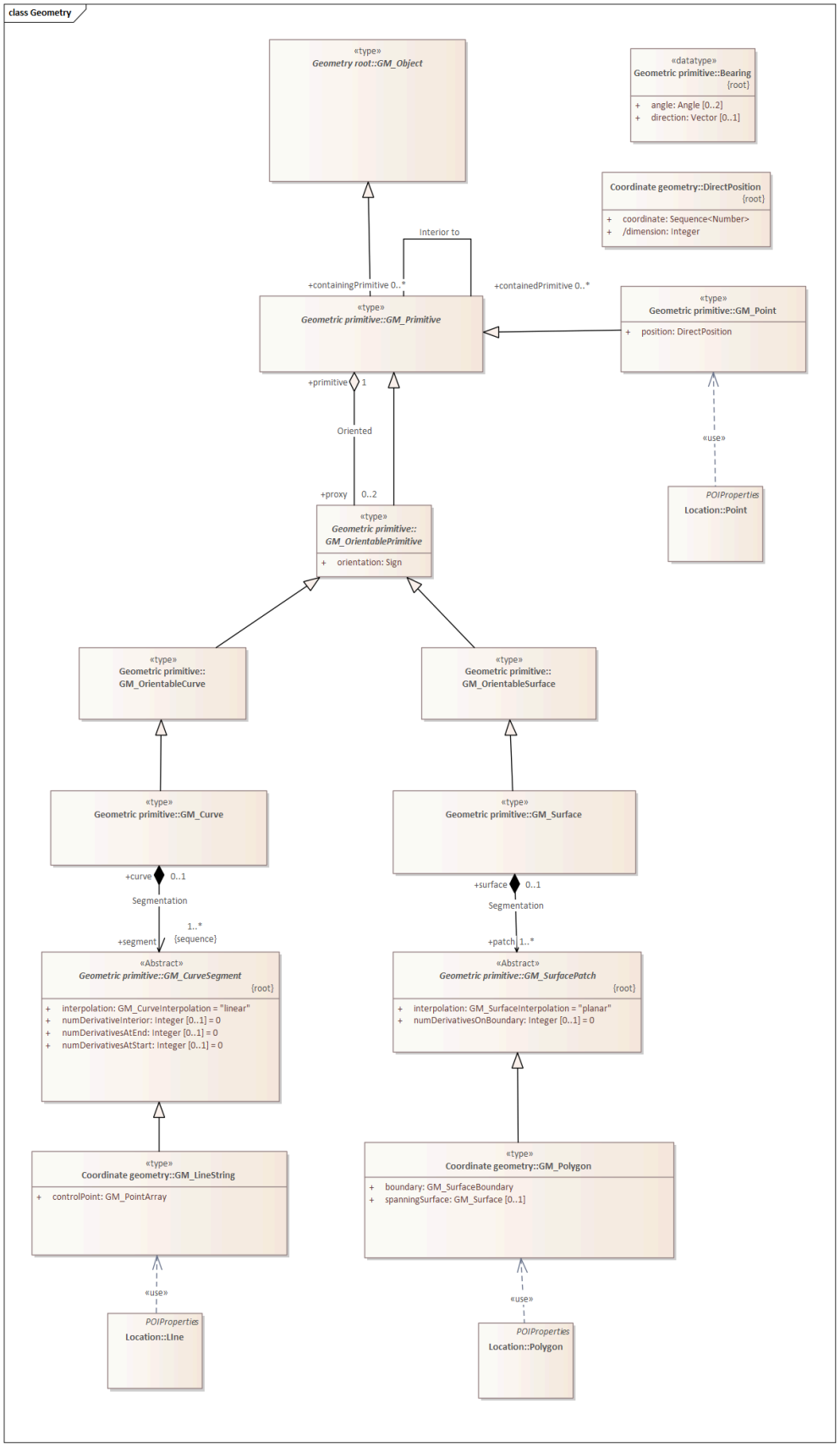



**Figure 4** — Geometry Model

The key classes described in this figure are:

**GM_Object:** Root class for all OGC geometries.

**GM_Point:** The geometric primitive for Points

**GM_LineString:** The geometric primitive for line strings.

**GM_Polygon:** The geometric primitive for areas.

| REQUIREMENT 2: REQUIREMENT – GEOMETRY | |
|---|---|
| LABEL | /req/core/geometry |
| STATEMENT | The POI Conceptual Model spatial geometry properties SHALL be compliant with the Geometry Model defined in ISO 19107. |
| A | A POI instance SHALL include a spatial geometry property using the SpatialAttributeType attribute type. |
| B | The spatial geometry properties of all POI instances SHALL be defined using one or more of the following classes:<br>1. GM_Point<br>2. GM_LineString<br>3. GM_Polygon |

## 6.3. POI ISO Extensions

This Standard extends the OGC Feature Model to support the concept of a Point of Interest. These extensions are illustrated in Figure 5.

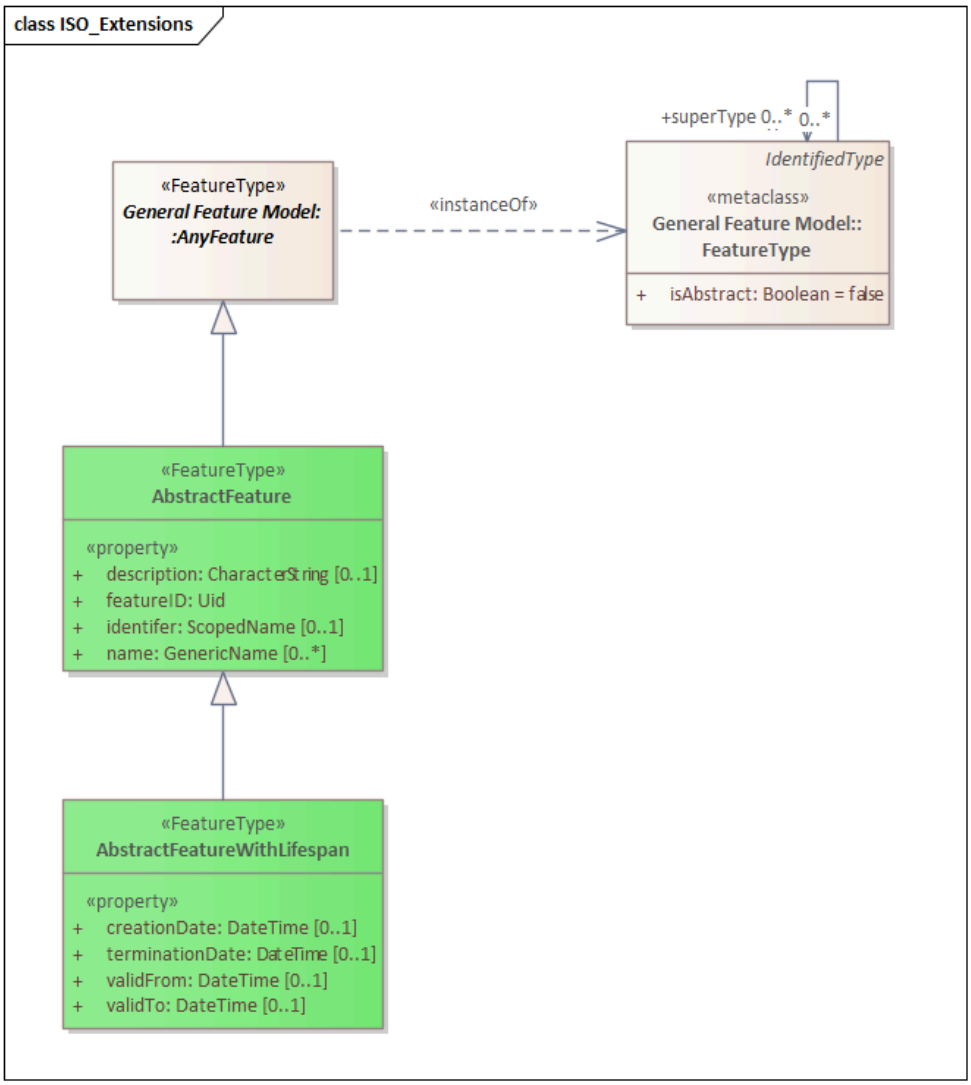



**Figure 5** — POI UML Model - ISO Extensions

These extensions include further refinement of the *AnyFeature* class through the addition of identification and temporal validity attributes.

**AbstractFeature:** The root Feature class for this standard. This class has been borrowed from the CityGML 3.0 Conceptual Model. *AbstractFeature* adds descriptive and identifying properties to *AnyFeature*. **AbstractFeatureWithLifespan:** Adds temporality to *AbstractFeature*. This class was borrowed from the CityGML 3.0 Conceptual Model.

