OGC API - Environmental Data Retrieval Standard

Open Geospatial Consortium

Submission Date: 2020-11-20

Approval Date: <yyyy-mm-dd>

Publication Date: <yyyy-mm-dd>

External identifier of this OGC® document: http://www.opengis.net/doc/is/ogcapi-edr-1/1.0

Internal reference number of this OGC® document: 19-086r2

Version: 0.9

Category: OGC® Implementation Specification

Editors: Mark Burgoyne, Dave Blodgett, Chuck Heazel, Chris Little

OGC API - Environmental Data Retrieval Standard

Document type:
OGC®ImplementationSpecification

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

The Environmental Data Retrieval (EDR) Application Programming Interface (API) provides a family of lightweight query interfaces to access Environmental Data resources by requesting data at a **Position**, within an **Area** or along a **Trajectory**. An Environmental Data resource is a collection of spatiotemporal data that can be sampled using the EDR query pattern geometries. These patterns are described in the Requirement Class Core section.

EDR API query patterns, such as Position, Area, Cube, Trajectory or Corridor, can be thought of as discrete sampling geometries, conceptually consistent with the feature of interest in the Sensor Observation Service (SOS). A typical EDR data resource is a multidimensional dataset that could be accessed via the Web Coverage Service (WCS). In contrast to SOS and WCS, EDR implements the technical baseline of the OGC API family of standards and aims to provide a single set of simple-to-use query patterns. Use cases for EDR range from real or virtual time-series observation retrievals, to sub-setting 4-dimensional data cubes along user-supplied sampling geometries. These query patterns do not attempt to satisfy the full scope of either SOS or WCS, but provide useful building blocks to allow the composition of APIs that satisfy a wide range of geospatial data use cases. By defining a set focused querying patterns (and no requirement to implement all of them), the EDR API should help to simplify the design of systems (as they can be performance tuned for the supported queries) making it easier to build robust and scalable infrastructure.

With the OGC API family of standards. the OGC community has extended its suite of standards to include Resource Oriented Architectures and Web Application Programming Interfaces (APIs). These standards are based on a shared foundation, the OGC API-Common standard, which defines the resources and access paths that are supported by all OGC APIs. These are listed in Table 1. This document extends that foundation to define the Environmental Data Retrieval API.

Table 1. Overview of Resources

Resource	Path	HTTP Method	Document Reference
Landing page	`/`	GET	API Landing Page
API definition	`/api`	GET	API Definition
Conformance classes	`/conformance`	GET	Declaration of Conformance Classes
Collections metadata	`/collections`	GET	Collections Metadata
Collection instance metadata	`/collections/{collection_id}`	GET	Collection Metadata

CAUTION

During the development phase, these components use a base URL of `https://github.com/opengeospatial/Environmental-Data-Retrieval-API/master`, but eventually they are expected to be available under the base URL `http://schemas.opengis.net/ogcapi-edr-1/1.0`.

The resources identified in Table 1 primarily support Discovery operations. Discovery operations allow clients to interrogate the API to determine its capabilities and obtain information (metadata) about a distribution of a resource. This includes the API definition of the server(s) as well as metadata about the resources provided by those servers.

This standard extends the Table 1 common query operations by defining simple, coordinate-based, queries which are applicable to many geospatial resource types. Other OGC API standards may define additional query capabilities specific to their resource type. EDR Query operations allow resources or values to be retrieved from the underlying geospatial resource data store. The information returned is based upon the selection criteria (query string) provided by the client.

ii. Keywords

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

ogcdoc, OGC document, property, geographic information, spatial data, spatial things, dataset, distribution, API, geojson, covJSON, html, OpenAPI, AsyncAPI, REST, Common, position, area, trajectory, corridor, time-series

iii. Preface

OGC Declaration

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

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

iv. Submitting organizations

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

- UK Met Office
- US Geological Service
- US National Weather Service
- Wuhan University
- Meteorological Service of Canada
- Finnish Meteorological Institute
- ESRI
- NASA
- Météo-France

v. Submitters

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

Name	**Affiliation**
Mark Burgoyne *(editor)*	Met Office
Chris Little *(editor)*	Met Office
Chuck Heazel *(editor)*	HeazelTech LLC
Dave Blodgett *(editor)*	USGS

Chapter 2. Scope

This specification identifies resources, captures compliance classes, and specifies requirements which are applicable to OGC Environmental Data Retrieval APIs.

This specification addresses two fundamental operations: discovery and query.

Discovery operations allow the API to be interrogated to determine its capabilities and retrieve information (metadata) about a distribution of a resource. This includes the API definition of the server as well as metadata about the Environmental Data resources provided by the server.

An Environmental Data resource is a **collection** of spatio-temporal data that can be sampled using OGC-API Environmental Data Retrieval query patterns.

Query operations allow other environmental data resources to be sampled from the underlying Environmental Data resource, or data store, based upon EDR query geometry and other selection criteria, defined by this standard and selected by the client.

Chapter 3. Conformance

Conformance with this standard shall be checked using the tests specified in Abstract Test Suite (Normative) of this document. The framework, concepts, and methodology for testing, and the criteria to claim conformance are specified in the OGC Compliance Testing Policies and Procedures and the OGC Compliance Testing web site.

The one Standardization Target for this standard is Web APIs.

OGC API - Common - Core defines an API module intended for re-use by other OGC Web API standards. This OGC API - EDR standard is an extension of OGC API - Common - Core. Conformance to this standard requires demonstrated conformance to the applicable Conformance Classes of OGC API - Common - Core.

This standard identifies nine (9) Conformance Classes. The Conformance Classes implemented by an API are advertised through the `/conformance` path on the landing page. Each Conformance Class is defined by one or more Requirements Classes. The test classes in Requirements Detail are organized by Requirements Class. The Requirements Classes define the functional requirements which will be tested through the associated Conformance Class.

The conformance classes and their constituent requirements classes for OGC API-EDR are:

- Core, one Requirement Class

The *Core Requirements Class* is the minimal useful service interface for an OGC API. The requirements specified in this Requirements Class are mandatory for all implementations of the EDR API.

The Core requirements class is specified in **Requirements Class Core** (Chapter 8).

- Collections, one Requirement Class

The API-Common Part 2: Collections standard extends the API-Common Part 1: Core to enable discovery and query access to collections of spatial resources.

The structure and organization of a collection of spatial data is very much dependent on the nature of that data and the expected access patterns. This is information which cannot be specified in a common manner. The OGC API-Common Part 2: Collections, specifies the requirements necessary to discover and understand a generic collection of spatial data.

The EDR API *Collection Requirements Class* extends the common requirements to those specific to the query and retrieval of collections of Environmental Data.

The EDR Collection requirements class is specified in **Requirements Class Collection** (Chapter 9).

- Encodings, three Requirements Classes
  - HTML
  - GeoJSON
  - CoverageJSON

Neither the *Core* nor *Collection* requirements classes mandate specific encodings or formats for representing resources. The *HTML*, *GeoJSON* and *CoverageJSON* requirements classes specify representations for these resources in frequently used encodings for spatial data on the web.

None of these encodings are mandatory. An implementation of the EDR API may decide to implement another encoding instead of, or in addition to, those listed. However, a common format does simplify interoperability so support for *CoverageJSON* is highly recommended as an established, efficient and effective format.

The Encoding Requirements Classes are specified in **Encoding Requirements Classes** (Chapter 10).

- OpenAPI 3.0, one Requirements Class

The OGC API-Common Part 1:Core standard does not mandate any encoding or format for the formal definition of the API. However, the prefered option is the OpenAPI 3.0 specification. Therefore the EDR APIs will be defined using *OpenAPI 3.0*.

The OpenAPI 3.0 Requirements Class is specified in **OpenAPI 3.0 Requirements Class** (Chapter 10).

- Queries, eight Requirements classes
  - Position
  - Radius
  - Area
  - Cube
  - Trajectory
  - Corridor
  - Items
  - Locations
  - Instances

The EDR API Queries Conformance class does not mandate any specific query patterns for querying resources. The *Position*, *Area* and *Trajectory* requirements classes specify query patterns for which there are ubiquitous use cases.

An implementation of the EDR API may decide to implement another query pattern instead of, or in addition to, those listed. However, a minimal query pattern of retrieving data at a position (with elevation and time) does simplify interoperability so support for the position query is highly recommended.

The Queries Requirements Classes are specified in **Queries Requirements Classes** (Chapter 9).

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.

- Open API Initiative: **OpenAPI Specification 3.0.3**, https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.3.md
- Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T.: **IETF RFC 2616, HTTP/1.1**, https://tools.ietf.org/rfc/rfc2616.txt
- Rescorla, E.: **IETF RFC 2818, HTTP Over TLS**, https://tools.ietf.org/rfc/rfc2818.txt
- Klyne, G., Newman, C.: **IETF RFC 3339, Date and Time on the Internet: Timestamps**, https://tools.ietf.org/rfc/rfc3339.txt
- Berners-Lee, T., Fielding, R., Masinter, L: **IETF RFC 3896, Uniform Resource Identifier (URI): Generic Syntax**, https://tools.ietf.org/rfc/rfc3896.txt
- Butler, H., Daly, M., Doyle, A., Gillies, S., Hagen, S., Schaub, T.: **IETF RFC 7946, The GeoJSON Format**, https://tools.ietf.org/rfc/rfc7946.txt
- Nottingham, M.: **IETF RFC 8288, Web Linking**, https://tools.ietf.org/rfc/rfc8288.txt
- W3C: **HTML5, W3C Recommendation**, https://www.w3.org/TR/html5/
- **Schema.org**: https://schema.org/docs/schemas.html
- Blower, J., Riechert, M., Roberts, B.: **Overview of the CoverageJSON format**, https://www.w3.org/TR/covjson-overview/
- Weibel, S., Kunze, J., Lagoze, C., Wolf, M.: **IETF RFC 2413, Dublin Core Metadata for Resource Discovery**, https://tools.ietf.org/rfc/rfc2413.txt
- Herring, J.: **Simple Feature Access - Part 1: Common Architecture**, http://portal.opengeospatial.org/files/?artifact_id=25355
- Lott, R.: **Well-Known Text representation of Coordinate Reference Systems**, http://docs.opengeospatial.org/is/18-010r7/18-010r7.html
- Portele, C., Vretanos, P, Heazel, C.: **OGC API - Features - Part 1: Core**, http://www.opengis.net/doc/IS/ogcapi-features-1/1.0

Chapter 5. Terms and Definitions

This document uses the terms defined in Sub-clause 5 of OGC API-Common Part 1 (OGC 19-072), 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.

The Glossary includes terms from other standards and specifications that, while not normative, are critical to accurately understand this specification.

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

5.1. area

region specified with a geographic envelope that may have vertical dimension

5.2. corridor

path of a moving point described by a one-parameter window around a set of points

5.3. cube

rectangular area, with a vertical extent

5.4. location

identifiable geographic place

SOURCE: ISO 19112

NOTE	A location is represented by one of a set of data types that describe a position, along with metadata about that data, including coordinates (from a coordinate reference system), a measure (from a linear referencing system), or an address (from an address system).

5.5. instance

version, release, or run of a given data collection

5.6. position

a place specified with a geographic point

5.7. radius

region specified with a geographic point and radial distance

5.8. trajectory

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

SOURCE: ISO 19141

Chapter 6. Conventions

6.1. Identifiers

The Architecture of the World Wide Web establishes the URI as the single global identification system for the Web. Therefore, URIs or URI Templates are used in OGC Web API standards to identify key entities in those standards.

The normative provisions in this standard are denoted by the URI:

http://www.opengis.net/spec/ogcapi-edr-1/1.0

All Requirements and Conformance Tests that appear in this document are denoted by partial URIs which are relative to this base.

A key requirement of Web API standards is the unambiguous identification of the resources they address. In an implementation of such a standard, URIs would be used to identify those resources. A standard, however, is not an implementation. A standard can identify potential resources, but not the resources themselves. Therefore, OGC Web API standards use URI Templates to identify resource categories. These resource categories are instantiated in the implementation of the standard.

The scope of each URI Template is specified in the standard. In some cases, API implementations are required to implement the template as a path in their API. In most cases they are optional.

Implementation of the URI Templates is recommended in that they provide a common look and feel to implementations of OGC Web API standards.

6.2. Link relations

To express relationships between resources, RFC 8288 (Web Linking) and registered link relation types are used wherever possible and denoted below with [IANA]. Additional link relation types are registered with the OGC Link Relation Registry. These are denoted below with [OGC].

The following link-relations are in common use by OGC Web API Standards.

- **alternate**: Refers to a substitute for this context. [IANA]
- **collection**: The target IRI points to a resource which represents the collection resource for the context IRI. [IANA]
- **conformance**: Refers to a resource that identifies the specifications that the link's context conforms to. [OGC]
- **data**: refers to the root resource of a dataset in an API. [OGC]
- **describedby**: Refers to a resource providing information about the link's context. [IANA]
- **item**: The target IRI points to a resource that is a member of the collection represented by the context IRI. [IANA]
- **items**: Refers to a resource that comprises members of the collection represented by the link's

context. [OGC]

- **license**: Refers to a license associated with this context. [IANA]
- **self**: Conveys an identifier for the link's context. [IANA]
- **service-desc**: Identifies service description for the context that is primarily intended for consumption by machines. [IANA]
  - API definitions are considered service descriptions.
- **service-doc**: Identifies service documentation for the context that is primarily intended for human consumption. [IANA]

Each resource representation includes an array of links. Implementations are free to add additional links for all resources provided by the API. For example, an **enclosure** link could reference a bulk download of a collection. Or a **related** link on a feature could reference a related feature.

A **license** link could be used for constraints on the data retrieved. Multiple **license** links could be provided for different content types. Alternatively, if all data retrieved via the API is available under the same license, the link MAY instead be added to the top-level links property of the response.

NOTE	The query patterns of the EDR API use the link relation **data**. It is envisaged that, in the future, this link relation may be replaced by **position**, **area**, and **trajectory** which would all be specializations of the currently used **data**.

6.3. Media Types

JSON media types that would typically be used in an OGC API that supports JSON are:

- `application/prs.coverage+json` for resources that include coverage content, and
- `application/geo+json` for feature collections and features, and
- `application/json` for all other resources.

XML media types that would typically occur in an OGC API that supports XML are:

- `application/gml+xml;version=3.2` for any GML 3.2 feature collections and features,
- `application/gml+xml;version=3.2;profile=http://www.opengis.net/def/profile/ogc/2.0/gml-sf0` for GML 3.2 feature collections and features conforming to the GML Simple Feature Level 0 profile,
- `application/gml+xml;version=3.2;profile=http://www.opengis.net/def/profile/ogc/2.0/gml-sf2` for GML 3.2 feature collections and features conforming to the GML Simple Feature Level 2 profile, and
- `application/xml` for all other resources.

The typical HTML media type for all "web pages" in an OGC API would be `text/html`.

The media types for an OpenAPI definition are `vnd.oai.openapi+json;version=3.0` (JSON) and `application/vnd.oai.openapi;version=3.0` (YAML).

NOTE	The OpenAPI media type has not been registered yet with IANA and may change.
NOTE	The CoverageJSON media type has not been registered yet with IANA and may change.

6.4. Examples

Most of the examples provided in this standard are encoded in JSON. JSON was chosen because it is widely understood by implementers and easy to include in a text document. This convention should NOT be interpreted as a requirement that JSON must be used. Implementors are free to use any format they desire as long as there is a Conformance Class for that format and the API advertises its support for that Conformance Class.

6.5. Schema

JSON Schema is used throughout this standard to define the structure of resources. These schema are typically represented using YAML encoding. This convention is for the ease of the user. It does not prohibit the use of another schema language or encoding. Nor does it indicate that JSON schema is required. Implementations should use a schema language and encoding appropriate for the format of the resource.

6.6. Use of HTTPS

For simplicity, this document generally refers to the HTTP protocol. This is not meant to exclude the use of HTTPS and simply is a shorthand notation for "HTTP or HTTPS". In fact, most servers are expected to use HTTPS, not HTTP.

6.7. API definition

6.7.1. General remarks

Good documentation is essential for every API so that developers can more easily learn how to use the API. In the best case, documentation would be available both in HTML for human consumption and in a machine readable format that can be best processed by software for run-time binding.

This standard specifies requirements and recommendations for APIs that share spatial resources and want to follow a standard way of doing so. In general, APIs will go beyond the requirements and recommendations stated in this standard. They will support additional operations, parameters, etc. that are specific to the API or the software tool used to implement the API.

6.7.2. Role of OpenAPI

This document uses OpenAPI 3.0 fragments as examples and to formally state requirements. Using OpenAPI 3.0 is not required for implementing an OGC API. Other API definition languages may be used along with, or instead of OpenAPI. However, any API definition language used should have an associated conformance class advertised through the `/conformance` path.

This approach is used to avoid lock-in to a specific approach to defining an API. This standard includes a conformance class for API definitions that follow the OpenAPI specification 3.0. Conformance classes for additional API definition languages will be added as the API landscape continues to evolve.

In this document, fragments of OpenAPI definitions are shown in YAML since YAML is easier to format than JSON and is typically used in OpenAPI editors.

6.7.3. References to OpenAPI components in normative statements

Some normative statements (requirements, recommendations and permissions) use a phrase that a component in the API definition of the server must be "based upon" a schema or parameter component in the OGC schema repository.

In this case, the following changes to the pre-defined OpenAPI component are permitted:

- If the server supports an XML encoding, `xml` properties may be added to the relevant OpenAPI schema components.
- The range of values of a parameter or property may be extended (additional values) or constrained (if a subset of all possible values are applicable to the server). An example for a constrained range of values is to explicitly specify the supported values of a string parameter or property using an `enum`.
- Additional properties may be added to the schema definition of a Response Object.
- Informative text may be changed or added, like comments or description properties.

For API definitions that do not conform to the OpenAPI Specification 3.0 the normative statement should be interpreted in the context of the API definition language used.

6.7.4. Paths in OpenAPI definitions

All paths in an OpenAPI definition are relative to the base URL of a server. Unlike Web Services, an API is decoupled from the server(s). Some ramifications of this are:

- An API may be hosted (replicated) on more than one server.
- Parts of an API may be distributed across multiple servers.

Example 1. URL of the OpenAPI definition

If the OpenAPI Server Object looks like this:

```
servers:
  - url: https://dev.example.org/
    description: Development server
  - url: https://data.example.org/
    description: Production server
```

The path `/mypath` in the OpenAPI definition of the API would be the URL `https://data.example.org/mypath` for the production server.

6.7.5. Reusable OpenAPI components

Reusable components for OpenAPI definitions for a OGC API are referenced from this document.

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 facilities which are applicable to all OGC APIs. Other OGC standards extend API-Common with facilities specific to a resource type.

This OGC API-EDR standard defines an API with two goals:

1. To allow clients to retrieve a subset of data created by the API in response to a standardized, coordinate orientated, query pattern;
2. To provide 'building blocks' allowing the construction of more complex applications.

The EDR API can be considered a 'Sampling API'. The query creates a discrete sampling geometry against the Environmental Data resource of a relatively persistent data store. The query and its response are transient resources, which could be made persistent for re-use if required.

The functionality provided by EDR query patterns could be realized through specific implementation of the SOS (and to some extent WCS) from the OGC Web Services family of (XML-based) standards. EDR introduces a streamlined JSON-based OGC API implementation of building blocks that could be used for many of the same use cases addressed by SOS and WCS in the past.

7.2. Resource Paths

Table 2 summarizes the access paths and relation types defined in this standard.

Table 2. Environmental Data API Paths

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

Path Template	Relation	Resource
{root}/collections/{collectionId}/items		Retrieve metadata about available items
Queries		
{root}/collections/{collectionId}/{queryType}		Retrieve data according to the query pattern
{root}/collections/{collectionId}/instances		Retrieve metadata about instances of a collection
{root}/collections/{collectionId}/instances/{instanceId}		Retrieve metadata from a specific instance of a collection which has the unique identifier `{instanceId}`

Where:

- `{root}` = Base URI for the API server
- `{collectionId}` = an identifier for a specific collection of data
- `{instanceId}` = an identifier for a specific version or instance of a collection of data
- `{queryType}` = an identifier for a specific query pattern to retrieve data from a specific collection of data

Chapter 8. Dependencies on API-Common Core and Collections

The OGC API-EDR standard is an extension of the OGC API-Common Part 1: Core and Part 2: Collections standards. Therefore, an implementation of API-EDR shall first satisfy the appropriate Requirements Classes from API-Common.

Requirement 1: /req/core/api-common OGC API-Common

8.1. Overview

The `Core` Requirements Class defines the requirements for locating, understanding, and accessing environmental data resources.

Requirements Class: OGC API-Environmental Data Retrieval Core

The `Core` Requirements Class is presented in five sections:

1. API Platform: a set of common capabilities
2. Collection Access: operations for accessing collections of environmental data.
3. Environmental Resources: operations for accessing environmental data resources
4. Query Resources: operations for accessing environmental data resources through queries
5. Parameters: parameters for use in the API-EDR operations.
6. General: general principles for use with this standard.

Table 3 Identifies the API-Common 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 3. Mapping API-EDR Sections to API-Common Requirements Classes

API-EDR Section	API-Common Requirements Class
API Landing Page	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core
API Definition	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core
Declaration of Conformance Classes	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core
Collections	http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections
OpenAPI 3.0	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/oas30
GeoJSON	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/json
CoverageJSON	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/json
HTML	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/html

8.2. Platform

API-Common defines a set of common capabilities which are applicable to any OGC Web API. Those capabilities provide the platform upon which resource-specific APIs can be built. This section describes those capabilities and any modifications needed to better support Environmental Data resources.

8.2.1. API landing page

The landing page provides links to start exploration of the resources offered by an API. Its most important component is a list of links. OGC API-Common already requires some common links. Those links are sufficient for this standard.

Table 4. Dependencies

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

Operation

The `Landing Page` operation is defined in the `Core` conformance class of API-Common. No modifications are needed to support `Environmental Data` resources. The `Core` conformance class specifies only one way of performing this operation:

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

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

Response

A successful response to the `Landing Page` operation is defined in API-Common. The schema for this resource is provided in Landing Page Response Schema.

Landing Page Response Schema

```
type: object
required:
  - links
properties:
  title:
    type: string
    example: Meteorological data server
  description:
    type: string
    example: Access to Meteorological data via a Web API that conforms to the OGC
      API Environmental Data Retrieval specification.
  links:
    type: array
    items:
      $ref: link.yaml
    example:
      - href: http://www.example.org/edr/api
```

```
      hreflang: en
      rel: service
      type: application/openapi+json;version=3.0
      title: ""
    - href: http://www.example.org/edr/conformance
      hreflang: en
      rel: data
      type: application/json
      title: ""
    - href: http://www.example.org/edr/collections
      hreflang: en
      rel: data
      type: application/json
      title: ""
keywords:
  type: array
  items: string
  example:
    - Temperature
    - Wind
    - Point
    - Trajectory
provider:
  type: object
  properties:
    name:
      description: Name of organization providing the service
      type: string
    url:
      description: Link to service providers website
      type: string
contact:
  type: object
  properties:
    email:
      description: Email address of service provider
      type: string
    phone:
      description: Phone number of service provider
      type: string
    fax:
      description: Fax number of service provider
      type: string
    hours:
      type: string
    instructions:
      type: string
    address:
      type: string
    postalCode:
      type: string
```

```
      city:
        type: string
      stateorprovince:
        type: string
      country:
        type: string
```

The following JSON fragment is an example of a response to an OGC API-EDR Landing Page operation.

Landing Page Example

```
{
  "title": "string",
  "description": "string",
  "links": [
    {
      "href": "http://data.example.org/",
      "rel": "self",
      "type": "application/json",
      "title": "this document"
    },
    {
      "href": "http://data.example.org/api",
      "rel": "service-desc",
      "type": "application/openapi+json;version=3.0",
      "title": "the API definition"
    },
    {
      "href": "http://data.example.org/conformance",
      "rel": "conformance",
      "type": "application/json",
      "title": "OGC conformance classes implemented by this API"
    },
    {
      "href": "http://data.example.org/collections",
      "title": "Metadata about the resource collections"
    }
  ]
}
```

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.

8.2.2. API definition

Every API is required to provide a definition document that describes the capabilities of that API. This definition document can be used by developers to understand the API, by software clients to

connect to the server, or by development tools to support the implementation of servers and clients.

Table 5. Dependencies

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

Operation

This operation is defined in the `Core` conformance class of API-Common. No modifications are needed to support `Environmental Data` resources. The `Core` conformance class describes two ways of performing this operation:

1. Issue a `GET` request on the `{root}/api` path
2. Follow the `service-desc` or `service-doc` link on the landing page

Only the link is required by API-Common.

Response

A successful response to the API Definition request is a resource which documents the design of the API. API-Common leaves the selection of format for the API Definition response to the API implementor. However, the options are limited to those which have been defined in the API-Common standard. At this time OpenAPI 3.0 is the only option provided.

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.

8.2.3. Declaration of conformance classes

To support "generic" clients that want to access multiple OGC API standards and extensions - and not "just" a specific API server, the API has to declare the conformance classes it claims to have implemented.

Table 6. Dependencies

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

Operation

This operation is defined in the `Core` conformance class of API-Common. No modifications are needed to support `Environmental Data` resources. The `Core` conformance class describes two ways of performing this operation:

1. Issue a `GET` request on the `{root}/conformance` path
2. Follow the `conformance` link on the landing page

Both techniques are required by API-Common.

Response

A successful response to the Conformance operation is a list of URLs. Each URL identifies an OGC Conformance Class for which this API claims conformance. The schema for this resource is defined in API-Common and provided for reference in Conformance Response Schema.

Requirement 2: /req/core/conformance Core conformance classes

Conformance Response Schema

```
type: object
required:
  - conformsTo
properties:
  conformsTo:
    type: array
    items:
      type: string
```

The following JSON fragment is an example of a response to an OGC API-EDR conformance operation.

Conformance Information Example

```
{
  "conformsTo": [
    "http://www.opengis.net/spec/ogcapi-edr-1/1.0/conf/core",
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core",
    "http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections",
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/oas3",
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/html",
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/json",
  ]
}
```

Chapter 9. Query, Environmental and Information Resources

Query resources are spatial queries which support the operation of the API or the access and use of the Environmental Data resources. The API-EDR standard has identified an initial set of common `queryTypes` to implement, described in the Query Resources section. This list may change as the standard is used and experience gained.

An Environmental Data resource is a collection of spatiotemporal data that can be sampled using the API-EDR query patterns.

9.1. Requirements Class: Collections

http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections	
Target type	Web API
Dependency	Requirements Class "OAPI Core"
Dependency	ISO 19107
Dependency	ISO 19108
Dependency	ISO 19111
Dependency	ISO 19108
Dependency	ISO 8601

Query resources related to Environmental Data resources (collections of spatiotemporal data) can be exposed using the path templates:

- `/collections/{collectionId}/{queryType}`
- `/collections/{collectionId}/instances/{instanceId}/{queryType}`

Where

`{collectionId}` = a unique identifier for a collection of geospatial data.

`{instanceId}` = a text string identifying the version or instance of the chosen collection.

`{queryType}` = a text string identifying the query pattern performed by the API.

The `instanceId` parameter allows support for multiple instances or versions of the same underlying datasource to be accessed by the API. This is applicable when the entire datasource has been regenerated rather than individual values in the datasource being changed. If only one instance of the datasource exists a value of `default` or `latest` could be used.

Information resources associated with a specific collection should be accessed through the `/collections` path. Information resources not associated with a specific collection should be accessed via the `/{instanceId}/{queryType}` path template.

The resources returned from each node in these templates are described in Table 7.

9.2. Information Resources

Table 7. Information Resource Paths

Path Template	Resource
/collections	The root resource describing the collections of geospatial data available from this API.
/collections/{collectionId}	Identifies a collection of geospatial data with the unique identifier `{collectionId}`
/collections/{collectionId}/{queryType}	Identifies an Information Resource of type {queryType} associated with the `{collectionId}` collection.

The OGC API-Common standards do not define any information resource types. However Table 8 provides a mapping of the initial query types proposed for the EDR API.

9.3. Query Resources

Table 8. Query Types

Path Template	Query Type	Description
/collections/{collectionId}/position	Position	Return data for the requested point location
/collections/{collectionId}/radius	Radius	Return data within a given radius of a position
/collections/{collectionId}/area	Area	Return data for the requested area
/collections/{collectionId}/cube	Cube	Return data for a spatial cube
/collections/{collectionId}/trajectory	Trajectory	Return data along a defined trajectory
/collections/{collectionId}/corridor	Corridor	Return data within a spatial corridor
/collections/{collectionId}/items	Item	Items associated with the `{collectionId}` collection.
/collections/{collectionId}/locations	Locations	Location identifers associated with the `{collectionId}` collection.
/collections/{collectionId}/instances	Instances	List the available instances of the collection

9.3.1. Shared query parameters

Query parameters are used in URLs to define the resources which are returned on a GET request. The following are defined as standard shared parameters for use.

Parameter coords

Requirement 3: /req/edr/coords-definition Parameter coords definition

Requirement 4: /req/edr/coords-response Parameter coords response

Parameter datetime

Requirement 5: /req/core/rc-datetime-parameter Datetime parameter

The `datetime` parameter is defined in API-Common. The following information is provided here as a convenience.

"Intersects" means that the time (instant or duration) specified in the parameter `datetime` includes a timestamp that is part of the temporal geometry of the resource (again, a time instant or duration). Time durations include the start and end times.

Example 2. A datetime

February 12, 2018, 23:20:52 GMT:

`datetime=2018-02-12T23%3A20%3A52Z`

For resources with a temporal property that is a timestamp (like `lastUpdate`), a `datetime` value would match all resources where the temporal property is identical.

For resources with a temporal property that is a date or a time interval, a `datetime` value would match all resources where the timestamp is on that day or within the time interval.

Example 3. Intervals

February 12, 2018, 00:00:00 GMT to March 18, 2018, 12:31:12 GMT:

`datetime=2018-02-12T00%3A00%3A00Z%2F2018-03-18T12%3A31%3A12Z`

February 12, 2018, 00:00:00 UTC or later:

`datetime=2018-02-12T00%3A00%3A00Z%2F..`

March 18, 2018, 12:31:12 UTC or earlier:

`datetime=..%2F2018-03-18T12%3A31%3A12Z`

A template for the definition of the parameter in YAML according to OpenAPI 3.0 is available at datetime.yaml.

Parameter parametername

Requirement 6: /req/edr/parameters-definition Parameter parametername definition

Requirement 7: /req/edr/parameters-response Parameter parametername response

Example 4. A single parameter

Only return values for the Maximum_temperature

```
parametername=Maximum_temperature
```

Example 5. Return multiple parameters

Values for the Maximum_temperature, Minimum_temperature and Total_precipitation

```
parametername=Maximum_temperature,Minimum_temperature,Total_precipitation
```

If a requested parameter doesn't exist in the collection the data for those parameters that do exist should be returned. If none of the requested parameters exist in the collection a `400` message SHOULD be returned.

Parameter crs

Requirement 8: /req/edr/outputCRS-definition Parameter crs definition

Requirement 9: /req/edr/outputCRS-response Parameter crs response

The value of the `crs` query parameter will be one of the name values described in the collection metadata for supported coordinate reference system transformations.

Parameter f

Requirement 10: /req/edr/outputFormat-definition Parameter `f` definition

Requirement 11: /req/edr/outputFormat-response Parameter `f` response

Example 6. Return data as coverageJSON

```
f=coverageJSON
```

If not specified, the query will return data in the native format of the collection. If the requested format system does not match an entry in the defined list of valid output formats for the collection, a `400` message SHOULD be returned.

9.3.2. Position query

The Position query returns data for the requested coordinate. Logic for identifying the best match for the coordinate will depend on the collection . The filter constraints are defined by the following query parameters:

Parameter coords

Requirement 3: /req/edr/coords-definition Parameter coords definition

Requirement 4: /req/edr/coords-response Parameter coords response

location(s) to return data for. The coordinates are defined by a Well Known Text (wkt) string. To retrieve a single location:

```
POINT(x y)
```

And for a list of locations:

```
MULTIPOINT((x y),(x1 y1),(x2 y2),(x3 y3))
```

And for a list of locations at defined heights:

```
MULTIPOINT((x y),(x1 y1),(x2 y2),(x3 y3))
```

See http://portal.opengeospatial.org/files/?artifact_id=25355 and https://en.wikipedia.org/wiki/Well-known_text_representation_of_geometry .

The coordinate values will depend on the CRS parameter. If this is not defined, the values will be assumed to WGS84 values (i.e x=longitude and y=latitude).

Example 7. Single location

Retrieve data for Greenwich, London:

```
coords=POINT(0 51.48)
```

Example 8. Multiple locations

Retrieve data for a list of locations: 38.9N 77W, 48.85N 2.35E, 39.92N 116.38E, 35.29S 149.1E, 51.5N 0.1W:

```
coords=MULTIPOINT((38.9 -77),(48.85 2.35),(39.92 116.38),(-35.29 149.1),(51.5 -0.1))
```

Parameter z

Requirement 12: /req/edr/z-definition Parameter z definition

Requirement 13: /req/edr/z-response Parameter z response

Define the vertical level to return data from i.e. z=level

If data at all available levels is required, z can be defined as ALL i.e. z=ALL

Example 9. A single vertical level

For example, if the 850hPa pressure level is being queried:

```
z=850
```

Example 10. Return data at all a levels defined by a list of vertical levels

Request data at levels 1000hPa, 900hPa, 850hPa, and 700hPa:

```
z=1000,900,850,700
```

Example 11. Return data for all levels between and including 2 defined levels

Request data for all levels between 2m and 100m:

```
z=2/100
```

Example 12. Return data at all available vertical levels

```
z=ALL
```

`When not specified the API MUST return data from all available levels`

Parameter datetime

Parameter parametername

Parameter crs

Parameter f

9.3.3. Radius query

The Radius query returns data within the defined radius of the requested coordinate. The filter constraints are defined by the following query parameters:

Parameter coords

Requirement 3: /req/edr/coords-definition Parameter coords definition

Requirement 4: /req/edr/coords-response Parameter coords response

location(s) to return data for, the coordinates are defined by a Well Known Text (wkt) string. To retrieve data for a single location :

POINT(x y)

A point at height `z` POINT(x y z)

And for a list of locations

MULTIPOINT((x y),(x1 y1),(x2 y2),(x3 y3))

And for a list of locations at defined heights

MULTIPOINT((x y z),(x1 y1 z1),(x2 y2 z2),(x3 y3 z3))

see http://portal.opengeospatial.org/files/?artifact_id=25355 and https://en.wikipedia.org/wiki/Well-known_text_representation_of_geometry

The coordinate values will depend on the CRS parameter. If this is not defined the values will be assumed to be WGS84 values (i.e x=longitude and y=latitude)

Example 13. Single location

retrieve data for Greenwich, London

```
coords=POINT(0 51.48)
```

Parameter within

Requirement 14: /req/edr/within-definition Parameter within definition

Requirement 15: /req/edr/within-response Parameter within response

Parameter withinunits

Requirement 16: /req/edr/withinUnits-definition Parameter withinUnits definition

Requirement 17: /req/edr/within-response Parameter withinUnits response

Parameter z

Define the vertical level to return data from i.e. z=level

if data at all available levels is required z can be defined as ALL i.e. z=ALL

Example 14. A single vertical level

For example if the 80m level is being queried

z=80

Example 15. Return data at all available vertical levels

z=ALL

`When not specified the API MUST return data from all available levels`

Parameter datetime

Parameter parametername

Parameter crs

Parameter f

9.3.4. Area query

The Area query returns data within the polygon defined by the `coords` parameter, the results are further filtered by the constraints defined by the following query parameters:

Parameter coords

Requirement 3: /req/edr/coords-definition Parameter coords definition

Requirement 4: /req/edr/coords-response Parameter coords response

Only data that has a geometry that intersects the area defined by the polygon are selected.

The polygon is defined using a Well Known Text string following

```
coords=POLYGON((x y,x1 y1,x2 y2,⋯,xn yn x y))
```

which are values in the coordinate system defined by the crs query parameter (if crs is not defined the values will be assumed to be WGS84 longitude/latitude coordinates).

For instance a polygon that roughly describes an area that contains South West England in WGS84 would look like:

```
coords=POLYGON((-6.1 50.3,-4.35 51.4,-2.6 51.6,-2.8 50.6,-5.3 49.9,-6.1,50.3))
```

see http://portal.opengeospatial.org/files/?artifact_id=25355 and https://en.wikipedia.org/wiki/Well-known_text_representation_of_geometry

```
The coords parameter will only support 2D POLYGON
```

Example 16. A polygon covering the UK

An area covering the UK in WGS84 (from 15°W to 5°E and from 60.95°S to 48.8°S)

```
coords=POLYGON((-15 48.8,-15 60.95,5 60.85,5 48.8,-15 48.8))
```

Example 17. Multiple areas

Selecting data for two different regions

```
coords=MULTIPOLYGON((-15 48.8,-15 60.95,5 60.85,5 48.8,-15 48.8),(-6.1 50.3,-4.35 51.4,-2.6 51.6,-2.8 50.6,-5.3 49.9,-6.1,50.3))
```

Parameter z

Requirement 12: /req/edr/z-definition Parameter z definition

Requirement 13: /req/edr/z-response Parameter z response

Define the vertical level to return data from i.e. z=level

if data at all available levels is required z can be defined as ALL i.e. z=ALL

Example 18. A single vertical level

For example if the 850hPa pressure level is being queried

z=850

Example 19. Return data at all a levels defined by a list of vertical levels

Request data at levels 1000hPa, 900hPa, 850hPa, and 700hPa

z=1000,900,850,700

Example 20. Return data for all levels between and including 2 defined levels

Request data for all levels between 2m and 100m

z=2/100

Example 21. Return data at all available vertical levels

z=ALL

`When not specified the API MUST return data from all available levels`

Parameter datetime

Parameter parametername

Parameter crs

Parameter f

9.3.5. Cube query

The `Cube` query returns a data cube defined by the `coords`, `minz` and `maxz` parameters, *Only WKT POLYGONS that define rectangles are valid inputs to the `coords` parameter. The results are further filtered by the constraints defined by the following query parameters:

Parameter coords

Requirement 3: /req/edr/coords-definition Parameter coords definition

Requirement 4: /req/edr/coords-response Parameter coords response

Only data that has a geometry that intersects the area defined by the cube are selected.

The cube's X Y coordinates are defined using Rectangular Polygon as Well Known Text

coords=POLYGON((x y,x1 y1,x2 y2, x3 y3, x y))

which are values in the coordinate system defined by the crs query parameter if crs is not defined the values will be assumed to be WGS84 longitude/latitude coordinates and heights will be assumed to be in metres above mean sea level

For instance a cube that roughly describes an area that contains South West England in WGS84 would look like

Example 22. A cube covering the South West of the UK

```
coords=POLYGON((-6.0 50.0,-4.35 50.0,-4.35 52.0,,-6.0 52.0,-6.0 50.0))
```

If the WKT does not define a Rectangle the service will generate a `400` error message

see http://portal.opengeospatial.org/files/?artifact_id=25355 and https://en.wikipedia.org/wiki/Well-known_text_representation_of_geometry

Parameter minz

Requirement 18: /req/edr/minz-definition Parameter minx definition

Requirement 19: /req/edr/minz-response Parameter minz response

Example 23. Define the bottom of the cube

denotes the minimum level to return data for

`minz=850`

would retrieve data for the 850 level and for data above the 850 level, if a `maxz` is not specified a `400` error should be thrown

Parameter maxz

Requirement 20: /req/edr/maxz-definition Parameter maxz definition

Requirement 21: /req/edr/maxz-response Parameter maxz response

Example 24. Define the top of the cube

denotes the maximum level to return data for

`maxz=300`

would retrieve data for the 300 level and for data below the 850 level, if a `minz` is not specified a `400` error should be thrown

Parameter resolutionx

Requirement 22: /req/edr/resolutionx-definition Parameter resolutionx definition

Requirement 23: /req/edr/resolutionx-response Parameter resolutionx response

If not specified the query will not interpolate the data along the x-axis of the cube and only return values for the defined `coords` parameter values.

Example 25. Define the resolution of data on the x-axis

denotes the number of intervals to retrieve data for across the defined x-axis of the cube

`resolutionx=10`

would retrieve 10 values along the width from the minimum x coordinate to maximum x coordinate (i.e. a value at both the minimum x and maximum x coordinates and 8 values between).

Parameter resolutiony

Requirement 24: /req/edr/resolutiony-definition Parameter resolutiony definition

Requirement 25: /req/edr/resolutiony-response Parameter resolutiony response

If not specified the query will not interpolate the data along the y-axis of the cube and only return values for the defined `coords` parameter values.

Example 26. Define resolution of data on the y-axis

denotes the number of intervals to retrieve data for across the defined y-axis of the cube

`resolutiony=10`

would retrieve 10 values along the width from the minimum y coordinate to maximum y coordinate (i.e. a value at both the minimum y and maximum y coordinates and 8 values between).

Parameter resolutionz

Requirement 26: /req/edr/resolutionz-definition Parameter resolutionz definition

Requirement 27: /req/edr/resolutionz-response Parameter resolutionz response

If not specified the query will not interpolate the data along the z-axis of the cube and only return values for the defined `coords` parameter values.

Example 27. Define resolution of data on the z-axis

denotes the number of intervals to retrieve data for the defined z-axis of the cube

`resolutionz=10`

would retrieve 10 values along the z-axis from the minimum z coordinate to maximum z coordinate (i.e. a value at both the minimum z and maximum z coordinates and 8 values between).

Parameter datetime

Parameter parametername

Parameter crs

Parameter f

9.3.6. Trajectory query

The Trajectory query returns data along the path defined by the `coords` parameter. **The logic to match the data for the requested path will depend on and be defined by the collection**. The results are further filtered by the constraints defined by the following query parameters:

Parameter coords

Requirement 3: /req/edr/coords-definition Parameter coords definition

Requirement 4: /req/edr/coords-response Parameter coords response

"Intersects" means that the geospatial shape specified by the parameter `coords`, includes a coordinate that is part of the (spatial) geometry of the resource. This includes the boundaries of the geometries.

The trajectory query supports the Linestring Well Known Text (WKT) geometry type, the trajectory query SHOULD support 2D, 3D and 4D queries allowing the definition of a vertical level value (z) and a time value (as an epoch time) therefore coordinates for geometries may be 2D (x, y), 3D (x, y, z) or 4D (x, y, z, t).

A 2D trajectory, on the surface of earth with no time or height dimensions: coords=LINESTRING(51.14 -2.98, 51.36 -2.87, 51.03 -3.15, 50.74 -3.48, 50.9 -3.36)

A 2D trajectory, on the surface of earth all at the same time and no height dimension, time value defined in ISO8601 format by the `datetime` query parameter : coords=LINESTRING(51.14 -2.98, 51.36 -2.87, 51.03 -3.15, 50.74 -3.48, 50.9 -3.36)&datetime=2018-02-12T23:00:00Z

A 2D trajectory, on the surface of earth with no time value but at a fixed height level, height defined in the collection height units by the `z` query parameter : coords=LINESTRING(51.14 -2.98, 51.36 -2.87, 51.03 -3.15, 50.74 -3.48, 50.9 -3.36)&z=850

A 2D trajectory, on the surface of earth all at the same time and at a fixed height level, time value defined in ISO8601 format by the `datetime` query parameter and height defined in the collection height units by the `z` query parameter : coords=LINESTRING(51.14 -2.98, 51.36 -2.87, 51.03 -3.15, 50.74 -3.48, 50.9 -3.36)&datetime=2018-02-12T23:00:00Z&z=850

A 3D trajectory, on the surface of the earth but over a time range with no height values: coords=LINESTRINGM(51.14 -2.98 1560507000, 51.36 -2.87 1560507600, 51.03 -3.15 1560508200, 50.74 -3.48 1560508500, 50.9 -3.36 1560510240)

A 3D trajectory, on the surface of the earth but over a time range with a fixed height value, height defined in the collection height units by the `z` query parameter : coords=LINESTRINGM(51.14 -2.98 1560507000, 51.36 -2.87 1560507600, 51.03 -3.15 1560508200, 50.74 -3.48 1560508500, 50.9 -3.36 1560510240)&z=200

A 3D trajectory, through a 3D volume with height or depth, but no defined time: coords=LINESTRINGZ(51.14 -2.98 0.1, 51.36 -2.87 0.2, 51.03 -3.15 0.3, 50.74 -3.48 0.4, 50.9 -3.36 0.5)

A 3D trajectory, through a 3D volume with height or depth, but at a fixed time value defined in ISO8601 format by the `datetime` query parameter: coords=LINESTRINGZ(51.14 -2.98 0.1, 51.36 -2.87

0.2, 51.03 -3.15 0.3, 50.74 -3.48 0.4, 50.9 -3.36 0.5)&datetime=2018-02-12T23:00:00Z

A 4D trajectory, through a 3D volume and over a time range: coords=LINESTRINGZM(51.14 -2.98 0.1 1560507000,51.36 -2.87 0.2 1560507600, 51.03 -3.15 0.3 1560508200, 50.74 -3.48 0.4 1560508500, 50.9 -3.36 0.5 1560510240)

If the coords specify a 4D trajectory i.e. coords=LINESTRINGZM(...) an error MUST be thrown by the server if the client application defines either the `z` or `datetime` query parameters

where `Z` in `LINESTRINGZ` and `LINESTRINGZM` refers to the height value. `If the specified CRS does not define the height units, the heights units will default to metres above mean sea level`

and the `M` in `LINESTRINGM` and `LINESTRINGZM` refers to the number of seconds that have elapsed since the Unix epoch, that is the time 00:00:00 UTC on 1 January 1970. See https://en.wikipedia.org/wiki/Unix_time

Example 28. A basic surface route

From Bristol to Exeter

```
coords=LINESTRING(51.14 -2.98,51.36 -2.87,51.03 -3.15,50.74 -3.48,50.9 -3.36)
```

Example 29. A basic surface route with defined datetime intervals

From Bristol to Exeter

```
coords=LINESTRING(51.14  -2.98  0  1560507000,51.36  -2.87  0  1560507600,51.03  -3.15  0  1560508200,50.74 -3.48 0 1560508500,50.9 -3.36 0 1560510240)
```

Parameter z

Used when the entire trajectory occurs at the same vertical height the z query parameter is used to define the height

Example 30. A single vertical level

for instance if the entire route is at the 850hPa pressure level

z=850

Parameter datetime

Parameter parametername

Parameter crs

Parameter f

9.3.7. Corridor query

The Corridor query returns data along and around the path defined by the `coords` parameter. **The logic to match the data for the requested path will depend on and be defined by the collection**. The results are further filtered by the constraints defined by the following query parameters:

Parameter coords

Requirement 3: /req/edr/coords-definition Parameter coords definition

Requirement 4: /req/edr/coords-response Parameter coords response

"Intersects" means that the geospatial shape specified by the parameter `coords`, includes a coordinate that is part of the (spatial) geometry of the resource. This includes the boundaries of the geometries.

The corridor query supports the Linestring Well Known Text (WKT) geometry type, the corridor query SHOULD support 2D, 3D and 4D queries allowing the definition of a vertical level value (z) and a time value (as an epoch time) therefore coordinates for geometries may be 2D (x, y), 3D (x, y, z) or 4D (x, y, z, t). The Linestring described by the coords parameter defines the center point of the corridor with the corridorHeight and corridorWidth query parameters defining the depth and breadth of the corridor.

A 2D corridor, on the surface of earth with no time or height dimensions: coords=LINESTRING(51.14 -2.98, 51.36 -2.87, 51.03 -3.15, 50.74 -3.48, 50.9 -3.36)

A 2D corridor, on the surface of earth all at the same time and no height dimension, time value defined in ISO8601 format by the `datetime` query parameter : coords=LINESTRING(51.14 -2.98, 51.36 -2.87, 51.03 -3.15, 50.74 -3.48, 50.9 -3.36)&datetime=2018-02-12T23:00:00Z

A 2D corridor, on the surface of earth with no time value but at a fixed height level, height defined in the collection height units by the `z` query parameter : coords=LINESTRING(51.14 -2.98, 51.36 -2.87, 51.03 -3.15, 50.74 -3.48, 50.9 -3.36)&z=850

A 2D corridor, on the surface of earth all at a the same time and at a fixed height level, time value defined in ISO8601 format by the `datetime` query parameter and height defined in the collection height units by the `z` query parameter : coords=LINESTRING(51.14 -2.98, 51.36 -2.87, 51.03 -3.15, 50.74 -3.48, 50.9 -3.36)&datetime=2018-02-12T23:00:00Z&z=850

A 3D corridor, on the surface of the earth but over a time range with no height values: coords=LINESTRINGM(51.14 -2.98 1560507000, 51.36 -2.87 1560507600, 51.03 -3.15 1560508200, 50.74 -3.48 1560508500, 50.9 -3.36 1560510240)

A 3D corridor, on the surface of the earth but over a time range with a fixed height value, height defined in the collection height units by the `z` query parameter : coords=LINESTRINGM(51.14 -2.98 1560507000, 51.36 -2.87 1560507600, 51.03 -3.15 1560508200, 50.74 -3.48 1560508500, 50.9 -3.36 1560510240)&z=200

A 3D corridor, through a 3D volume with height or depth, but no defined time: coords=LINESTRINGZ(51.14 -2.98 0.1, 51.36 -2.87 0.2, 51.03 -3.15 0.3, 50.74 -3.48 0.4, 50.9 -3.36 0.5)

A 3D corridor, through a 3D volume with height or depth, but at a fixed time, time value defined in ISO8601 format by the `datetime` query parameter: coords=LINESTRINGZ(51.14 -2.98 0.1, 51.36 -2.87 0.2, 51.03 -3.15 0.3, 50.74 -3.48 0.4, 50.9 -3.36 0.5)&datetime=2018-02-12T23:00:00Z

A 4D corridor, through a 3D volume but over a time range: coords=LINESTRINGZM(51.14 -2.98 0.1 1560507000,51.36 -2.87 0.2 1560507600, 51.03 -3.15 0.3 1560508200, 50.74 -3.48 0.4 1560508500, 50.9 -3.36 0.5 1560510240)

If the coords specify a 4D corridor i.e. coords=LINESTRINGZM(...) an error MUST be thrown by the server if the client application defines either the `z` or `datetime` query parameters

where **Z** in `LINESTRINGZ` and `LINESTRINGZM` refers to the height value. `If the specified CRS does not define the height units, the heights units will default to metres above mean sea level`

and the **M** in `LINESTRINGM` and `LINESTRINGZM` refers to the number of seconds that have elapsed since the Unix epoch, that is the time 00:00:00 UTC on 1 January 1970. See https://en.wikipedia.org/wiki/Unix_time

Example 31. A basic surface route

From Bristol to Exeter

```
coords=LINESTRING(51.14 -2.98,51.36 -2.87,51.03 -3.15,50.74 -3.48,50.9 -3.36)
```

Example 32. A basic surface route with defined datetime intervals

From Bristol to Exeter

```
coords=LINESTRING(51.14  -2.98  0  1560507000,51.36  -2.87  0  1560507600,51.03  -3.15  0  1560508200,50.74 -3.48 0 1560508500,50.9 -3.36 0 1560510240)
```

Parameter z

Used when the entire corridor occurs at the same vertical height the z query parameter is used to define the height

Example 33. A single vertical level

for instance if the entire route is at the 850hPa pressure level

z=850

Parameter datetime

resolutionx

Example 34. A interpolate corridor values along the width

Return 10 values across the defined corridor width `resolutionx=10`

Example 35. A get values across the width of the corridor at stored resolution

Return values at the stored resolution `resolutionx=0`

resolutionz

Example 36. A interpolate corridor values over the height

Return 8 values over the defined corridor height `resolutionz=8`

Example 37. A get values over the height of the corridor at stored resolution

Return values at the stored resolution `resolutionz=0`

corridorHeight

corridorWidth

Parameter parametername

Parameter crs

Parameter f

9.3.8. Items query

The EDR Items query is an OGC API - Features endpoint that may be used to catalog pre-existing EDR sampling features. The pre-existence of an EDR sampling feature may be because of the existence of a monitoring location, because a particular query has been cached for later use, or for an array of application-specific use cases (e.g. a catalog of spatio-temporal domains of anomalies in a dataset). A GeoJSON-compatible JSON-Schema has been specified to document an EDR query endpoint and valid query parameters including time range, parameters, and spatial characteristics.

Recommendation 1	**/rec/core/edr-geojson**
A	If a collection using other EDR queries uses the `items` query, implementations SHOULD consider support for the EDR GeoJSON Schema as an encoding.

Parameter itemID

If an itemID is not specified, the query will return a list of the available itemID's. This behavior is specified in in OGC API - Features. all other parameters for use with the Items query are defined by OGC API - Features.

Example 38. List available items

`/collections/{collectionId}/items`

e.g. return query parameters to retrieve trapical storms using OGC API - Features and the EDR FeatureCollection GeoJSON schema. Each item would include an area query end point, a time range, a list of available parameters, and a representative geojson geometry.

`/collections/tropical_storms/items`

e.g. return query parameters to retrieve monitoring data from a collection end point, a time range, a list of available parameters and a representative geojson geometry.

`/collections/stream_gages/items`

Example 39. itemID

`/collections/{collectionId}/items/{itemID}`

e.g. return information for the requested item with an id of KIAD_2020-05-19T00Z from the Metar collection. Returned data would include a location query end point, time range, a list of available parameters, and a representative geometry for the KIAD METAR station.

`/collections/metar/items/KIAD_2020-05-19T00Z`

e.g. return information for the requested item with an id of warning_12345 from the forecast collection. Returned data would include an area query end point, time range, a list of available parameters and a representative geometry for the warning_12345 warning area.

`/collections/forecast/items/warning_12345`

9.3.9. Locations query

Parameter locationID

With the locations query a location is defined by a unique identifier, this is a string value. It can be anything as long as it is unique for the required location, for instance a GeoHash `gbsvn` or a WMO station id like `03772`, what3words like `bolt.lime.metro` or place name like `Devon`. The metadata returned by the API must supply a geospatial definition for the identifier.

Example 40. get list of location id's

/collections/{collectionID}/locations/

return a list of location identifiers and relevant metadata for the metar collection

/collections/metar/locations/

Valid locationIDs can also be discovered via another mechanism such as the `items` query.

Example 41. locationID

/collections/{collectionID}/locations/{locationID}

return all available data for the metar collection for the requested location identifier, where the location is defined by the Heathrow METAR id

/collections/metar/locations/EGLL

Parameter datetime

Parameter parametername

Parameter crs

Parameter f

9.3.10. Instances query

It is not unusual in the scientific world for there to be multiple versions or instances of the same collection, where the same information is reprocessed or regenerated. Although they could be described as new collections the instance query type allows this data to be described as different views of the same collection.

Parameter instanceId

A unique identifier for the instance of the collection

/collections/{collectionId}/instance/{instanceId}

1. Return the Raw data instance metadata (instanceId = raw) for the Metar ((collectionId = metar) collection

```
/collections/metar/instance/raw
```

1. Return the Level 1 Quality controlled data instance (instanceId = qc_lvl_1) metadata for the Metar (collectionId = metar) collection

```
/collections/metar/instance/qc_lvl_1
```

Parameter queryType

The queryType options are exactly the same as those available to collections that don't have multiple instances and support the same query parameters and functionality. See the <query-resource-table>> for the mappings of the intial query types proposed for the EDR API.

/collections/{collectionId}/instance/{instanceId}/{queryType}

see the Query Resources section for details of the query parameters supported by the queryTypes.

1. A point query on an Raw data instance(instanceId = raw) for the Metar ((collectionId = metar) collection

```
/collections/metar/instance/raw/point
```

1. A trajectory query on an Raw data instance(instanceId = raw) for the Metar ((collectionId = metar) collection

```
/collections/metar/instance/raw/trajectory
```

Chapter 10. General Requirements

The following general requirements and recomendations apply to all OGC APIs.

10.1. HTTP 1.1

The standards used for Web APIs are built on the HTTP protocol. Therefore, conformance with HTTP or a closely related protocol is required.

Requirement 32: /req/core/http HTTP

10.2. HTTP Status Codes

Table 9 lists the main HTTP status codes that clients should be prepared to receive. This includes support for specific security schemes or URI redirection. In addition, other error situations may occur in the transport layer outside of the server.

Table 9. Typical HTTP status codes

Status code	Description
`200`	A successful request.
`202`	A successful request, but the response is still being generated. The response will include a `Retry-After` header field giving a recommendation in seconds for the client to retry.
`304`	An entity tag was provided in the request and the resource has not been changed since the previous request.
`308`	The server cannot process the data through a synchronous request. The response includes a `Location` header field which contains the URI of the location the result will be available at once the query is complete Asynchronous queries.
`400`	The server cannot or will not process the request due to an apparent client error. For example, a query parameter had an incorrect value.
`401`	The request requires user authentication. The response includes a `WWW-Authenticate` header field containing a challenge applicable to the requested resource.
`403`	The server understood the request, but is refusing to fulfill it. While status code `401` indicates missing or bad authentication, status code `403` indicates that authentication is not the issue, but the client is not authorised to perform the requested operation on the resource.
`404`	The requested resource does not exist on the server. For example, a path parameter had an incorrect value.
`405`	The request method is not supported. For example, a POST request was submitted, but the resource only supports GET requests.

Status code	Description
406	Content negotiation failed. For example, the Accept header submitted in the request did not support any of the media types supported by the server for the requested resource.
413	Request entity too large. For example the query would involve returning more data than the server is capable of processing, the implementation should return a message explaining the query limits imposed by the server implementation.
500	An internal error occurred in the server.

More specific guidance is provided for each resource, where applicable.

Permission 1	/per/core/additional-status-codes
A	Servers MAY support other capabilities of the HTTP protocol and, therefore, MAY return other status codes than those listed in Table 9, too.

10.3. Web Caching

Entity tags are a mechanism for web cache validation and for supporting conditional requests to reduce network traffic. Entity tags are specified by HTTP/1.1 (RFC 2616).

Recommendation 2	/rec/core/etag
A	The service SHOULD support entity tags and the associated headers as specified by HTTP/1.1.

10.4. Support for Cross-Origin Requests

Access to data from a HTML page is by default prohibited for security reasons, if the data is located on another host than the webpage ("same-origin policy"). A typical example is a web-application accessing feature data from multiple distributed datasets.

Recommendation 3	/rec/core/cross-origin
A	If the server is intended to be accessed from the browser, cross-origin requests SHOULD be supported. Note that support can also be added in a proxy layer on top of the server.

Two common mechanisms to support cross-origin requests are:

- Cross-origin resource sharing (CORS)
- JSONP (JSON with padding)

10.5. Asynchronous queries

It will not always be possible to respond to queries synchronously. This standard does not specify how to handle any asynchrony. Different services may propose different best practices.

For example, if the query requires handling asynchronously, one option, but there are others, is that the system could respond with a HTTP code of `308` and include a `Location` response header field with the URI of the location of the data once the query has completed. If the user queries the URI of the product of the query before the data is available that response should respond with a HTTP code of `202` and include a `Retry-after` response header field with a suggested interval in seconds to retry the data retrieval.

10.6. Coordinate Reference Systems

As discussed in Chapter 9 of the W3C/OGC Spatial Data on the Web Best Practices document, how to express and share the location of resources in a consistent way is one of the most fundamental aspects of publishing geospatial data and it is important to be clear about the coordinate reference system that the coordinates use.

For the reasons discussed in the Best Practices, EDR APIs MUST always support WGS84 longitude and latitude as a coordinate reference system.

Requirement 33: /req/core/crs84 CRS84

The implementations compliant with the OGC API Common Part 1: Core are only required to support publishing geometries in coordinate reference system `http://www.opengis.net/def/crs/OGC/1.3/CRS84`.

10.7. Encodings

While the OGC API Common standard does not specify any mandatory encoding, the following encodings are recommended. See Clause 7 (Overview) for a discussion of this issue.

HTML encoding recomendation:

Recommendation 4	**/rec/core/html**
A	To support browsing a API with a web browser and to enable search engines to crawl and index the dataset, implementations SHOULD consider to support an HTML encoding.

GeoJSON encoding recommendation:

Recommendation 5	**/rec/core/geojson**

A	If the resource can be represented for the intended use in GeoJSON, implementations SHOULD consider to support GeoJSON as an encoding.

CoverageJSON encoding recommendation:

Recommendation 6	**/rec/core/covjson**
A	If the resource can be represented for the intended use in CoverageJSON, implementations SHOULD consider to support CoverageJSON as an encoding.

Requirement `/req/core/http` implies that the encoding of a response is determined using content negotiation as specified by the HTTP RFC.

The section Media Types includes guidance on media types for encodings that are specified in this document.

Note that any API that supports multiple encodings will have to support a mechanism to create encoding-specific URIs for resources in order to express links, for example, to alternative representations of the same resource. This document does not mandate any particular approach to how this is supported by the API.

As clients simply need to dereference the URI of the link, the implementation details and the mechanism how the encoding is included in the URI of the link are not important. Developers interested in the approach of a particular implementation, for example, to manipulate ("hack") in the browser address bar, can study the API definition.

NOTE

Two common approaches are:

- an additional path for each encoding of each resource (this can be expressed, for example, using format specific suffixes like `.html`);
- an additional query parameter (for example, `accept` or `f`) that overrides the Accept header of the HTTP request.

10.8. Link Headers

Recommendation 7	**/rec/core/link-header**
A	Links included in payload of responses SHOULD also be included as `Link` headers in the HTTP response according to RFC 8288, Clause 3.

B	This recommendation does not apply, if there are a large number of links included in a response or a link is not known when the HTTP headers of the response are created.

10.9. OpenAPI 3.0

10.9.1. Basic requirements

Requirements Class: OpenAPI 3.0

The OpenAPI 3.0 Requirements Class used in OGC API Common is applicable to the EDR API as well. So an implementation of EDR API which supports OpenAPI 3.0 as an API Description format must also comply with the OGC API-Common oas30 Conformance Class.

Requirement 34: /req/oas30/oas-common OpenAPI 3.0 API Common

Implementations must also advertise conformance with this Requirements Class.

Requirement 35 |/req/oas30/conformance OpenAPI 3.0 Conformance

Two example OpenAPI documents are included in Annex B.

10.9.2. Complete definition

Requirement 36: /req/oas30/completeness OpenAPI 3.0 Completeness

Note, for example, that APIs which are access-controlled (see Security), support web cache validation, CORS or that use HTTP redirection will make use of additional HTTP status codes beyond regular codes such as `200` for successful GET requests and `400`, `404` or `500` for error situations. See HTTP Status Codes.

Clients have to be prepared to receive responses not documented in the OpenAPI definition. For example, additional errors may occur in the transport layer outside of the server.

10.9.3. Exceptions

Requirement 37: /req/oas30/exceptions-codes OpenAPI 3.0 Exception codes

Example 42. An exception response object definition

```
description: An error occurred.
content:
  application/json:
    schema:
      $ref:
https://raw.githubusercontent.com/opengeospatial/OAPI/openapi/schemas/exception.ya
ml
  text/html:
    schema:
      type: string
```

10.10. Security

Requirement 38: /req/oas30/security OpenAPI 3.0 Security

The OpenAPI specification currently supports the following security schemes:

- HTTP authentication,
- an API key (either as a header or as a query parameter),
- OAuth2's common flows (implicit, password, application and access code) as defined in RFC6749, and
- OpenID Connect Discovery.

Annex A: Requirements Detail

A.1. Introduction

For clarity, the complete requirements test class descriptions are omitted in the body of this specification. This annex contains the complete requirements test classes.

A.2. Requirements Test Class Core

Requirements Class: OGC API-Environmental Data Retrieval Core

http://www.opengis.net/spec/ogcapi-edr-1/1.0/conf/core	
Target type	Web API
Dependency	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core
Dependency	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/collections

Requirement 1: /req/core/api-common OGC API-Common

The API implementation SHALL demonstrate conformance with the following Requirements Classes of the OGC API-Common version 1.0 Standard.	
A	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core
B	http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections

Requirement 2: /req/core/conformance Core conformance classes

The list of Conformance Classes advertised by the API SHALL include:	
A	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core
B	http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections
C	http://www.opengis.net/spec/ogcapi-edr-1/1.0/conf/core

Requirement 3: /req/edr/coords-definition Parameter coords definition

A	Each geometry based resource (Position, Radius, Area, Cube , Trajectory, Corridor) collection operation SHALL support a parameter `coords` with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: coords`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

B	The `coords` string value will be a Well Known Text of representation geometry as defined in Simple Feature Access - Part 1: Common Architecture. The representation type will depend on the queryType of the API

Requirement 4: /req/edr/coords-response Parameter coords response

A	Only those resources that have a spatial geometry that intersects the area defined by the `coords` parameter SHALL be part of the result set.
B	The coordinates SHALL consist of a Well Known Text (WKT) geometry string

C	The coordinate reference system of the values SHALL be interpreted as WGS84 longitude/latitude
	
	`WKT: GEOGCS["WGS 84", DATUM["WGS_1984", SPHEROID["WGS 84", 6378137, 298.257223563, AUTHORITY["EPSG", "7030"]], AUTHORITY["EPSG", "6326"]], PRIMEM["Greenwich", 0 , AUTHORITY["EPSG", "8901"]], UNIT["degree", 0.01745329251994328, AUTHORITY["EPSG", "9122"]], AUTHORITY["EPSG", "4326"]]`
	
	`EPSG: http://www.opengis.net/def/crs/OGC/1.3/CRS84`
	
	unless a different coordinate reference system is specified in a parameter `crs`.

Requirement 5: /req/core/rc-datetime-parameter Datetime parameter

A	An EDR API SHALL support the Date-Time (datetime) parameter for `/collections` and `/collections/{collectionid}` requests.
	
	`name: datetime`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

B	Requests which include the Date-Time parameter SHALL comply with API-Common requirement `/req/core/rc-time-definition`.
C	Responses to Date-Time requests SHALL comply with API-Common requirement `/req/core/rc-time-response`.

Requirement 6: /req/edr/parameters-definition Parameter parametername definition

A	Each resource collection operation SHALL support a parameter `parametername`with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: parametername`
	`in: query`
	`required: true`
	`explode: false`
	`schema:`
	`    minItems: 1`
	`    type: array`
	`    items:`
	`        type: string`

Requirement 7: /req/edr/parameters-response Parameter parametername response

A	If the `parametername` parameter is provided, only those parameters named SHALL be returned. If the `parametername` parameter is not specified all parameters in the collection SHALL be returned.

B	The `parametername` parameter SHALL consist of a comma delimited string value based on an enumerated list of options listed in the collections metadata

Requirement 8: /req/edr/outputCRS-definition Parameter crs definition

A	Each resource collection operation SHALL support a parameter `crs` with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: crs`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 9: /req/edr/outputCRS-response Parameter crs response

A	If the `crs` parameter is provided, the returned information should be reprojected (if required) to the defined coordinate system. If the `crs` parameter is not specifed the data will be returned in its native projection.
B	The `crs` parameter SHALL consist of an identifier selected from the enumerated list of valid values supplied in the collections metadata.
C	if an unsupported `crs` value is requested a `400` error message SHOULD be returned .

Requirement 10: /req/edr/outputFormat-definition Parameter `f` definition

A	Each resource collection operation SHALL support a parameter `f` with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: f`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 11: /req/edr/outputFormat-response Parameter `f` response

A	If the `f` parameter is provided, the returned information should be transformed to the defined data format.
B	The `f` parameter SHALL consist of an string value based on an enumerated list of available options provided in the collections metadata.
C	If an unsupported `f` value is requested a `400' error message should be returned. =======

Requirement 12: /req/edr/z-definition Parameter z definition

A	Each resource collection operation MAY support a parameter `z`with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: z`
	`in: query`
	`required: true`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 13: /req/edr/z-response Parameter z response

A	If the `z` parameter is provided, only resources that have a vertical geometry that intersects the vertical information in the `z` parameter SHALL be part of the result set.

B	The `z` can be defined as a height range by specifying a min-level and max-level seperated by a forward slash "/"
C	A list of `z` can be defined be specifying a comma delimited list of values level1, level2, level3 etc ``` single-level = level interval-closed = min-level "/" max-level level-list = level1 "," level2 "," level3 ```

Requirement 14: /req/edr/within-definition Parameter within definition

A	Each resource collection operation MAY support a parameter `within`with the following characteristics (using an OpenAPI Specification 3.0 fragment):
B	If the instance metadata does not provide `withinUnits` values the API SHALL NOT support `within` queries: ``` name: within in: query required: false schema: type: string style: form explode: false ```

Requirement 15: /req/edr/within-response Parameter within response

A	If the `within` parameter is provided, all selected information within the specified radius SHALL be part of the result set.
B	If a `withinunits` parameter is not provided, a `400` error WILL be returned.

Requirement 16: /req/edr/withinUnits-definition Parameter withinUnits definition

A	Each resource collection operation MAY support a parameter `withinunits`with the following characteristics (using an OpenAPI Specification 3.0 fragment):

B	A withinunits value MUST be one of the values defined in the instance metadata:
	
	`name: withinunits`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 17: /req/edr/within-response Parameter withinUnits response

A	**The `withinunits` parameter defines the distance units of the `within` query parameter value .**

Requirement 18: /req/edr/minz-definition Parameter minx definition

A	Each resource collection operation MUST support a parameter `minz`with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: minz`
	`in: query`
	`required: true`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 19: /req/edr/minz-response Parameter minz response

A	Only resources with a vertical geometry coordinate that is greater than or includes the vertical information in the `minz` parameter SHALL be part of the result set. `minz  = level`

Requirement 20: /req/edr/maxz-definition Parameter maxz definition

A	Each resource collection operation MUST support a parameter `maxz`with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: maxz`
	`in: query`
	`required: true`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 21: /req/edr/maxz-response Parameter maxz response

A	Only resources with a vertical geometry coordinate that is less than or includes the vertical information in the `maxz` parameter SHALL be part of the result set. `maxz  = level`

Requirement 22: /req/edr/resolutionx-definition Parameter resolutionx definition

A	Each resource collection operation MAY support a parameter `resolutionx` with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: resolutionx`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 23: /req/edr/resolutionx-response Parameter resolutionx response

A	If the `resolutionx` parameter is provided, it denotes the number of positions to retrieve data for, across the width of the corridor path, including its minimum and maximum width coordinates. [interpolated corridor example] | *../../images/REQ_rc-resolutionx-*

B	A `resolutionx` value of 0 SHALL return all available data at the stored resolution between (and including) the minimum and maximum coordinates of the defined corridor. [native resolution corridor example] | *../../images/REQ_rc-*

C	If `resolutionx` is not specified, the API SHOULD return all available data at a resolution determined by the server, including the minimum and maximum coordinates of the defined corridor.

```
resolutionx = number of intervals + 1
```

Requirement 24: /req/edr/resolutiony-definition Parameter resolutiony definition

A	Each resource collection operation MAY support a parameter `resolutiony` with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	`name: resolutiony`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 25: /req/edr/resolutiony-response Parameter resolutiony response

A	If the `resolutiony` parameter is provided, denotes the number of intervals to retrieve data for along the path between the minimum and maximum y coordinates
B	The total number of intervals includes the values for the minimum and maximum coordinates
C	A `resolutiony` value of 0 MUST return all available data at the native y resolution between the minimum and maximum coordinates
D	IF `resolutiony` is not specified data should be returned for just the locations specified in the requested coordinates (**ONLY IF** interpolation is supported by the API) `resolutiony = number of intervals`

Requirement 26: /req/edr/resolutionz-definition Parameter resolutionz definition

A	Each resource collection operation MAY support a parameter `resolutionz` with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: resolutionz`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 27: /req/edr/resolutionz-response Parameter resolutionz response

A	If the `resolutionz` parameter is provided, it denotes the number of positions to retrieve data for, over the depth of the corridor path including its minimum and maximum width coordinates. [interpolated corridor example] | *../../images/REQ_rc-resolutionz-*

B	A `resolutionz` value of 0 SHALL return all available data at the stored vertical resolution between (and including) the minimum and maximum coordinates of the defined corridor. [native resolution corridor example] | *../../images/REQ_rc-*

C	If `resolutionz` is not specified the API SHOULD return all available data at a resolution determined by the server, including the minimum and maximum coordinates of the defined corridor. `resolutionz = number of intervals + 1`

Requirement 28: /req/edr/corridorHeight-definition Parameter corridorHeight definition

A	Each resource collection operation SHALL support a parameter `corridorz`with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: corridorz`
	`in: query`
	`required: true`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 29: /req/edr/corridorHeight-response Parameter corridorHeight response

A	If the `corridorz` parameter is defined the result set SHALL contain values derieved based on the chosen interpolation algorithm at the number of specifed intervals.
B	The `corridorz` information is a the corridor height and the corridor units. `corridorz = height "/" units`
C	The height of corridor parameter is the total height of the required corridor.
D	The coordinates of the `coords` parameter define the centre point of the corridor.
E	If an unsupported units value is requested a `400` error should be returned.

Requirement 30: /req/edr/corridorWidth-definition Parameter corridorWidth definition

A	Each resource collection operation SHALL support a parameter `corridorwidth`with the following characteristics (using an OpenAPI Specification 3.0 fragment):
	
	`name: corridorwidth`
	`in: query`
	`required: true`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Requirement 31: /req/edr/corridorWidth-response Parameter corridorWidth response

A	The `corridorwidth` information is the total width of the required corridor.
B	The supported `corridorwidth` width units will be supplied by the query metatdata information. `corridorwidth = width "/" units`
C	If the width value is the total width of the corridor.
D	The coordinates of the `coords` parameter define the centre point of the corridor.
E	If an unsupported units value is requested a `400` error should be returned.

Requirement 32: /req/core/http HTTP

A	The API SHALL conform to HTTP 1.1.
B	If the API supports HTTPS, then the API SHALL also conform to HTTP over TLS.

Requirement 33: /req/core/crs84 CRS84

A	Unless the client explicitly requests a different coordinate reference system, all spatial geometries SHALL be in the CRS84 (WGS 84 longitude/latitude) coordinate reference system for geometries without height information and CRS84h (WGS 84 longitude/latitude plus ellipsoidal height) for geometries with height information.

A.3. Requirements Test Class OpenAPI 3.0

Requirements Class: OpenAPI 3.0

http://www.opengis.net/spec/ogcapi-edr-1/1.0/conf/oas30	
Target type	Web API
Dependency	Conformance Class "Core"
Dependency	OGC API-Common Standard 1.0
Dependency	OpenAPI Specification 3.0.2

Requirement 34: /req/oas30/oas-common OpenAPI 3.0 API Common

Extends	/req/core/api-common
A	The API SHALL demonstrate conformance with the following Requirements Class of the OGC API-Common version 1.0 Standard. http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/oas30.

Requirement 35 |/req/oas30/conformance OpenAPI 3.0 Conformance

The list of Conformance Classes advertised by the API SHALL include:	
A	http://www.opengis.net/spec/ogcapi-coverages-1/1.0/conf/oas30

Requirement 36: /req/oas30/completeness OpenAPI 3.0 Completeness

A	The OpenAPI definition SHALL specify for each operation all HTTP Status Codes and Response Objects that the API uses in responses.
B	This includes the successful execution of an operation as well as all error situations that originate from the server.

Requirement 37: /req/oas30/exceptions-codes OpenAPI 3.0 Exception codes

A	For error situations that originate from an API server, the API definition SHALL cover all applicable HTTP Status Codes.

Requirement 38: /req/oas30/security OpenAPI 3.0 Security

A	For cases, where the operations of the API are access-controlled, the security scheme(s) and requirements SHALL be documented in the OpenAPI definition.

Annex B: Abstract Test Suite (Normative)

B.1. Introduction

The Abstract Test Suite (ATS) is a compendium of test assertions applicable to implementations of the EDR API. An ATS provides a basis for developing an Executable Test Suite to verify that the implementation under test conforms to all the relevant functional specifications.

The abstract test cases (assertions) are organized into test groups that correspond to distinct conformance test classes defined in the EDR API specification.

OGC Web APIs are not a Web Services in the traditional sense. Rather, they define the behavior and content of a set of Resources exposed through a Web Application Programing Interface (Web API). Therefore, an API may expose resources in addition to those defined by the standard. A test engine must be able to traverse the API, identify and validate test points, and ignore resource paths which are not to be tested.

B.2. Conformance Class Core

Conformance Class	
http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core

B.2.1. General Tests

HTTP

Abstract Test 1	**/conf/core/http**
Test Purpose	Validate that the resource paths advertised through the API conform with HTTP 1.1 and, where approprate, TLS.
Requirement	/req/core/http
Test Method	1. All compliance tests shall be configured to use the HTTP 1.1 protocol exclusively. 2. For APIs which support HTTPS, all compliance tests shall be configured to use HTTP over TLS (RFC 2818) with their HTTP 1.1 protocol.

B.2.2. Landing Page {root}/

Abstract Test 2	**/conf/core/root-op**
Test Purpose	Validate that a landing page can be retrieved from the expected location.
Requirement	/req/core/root-op
Test Method	1. Issue an HTTP GET request to the URL {root}/ 2. Validate that a document was returned with a status code 200 3. Validate the contents of the returned document using test /conf/core/root-success.

Abstract Test 3	**/conf/core/root-success**
Test Purpose	Validate that the landing page complies with the require structure and contents.
Requirement	/req/core/root-success
Test Method	Validate the landing page for all supported media types using the resources and tests identified in Table 10 For formats that require manual inspection, perform the following: a. Validate that the landing page includes a "service-desc" and/or "service-doc" link to an API Definition b. Validate that the landing page includes a "conformance" link to the conformance class declaration c. Validate that the landing page includes a "data" link to the Feature contents.

The landing page may be retrieved in a number of different formats. The following table identifies the applicable schema document for each format and the test to be used to validate the landing page against that schema. All supported formats should be exercised.

Table 10. Schema and Tests for Landing Pages

Format	**Schema Document**	**Test ID**
HTML	landingPage.json	/conf/html/content
JSON	landingPage.json	/conf/geojson/content

B.2.3. API Definition Path {root}/api (link)

Abstract Test 4	**/conf/core/api-definition-op**
Test Purpose	Validate that the API Definition document can be retrieved from the expected location.
Requirement	/req/core/api-definition-op
Test Purpose	Validate that the API Definition document can be retrieved from the expected location.
Test Method	1. Construct a path for each API Definition link on the landing page 2. Issue a HTTP GET request on each path 3. Validate that a document was returned with a status code 200 4. Validate the contents of the returned document using test /conf/core/api-definition-success.

Abstract Test 5	**/conf/core/api-definition-success**
Test Purpose	Validate that the API Definition complies with the required structure and contents.
Requirement	/req/core/api-definition-success
Test Method	Validate the API Definition document against an appropriate schema document.

B.2.4. Conformance Path {root}/conformance

Abstract Test 6	**/conf/core/conformance-op**
Test Purpose	Validate that a Conformance Declaration can be retrieved from the expected location.
Requirement	/req/core/conformance-op

Test Method	1. Construct a path for each "conformance" link on the landing page as well as for the {root}/conformance path. 2. Issue an HTTP GET request on each path 3. Validate that a document was returned with a status code 200 4. Validate the contents of the returned document using test /conf/core/conformance-success.

Abstract Test 7	**/conf/core/conformance-success**
Test Purpose	Validate that the Conformance Declaration response complies with the required structure and contents.
Requirement	/req/core/conformance-success
Test Method	1. Validate the response document against OpenAPI 3.0 schema confClasses.yaml 2. Validate that the document includes the conformance class "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/core" 3. Validate that the document list all OGC API conformance classes that the API implements.

B.3. Conformance Class Collections

Conformance Class	
http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/collections	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/collections
Dependency	Conformance Class "OAPI Core"

B.3.1. General Tests

CRS 84

Abstract Test 8	**/conf/core/crs84**
Test Purpose	Validate that all spatial geometries provided through the API are in the CRS84 spatial reference system unless otherwise requested by the client.

Requirement	/req/core/crs84
Test Method	1. Do not specify a coordinate reference system in any request. All spatial data should be in the CRS84 reference system. 2. Validate retrieved spatial data using the CRS84 reference system.

B.3.2. Environmental Data Collections {root}/collections

Abstract Test 9	**/conf/collections/rc-md-op**
Test Purpose	Validate that information about the Collections can be retrieved from the expected location.
Requirement	/req/collections/rc-md-op
Test Method	1. Issue an HTTP GET request to the URL {root}/collections 2. Validate that a document was returned with a status code 200 3. Validate the contents of the returned document using test /conf/collections/rc-md-success.

Abstract Test 10	**/conf/rc-md-success**
Test Purpose	Validate that the Collections content complies with the required structure and contents.
Requirement	/req/collections/rc-md-success, /req/edr/rc-crs
Test Method	1. Validate that all response documents comply with /conf/collections/rc-collection-info-links 2. In case the response includes a "crs" property, validate that it includes a valid Well Known Text definition 3. Validate the collections content for all supported media types using the resources and tests identified in Table 11

The Collections content may be retrieved in a number of different formats. The following table identifies the applicable schema document for each format and the test to be used to validate the against that schema. All supported formats should be exercised.

Table 11. Schema and Tests for Collections content

Format	Schema Document	Test ID
HTML	collections.json	/conf/html/content
JSON	collections.json	/conf/geojson/content

B.3.3. Environmental Data Collection {root}/collections/{collectionId}

Abstract Test 11	/conf/collections/src-md-op
Test Purpose	Validate that the Collection content can be retrieved from the expected location.
Requirement	/req/collections/src-md-op
Test Method	For every Feature Collection described in the Collections content, issue an HTTP GET request to the URL `/collections/{collectionId}` where `{collectionId}` is the `id` property for the collection. . Validate that a Collection was returned with a status code 200 . Validate the contents of the returned document using test /conf/collections/src-md-success.

Abstract Test 12	/conf/collections/src-md-success
Test Purpose	Validate that the Collection content complies with the required structure and contents.
Requirement	/req/collections/src-md-success
Test Method	Verify that the content of the response is consistent with the content for this Resource Collection in the `/collections` response. That is, the values for `id`, `title`, `description` and `extent` are identical.

B.3.4. Second Tier Collections Tests

These tests are invoked by other tests.

Collection Extent

Abstract Test 13	/conf/core/rc-extent
Test Purpose	Validate that the `extent` property if it is present
Requirement	/req/core/rc-extent

Test Method	1. Verify that the `extent` provides bounding boxes that include all spatial geometries 2. Verify that if the `extent` provides time intervals that include all temporal geometries in this collection. 3. A temporal extent of `null` indicates an open time interval. 4. Verify that if the `extent` provides vertical intervals that include all vertical geometries in this collection. 5. A vertical extent of `null` indicates an open vertical interval.

Collection Queries

Abstract Test 14	**/conf/edr/rc-collection-info**
Test Purpose	Validate that each collection provided by the server is described in the Collections Metadata.
Requirement	/req/edr/rc-collection-info
Test Method	1. Verify that all collections listed in the `collections` array of the Collections Metadata exist. 2. Verify that each collection entry includes an identifier. 3. Verify that each collection entry includes links in accordance with /core/rc-collection-info-links. 4. Verify that if the collection entry includes an extent property, the property complies with /core/rc-extent 5. Verify that if the collection entry includes an crs property, the property complies with /edr/rc-crs 6. Verify that if the collection entry includes an parameters property, the property complies with /edr/rc-parameters 7. Validate each collection entry for all supported media types using the resources and tests identified in Table 12

The collection entries may be encoded in a number of different formats. The following table identifies the applicable schema document for each format and the test to be used to validate the against that schema. All supported formats should be exercised.

Table 12. Schema and Tests for Collection Entries

Format	Schema Document	Test ID
HTML	collectionInfo.json	/conf/html/content
JSON	collectionInfo.json	/conf/json/content

Abstract Test 15	/conf/edr/rc-md-query-links
Test Purpose	Validate that each Collection metadata entry in the Collections Metadata document includes all required links.
Requirement	/req/edr/rc-md-query-links
Test Method	1. Verify that each Collection item in the Collections Metadata document includes a `link` property for each supported encoding. 2. Verify that the `links` properties of the collection includes an item for each supported encoding with at least one link to either a query resource (relation: `data`) or instance resource (relation: `collection`). 3. Verify that all links include the `rel` and `type` link parameters.

Collection Links

Abstract Test 16	/conf/core/rc-collection-info-links
Test Purpose	Validate that the required links are included in the Collections Metadata document.
Requirement	/req/core/rc-collection-info-links
Test Method	Verify that the response document includes: 1. a link to this response document (relation: `self`), 2. a link to the response document in every other media type supported by the server (relation: `alternate`). Verify that all links include the `rel` and `type` link parameters.

B.4. Conformance Class JSON

Conformance Class	
http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/json	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/json
Dependency	Conformance Class "OAPI Core"

B.4.1. JSON Definition

Abstract Test 17	/conf/json/definition
Test Purpose	Verify support for JSON
Requirement	/req/json/definition
Test Method	1. A resource is requested with response media type of `application/json` 2. All `200`-responses SHALL support the media type:- `application/json`

B.5. Conformance Class GeoJSON

Conformance Class	
http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/json	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/json
Dependency	Conformance Class "OAPI Core"

B.5.1. JSON Definition

Abstract Test 18	/conf/json/definition
Test Purpose	Verify support for JSON
Requirement	/req/json/definition
Test Method	1. A resource is requested with response media type of `application/json` 2. All `200`-responses SHALL support the media type:- `application/json`

B.5.2. GeoJSON Content

Abstract Test 19	/conf/geojson/content
Test Purpose	Verify the content of a GeoJSON document given an input document and schema.

Requirement	/req/geojson/content
Test Method	1. Validate that the document is a GeoJSON document. 2. Validate the document against the schema using an JSON Schema validator. 3. Validate the document against the schema using an GeoJSON Schema validator.

B.6. Conformance Class EDR GeoJSON

Conformance Class	
http://www.opengis.net/spec/ogcapi-edr-1/1.0/conf/core/edr-geojson	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/edr-geojson
Dependency	Conformance Class "OAPI Core"

B.6.1. EDR GeoJSON Definition

Abstract Test 20	**/conf/edr-geojson/definition**
Test Purpose	Verify support for the EDR GeoJSON Schema
Requirement	/req/edr-geojson/definition
Test Method	1. A resource is requested with response media type of `application/json` and adheres to the EDR Feature Collection GeoJSON Schema. 2. All `200`-responses SHALL support the following media types: ◦ `application/json` for resources

B.6.2. EDR GeoJSON Content

Abstract Test 21	**/conf/edr-geojson/content**
Test Purpose	Verify the content of a EDR GeoJSON document given an input document and schema.
Requirement	/req/edr-geojson/content

Test Method	1. Validate that the document is an EDR GeoJSON document. 2. Validate the document against the schema using an JSON Schema validator. 3. Validate the document against the schema using an EDR GeoJSON Schema validator.

B.7. Conformance Class CoverageJSON

Conformance Class	
http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/covjson	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/covjson
Dependency	Conformance Class "OAPI Core"

B.7.1. CoverageJSON Definition

Abstract Test 22	**/conf/covjson/definition**
Test Purpose	Verify support for CoverageJSON
Requirement	/req/covjson/definition
Test Method	1. A resource is requested with response media type of `application/prs.coverage+json` 2. All `200`-responses SHALL support the following media types: ◦ `application/prs.coverage+json` for resources

B.7.2. CoverageJSON Content

Abstract Test 23	**/conf/covjson/content**
Test Purpose	Verify the content of a CoverageJSON document given an input document and schema.
Requirement	/req/covjson/content

Test Method	1. Validate that the document is a CoverageJSON document. 2. Validate the document against the schema using an JSON Schema validator. 3. Validate the document against the schema using an CoverageJSON Schema validator.

B.8. Conformance Class HTML

Conformance Class	
http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/html	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/html
Dependency	Conformance Class "OAPI Core"

B.8.1. HTML Definition

Abstract Test 24	**/conf/html/definition**
Test Purpose	Verify support for HTML
Requirement	/req/html/definition
Test Method	Verify that every `200`-response of every operation of the API where HTML was requested is of media type `text/html`

B.8.2. HTML Content

Abstract Test 25	**/conf/html/content**
Test Purpose	Verify the content of an HTML document given an input document and schema.
Requirement	/req/html/content
Test Method	1. Validate that the document is an HTML 5 document 2. Manually inspect the document against the schema.

B.9. Conformance Class OpenAPI 3.0

Conformance Class	
http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/oas3	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/oas3
Dependency	Conformance Class "OAPI Core"

Abstract Test 26	**/conf/oas30/completeness**
Test Purpose	Verify the completeness of an OpenAPI document.
Requirement	/req/oas30/completeness
Test Method	Verify that for each operation, the OpenAPI document describes all HTTP Status Codes and Response Objects that the API uses in responses.

Abstract Test 27	**/conf/oas30/exceptions-codes**
Test Purpose	Verify that the OpenAPI document fully describes potential exception codes.
Requirement	/req/oas30/exceptions-codes
Test Method	Verify that for each operation, the OpenAPI document describes all HTTP Status Codes that may be generated.

Abstract Test 28	**/conf/oas30/oas-definition-1**
Test Purpose	Verify that JSON and HTML versions of the OpenAPI document are available.
Requirement	/req/oas30/oas-definition-1
Test Method	1. Verify that an OpenAPI definition in JSON is available using the media type `application/vnd.oai.openapi+json;version=3.0` and link relation `service-desc` 2. Verify that an HTML version of the API definition is available using the media type `text/html` and link relation `service-doc`.

Abstract Test 29	/conf/oas30/oas-definition-2
Test Purpose	Verify that the OpenAPI document is valid JSON.
Requirement	/req/oas30/oas-definition-2
Test Method	Verify that the JSON representation conforms to the OpenAPI Specification, version 3.0.

Abstract Test 30	/conf/oas30/oas-impl
Test Purpose	Verify that all capabilities specified in the OpenAPI definition are implemented by the API.
Requirement	/req/oas30/oas-impl
Test Method	1. Construct a path from each URL template including all server URL options and all enumerated path parameters. 2. For each path defined in the OpenAPI document, validate that the path performs in accordance with the API definition and the API-Features standard.

Abstract Test 31	/conf/oas30/security
Test Purpose	Verify that any authentication protocols implemented by the API are documented in the OpenAPI document.
Requirement	/req/oas30/security
Test Method	1. Identify all authentication protocols supported by the API. 2. Validate that each authentication protocol is described inthe OpenAPI document by a Security Schema Object and its use specified by a Security Requirement Object.

B.10. Conformance Class Queries

Conformance Class	
http://www.opengis.net/spec/ogcapi-edr-1/1.0/conf/queries	
Target type	Web API
Requirements Class	http://www.opengis.net/spec/ogcapi-edr-1/1.0/conf/queries

Dependency	Conformance Class "OGC API Common Core"
Dependency	Conformance Class "OGC API Common Collections"

B.10.1. Query Pattern Tests

Position

Abstract Test 32	**/conf/position**
Test Purpose	Validate that an error is returned by a `Position` query if no query parameters are specified.
Requirement	/req/queries/position
Test Method	1. No query parameters are specified 2. Validate that a document was returned with a status code 400.
Abstract Test 33	**/conf/position**
Test Purpose	Validate that an error is returned by a `Position` query when the coords query parameter is not specified.
Requirement	/req/queries/position
Test Method	1. coords query parameter is not specified 2. Validate that a document was returned with a status code 400.
Abstract Test 34	**/conf/position**
Test Purpose	Validate that an error is returned by a `Position` query when the coords query parameter does not contain a valid POINT Well Known Text value.
Requirement	/req/queries/position
Test Method	1. Check coords query parameter is a valid Well Known Text Point value 2. Validate that a document was returned with a status code 400.
Abstract Test 35	**/conf/position**
Test Purpose	Validate that resources can be identified and extracted from a Collection with a `Position` query using query parameters.

Requirement	/req/queries/position
Test Method	1. Test with valid query parameters 2. Validate that a document was returned with a status code 200. Repeat these tests using the following parameter tests: **Coordinates** • Parameter /conf/collections/rc-coords-definition • Response /conf/collections/rc-coords-response **VerticalLevel** • Parameter /conf/collections/rc-z-definition • Response /conf/collections/rc-z-response **Parameters** * Parameter /conf/collections/rc-parametername-definition * Response /conf/collections/rc-parametername-response **DateTime** • Parameter /conf/collections/rc-time-definition • Response /conf/collections/rc-time-response Execute requests with combinations of the "coords","time","parametername","z","crs" and "f" query parameters and verify that only information that matches the selection criteria is returned.

Abstract Test 36	**/conf/edr/rc-coords-definition**
Test Purpose	Validate that the coords query parameters are constructed correctly.
Requirement	/req/edr/rc-coords-definition

Test Method	Verify that the `coords` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: coords`
	`in: query`
	`required: true`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`
	
	Use a coords value in all requests:
	
	• A valid Well-known text representation of geometry string

Abstract Test 37	**/conf/edr/rc-coords-response**
Test Purpose	Validate that the `coords` query parameters are processed corrrectly.
Requirement	/req/edr/rc-coords-response
Test Method	1. Verify that only resources that have a spatial geometry that intersects the coordinates are returned as part of the result set.
Test Method	1. Verify `coords` values are valid for the specified coordinate reference system 2. Verify that the coordinate reference system of the geometries are valid for the parameter defined by `crs`. If the `crs` parameter is not defined the geometries must be valid for WGS 84 longitude/latitude (`http://www.opengis.net/def/crs/OGC/1.3/CRS84` or `http://www.opengis.net/def/crs/OGC/0/CRS84h`).

Abstract Test 38	**/conf/edr/rc-z-definition**
Test Purpose	Validate that the vertical level query parameters are constructed correctly.
Requirement	/req/edr/rc-z-definition

Test Method	Verify that the `z` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: z`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 39	**/conf/edr/rc-z-response**
Test Purpose	Validate that the vertical level query parameters are processed correctly.
Requirement	/req/edr/rc-z-response
Test Method	1. Verify that only resources that have a vertical geometry that intersects the vertical information in the `z` parameter were included in the result set 2. Validate that the vertical level parameter complies with the syntax described in /req/edr/REQ_rc-z-response.

Abstract Test 40	**/conf/core/rc-time-definition**
Test Purpose	Validate that the dateTime query parameters are constructed correctly.
Requirement	/req/core/rc-time-definition

Test Method	Verify that the `time` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: time`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 41	**/conf/core/rc-time-response**
Test Purpose	Validate that the dataTime query parameters are processed correctly.
Requirement	/req/core/rc-time-response
Test Method	1. Verify that only resources that have a temporal geometry that intersects the temporal information in the `datetime` parameter were included in the result set 2. Verify that all resources in the collection that are not associated with a temporal geometry are included in the result set 3. Validate that the dateime parameter complies with the syntax described in /req/core/rc-time-response.

Abstract Test 42	**/conf/collections/rc-parametername-definition**
Test Purpose	Validate that the parametername query parameters are constructed correctly.
Requirement	/req/edr/rc-parametername-definition

Test Method	Verify that the `parametername` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: parametername`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 43	**/conf/edr/rc-parametername-response**
Test Purpose	Validate that the parametername query parameters are processed correctly.
Requirement	/req/edr/rc-parametername-response

Test Method	1. Verify that only resources for the requested parameters were included in the result set 2. Validate that the parametername parameter complies with the syntax described in /req/edr/rc-parameters-response.

Abstract Test 44	**/conf/edr/rc-outputCRS-definition**
Test Purpose	Validate that the crs query parameters are constructed correctly.
Requirement	/req/edr/rc-outputCRS-definition

Test Method	Verify that the `crs` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: crs`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 45	**/conf/edr/rc-outputCRS-response**
Test Purpose	Validate that the crs query parameters are processed correctly.
Requirement	/req/edr/rc-outputCRS-response
Test Method	1. Verify that the geometry of the resources returned are valid for the requested coordinate reference system
Test Method	1. Verify that all crs values defined in the collections metadata are supported by the collection 2. Validate that the crs parameter complies with the syntax described in /req/collections/edr/rc-crs-response.

Abstract Test 46	**/conf/edr/rc-outputformat-definition**
Test Purpose	Validate that the `f` query parameter is constructed correctly.
Requirement	/req/edr/rc-outputformat-definition

Test Method	Verify that the `f` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: f`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 47	**/conf/collections/rc-outputformat-response**
Test Purpose	Validate that the `f` query parameters are processed correctly.
Requirement	/req/collections/rc-outputformat-response
Test Method	1. Verify that the response is returned in the requested data format
Test Method	1. Verify that all output format values defined in the collections metadata are supported by the collection 2. Validate that the `f` parameter complies with the syntax described in /req/collections/edr/rc-output-response.

Area

Abstract Test 48	**/conf/area**
Test Purpose	Validate that an error is returned by a `Area` query if no query parameters are specified.
Requirement	/req/queries/area
Test Method	1. No query parameters are specified 2. Validate that a document was returned with a status code 400.
Abstract Test 49	**/conf/area**
Test Purpose	Validate that an error is returned by a `Area` query when the coords query parameter is not specified.

Requirement	/req/queries/area
Test Method	1. coords query parameter is not specified 2. Validate that a document was returned with a status code 400.
Abstract Test 50	**/conf/area**
Test Purpose	Validate that an error is returned by a `Area` query when the coords query parameter does not contain a valid POLYGON Well Known Text value.
Requirement	/req/queries/position
Test Method	1. Check coords query parameter is a valid Well Known Text Polygon value 2. Validate that a document was returned with a status code 400.
Abstract Test 51	**/conf/area**
Test Purpose	Validate that resources can be identified and extracted from a Collection with a `Area` query using query parameters.
Requirement	/req/queries/area

Test Method	1. Test with valid query parameters 2. Validate that a document was returned with a status code 200. Repeat these tests using the following parameter tests: **Coordinates** • Parameter /conf/collections/rc-coords-definition • Response /conf/collections/rc-coords-response **VerticalLevel** • Parameter /conf/collections/rc-z-definition • Response /conf/collections/rc-z-response **Parameters** * Parameter /conf/collections/rc-parametername-definition * Response /conf/collections/rc-parametername-response **DateTime** • Parameter /conf/collections/rc-time-definition • Response /conf/collections/rc-time-response Execute requests with combinations of the "coords","time","parametername","z","crs" and "f" query parameters and verify that only information that matches the selection criteria is returned.

Abstract Test 52	**/conf/edr/rc-coords-definition**
Test Purpose	Validate that the coords query parameters are constructed correctly.
Requirement	/req/edr/rc-coords-definition

Test Method	Verify that the `coords` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: coords`
	`in: query`
	`required: true`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`
	
	Use a coords value in all requests:
	
	• A valid Well-known text representation of geometry string

Abstract Test 53	**/conf/edr/rc-coords-response**
Test Purpose	Validate that the `coords` query parameters are processed corrrectly.
Requirement	/req/edr/rc-coords-response
Test Method	1. Verify that only resources that have a spatial geometry that intersects the coordinates are returned as part of the result set.
Test Method	1. Verify `coords` values are valid for the specified coordinate reference system 2. Verify that the coordinate reference system of the geometries are valid for the parameter defined by `crs`. If the `crs` parameter is not defined the geometries must be valid for WGS 84 longitude/latitude (`http://www.opengis.net/def/crs/OGC/1.3/CRS84` or `http://www.opengis.net/def/crs/OGC/0/CRS84h`).

Abstract Test 54	**/conf/edr/rc-z-definition**
Test Purpose	Validate that the vertical level query parameters are constructed correctly.
Requirement	/req/edr/rc-z-definition

Test Method	Verify that the `z` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: z`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 55	**/conf/edr/rc-z-response**
Test Purpose	Validate that the vertical level query parameters are processed correctly.
Requirement	/req/edr/rc-z-response
Test Method	1. Verify that only resources that have a vertical geometry that intersects the vertical information in the `z` parameter were included in the result set 2. Validate that the vertical level parameter complies with the syntax described in /req/edr/REQ_rc-z-response.

Abstract Test 56	**/conf/core/rc-time-definition**
Test Purpose	Validate that the dateTime query parameters are constructed correctly.
Requirement	/req/core/rc-time-definition

Test Method	Verify that the `time` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: time`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 57	**/conf/core/rc-time-response**
Test Purpose	Validate that the dataTime query parameters are processed correctly.
Requirement	/req/core/rc-time-response
Test Method	1. Verify that only resources that have a temporal geometry that intersects the temporal information in the `datetime` parameter were included in the result set 2. Verify that all resources in the collection that are not associated with a temporal geometry are included in the result set 3. Validate that the dateime parameter complies with the syntax described in /req/core/rc-time-response.

Abstract Test 58	**/conf/collections/rc-parametername-definition**
Test Purpose	Validate that the parametername query parameters are constructed correctly.
Requirement	/req/edr/rc-parametername-definition

Test Method	Verify that the `parametername` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: parametername`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 59	**/conf/edr/rc-parametername-response**
Test Purpose	Validate that the parametername query parameters are processed correctly.
Requirement	/req/edr/rc-parametername-response

Test Method	1. Verify that only resources for the requested parameters were included in the result set 2. Validate that the parametername parameter complies with the syntax described in /req/edr/rc-parameters-response.

Abstract Test 60	**/conf/edr/rc-outputCRS-definition**
Test Purpose	Validate that the crs query parameters are constructed correctly.
Requirement	/req/edr/rc-outputCRS-definition

Test Method	Verify that the `crs` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: crs`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 61	**/conf/edr/rc-outputCRS-response**
Test Purpose	Validate that the crs query parameters are processed correctly.
Requirement	/req/edr/rc-outputCRS-response
Test Method	1. Verify that the geometry of the resources returned are valid for the requested coordinate reference system
Test Method	1. Verify that all crs values defined in the collections metadata are supported by the collection 2. Validate that the crs parameter complies with the syntax described in /req/collections/edr/rc-crs-response.

Abstract Test 62	**/conf/edr/rc-outputformat-definition**
Test Purpose	Validate that the `f` query parameter is constructed correctly.
Requirement	/req/edr/rc-outputformat-definition

Test Method	Verify that the `f` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: f`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 63	**/conf/collections/rc-outputformat-response**
Test Purpose	Validate that the `f` query parameters are processed correctly.
Requirement	/req/collections/rc-outputformat-response
Test Method	1. Verify that the response is returned in the requested data format
Test Method	1. Verify that all output format values defined in the collections metadata are supported by the collection 2. Validate that the `f` parameter complies with the syntax described in /req/collections/edr/rc-output-response.

Trajectory

Abstract Test 64	**/conf/trajectory**
Test Purpose	Validate that an error is returned by a `Trajectory` query if no query parameters are specified.
Requirement	/req/queries/trajectory
Test Method	1. No query parameters are specified 2. Validate that a document was returned with a status code 400.
Abstract Test 65	**/conf/trajectory**
Test Purpose	Validate that an error is returned by a `Trajectory` query when the coords query parameter is not specified.

Requirement	/req/queries/trajectory
Test Method	1. coords query parameter is not specified 2. Validate that a document was returned with a status code 400.
Abstract Test 66	**/conf/trajectory**
Test Purpose	Validate that an error is returned by a `Trajectory` query when the coords query parameter does not contain a valid LINESTRING Well Known Text value.
Requirement	/req/queries/trajectory
Test Method	1. Check coords query parameter is a valid Well Known Text Linestring value 2. Validate that a document was returned with a status code 400.
Abstract Test 67	**/conf/trajectory**
Test Purpose	Validate that an error is returned by a `Trajectory` query when the coords query parameter does not contain a valid LINESTRINGM Well Known Text value.
Requirement	/req/queries/trajectory
Test Method	1. Check coords query parameter with time parameter is a valid Well Known Text LINESTRINGM value, the `M` coordinate must be a valid Epoch value (as known as UNIX time) 2. Validate that a document was returned with a status code 400.
Abstract Test 68	**/conf/trajectory**
Test Purpose	Validate that an error is returned by a `Trajectory` query when the `coords` query parameter is a LINESTRINGZ coordinate and the `z`query parameter is specified
Requirement	/req/queries/trajectory
Test Method	1. Check coords query parameter that the system throws an error when a vertical level is specified in both the `coords` and `z` parameters 2. Validate that a document was returned with a status code 400.

Abstract Test 69	**/conf/trajectory**
Test Purpose	Validate that an error is returned by a `Trajectory` query when the `coords` query parameter is a LINESTRINGZM coordinate and the `z`query parameter is specified
Requirement	/req/queries/trajectory
Test Method	1. Check coords query parameter that the system throws an error when a vertical level is specified in both the `coords` and `z` parameters 2. Validate that a document was returned with a status code 400.
Abstract Test 70	**/conf/trajectory**
Test Purpose	Validate that an error is returned by a `Trajectory` query when the coords query parameter does not contain a valid LINESTRINGMZ Well Known Text value.
Requirement	/req/queries/trajectory
Test Method	1. Check coords query parameter with time parameter is a valid Well Known Text LINESTRINGMZ value, the `Z` coordinate must be a within the range of vertical levels advertised in the Collection metadata 2. