Open Geospatial Consortium

Submission Date: <yyyy-mm-dd>

Approval Date: <yyyy-mm-dd>

Publication Date: <yyyy-mm-dd>

External identifier of this OGC® document: http://www.opengis.net/doc/{doc-type}/{standard}/{m.n}

Internal reference number of this OGC® document: YY-nnnrx

Version: 0.01

Category: OGC® Implementation Specification

Editor: Kyoung-Sook Kim, Taehoon Kim, Mahmoud SAKR

OGC API - Moving Features Standard

Document type:	OGC® Implementation Specification
Document stage:	Draft
Document language:	English

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD.

THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications. This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

Table of Contents

Chapter 1. Introduction

i. Abstract

<Insert Abstract Text here>

ii. Keywords

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

ogcdoc, OGC document, OGC MovingFeature, MovingFeatures JSON, MovingFeature Access, API, OpenAPI, REST, trajectory

iii. Preface

NOTE	Insert Preface Text here. Give OGC specific commentary: describe the technical content, reason for document, history of the document and precursors, and plans for future work. > 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. Submitting organizations

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

- Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology
- Université libre de Bruxelles

v. Submitters

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

Name	Organization
Kyoung-Sook KIM	Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology
Taehoon KIM	Artificial Intelligence Research Center, National Institute of Advanced Industrial Science and Technology
Mahmoud SAKR	Université libre de Bruxelles

Chapter 2. Scope

NOTE Insert Scope text here. Give the subject of the document and the aspects of that scope covered by the document.

Chapter 3. Conformance

This Standard defines XXXX.

Requirements for N standardization target types are considered: * AAAA * BBBB

Conformance with this Standard shall be checked using all the relevant tests specified in Annex A (normative) of this document. The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in the OGC Compliance Testing Policies and Procedures and the OGC Compliance Testing web site.

In order to conform to this OGC® Standard, a software implementation shall choose to implement: * Any one of the conformance levels specified in Annex A (normative). * Any one of the Distributed Computing Platform profiles specified in Annexes TBD through TBD (normative).

All requirements-classes and conformance-classes described in this document are owned by the Standard(s) identified.

Chapter 4. References

The following normative documents contain provisions that, through reference in this text, constitute provisions of this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

Insert References here. If there are no references, state “There are no normative references”.

References are to follow the Springer LNCS style, with the exception that optional information may be appended to references: DOIs are added after the date and web resource references may include an access date at the end of the reference in parentheses. See examples from Springer and OGC below.

Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. J. Mol. Biol. 147, 195–197 (1981)

May, P., Ehrlich, H.C., Steinke, T.: ZIB Structure Prediction Pipeline: Composing a Complex Biological Workflow through Web Services. In: Nagel, W.E., Walter, W.V., Lehner, W. (eds.) Euro-Par 2006. LNCS, vol. 4128, pp. 1148–1158. Springer, Heidelberg (2006)

Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)

Czajkowski, K., Fitzgerald, S., Foster, I., Kesselman, C.: Grid Information Services for Distributed Resource Sharing. In: 10th IEEE International Symposium on High Performance Distributed Computing, pp. 181–184. IEEE Press, New York (2001)

NOTE

Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration. Technical report, Global Grid Forum (2002)

National Center for Biotechnology Information, http://www.ncbi.nlm.nih.gov

ISO / TC 211: ISO 19115-1:2014 Geographic information — Metadata — Part 1: Fundamentals (2014)

ISO / TC 211: ISO 19157:2013 Geographic information — Data quality (2013)

ISO / TC 211: ISO 19139:2007 Geographic information — Metadata — XML schema implementation (2007)

ISO / TC 211: ISO 19115-3: Geographic information — Metadata — Part 3: XML schemas (2016)

OGC: OGC 15-097 OGC Geospatial User Feedback Standard. Conceptual Model (2016)

OGC: OGC 12-019, OGC City Geography Markup Language (CityGML) Encoding Standard (2012)

OGC: OGC 14-005r3, OGC IndoorGML (2014)

Chapter 5. Terms and Definitions

This document used 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 standard 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.

application programming interface (API)

a formally defined set of types and methods which establish a contract between client code which uses the API and implementation code which provides the API

coordinate

one of a sequence of numbers designating the position of a point

Note 1 to entry: In a spatial coordinate reference system, the coordinate numbers are qualified by units.

[source: ISO 19111]

coordinate reference system (CRS)

coordinate system that is related to an object by a datum

Note 1 to entry: Geodetic and vertical datums are referred to as reference frames.

Note 2 to entry: For geodetic and vertical reference frames, the object will be the Earth. In planetary applications, geodetic and vertical reference frames may be applied to other celestial bodies.

[source: ISO 19111]

dataset

identifiable collection of data

[source: ISO 19115-1]

datatype

specification of a value domain with operations allowed on values in this domain

Examples: `Integer`, `Real`, `Boolean`, `String` and `Date`.