### 6.3.1. Abstract Feature

| REQUIREMENT 3: REQUIREMENT — ABSTRACT FEATURE | |
|---|---|
| **LABEL** | /req/core/abstractfeature |

| REQUIREMENT 3: REQUIREMENT — ABSTRACT FEATURE | |
|---|---|
| STATEMENT | An encoding of the AbstractFeature class SHALL be a compliant extension of the Any Feature class defined in ISO 19109. |
| A | An encoding of the AbstractFeature class SHALL comply with requirement /req/core/abstractfeature-description. |
| B | An encoding of the AbstractFeature class SHALL comply with requirement /req/core/abstractfeature-featureid. |
| C | An encoding of the AbstractFeature class SHALL comply with requirement /req/core/abstractfeature-identifier. |
| D | An encoding of the AbstractFeature class SHALL comply with requirement /req/core/abstractfeature-name. |

| REQUIREMENT 4: REQUIREMENT — ABSTRACT FEATURE DESCRIPTION | |
|---|---|
| LABEL | /req/core/abstractfeature-description |
| STATEMENT | An encoding of the AbstractFeature class SHALL include zero or one `description` attributes. |
| A | Encodings of the `description` attribute SHALL be a valid implementation of the CharacterString class from ISO 19103. |

| REQUIREMENT 5: REQUIREMENT — ABSTRACT FEATURE FEATURE ID | |
|---|---|
| LABEL | /req/core/abstractfeature-featureid |
| STATEMENT | An encoding of the AbstractFeature class SHALL include one `featureID` attributes. |
| A | Encodings of the `featureID` attribute SHALL be a valid implementation of the ID class from ISO 19103. |

| REQUIREMENT 6: REQUIREMENT — ABSTRACT FEATURE IDENTIFIER | |
|---|---|
| LABEL | /req/core/abstractfeature-identifier |
| STATEMENT | An encoding of the AbstractFeature class SHALL include zero or one `identifier` attributes. |
| A | Encodings of the `identifier` attribute SHALL be a valid implementation of the ScopedName class from ISO 19103. |

| REQUIREMENT 7: REQUIREMENT — ABSTRACT FEATURE NAME | |
|---|---|
| LABEL | /req/core/abstractfeature-name |
| STATEMENT | An encoding of the AbstractFeature class SHALL include zero or more `name` attributes. |
| A | Encodings of the `name` attribute SHALL be a valid implementation of the Generic Name class from ISO 19103. |

### 6.3.2. Abstract Feature with Lifespan

| REQUIREMENT 8: REQUIREMENT — FEATURE WITH LIFESPAN | |
|---|---|
| LABEL | |
| STATEMENT | |
| A | |

## REQUIREMENT 8: REQUIREMENT — FEATURE WITH LIFESPAN

B

C

D

## REQUIREMENT 8: REQUIREMENT — FEATURE WITH LIFESPAN

## REQUIREMENT 9: REQUIREMENT — FEATURE WITH LIFESPAN CREATION DATE

| | |
|---|---|
| **LABEL** | /req/core/featurewithlifespan-creationdate |
| **STATEMENT** | An encoding of the AbstractFeatureWithLifespan class SHALL include zero or one `creationDate` attributes. |
| **A** | Encodings of the `creationDate` attribute SHALL be a valid implementation of the DateTime class from ISO 19103. |

## REQUIREMENT 10: REQUIREMENT — FEATURE WITH LIFESPAN TERMINATION DATE

| | |
|---|---|
| **LABEL** | /req/core/featurewithlifespan-terminationdate |
| **STATEMENT** | An encoding of the AbstractFeatureWithLifespan class SHALL include zero or one `terminationDate` attributes. |
| **A** | Encodings of the `terminationDate` attribute SHALL be a valid implementation of the DateTime class from ISO 19103. |

## REQUIREMENT 11: REQUIREMENT — FEATURE WITH LIFESPAN VALID FROM

| | |
|---|---|
| **LABEL** | /req/core/featurewithlifespan-validfrom |
| **STATEMENT** | An encoding of the AbstractFeatureWithLifespan class SHALL include zero or one `validFrom` attributes. |

| REQUIREMENT 11: REQUIREMENT – FEATURE WITH LIFESPAN VALID FROM | |
|---|---|
| A | Encodings of the `validFrom` attribute SHALL be a valid implementation of the DateTime class from ISO 19103. |

| REQUIREMENT 12: REQUIREMENT – FEATURE WITH LIFESPAN VALID TO | |
|---|---|
| LABEL | /req/core/featurewithlifespan-validto |
| STATEMENT | An encoding of the AbstractFeatureWithLifespan class SHALL include zero or one `validTo` attributes. |
| A | Encodings of the `validTo` attribute SHALL be a valid implementation of the Date Time class from ISO 19103. |

## 6.4. POI Class Model

The following classes form the core of the POI model. These classes are the same for all POIs.

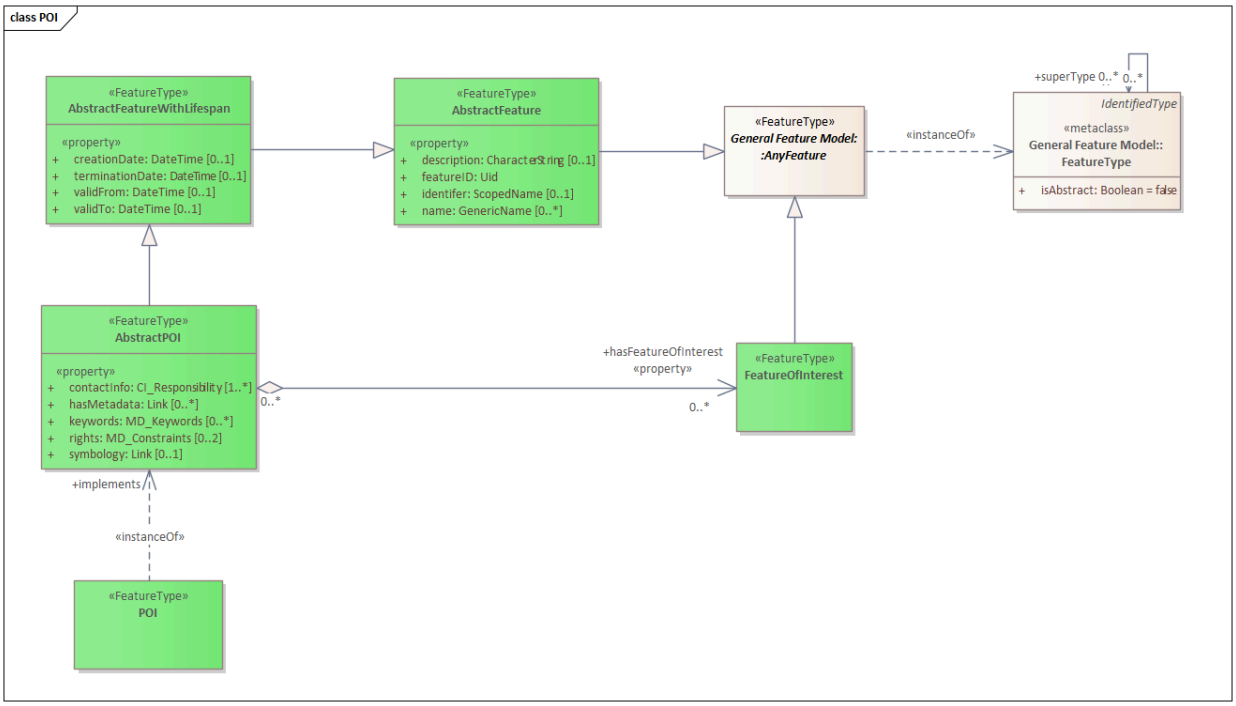



**Figure 6** — POI UML Model - Core

- AbstractPOI: The abstract model for a Point of Interest. All POI instances will contain these attributes.

- POI: A POI instance.
- FeatureOfInterest: This is an OGC Feature which has been defined independently from the POI. Conceptually, the purpose of the POI is to provide a user friendly synopsis of this Feature.

### 6.4.1. Abstract POI

| REQUIREMENT 13: REQUIREMENT — ABSTRACT POI | |
|---|---|
| LABEL | /req/core/abstract-poi |
| STATEMENT | An instantiation of the Abstract POI class SHALL comply with requirement /req/core/poi-feature-with-lifespan. |
| A | An instantiation of the Abstract POI class SHALL comply with requirement /req/core/poi-contactInfo. |
| B | An instantiation of the Abstract POI class SHALL comply with requirement /req/core/poi-featureOfInterest. |
| C | An instantiation of the Abstract POI class SHALL comply with requirement /req/core/poi-metadata. |
| D | An instantiation of the Abstract POI class SHALL comply with requirement /req/core/poi-keywords. |
| E | An instantiation of the Abstract POI class SHALL comply with requirement /req/core/poi-rights. |
| F | An instantiation of the Abstract POI class SHALL comply with requirement /req/core/poi-symbology. |
| G | An instantiation of the Abstract POI class SHALL comply with requirement /req/core/poi-payload. |

| REQUIREMENT 14: REQUIREMENT — POI CONTACT INFORMATION | |
|---|---|
| LABEL | /req/core/poi-contactInfo |
| STATEMENT | An encoding of the Abstract POI class SHALL include one or more `contactInfo` attributes. |
| A | Encodings of the `contactInfo` attribute SHALL be a valid implementation of the CI_Responsibility class from ISO 19115-1:2014 |

## REQUIREMENT 15: REQUIREMENT — POI FEATURE OF INTEREST

| | |
|---|---|
| **LABEL** | /req/core/poi-featureOfInterest |
| **STATEMENT** | An encoding of the Abstract POI class SHALL include zero or more associated instances of the `FeatureOfInterest` class. |
| **A** | The target of the `hasFeatureOfIntestest` aggregation SHALL be a valid implementation of the Feature class from ISO 19109:2015 |

## REQUIREMENT 16: REQUIREMENT — POI METADATA

| | |
|---|---|
| **LABEL** | /req/core/poi-metadata |
| **STATEMENT** | An encoding of the Abstract POI class SHALL include zero or more `metadata` attributes. |
| **A** | An implementation of the `metadata` association SHALL comply with requirement /req/core/link. |

## REQUIREMENT 17: REQUIREMENT — POI KEYWORDS

| | |
|---|---|
| **LABEL** | /req/core/poi-keywords |
| **STATEMENT** | An encoding of the Abstract POI class SHALL include zero or more `keywords` attributes. |
| **A** | Encodings of the `keywords` attribute SHALL be a valid implementation of the MD_Keywords class from ISO 19115-1:2014 |

## REQUIREMENT 18: REQUIREMENT — POI RIGHTS

| | |
|---|---|
| **LABEL** | /req/core/poi-rights |
| **STATEMENT** | An encoding of the Abstract POI class SHALL include zero, one, or two `rights` attributes. |
| **A** | Encodings of the `rights` attribute SHALL be a valid implementation of the MD_Constraints class from ISO 19115-1:2014 |

## REQUIREMENT 19: REQUIREMENT — POI SYMBOLOGY

| | |
|---|---|
| **LABEL** | /req/core/poi-symbology |

| REQUIREMENT 19: REQUIREMENT — POI SYMBOLOGY | |
|---|---|
| STATEMENT | An encoding of the Abstract POI class SHALL include zero or one `symbology` attributes. |
| A | Encodings of a `symbology` attribute SHALL comply with requirement /req/core/link. |

| REQUIREMENT 20: REQUIREMENT — POI PAYLOAD | |
|---|---|
| LABEL | /req/core/poi-payload |
| STATEMENT | An encoding of the Abstract POI class SHALL include zero or more associated instances of the `POI_Payload` class. |
| A | An encoding of the Abstract POI class SHALL comply with requirement /req/core/poi-payload. |

| REQUIREMENT 21: REQUIREMENT — LINK CLASS | |
|---|---|
| LABEL | /req/core/poi-link |
| STATEMENT | TBD. |

### 6.4.2. POI

| REQUIREMENT 22: REQUIREMENT — POI CLASS | |
|---|---|
| LABEL | /req/core/poi |
| STATEMENT | An encoding of the POI class SHALL comply with requirement /req/core/req-poi-abstract-poi. |

### 6.4.3. Feature of Interest

| REQUIREMENT 23: REQUIREMENT FEATURE OF INTEREST | |
|---|---|
| LABEL | /req/core/feature-of-interest |

| REQUIREMENT 23: REQUIREMENT FEATURE OF INTEREST | |
|---|---|
| **STATEMENT** | An encoding of the FeatureOfInterest class SHALL be a compliant extension of the Any Feature class defined in ISO 19109. |

## 6.5. POI Payload

A POI is a representation of a Feature. The POI class provides a standard way to identify and manage a POI. However, it does not provide any information about the Feature it is representing. This information is difficult to standardize since it is dependent on the data model of the Feature store being described.

Therefore, the POI model is designed to be extended with properties specific to a Feature or a Feature Collection. The POI Payload is a container for representations of Feature properties. The syntax of those representations is provided by the Payload Schema class. Where appropriate, the semantics can also be provided through the Payload Definition class. Since the schema and definitions may be the same for a large number of Features, these classes should be instantiated as referenceable resources, allowing one instance to be used by a number of POIs.

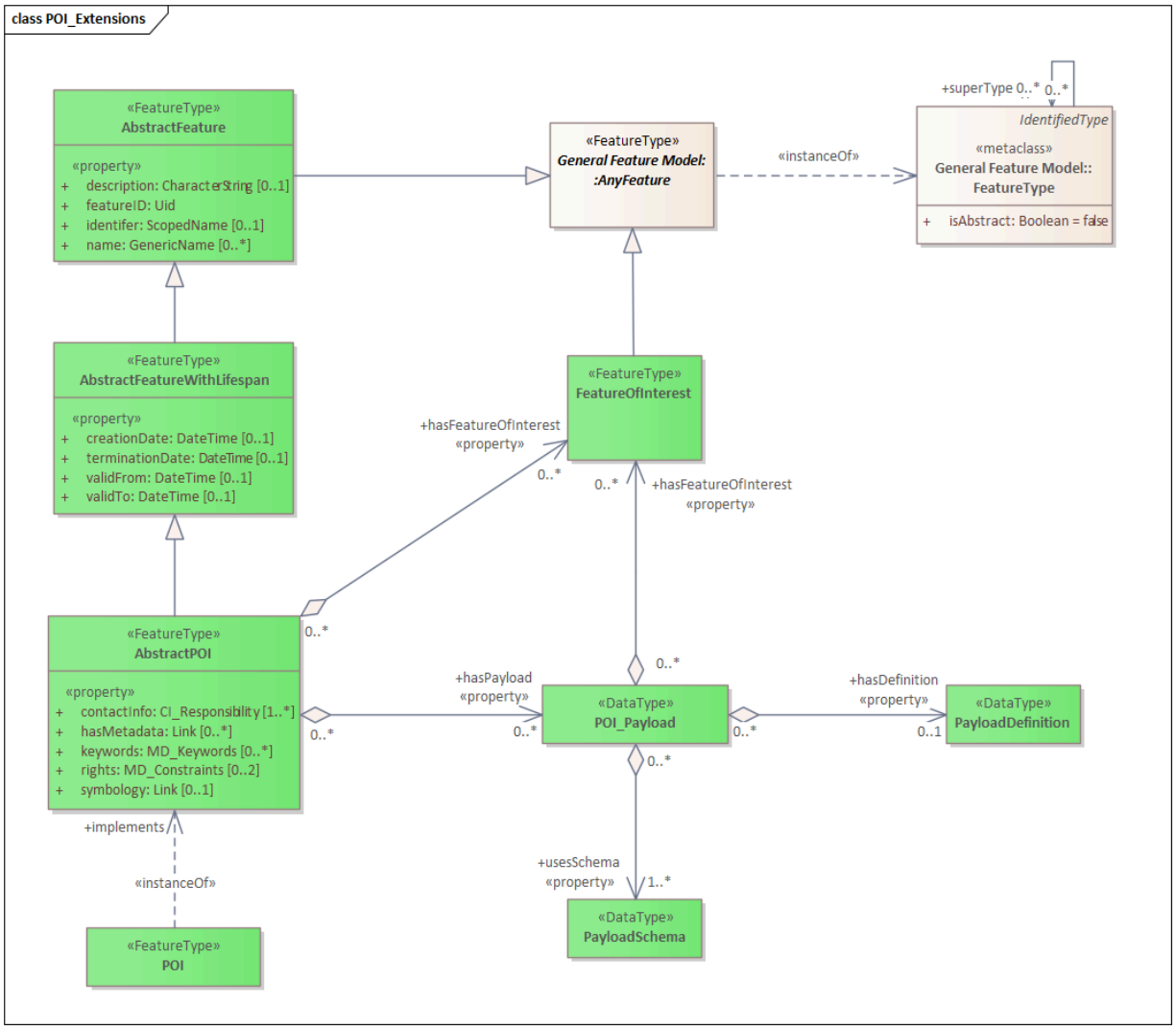



**Figure 7** — POI UML Model - Payload

- POI_Payload: The abstract model for a Point of Interest. All POI instances will contain these attributes.
- PayloadSchema: The Payload Schema Class represents a syntactic model (schema) for a POI payload.
- PayloadDefinition: The Payload Definition Class represents a semantic model (ontology) for a POI payload.

In the interest of interoperability, the POI Payload should be constructed using data types and concepts which are already in wide use by the Geospatial community. A collection of data types and concepts defined by the ISO 19103, ISO 19107, ISO 19109, and ISO 19115 Standards is provided in Annex B.

## REQUIREMENT 24: REQUIREMENT — POI-PAYLOAD

| | |
|---|---|
| **LABEL** | /req/core/poi-payload |
| **STATEMENT** | An instantiation of the POI_Payload class SHALL comply with requirement /req/core/poi_payload/featureofinterest. |
| **A** | An instantiation of the POI_Payload class SHALL comply with requirement /req/core/poi_payload/hasDefinition. |
| **B** | An instantiation of the POI_Payload class SHALL comply with requirement /req/core/poi_payload/usesSchema. |

## REQUIREMENT 25: REQUIREMENT — POI PAYLOAD FEATURE OF INTEREST

| | |
|---|---|
| **LABEL** | /req/core/poi_payload/featureofinterest |
| **STATEMENT** | An encoding of the Abstract POI_Payload class SHALL include zero or more `hasFeatureOfInterest` aggregations. |
| **A** | The target of a `hasFeatureOfIntestest` aggregation SHALL be a valid implementation of the Feature class from ISO 19109:2015 |

## REQUIREMENT 26: REQUIREMENT — POI PAYLOAD HAS DEFINITION

| | |
|---|---|
| **LABEL** | /req/core/poi_payload/hasDefinition |
| **STATEMENT** | An encoding of the Abstract POI_Payload class SHALL include no more than one `hasDefinition` aggregations. |
| **A** | The target of a `hasDefinition` aggregation SHALL be a valid ontology for the implementing technology |

## REQUIREMENT 27: REQUIREMENT — POI PAYLOAD USES SCHEMA

| | |
|---|---|
| **LABEL** | /req/core/poi_payload/usesSchema |
| **STATEMENT** | An encoding of the Abstract POI_Payload class SHALL include one or more `usesSchema` aggregations. |
| **A** | The target of a `usesSchema` aggregation SHALL be a valid schema for the implementing technology. |

## 6.6. POI Data Dictionary

The POI UML model is the normative definition of the POI Conceptual Model. The Data Dictionary tables in this section were software generated from the UML model. As such, this section provides a normative representation of the POI Conceptual Model.

### Table 1

**AbstractFeature**

| | |
|---|---|
| Definition: | AbstractFeature is the abstract superclass of all feature types within the PoI Model. |
| Subclass of: | AnyFeature |
| Stereotype: | «FeatureType» |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| description «property» | CharacterString [0..1] | Provides further information on the feature. |
| featureID «property» | Uid [1..1] | Specifies the unique identifier of the feature that is valid in the instance document within which it occurs. |
| identifer «property» | ScopedName [0..1] | Specifies the unique identifier of the feature that is valid globally. |
| name «property» | GenericName [0..*] | Specifies the name of the feature. |

### Table 2

**AbstractFeatureWithLifespan**

| | |
|---|---|
| Definition: | AbstractFeatureWithLifespan is the base class for all PoI features. This class allows the optional specification of the real-world and database times for the existence of each feature. |
| Subclass of: | AbstractFeature |
| Stereotype: | «FeatureType» |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| creationDate «property» | DateTime [0..1] | Indicates the date at which a POI feature was added to the containing model. |
| terminationDate «property» | DateTime [0..1] | Indicates the date at which a POI feature was removed from the containing model. |
| validFrom «property» | DateTime [0..1] | Indicates the date at which a POI feature started to exist in the real world. |
| validTo «property» | DateTime [0..1] | Indicates the date at which a POI feature ceased to exist in the real world. |

## Table 3

**AbstractPOI**

| | |
|---|---|
| Definition: | A Point of Interest (POI) is a Feature which provides a concise summary of one or more associated Features. Its purpose is to provide easy access to key information about one or more real-world objects without the need to access or understand the underlying Feature data set. |
| Subclass of: | AbstractFeatureWithLifespan |
| Stereotype: | «FeatureType» |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| hasPayload | POI_Payload [0..*] | Indicates a payload associated with this POI. |
| hasFeature OfInterest | FeatureOfInterest [0..*] | One or more Features which are represented by this POI. |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| contactInfo «property» | CI_Responsibility [1..*] | Contact information for the creators and maintainers of this POI. |
| hasMetadata «property» | Link [0..*] | An association with zero or more metadata records providing additional information about this POI and/or the associated Features of Interest. |
| keywords «property» | MD_Keywords [0..*] | Keywords used to aid in discovery of POIs of interest. |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| rights «property» | MD_Constraints [0..2] | Legal and security constraints applicable to this POI. |
| symbology «property» | Link [0..1] | A reference to information about rendering this POI. |

## Table 4

**FeatureOfInterest**

| | |
|---|---|
| Definition: | The thing whose property is being estimated or calculated in the course of an Observation to arrive at a Result, or whose property is being manipulated by an Actuator, or which is being sampled or transformed in an act of Sampling. (SOSA) |
| Subclass of: | AnyFeature |
| Stereotype: | «FeatureType» |

## Table 5

**PayloadDefinition**

| | |
|---|---|
| Definition: | The semantic model (ontology) for a POI payload. |
| Subclass of: | none |
| Stereotype: | «DataType» |

## Table 6

**PayloadSchema**

| | |
|---|---|
| Definition: | A model of the syntax of the POI payload. |
| Subclass of: | none |

| | |
|---|---|
| Stereotype: | «DataType» |

## Table 7

**POI**

| | |
|---|---|
| Definition: | An instance of a POI. Implements the AbstractPOI class. |
| Subclass of: | none |
| Stereotype: | «FeatureType» |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| implements | AbstractPOI [] | Identifies the abstract POI implemented by this POI |

## Table 8

**POI_Payload**

| | |
|---|---|
| Definition: | A representation of properties of the FoI which are to be included in the POI. |
| Subclass of: | none |
| Stereotype: | «DataType» |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| hasDefinition | PayloadDefinition [0..1] | A reference to the semantic model of this POI payload. |
| representsFoI | FeatureOfInterest [1..1] | Indicates the Feature of Interest which is being summarized in this payload. |
| usesSchema | PayloadSchema [1..*] | A reference to the schema for this POI payload. |

### 6.6.1. POI Data Types

The following data types are used in the POI UML model.

Table 9

| **CharacterString** | |
|---|---|
| Definition: | Characterstring is a family of datatypes which represent strings of symbols from standard character-sets. The semantics of CharacterString is in accordance with ISO/IEC 11404:2007 clause 10.1.5. (ISO 19103) |
| Subclass of: | none |
| Stereotype: | «Type» |

Table 10

| **Integer** | |
|---|---|
| Definition: | An exact integer value, with no fractional part. (ISO 19103) |
| Subclass of: | none |
| Stereotype: | «Type» |

Table 11

| **Link** | |
|---|---|
| Definition: | A data type which allows elements to be inserted into a document in order to create and describe links between resources. (derived from XML Linking Language (XLink) Version 1.1) |
| Subclass of: | none |
| Stereotype: | «Type» |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| href | CharacterString | Supplies the URI to a remote resource (or resource fragment). |
| hreflang | CharacterString [0..1] | A hint indicating what the language of the result of dereferencing the link should be. |
| rel | CharacterString | The type or semantics of the relation. |
| title | CharacterString [0..1] | Used to label the destination of a link such that it can be used as a human-readable identifier. |
| type | CharacterString [0..1] | A hint indicating what the media type of the result of dereferencing the link should be. |

Table 12

| **Uid** | |
|---|---|
| Definition: | Uid is a basic type that represents a unique identifier. |
| Subclass of: | none |
| Stereotype: | «Type» |

A

# ANNEX A (INFORMATIVE) ABSTRACT TEST SUITE (NORMATIVE)

# A ANNEX A (INFORMATIVE) ABSTRACT TEST SUITE (NORMATIVE)

## A.1. Conformance Class Core

| CONFORMANCE CLASS A.1 | |
|---|---|
| IDENTIFIER | `http://www.opengis.net/spec/poi/1.0/conf/core` |
| REQUIREMENTS CLASS | Requirements Class core |
| TARGET TYPE | Implementation Specification |
| NOTE | If the Implementaion Specification is based on a know conformant Standard (ex: GML) then an explicit test is not required. |

### A.1.1. General Feature Model

| ABSTRACT TEST A.1 | |
|---|---|
| IDENTIFIER | `/conf/core/generalfeaturemodel` |
| REQUIREMENT | /req/core/generalfeaturemodel |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the POI Implementation Specification is conformant with the ISO 19109 General Feature Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Inspect the POI Implementation Specification for the following: |

| ABSTRACT TEST A.1 | |
|---|---|
| A | Validate that the Implementation Specification includes an Abstract Test Suite (Annex A). |
| B | Validate that the Abstract Test Suite tests for conformance with the General Feature Model defined in ISO 19109. |

## A.1.2. Geometry

| ABSTRACT TEST A.2 | |
|---|---|
| IDENTIFIER | `/conf/core/geometry` |
| REQUIREMENT | /req/core/geometry |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | To validate that the POI Implementation Specification is conformant with the ISO 19107 Geometry Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Inspect the POI Implementation Specification for the following: |
| A | Validate that the Implementation Specification includes an Abstract Test Suite (Annex A). |
| B | Validate that all geometries used in the Implementation Specification conform with the geometry model defined in ISO 19107. |
| C | Validate that the Abstract Test Suite tests each POI Feature for the presence of a `SpatialAttributeType` property of type GM_Point, GM_LineString, or GM_Polygon. |

## A.1.3. Abstract Feature

| ABSTRACT TEST A.3 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature` |
| REQUIREMENT | /req/core/abstractfeature |
| PREREQUISITES | Abstract test A.1: `/conf/core/generalfeaturemodel`<br>Abstract test A.2: `/conf/core/geometry` |

| ABSTRACT TEST A.3 | |
|---|---|
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the AbstractFeature Class as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Inspect the POI Implementation Specification to validate that the AbstractFeature class is properly implemented. |
| A | Validate that the implementation of the AbstractFeature class is also a valid implementation of the AnyFeature class defined in the ISO 19109 General Feature Model. |
| B | Validate correct implementation of the `description` attribute using test /conf/core/abstractfeature-description. |
| C | Validate correct implementation of the `featureId` attribute using test /conf/core/abstractfeature-featureid. |
| D | Validate correct implementation of the `identifier` attribute using test /conf/core/abstractfeature-identifier. |
| E | Validate correct implementation of the `name` attribute using test /conf/core/abstractfeature-name. |

### A.1.3.1. Abstract Feature-Description

| ABSTRACT TEST A.4 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature-description` |
| REQUIREMENT | /req/core/abstractfeature-description |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `description` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `description` attribute. |
| A | Validate that no more than one 'description' attribute is allowed in an AbstractFeature class. |
| B | Validate that the `description` attribute is a valid implementation of the CharacterString class from ISO 19103. |

### A.1.3.2. Abstract Feature-FeatureId

| ABSTRACT TEST A.5 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature-featureid` |
| REQUIREMENT | /req/core/abstractfeature-featureid |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `featureId` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `featureId` attribute. |
| A | Validate that one and only one 'featureId' attribute is allowed in an AbstractFeature class. |
| B | Validate that the `featureId` attribute is a valid implementation of the ID class from ISO 19103. |

**NOTE:** Validate use of ID class from ISO 19103

### A.1.3.3. Abstract Feature-Identifier

| ABSTRACT TEST A.6 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature-identifier` |
| REQUIREMENT | /req/core/abstractfeature-identifier |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `identifier` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `identifier` attribute. |

| ABSTRACT TEST A.6 | |
|---|---|
| A | Validate that no more than one 'identifer' attribute is allowed in an AbstractFeature class. |
| B | Validate that the `identifier` attribute is a valid implementation of the ScopedName class from ISO 19103. |

### A.1.3.4. Abstract Feature-Name

| ABSTRACT TEST A.7 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature-name` |
| REQUIREMENT | /req/core/abstractfeature-name |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `name` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `name` attribute. |
| A | Validate that zero or more 'name' attributes are allowed in an AbstractFeature class. |
| B | Validate that the `name` attribute is a valid implementation of the GenericName class from ISO 19103. |

## A.1.4. Abstract Feature with Lifespan

| ABSTRACT TEST A.8 | |
|---|---|
| IDENTIFIER | `/conf/core/featurewithlifespan` |
| REQUIREMENT | /req/core/featurewithlifespan |
| PREREQUISITE | Abstract test A.3: `/conf/core/abstractfeature` |
| TARGET TYPE | Implementation Specification |

| ABSTRACT TEST A.8 | |
|---|---|
| TEST PURPOSE | Validate that the Implementation Specification implements the AbstractFeaturewithLifespan Class as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Inspect the POI Implementation Specification to validate that the AbstractFeatureWithLifespan class is properly implemented. |
| A | Validate that the implementation of the AbstractFeatureWithLifespan class is also a valid implementation of the AbstractFeature class using test /conf/core/abstractfeature. |
| B | Validate correct implementation of the `creationDate` attribute using test /conf/core/abstractfeature-creationdate. |
| C | Validate correct implementation of the `terminationDate` attribute using test /conf/core/abstractfeature-terminationdate. |
| D | Validate correct implementation of the `validFrom` attribute using test /conf/core/abstractfeature-validfrom. |
| E | Validate correct implementation of the `validTo` attribute using test /conf/core/abstractfeature-validto. |

### A.1.4.1. Creation Date

| ABSTRACT TEST A.9 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature-creationdate` |
| REQUIREMENT | /req/core/abstractfeature-creationdate |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `creationDate` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `creationDate` attribute. |
| A | Validate that no more than one `creationDate` attribute is allowed in an AbstractFeatureWith Lifespan class. |
| B | Validate that the `creationDate` attribute is a valid implementation of the DateTime type from ISO 19103. |

### A.1.4.2. Termination Date

| ABSTRACT TEST A.10 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature-terminationdate` |
| REQUIREMENT | /req/core/abstractfeature-terminatyiondate |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `terminationDate` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `terminationDate` attribute. |
| A | Validate that no more than one `terminationDate` attribute is allowed in an AbstractFeature WithLifespan class. |
| B | Validate that the `terminationDate` attribute is a valid implementation of the DateTime type from ISO 19103. |

### A.1.4.3. Valid From

| ABSTRACT TEST A.11 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature-validfrom` |
| REQUIREMENT | /req/core/abstractfeature-validfrom |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `validFrom` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `validFrom` attribute. |
| A | Validate that no more than one `validFrom` attribute is allowed in an AbstractFeatureWith Lifespan class. |

| ABSTRACT TEST A.11 | |
|---|---|
| B | Validate that the `validFrom` attribute is a valid implementation of the DateTime type from ISO 19103. |

### A.1.4.4. Valid To

| ABSTRACT TEST A.12 | |
|---|---|
| IDENTIFIER | `/conf/core/abstractfeature-validto` |
| REQUIREMENT | /req/core/abstractfeature-validto |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `validTo` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `validTo` attribute. |
| A | Validate that no more than one 'validTo' attribute is allowed in an AbstractFeatureWith Lifespan class. |
| B | Validate that the `validTo` attribute is a valid implementation of the DateTime type from ISO 19103. |

## A.1.5. Abstract POI

| ABSTRACT TEST A.13 | |
|---|---|
| IDENTIFIER | `/conf/core/astract-poi` |
| REQUIREMENT | /req/core/abstract-poi |
| PREREQUISITE | Abstract test A.8: `/conf/core/featurewithlifespan` |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | To validate that the Implementation Specification implements the AbstractPOI class as defined in the POI Conceptual Model. |

| ABSTRACT TEST A.13 | |
|---|---|
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate that the Implementation Specification correctly implements the AbstractPOI class: |
| A | Validate that the implementation of the AbstractPOI class is also a valid implementation of the AbstractFeatureWithLifespan class using test /conf/core/featurewithlifespan. |
| B | Validate correct implementation of the `contactInfo` attribute using test <<ats_core_poi-contactinfo,/conf/core/poi-contactinfo>. |
| C | Validate correct implementation of the `hasFeatureOfInterest` aggregation using the test /conf/core/poi-featureOfInterest. |
| D | Validate correct implementation of the `hasMetadata` association using the test /conf/core/POI-Metadata. |
| E | Validate correct implementation of the `hasPayload` aggregation using the test /conf/core/poi-payload. |
| F | Validate correct implementation of the `keywords` attribute using the test /conf/core/poi-keywords. |
| G | Validate correct implementation of the `rights` attribute using the test /conf/core/poi-rights. |
| H | Validate correct implementation of the `symbology` association using the test /conf/core/poi-symbology. |

### A.1.5.1. Abstract POI ContactInfo

| ABSTRACT TEST A.14 | |
|---|---|
| IDENTIFIER | `/conf/core/poi-contactinfo` |
| REQUIREMENT | /req/core/poi-contactInfo |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `contactInfo` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `contactInfo` attribute. |
| A | Validate that zero or more `contactInfo` attributes are allowed in an AbstractPOI class. |

| ABSTRACT TEST A.14 | |
|---|---|
| B | Validate that the `contactInfo` attribute is a valid implementation of the CI_Responsibility class from ISO 19115-1:2014. |

### A.1.5.2. Abstract POI Feature Of Interest

| ABSTRACT TEST A.15 | |
|---|---|
| IDENTIFIER | `/conf/core/poi-featureofinterest` |
| REQUIREMENT | /req/core/poi-featureofinterest |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `hasFeatureOfinterest` aggregation as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and target class of the `hasFeatureOfInterest` aggregation. |
| A | Validate that zero or more `hasFeatureOfIntestest` aggregations are allowed in an Abstract POI class. |
| B | Validate that the target of the `hasFeatureOfIntestest` aggregation is a valid implementation of the Feature class from ISO 19109:2015. |

### A.1.5.3. Abstract POI Metadata

| ABSTRACT TEST A.16 | |
|---|---|
| IDENTIFIER | `/conf/core/poi-metadata` |
| REQUIREMENT | /req/core/poi-metadata |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `hasMetadata` association as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |

| ABSTRACT TEST A.16 | |
|---|---|
| DESCRIPTION | Validate the cardinality and encoding of the `hasMetadata` association. |
| A | Validate that zero or more `hasMetadata` associations are allowed in an AbstractPOI class. |
| B | Validate that the `hasMetadata` association is implemented as described in the Conceptual Model using the /conf/core/link test. |

### A.1.5.4. Abstract POI Keywords

| ABSTRACT TEST A.17 | |
|---|---|
| IDENTIFIER | `/conf/core/poi-keywords` |
| REQUIREMENT | /req/core/poi-keywords |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `keywords` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `keywords` attribute. |
| A | Validate that zero or more `keywords` attributes are allowed in an AbstractPOI class. |
| B | Validate that the `keywords` attribute is a valid implementation of the MD_Keywords class from ISO 19115-1:2014. |

### A.1.5.5. Abstract POI Rights

| ABSTRACT TEST A.18 | |
|---|---|
| IDENTIFIER | `/conf/core/poi-rights` |
| REQUIREMENT | /req/core/poi-rights |
| TARGET TYPE | Implementation Specification |

| ABSTRACT TEST A.18 | |
|---|---|
| TEST PURPOSE | Validate that the Implementation Specification implements the `rights` attribute as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and type of the `rights` attribute. |
| A | Validate that zero, one, or two `rights` attributes are allowed in an AbstractPOI class. |
| B | Validate that the `contactInfo` attribute is a valid implementation of the MD_Constraints class from ISO 19115-1:2014. |

### A.1.5.6. Abstract POI Symbology

| ABSTRACT TEST A.19 | |
|---|---|
| IDENTIFIER | `/conf/core/poi-symbology` |
| REQUIREMENT | /req/core/poi-symbology |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `symbology` association as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and encoding of the `symbology` association. |
| A | Validate that zero or one `symbology` associations are allowed in an AbstractPOI class. |
| B | Validate that the `symbology` association is implemented as described in the Conceptual Model using the /conf/core/link test. |

### A.1.5.7. Abstract POI Payload Association

| ABSTRACT TEST A.20 | |
|---|---|
| IDENTIFIER | `/conf/core/poi-haspayload` |

| ABSTRACT TEST A.20 | |
|---|---|
| REQUIREMENT | /req/core/poi-haspayload |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `hasPayload` aggregation as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and target class of the `hasPayload` aggregation. |
| A | Validate that zero of more `hasPayload` aggregation are allowed in an AbstractPOI class. |
| B | Validate that the target of the `hasPayload` aggregation is a valid implementation of the POI_Payload class using the /conf/core/poi-payload test. |

### A.1.5.8. Link

| ABSTRACT TEST A.21 | |
|---|---|
| IDENTIFIER | `/conf/core/link` |
| REQUIREMENT | /req/core/link |
| TARGET TYPE | Implementation Specification |
| NOTE | More needed. |
| TEST PURPOSE | Validate that the Implementation Specification implements the Link class as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate that the association being tested is a valid implementation of the Link class from |

## A.1.6. POI Payload

| ABSTRACT TEST A.22 | |
|---|---|
| IDENTIFIER | `/conf/core/poi-payload` |
| REQUIREMENT | /req/core/poi-payload |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | To validate that the Implementation Specification implements the POI_Payload Class as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate that the Implementation Specification correctly implements the POI_Payload class: |
| A | Validate correct implementation of the `usesSchema` aggregation using the test /conf/core/poi_payload/usesSchema. |
| B | Validate correct implementation of the `hasDefinition` aggregation using the test /conf/core/poi_payload/hasDefinition. |
| C | Validate correct implementation of the `hasFeatureOfInterest` aggregation using the test /conf/core/poi_payload/featureOfInterest. |

### A.1.6.1. POI_Payload Uses Schema

| ABSTRACT TEST A.23 | |
|---|---|
| IDENTIFIER | `/conf/core/poi_payload/usesSchema` |
| REQUIREMENT | /req/core/poi_payload/usesSchema |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `usesSchema` aggregation as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and target class of the `usesSchema` aggregation. |
| A | Validate that at least one `usesSchema` aggregation is allowed in an POI_Payload class. |
| B | Validate that the target of the `usesSchema` aggregation is a valid schema for the implementing technology. |

### A.1.6.2. POI_Payload Has Definition

| ABSTRACT TEST A.24 | |
|---|---|
| IDENTIFIER | `/conf/core/poi_payload/hasDefinition` |
| REQUIREMENT | /req/core/poi_payload/hasDefinition |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `hasDefinition` aggregation as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and target class of the `hasDefinition` aggregation. |
| A | Validate that no more than one `hasDefinition` aggregation is allowed in an POI_Payload class. |
| B | Validate that the target of the `hasDefinition` aggregation is a valid ontology for the implementing technology. |

### A.1.6.3. POI_Payload Feature Of Interest

| ABSTRACT TEST A.25 | |
|---|---|
| IDENTIFIER | `/conf/core/poi_payload/featureofinterest` |
| REQUIREMENT | /req/core/poi_payload/featureofinterest |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | Validate that the Implementation Specification implements the `hasFeatureOfinterest` aggregation as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate the cardinality and target class of the `hasFeatureOfInterest` aggregation. |
| A | Validate that zero or more `hasFeatureOfIntestest` aggregations are allowed in an POI_Payload class. |

| ABSTRACT TEST A.25 | |
|---|---|
| B | Validate that the target of the `hasFeatureOfInterest` aggregation is a valid implementation of the Feature class from ISO 19109:2015. |

## A.1.7. POI

| ABSTRACT TEST A.26 | |
|---|---|
| IDENTIFIER | `/conf/core/poi` |
| REQUIREMENT | /req/core/poi |
| PREREQUISITE | /conf/core/abstract-poi |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | To validate that the Implementation Specification implements the POI Class as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |
| DESCRIPTION | Validate that the Implementation Specification correctly implements the POI class: |
| A | Validate that the implementation of the POI class is also a valid implementation of the Abstract_POI class using test /conf/core/abstract-poi. |

## A.1.8. Feature of Interest

| ABSTRACT TEST A.27 | |
|---|---|
| IDENTIFIER | `/conf/core/feature-of-interest` |
| REQUIREMENT | /req/core/feature-of-interest |
| TARGET TYPE | Implementation Specification |
| TEST PURPOSE | To validate that the Implementation Specification implements the Feature of Interest Class as defined in the POI Conceptual Model. |
| TEST-METHOD-TYPE | Manual Inspection |

| ABSTRACT TEST A.27 | |
|---|---|
| **DESCRIPTION** | Validate that the Implementation Specification correctly implements the Feature of Interest: |
| **A** | Validate that the implementation of the FeatureOfInterest class is a compliant extension of the AnyFeature class defined in ISO 19109. |

B

# ANNEX B (INFORMATIVE) ISO DATA DICTIONARY

# B ANNEX B (INFORMATIVE) ISO DATA DICTIONARY

ISO Technical Committee 211 maintains a harmonized UML model which covers many of their standards. All of the TC211 Standards which are relevant to the POI Standard are included. Therefore the full UML model for POI consists of the classes defined in the POI UML model as well as those which referenced from the TC211 Hamonized UML model.

The Data Dictionary tables in this section were software generated from the TC211 Hamonized UML model. As such, this section provides a normative representation of the TC211 classes which are leveraged by the POI Conceptual Model.

Note that some of the properties in the ISO model are not populated. Since the model is normative, the missing information cannot be included in this document until it is first included in the ISO model by TC211.

## B.1. General Feature Model

The following classes are defined in (ISO 19109:2015)

**Table B.1** — Any Feature Class

| **AnyFeature** | |
|---|---|
| Definition: | The class AnyFeature is an instance of the «metaclass» FeatureType (ISO 19109). It represents the set of all classes which are feature types.<br>+ In an implementation this abstract class shall be substituted by a concrete class representing a feature type from an application schema associated with a domain of discourse (ISO 19109, ISO 19101). |
| StereoType: | «FeatureType» |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | FeatureType [1..1] | |

**Table B.2** — Feature Type Class

| FeatureType | |
|---|---|
| Definition: | feature: abstraction of real world phenomena<br>NOTE: A feature may occur as a type or an instance. Feature type or feature instance should be used when only one is meant.<br>This class describes how a feature class shall be constructed in an Application Schema. In accordance with the conformance clause of the standard, instances of this class are instanciated as feature classes in an Application Schema |
| Subclass Of: | IdentifiedType |
| StereoType: | «Metaclass» |
| Constraint: | name is mandatory (Invariant): |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | NS_AvoidList [0..*] | |
| superType | FeatureType [0..*] | |
| featureType Metadata | MD_Metadata [0..*] | |
| carrier OfCharacteristics | PropertyType [0..*] | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| isAbstract | Boolean | |

# B.2. Geometry

The following classes are defined in ISO 19107:2003

**Table B.3** — GM_Object Class

| GM_Object | |
|---|---|
| Definition: | GM_Object is the root class of the geometric object taxonomy and supports interfaces common to all geographically referenced geometric objects. GM_Object instances are sets of direct positions in a particular coordinate reference system. A GM_Object can be regarded as an infinite set of points that satisfies the set operation interfaces for a set of direct positions, TransfiniteSet<DirectPosition>. Since an infinite collection class cannot be implemented directly, a Boolean test for inclusion shall be provided by the GM_Object interface. This international standard concentrates on vector geometry classes, but future work may use GM_Object as a root class without modification. NOTE As a type, GM_Object does not have a well-defined default state or value representation as a data type. Instantiated subclasses of GM_Object will. |
| Subclass Of: | none |
| StereoType: | «type» |
| Constraint: | dimension() > boundary().dimension (Invariant): |
| Constraint: | boundary().notEmpty() implies boundary().dimension() = dimension() -1 (Invariant): |
| Constraint: | boundary().isEmpty() = isCycle() (Invariant): |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | Geometry [1..1] | |
| | TransfiniteSet<Direct Position> [1..1] | |
| | CV_DomainObject [1. .1] | |
| CRS | CRS [0..1] | |
| CRS | SC_CRS [0..1] | |

## **Table B.4** — GM_Point Class

**GM_Point**

| | |
|---|---|
| Definition: | GM_Point is the basic data type for a geometric object consisting of one and only one point. |
| Subclass Of: | GM_Primitive |
| StereoType: | «type» |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | Point [1..1] | |
| composite | GM_CompositePoint [0..*] | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| position | DirectPosition [1..1] | The attribute “position” shall be the DirectPosition of this GM_Point. GM_Point::position [1] : DirectPosition NOTE In most cases, the state of a GM_Point is fully determined by its position attribute. The only exception to this is if the GM_Point has been subclassed to provide additional non-geometric information such as symbology. |

## **Table B.5** — GM_LineString Class

**GM_LineString**

| | |
|---|---|
| Definition: | A GM_LineString (Figure 16) consists of sequence of line segments, each having a parameterization like the one for GM_LineSegment (See 6.4.11). The class essentially combines a Sequence<GM_LineSegments> into a single object, with the obvious savings of storage space. |
| Subclass Of: | GM_Primitive |
| StereoType: | «type» |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| controlPoint | GM_PointArray [1..1] | |

**Table B.6** — GM_Polygon Class

| GM_Polygon | |
|---|---|
| Definition: | A GM_Polygon (Figure 21) is a surface patch that is defined by a set of boundary curves and an underlying surface to which these curves adhere. The default is that the curves are coplanar and the polygon uses planar interpolation in its interior. |
| Subclass Of: | GM_Primitive |
| StereoType: | «type» |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| surface | GM_PolyhedralSurface [0..1] | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| boundary | GM_SurfaceBoundary | |
| spanningSurface | GM_Surface [0..1] | |

# B.3. Citation and responsible party information

The following classes are defined in (ISO 19115-1 Edition 1)

**Table B.7** — CI_Contact Class

| CI_Contact |
|---|

| | |
|---|---|
| Definition: | information required to enable contact with the responsible person and/or organisation |
| StereoType: | None |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| address | CI_Address [0..*] | physical and email address at which the organisation or individual may be contacted |
| contact Instructions | CharacterString [0..1] | supplemental instructions on how or when to contact the individual or organisation |
| contactType | CharacterString [0..1] | type of contact |
| hoursOfService | CharacterString [0..*] | time period (including time zone) when individuals can contact the organisation or individual |
| onlineResource | CI_OnlineResource [0..*] | on-line information that can be used to contact the individual or organisation |
| phone | CI_Telephone [0..*] | telephone numbers at which the organisation or individual may be contacted |

### **Table B.8** — CI_Individual Class

**CI_Individual**

| | |
|---|---|
| Definition: | information about the party if the party is an individual |
| Subclass Of: | CI_Party |
| StereoType: | None |
| Constraint: | count (name + positionName) > 0 (Invariant): |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | CI_Organisation [1..1] | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| positionName | CharacterString [0..1] | position of the individual in an organisation |

### **Table B.9** — CI_Organisation Class

**CI_Organisation**

| | |
|---|---|
| Definition: | information about the party if the party is an organisation |
| Subclass Of: | CI_Party |
| StereoType: | None |
| Constraint: | count (name + logo) > 0 (Invariant): |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| individual | CI_Individual [0..*] | an individual in the named organisation |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| logo | MD_BrowseGraphic [0..*] | Graphic identifying organization |

### **Table B.10** — CI_Party Class

**CI_Party**

| | |
|---|---|
| Definition: | information about the individual and/or organisation of the party |
| StereoType: | «abstract» |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | CI_Responsibility [] | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| contactInfo | CI_Contact [0..*] | contact information for the party |
| name | CharacterString [0..1] | name of the party (individual or organization) |

**Table B.11** — CI_Responsibility Class

**CI_Responsibility**

| | |
|---|---|
| Definition: | information about the party and their role |
| StereoType: | None |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| party | CI_Party [1..*] | information about the party |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| extent | EX_Extent [0..*] | spatial or temporal extent of the role |
| role | CI_RoleCode [1..1] | function performed by the responsible party |

# B.4. Constraint information

The following classes are defined in (ISO 19115-1 Edition 1)

## Table B.12

**MD_Constraints**

| | |
|---|---|
| Definition: | restrictions on the access and use of a resource or metadata |
| Subclass Of: | None |
| StereoType: | |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | MD_Identification [] | |
| | MD_Metadata [] | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| constraintApplicationScope | MD_Scope [0..1] | Spatial and temporal extent of the application of the constraint restrictions |
| graphic | MD_BrowseGraphic [0..*] | graphic /symbol indicating the constraint |
| reference | CI_Citation [0..*] | citation/URL for the limitation or constraint, eg. copyright statement, license agreement, etc |
| releasability | MD_Releasability [0..1] | information concerning the parties to whom the resource can or cannot be released |
| responsibleParty | CI_Responsibility [0..*] | party responsible for the resource constraints |
| useLimitation | CharacterString [0..*] | limitation affecting the fitness for use of the resource or metadata. Example, “not to be used for navigation” |

## Table B.13

**MD_LegalConstraints**

| | |
|---|---|
| Definition: | restrictions and legal prerequisites for accessing and using the resource or metadata |

| | |
|---|---|
| Subclass Of: | None |
| StereoType: | |
| Constraint: | otherConstraints: only documented if accessConstraints or useConstraints = “other Restrictions” (Invariant): |
| Constraint: | If MD_LegalConstraints used then count of (accessConstraints + useConstraints + other Constraints + useLimitation + releasability ) > 0 (Invariant): |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| access Constraints | MD_RestrictionCode [0..*] | access constraints applied to assure the protection of privacy or intellectual property, and any special restrictions or limitations on obtaining the resource or metadata |
| otherConstraints | CharacterString [0..*] | other restrictions and legal prerequisites for accessing and using the resource or metadata |
| useConstraints | MD_RestrictionCode [0..*] | constraints applied to assure the protection of privacy or intellectual property, and any special restrictions or limitations or warnings on using the resource or metadata |

## Table B.14

**MD_Releasability**

| | |
|---|---|
| Definition: | information about resource release constraints |
| Subclass Of: | None |
| StereoType: | |
| Constraint: | count (addressee + statement) > 0 (Invariant): |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| addressee | CI_Responsibility [0..*] | party to which the release statement applies |
| dissemination Constraints | MD_RestrictionCode [0..*] | component in determining releasability |
| statement | CharacterString [0..1] | release statement |

Table B.15

| MD_SecurityConstraints | |
|---|---|
| Definition: | handling restrictions imposed on the resource or metadata for national security or similar security concerns |
| Subclass Of: | None |
| StereoType: | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| classification | MD_Classification Code | name of the handling restrictions on the resource or metadata |
| classification System | CharacterString [0..1] | name of the classification system |
| handling Description | CharacterString [0..1] | additional information about the restrictions on handling the resource or metadata. |
| userNote | CharacterString [0..1] | explanation of the application of the legal constraints or other restrictions and legal prerequisites for obtaining and using the resource or metadata |

# B.5. Identification information

The following classes are defined in (ISO 19115-1 Edition 1)

**Table B.16** — MD_KeywordClass Class

| MD_KeywordClass | |
|---|---|
| Definition: | specification of a class to categorize keywords in a domain-specific vocabulary that has a binding to a formal ontology |
| StereoType: | None |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | MD_Keywords [] | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| className | CharacterString [1..1] | character string to label the keyword category in natural language |
| concept Identifier | URI [0..1] | URI of concept in ontology specified by the ontology attribute; this concept is labeled by the className: CharacterString. |
| ontology | CI_Citation [1..1] | a reference that binds the keyword class to a formal conceptualization of a knowledge domain for use in semantic processingNOTE: Keywords in the associated MD_Keywords keyword list must be within the scope of this ontology |

**Table B.17** — MD_Keywords Class

**MD_Keywords**

| | |
|---|---|
| Definition: | keywords, their type and reference source NOTE: When the resource described is a service, one instance of MD_Keyword shall refer to the service taxonomy defined in ISO 19119, 8.3) |
| StereoType: | None |
| Constraint: | When the resource described is a service, one instance of MD_Keyword shall refer to the service taxonomy defined in ISO 19119 (Invariant): |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| | MD_Identification [] | |
| keywordClass | MD_KeywordClass [0..1] | association of a MD_Keywords instance with a MD_KeywordClass to provide user-defined categorization of groups of keywords that extend or are orthogonal to the standardized KeywordTypeCodes and are associated with an ontology that allows additional semantic query processing |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| keyword | CharacterString [1..*] | commonly used word(s) or formalised word(s) or phrase(s) used to describe the subject |
| thesaurusName | CI_Citation [0..1] | name of the formally registered thesaurus or a similar authoritative source of keywords |
| type | MD_KeywordTypeCode [0..1] | subject matter used to group similar keywords |

# B.6. Name types

The following classes are defined in (ISO 19103:2015)

**Table B.18** — Generic Name Class

| GenericName | |
|---|---|
| Definition: | Generic Name is the abstract class for all names in a NameSpace. Each instance of a GenericName is either a LocalName or a ScopedName. A LocalName references a local object directly accessible from the NameSpace. A ScopedName is a composite of a Local Name for locating another NameSpace and a GenericName valid in that NameSpace. |
| StereoType: | interface |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| scope | NameSpace [1..1] | |

**Table B.19** — Local Name Class

| LocalName | |
|---|---|
| Definition: | A LocalName references a local object directly accessible from the NameSpace. |
| Subclass Of: | GenericName |

| | |
|---|---|
| StereoType: | interface |

### Table B.20 — Member Name Class

**MemberName**

| | |
|---|---|
| Definition: | A MemberName is a LocalName that references either an attribute slot in a record or record Type or an attribute, operation, or association role in an object instance or type description in some form of schema. |
| Subclass Of: | LocalName |
| StereoType: | interface |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| aName | CharacterString [1..1] | The stored value “aName” is the returned value for the “aName()” operation. |
| attributeType | TypeName [1..1] | The allowable type for this member. |

### Table B.21 — Namespace Class

**NameSpace**

| | |
|---|---|
| Definition: | A Name Space is a domain in which “names” given by character strings (possibly under local constrains constraints enforced by the Name Space) can be mapped to objects via a get Obejct operation. Examples include objects which form a Name Space for their attributes, operations and associations, or Schemas that form Name Spaces for their included data types or classes. Not all methods for NameSpaces need to be made publicly accessible. |
| StereoType: | interface |

| ROLE NAME | TARGET CLASS AND MULTIPLICITY | DEFINITION |
|---|---|---|
| name | GenericName [0..*] | |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| acceptableClassList | TypeName [1..1] | |
| isGlobal | Boolean [1..1] | |

### **Table B.22** — Scoped Name Class

| **ScopedName** | |
|---|---|
| Definition: | ScopedName is a composite of a LocalName for locating another NameSpace and a Generic Name valid in that NameSpace. ScopedName contains a LocalName as head and a Generic Name, which might be a LocalName or a ScopedName, as tail. |
| Subclass Of: | GenericName |
| StereoType: | interface |

### **Table B.23** — Type Name Class

| **TypeName** | |
|---|---|
| Definition: | A TypeName is a LocalName that references either a recordType or object type in some form of schema. The stored value “aName” is the returned value for the “aName()” operation. This is the types name. |
| Subclass Of: | LocalName |
| StereoType: | interface |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| aName | CharacterString [1..1] | The stored value “aName” is the returned value for the “aName()” operation. |

# B.7. Primitive types

The following classes are defined in (ISO 19103:2015)

## B.7.1. Date and Time

**Table B.24** — Date Class

**Date**

| | |
|---|---|
| Definition: | |
| StereoType: | interface |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| century | CharacterString [1..1] | |
| day | CharacterString [1..1] | |
| month | CharacterString [1..1] | |
| year | CharacterString [1..1] | |

**Table B.25** — DateTime Class

**DateTime**

| | |
|---|---|
| Definition: | |
| Subclass Of: | Date and Time |
| StereoType: | interface |

**Table B.26** — Time Class

| Time | |
|---|---|
| Definition: | |
| StereoType: | interface |

| ATTRIBUTE | VALUE TYPE AND MULTIPLICITY | DEFINITION |
|---|---|---|
| hour | CharacterString [1..1] | |
| minute | CharacterString [1..1] | |
| second | CharacterString [1..1] | |
| timeZone | CharacterString [1..1] | |

C

# ANNEX C (INFORMATIVE) REVISION HISTORY

# C ANNEX C (INFORMATIVE) REVISION HISTORY

Table C.1

| DATE | RELEASE | EDITOR | PRIMARY CLAUSES MODIFIED | DESCRIPTION |
|---|---|---|---|---|
| 2021-06-17 | 0.0.1 | Matthew Purss | all | initial version |
| 2021-07-08 | 0.0.1 | Matthew Purss | Clause 1 | initial scope text inserted from original POI draft standard |
| 2021-07-09 | 0.0.1 | Matthew Purss | Clause 4 | initial terms inserted from original POI draft standard (and reformatted to meet formal definition requirements) |

# BIBLIOGRAPHY

# BIBLIOGRAPHY


[1] ISO: ISO 1087-1, *Terminology work — Vocabulary — Part 1: Theory and application*. International Organization for Standardization, Geneva https://www.iso.org/standard/20057.html.

[2] ISO/IEC: ISO/IEC 2382, *Information technology — Vocabulary*. International Organization for Standardization, International Electrotechnical Commission, Geneva https://www.iso.org/standard/63598.html.

[3] ISO: ISO 11404:2007, ISO (2007).

[4] ISO: ISO 19104, *Geographic information — Terminology*. International Organization for Standardization, Geneva https://www.iso.org/standard/63541.html.

[5] ISO: ISO 19111:2019, *Geographic information — Referencing by coordinates*. International Organization for Standardization, Geneva (2019). https://www.iso.org/standard/74039.html.

[6] ISO: ISO 19112, *Geographic information — Spatial referencing by geographic identifiers*. International Organization for Standardization, Geneva https://www.iso.org/standard/70742.html.

[7] ISO: ISO 19117:2012, *Geographic information — Portrayal*. International Organization for Standardization, Geneva (2012). https://www.iso.org/standard/46226.html.

[8] ISO: ISO 19118, *Geographic information — Encoding*. International Organization for Standardization, Geneva https://www.iso.org/standard/44212.html.

[9] ISO: ISO 19119:2016, *Geographic information — Services*. International Organization for Standardization, Geneva (2016). https://www.iso.org/standard/59221.html.

[10] ISO: ISO 19133, *Geographic information — Location-based services — Tracking and navigation*. International Organization for Standardization, Geneva https://www.iso.org/standard/32551.html.

[11] ISO: ISO 19136-1, *Geographic information — Geography Markup Language (GML) — Part 1: Fundamentals*. International Organization for Standardization, Geneva https://www.iso.org/standard/75676.html.

[12] ISO: ISO 19143, *Geographic information — Filter encoding*. International Organization for Standardization, Geneva https://www.iso.org/standard/42137.html.

[13] ISO: ISO 19150-1, *nofetch*. ISO

[14] ISO: ISO 19150-2, *Geographic information — Ontology — Part 2: Rules for developing ontologies in the Web Ontology Language (OWL)*. International Organization for Standardization, Geneva https://www.iso.org/standard/57466.html.

[15] ISO: ISO 19150-4, *Geographic information — Ontology — Part 4: Service ontology*. International Organization for Standardization, Geneva https://www.iso.org/standard/72177.html.

[16] ISO: ISO 19155, *Geographic information — Place Identifier (PI) architecture*. International Organization for Standardization, Geneva https://www.iso.org/standard/32573.html.

[17] ISO: ISO 19156:2011, *Geographic information — Observations and measurements*. International Organization for Standardization, Geneva (2011). https://www.iso.org/standard/32574.html.

[18] ISO: ISO 19160-4, *Addressing — Part 4: International postal address components and template language*. International Organization for Standardization, Geneva https://www.iso.org/standard/83470.html.

[19] ISO/IEC: ISO/IEC 19501, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*. International Organization for Standardization, International Electrotechnical Commission, Geneva https://www.iso.org/standard/32620.html.

[20] Object Management Group, Model Driven Architecture Guide rev. 2.0