Note 1 to entry: Data types include primitive predefined types and user definable types.

[source: ISO 19103]

dynamic attribute

characteristic of a feature in which its value varies with time

[source: OGC 16-140]

feature

abstraction of a real world phenomena

Note 1 to entry: A feature can occur as a type or an instance. Feature type or feature instance should be used when only one is meant.

[source: ISO 19109]

feature attribute

characteristic of a feature

Note 1 to entry: A feature attribute can occur as a type or an instance. Feature attribute type or feature attribute instance is used when only one is meant.

[source: ISO 19109]

feature table

table where the columns represent feature attributes, and the rows represent features

[source: OGC 06-104]

geographic feature

representation of real world phenomenon associated with a location relative to the Earth

[source: ISO 19101-2]

geometric object

spatial object representing a geometric set

[source: ISO 19107:2003]

moving feature

feature whose location changes over time

Note 1 to entry: Its base representation uses a local origin and local coordinate vectors of a geometric object at a given reference time.

Note 2 to entry: The local origin and ordinate vectors establish an engineering coordinate reference system (ISO 19111), also called a local frame or a local Euclidean coordinate system.

property

facet or attribute of an object referenced by a name

[source: ISO 19143]

trajectory

path of a moving point described by a one parameter set of points

[source: ISO 19141]

Chapter 6. Conventions

This section provides details and examples for any conventions used in the document. Examples of conventions are symbols, abbreviations, use of XML schema, or special notes regarding how to read the document.

6.1. Identifiers

The normative provisions in this Standard are denoted by the URI

http://www.opengis.net/spec/{standard}/{m.n}

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

Chapter 7. Overview

7.1. General

The OGC API standards enable access to resources using the HTTP protocol and its associated operations (GET, PUT, POST, etc.) OGC API-Common defines a set of features which are applicable to all OGC APIs. Other OGC standards extend API-Common with features specific to a resource type. This OGC API-MovingFeatures standard defines an API with two goals:

1. Provide access to ***Moving Features*** conformance to the OGC Moving Features JSON encoding standard.
2. Provide functionality comparable to that of the OGC Moving Features Access standard.

Resources exposed through an OGC API may be accessed through a Universal Resource Identifier (URI). URIs are composed of three sections:

- Dataset distribution API: The endpoint corresponding to a dataset distribution, where the landing page resource as defined in OGC API-Common-Part 1: Core is available (subsequently referred to as Base URI or `{root}`)
- Access Paths: Unique paths to Resources
- Query Parameters: Parameters to adjust the representation of a Resource or Resources like encoding format or subsetting

Access Paths are used to build resource identifiers. It is recommended, but not required. Most resources are also accessible through links on previously accessed resources. Unique relation types are used for each resource.

Table 1 summarizes the access paths and relation types defined in this standard.

Table 1. Moving Features API Paths

Path Template	Relation	Resource
Common		
{root}/	none	Landing page for this dataset distribution
{root}/api	`service-desc` or `service-doc`	API Description (optional)
{root}/conformance	`conformance`	Conformance Classes
Collections		
{root}/collections	`data`	Metadata describing the MovingFeatureCollection of data available from this API.
{root}/collections/{collectionId}		Metadata describing the MovingFeatureCollection of data which has the unique identifier `{collectionId}`
Moving Features		

Path Template	Relation	Resource
{root}/collections/{collectionId}/items	items	Retrieve metadata about available items
{root}/collections/{collectionId}/items/{mFeatureId}	item	
{root}/collections/{collectionId}/items/{mFeatureId}/temporalGeometry	items	
{root}/collections/{collectionId}/items/{mFeatureId}/temporalGeometry/{tGeometryId}	item	
{root}/collections/{collectionId}/items/{mFeatureId}/temporalProperties	items	
{root}/collections/{collectionId}/items/{mFeatureId}/temporalProperties/{tPropertyName}	item	
Processes		

Where:

- `{root}` = Base URI for the API server
- `{collectionId}` = an identifier for a specific MovingFeatureCollection of data
- `{mFeatureId}` = an identifier for a specific MovingFeatures of a specific MovingFeatureCollection of data
- `{tGeometryId}` = an identifier for a specific TemporalGeometry of a specific MovingFeatures of data
- `{tPropertyName}` = an identifier for a specific TemporalProperty of a specific MovingFeatures of data

7.2. API Behavior Model

T.B.D

7.3. Search

The core search capability is based on OGC API-Common and thus supports:

- bounding box searches,
- time instant or time period searches,

- and equality predicates (i.e. *property=value*).

OGC API-MovingFeatures extends these core search capabilities to include:

- keyword searches.

7.4. Dependencies

The OGC API-MovingFeatures (shortly, API-MF) standard is an extension of the OGC API-Common and the OGC API-Features standards. Therefore, an implementation of API-MF shall first satisfy the appropriate Requirements Classes from API-Common and API-Features. Table 2, identifies the OGC API - Common and OGC API - Features Requirements Classes which are applicable to each section of this Standard. Instructions on when and how to apply these Requirements Classes are provided in each section.

Table 2. Required OGC API - Common and OGC API - Features Requirements Classes

API - Record Section	OGC API - Common, OGC API - Features Requirements Class
API Landing Page	http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core
API Definition	http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core
Declaration of Conformance Classes	http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core
MovingFeatureCollection	http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core
MovingFeatures	http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core
TemporalGeometry	http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core
TemporalProperties	http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core
OpenAPI 3.0	http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/oas30
JSON	http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/geojson

Chapter 8. Requirements Class "Collection Catalog"

8.1. Overview

Requirements Class	
http://www.opengis.net/spec/ogcapi-movingfeatures-1/1.0/req/movingfeature-collection	
Target type	Web API

The `Collection Catalog` requirements class defines the requirements for a collection. A collection is an object that provides information about and access to a set of related MovingFeature.

The schema for the collection object presented in this clause is an extension of the collection schema defined in OGC API-Common and OGC API-Features.

8.2. Schema of a collection metadata

Table 3 defines the set of properties that may be used to describe a collection.

Table 3. Table of collection properties

Property	Requirement	Description
id	M	A unique identifier to the collection.
title	M	A human-readable name given to the collection.
description	O	A free-text description of the collection.
links	M	A list of links for navigating the API (e.g. link to previous or next pages; links to alternative representations, etc.)
extent	O	The spatio-temporal coverage of the collection.
itemType	M	Fixed to the value "movingfeature".
crs	O	A list of coordinate reference system used for spatial-temporal values.
updateFrequency	M	A time interval of sampling location.

NOTE The properties *id*, *title*, *description*, *links*, *extent*, *itemsType* and *crs* were inherited from OGC API-Common and OGC API-Features.

Requirement 1	/req/core/mandatory-collection A collection object SHALL contain all the mandatory properties listed in Table 3.

8.2.1. Update Frequency

An update frequency is one of the most important properties of moving feature collection. It is determined by a data source. It can use to determine the continuous of the moving feature's trajectory.

8.3. Collections

The `Collections` operation returns a set of metadata which describes the collections available from this API. The catalog of collections returned to the response can be limited using the bbox, datetime, and limit parameters.

Table 4. Dependencies

http://www.opengis.net/spec/ogcapi-common-1/1.0/req/collections

8.3.1. Operation

This operation is defined in the `Collections` conformance class of API-Common. No modifications are needed to support `MovingFeature` resources.

1. Issue a `GET` request on `{root}/collections` path

Support for GET on the `{root}/collections` path is required by API-Common.

Requirement 2	/req/core/mfc-op
A	A MovingFeatures API implementation SHALL comply with the API - Common requirement `http://www.opengis.net/spec/ogcapi_common-2/1.0/req/collections`.
B	For API - MovingFeatures, the API - Common `/rec/collections/rc-md-item-type` recommendation SHALL apply to collections where the value of the `itemType` property is specified as `movingfeature`.

8.3.2. Response

A successful response to the `Collections` Operation is a document that contains summary metadata for each collection accessible though the API. In a typical API deployment, the `Collections` Response will list collections of all offered resource types. The collections where the value of the `itemType` property is `movingfeature` is collections of moving features.

Collections Response Schema

```
type: object
required:
  - collections
  - links
properties:
  collections:
    type: array
    items:
      $ref: collectionInfo.yaml
  links:
    type: array
    items:
      $ref: link.yaml
```

The following JSON payload is an example of a response to an OGC API-MovingFeatures Collections operation.

Collections Example


```
{
  "collections": [
    {
      "id": "Miraikan",
      "itemType": "movingfeature",
      "extent": {
        "spatial": {
          "bbox": [
            -180, -90, 190, 90
          ],
          "crs": [
            "http://www.opengis.net/def/crs/OGC/1.3/CRS84"
          ]
        },
        "temporal": {
          "interval": [
            "2011-11-11T12:22:11Z","2012-11-24T12:32:43Z"
          ],
          "trs": [
            "http://www.opengis.net/def/uom/ISO-8601/0/Gregorian"
          ]
        }
      },
      "links": [
        {
          "href": "https://pntml.io/mf/collections/Miraikan",
          "rel": "self"
        }
      ]
    }
  ],
  "links": [
    {
      "href": "https://pntml.io/mf/collections",
      "rel": "self"
    }
  ]
}
```


8.3.3. Error situations

The requirements for handling unsuccessful requests are provided in [http-response]. General guidance on HTTP status codes and how they should be handled is provided in [http-status-codes].

8.4. Collection Information

Collection Information is the set of metadata that describes a single collection. An abbreviated copy

of this information is returned for each Collection in the `/collections` response.

Table 5. Dependencies

http://www.opengis.net/spec/ogcapi_common-1/1.0/req/collections

8.4.1. Operation

This operation is defined in the `Collection` conformance class of API-Common. No modifications are required to support `MovingFeature` resources.

1. Issue a `GET` request on the `{root}/collections/{collectionId}` path

The {collectionId} parameter is the unique identifier for a single collection offered by the API. The list of valid values for `{collectionId}` is provided in the `/collections` response.

Support for the `/collections/{collectionId}` path is required by OGC API-Common.

8.4.2. Response

A successful response to the Collection operation is a set of metadata that describes the collection identified by the `{collectionId}` parameter.

Requirement 3	**/req/core/mfc-response**
A	A successful execution of the operation SHALL be reported as a response with an HTTP status code `200`.
B	The response SHALL only include moving features selected by the request.
C	Each moving feature in the response SHALL include the mandatory properties listed in Table 3.

Collection Information Response Schema

```
type: object
required:
  - id
  - links
  - itemType
properties:
  id:
    description: identifier of the collection used, for example, in URIs
    type: string
  title:
    description: human readable title of the collection
    type: string
    example: address
  description:
    description: a description of the features in the collection
    type: string
    example: An address.
  links:
    type: array
    items:
      $ref: link.yaml
  extent:
    $ref: extent.yaml
  itemType:
    description: indicator about the type of the items in the collection.
    type: string
    enum:
      - feature
      - movingfeature
```

The following JSON payload is an example of a response to an OGC API-MovingFeatures Collection Information operation.

Collection Information Example


```
{
  "id": "Miraikan",
  "itemType": "movingfeature",
  "extent": {
    "spatial": {
      "bbox": [
        -180, -90, 190, 90
      ],
      "crs": [
        "http://www.opengis.net/def/crs/OGC/1.3/CRS84"
      ]
    },
    "temporal": {
      "interval": [
        "2011-11-11T12:22:11Z","2012-11-24T12:32:43Z"
      ],
      "trs": [
        "http://www.opengis.net/def/uom/ISO-8601/0/Gregorian"
      ]
    }
  },
  "links": [
    {
      "href": "https://pntml.io/mf/collections/Miraikan",
      "rel": "self"
    }
  ]
}
```


8.4.3. Error situations

The requirements for handling unsuccessful requests are provided in [http-response]. General guidance on HTTP status codes and how they should be handled is provided in [http-status-codes].

Chapter 9. Requirements Class "MovingFeature"

9.1. Overview

Requirements Class	
http://www.opengis.net/spec/ogcapi-movingfeatures-1/1.0/req/movingfeature	
Target type	Web API

The `MovingFeature` requirements class defines the requirements for a moving feature.

Table 6. Dependencies

http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-op
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-limit-definition
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-limit-default-minimum-maximum
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-limit-response-1
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-bbox-definition
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-bbox-response
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-time-definition
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-time-response
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-timeStamp
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-numberMatched
http://www.opengis.net/spec/ogcapi-features-1/1.0/req/core/fc-numberReturned

9.2. Features

The MF-API Items query is an OGC API-Features endpoint that may be used to catalog pre-existing moving features. The `Features` operation returns a set of features which describes the moving feature available from this API. If a `mFeatureID` is not specified, the query will return a list of the available mFeatureID's. This behavior is specified in OGC API-Features. all other parameters for use with the Items query are defined by OGC API-Features. The list of features returned to the response can be limited using the bbox, datetime, and limit parameters.

9.2.1. Operation

This operation is defined in the `Features` conformance class of API-Features. No modifications are needed to support `MovingFeature` resources.

1. Issue a `GET` request on `{root}/collections/{collledctionID}/items` path

Support for GET on the `{root}/collections/{collledctionID}/items` path is required by API-Features.

9.2.2. Response

A successful response to the `Features` Operation is a document that contains the static data of moving features. In a typical API deployment, the `Features` Response will list features of all offered resource types.

Features Response Schema

```
type: object
required:
  - type
  - features
properties:
  type:
    type: string
    enum:
      - FeatureCollection
  features:
    type: array
    items:
      $ref: featureGeoJSON.yaml
  links:
    type: array
    items:
      $ref: link.yaml
  timeStamp:
    type: string
    format: date-time
  numberMatched:
    type: integer
    minimum: 0
  numberReturned:
    type: integer
    minimum: 0
```

The following JSON payload is an example of a response to an OGC API-MovingFeatures Features operation.

Features Example


```
{
  "type": "FeatureCollection",
  "features":[
    {
      "type": "Feature",
      "id": "feature-000001",
      "bbox":[
        -122.59750209, 37.48803556, -122.2880486, 37.613537207
      ],
      "time":[
        "2011-07-14T22:01:01Z",
        "2011-07-15T01:11:22Z"
      ],
      "geometry":{
        "type": "Polygon",
        "coordinates":[
          [-122.308150179, 37.488035566],
          [-122.597502109, 37.538869539],
          [-122.576687533, 37.613537207],
          [-122.2880486, 37.562818007],
          [-122.308150179, 37.488035566]
        ]
      },
      "properties":{
        "label": "car"
      }
    }
  ],
  "links":[
    {
      "href": "https://pntml.io/mf/collections/Mirakan/mfeatures",
      "rel": "self"
    },
    {
      "href": "https://pntml.io/mf/collections/Mirakan/mfeatures?resourceKey=06f11bfa-
9fa9-452d-b9a8-249d6aee9d4f",
      "rel": "next"
    }
  ],
  "timeStamp": "2020-01-01T12:00:00Z",
  "numberMatched": 1,
  "numberReturned": 1
}
```


9.2.3. Error situations

The requirements for handling unsuccessful requests are provided in [http-response]. General guidance on HTTP status codes and how they should be handled is provided in [http-status-codes].

Chapter 10. Requirements Class "TemporalGeometry"

10.1. Overview

10.1.1. Operation

10.1.2. Response

10.1.3. Parameters

10.1.4. Error situations

Chapter 11. Requirements Class "TemporalProperties"

11.1. Overview

11.1.1. Operation

11.1.2. Response

11.1.3. Parameters

11.1.4. Error situations

Chapter 12. Query and Information Resources

Chapter 13. General Requirements

Annex A: Requirements Detail

NOTE	Ensure that there is a conformance class for each requirements class and a test for each requirement (identified by requirement name and number)

A.1. Conformance Class A

A.1.1. Requirement 1

Test id:	/conf/conf-class-a/req-name-1
Requirement:	/req/req-class-a/req-name-1
Test purpose:	Verify that…
Test method:	Inspect…

A.1.2. Requirement 2

Annex B: Abstract Test Suite (Normative)

Annex C: Examples (Informative)

Annex D: Relationship with other OGC/ISO standards (Informative)

This specification is built upon the following OGC/ISO standards. The geometry concept is presented first, followed by the feature concept. Note that a feature is *not* a geometry, but a feature often contains a geometry as one of its attributes. However it is legal to build features without geometry attribute, or with more than one geometry attributes.

D.1. Static geometries, features and accesses

The following standards define static objects, without time-varying properties.

D.1.1. Geometry (ISO 19107)

The ISO 19107, *Geographic information — Spatial schema* standard defines a `GM_Object` base type which is the root of all geometric objects. Some examples of `GM_Object` subtypes are `GM_Point`, `GM_Curve`, `GM_Surface` and `GM_Solid`. A `GM_Object` instance can be regarded as an infinite set of points in a particular coordinate reference system. The standard provides a `GM_CurveInterpolation` code list to identify how those points are computed from a finite set of points. Some interpolation methods listed by ISO 19107 are (non-exhaustive list):

linear

Positions on a straight line between each consecutive pair of control points.

geodesic

Positions on a geodesic curve between each consecutive pair of control points. A geodesic curve is a curve of shortest length. The geodesic shall be determined in the coordinate reference system of the curve.

circularArc3Points

For each set of three consecutive control points, a circular arc passing from the first point through the middle point to the third point. Note: if the three points are co-linear, the circular arc becomes a straight line.

elliptical

For each set of four consecutive control points, an elliptical arc passing from the first point through the middle points in order to the fourth point. Note: if the four points are co-linear, the arc becomes a straight line. If the four points are on the same circle, the arc becomes a circular one.

cubicSpline

The control points are interpolated using initial tangents and cubic polynomials, a form of degree 3 polynomial spline.

The UML below shows the `GM_Object` base type with its operations (e.g. `distance(⋯)` for computing the distance between two geometries). `GM_Curve` (not shown in this UML) is a subtype of

GM_Primitive. All operations assume static objects, without time-varying coordinates or attributes.

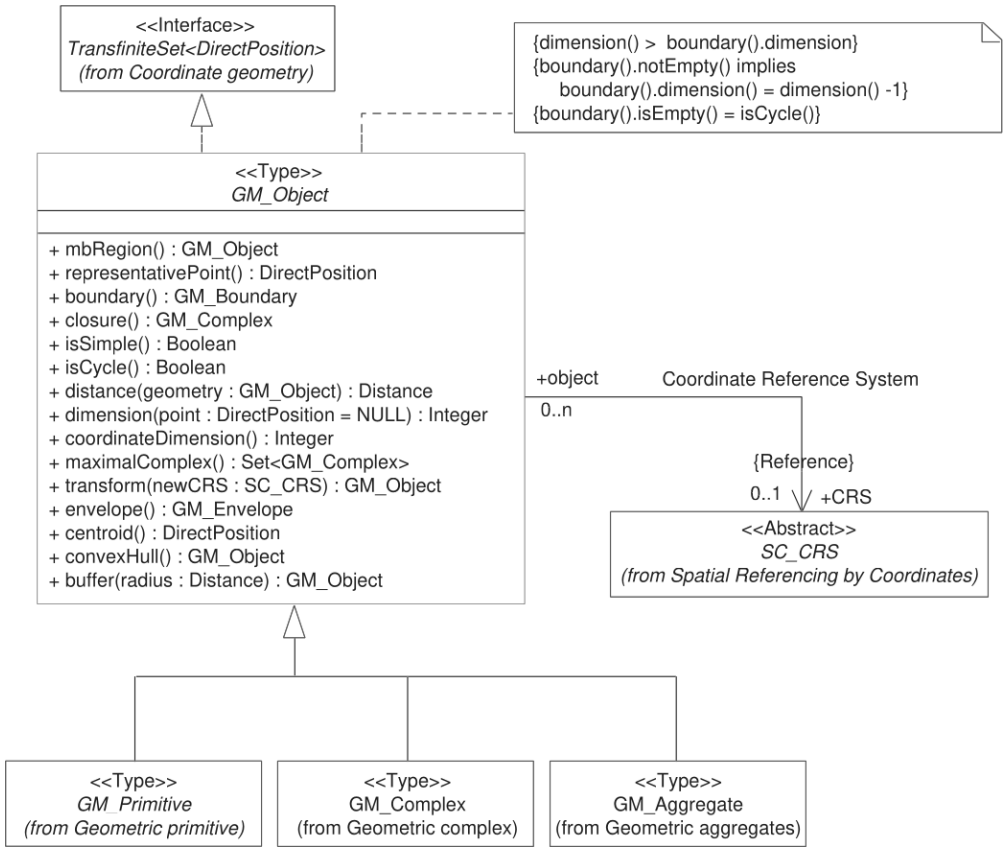

Figure 1. GM_Object from ISO 19107:2003 figure 6

TODO: above discussion is based on ISO 19107:2003. It needs to be updated for latest revisions.

TODO: provide a simplified version of this UML.

Geometry, topology and temporal-objects (GM_Object, TP_Object, TM_Object) are not abstractions of real-world phenomena. These types can provide types for feature properties as described in the next section, but cannot be specialized to features.

D.1.2. Features (ISO 19109)

The ISO 19109, *Geographic information — Rules for application schema* standard defines types for the definition of features. A feature is an abstraction of a real-world phenomena. The terms “feature type” and “feature instance” are used to separate the following concepts of “feature”:

Feature type

The whole collection of real-world phenomena classified in a concept. For example the “bridge” feature type is the abstraction of the collection of all real-world phenomena that is classified into the concept behind the term “bridge”.

Feature instance

A certain occurrence of a feature type. For example “Tower Bridge” feature instance is the abstraction of a certain real-world bridge in London.

In object-oriented modelling, feature types are equivalent to classes and feature instances are equivalent to objects,

The UML below shows the General Feature Model. `FeatureType` is a metaclass that is instantiated as classes that represent individual feature types. A `FeatureType` instance contains the list of properties (attributes, associations and operations) that feature instances of that type can contain. Geometries are properties like any other, without any special treatment. All properties are static, without time-varying values.

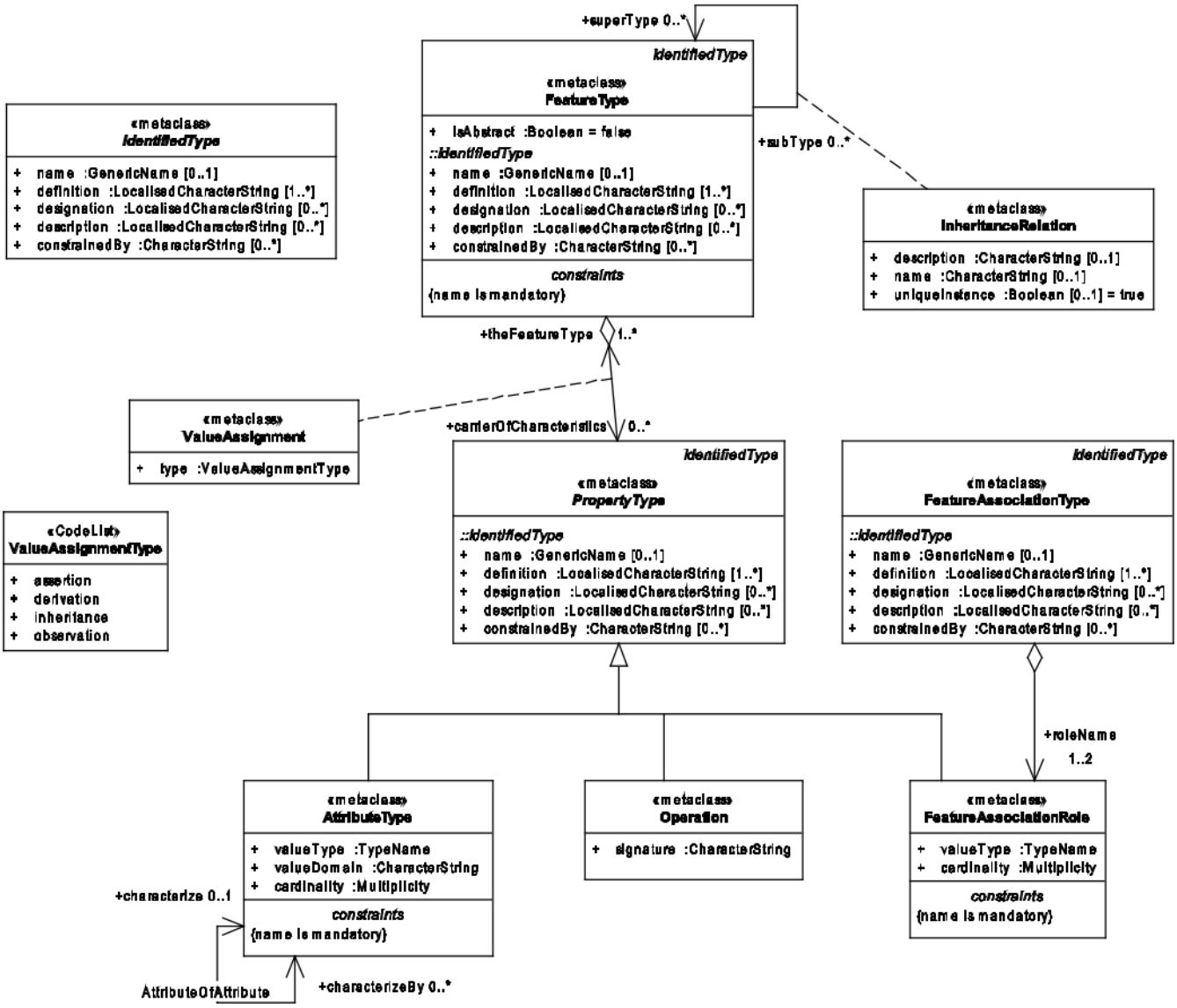

Figure 2. General Feature Model from ISO 19109:2009 figure 5

TODO: provide a simplified version of this UML.

D.1.3. Simple Features SQL

The Simple Feature Access — Part 2: SQL Option standard describes a feature access implementation in SQL based on a profile of ISO 19107. This standard defines *feature table* as a table where the columns represent feature attributes, and the rows represent feature instances. The geometry of a feature is one of its feature attributes.

D.1.4. Filter Encoding (ISO 19143)

The ISO 19143, *Geographic information — Filter encoding* standard (also OGC standard) provides types for constructing queries. These objects can be transformed into a SQL “SELECT … FROM …

WHERE … ORDER BY …” statement to fetch data stored in a SQL-based relational database. Similarly, the same objects can be transformed into an XQuery expression in order to retrieve data from XML document. The UML below shows the objects used for querying a subset based on spatial operations such as “contains” or “intersects”.

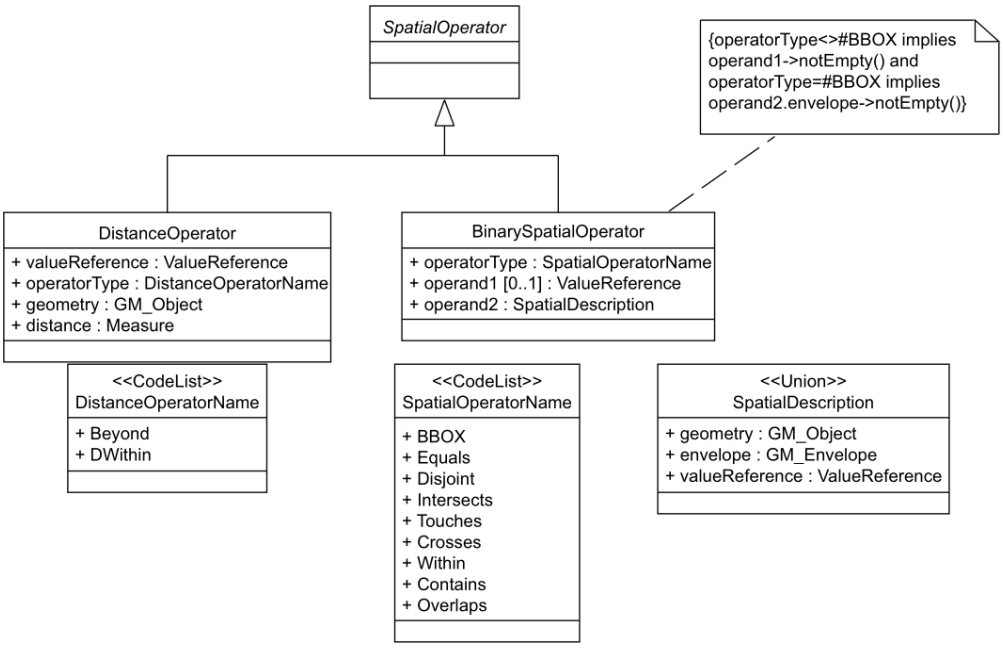

Figure 3. Spatial operators from ISO 19143 figure 6

D.1.5. Features web API

The OGC 17-069, *Features — Part 1: Core* standard specifies the fundamental building blocks for interacting with features using Web API. This base standards allow to get all features available on a server, or to get feature instances by their identifier.

D.1.6. Features Filtering web API

The OGC TBD, *Features — Part 3: Filtering and the Common Query Language (CQL)* standard extends the Feature web API with capabilities to encode more sophisticated queries. The conceptual model is close to ISO 19143.

D.2. Temporal geometries and moving Features

D.2.1. Moving Features (ISO 19141)

The ISO 19141, *Geographic information — Schema for moving features* standard extends the ISO 19107 spatial schema for addressing features whose locations change over time. Despite the “Moving Features” name, that standard is more about “Moving geometries”. The UML below shows how the `MF_Trajectory` type extends the “static” types from ISO 19107.

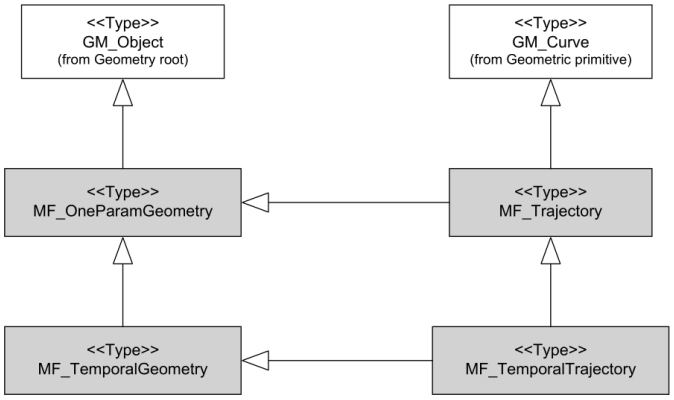

Figure 4. Trajectory type from ISO 19141 figure 3

Trajectory inherits some operations shown below. Those operations are in addition to the operations inherited from `GM_Object`. For example the `distance(⋯)` operation from ISO 19107 is now completed by a `nearestApproach(⋯)` operation.

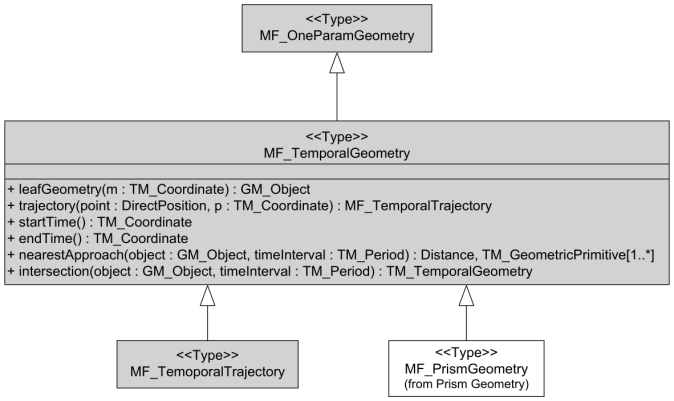

Figure 5. Temporal geometry from ISO 19141 figure 6

D.2.2. Moving Features XML encoding (OGC 18-075)

The OGC 18-075 *Moving Features Encoding Part I: XML Core* standard takes a subset of ISO 19141 specification and encodes it in XML format. But that standard also completes ISO 19141 by allowing to specify attributes whose value change over time. This extension to above *General Feature Model* is shown below:

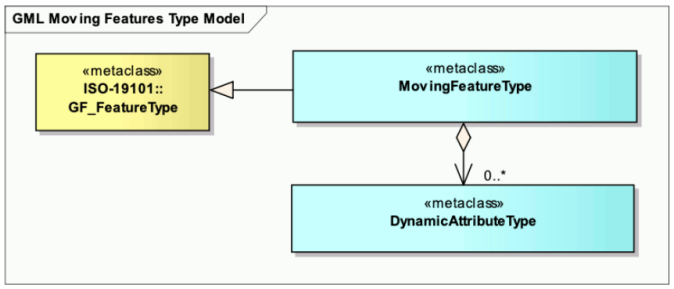

Figure 6. Dynamic attribute from OGC 18-075 figure 3

D.2.3. Moving Features JSON encoding (OGC 19-045)

The OGC 19-045 *Moving Features Encoding Extension — JSON* standard takes a subset of ISO 19141 specification and encodes it in JSON format. The specification provides various UML diagrams summarizing ISO 19141.

D.2.4. Moving Feature Access

The OGC 16-120, *Moving Features Access* standard (TODO)

Annex E: Revision History

Date	Release	Editor	Primary clauses modified	Description
2021-07-06	0.01	Taehoon Kim	all	initial version (dummy)

Annex F: Bibliography

[1] OGC: OGC Testbed 12 Annex B: Architecture. (2015).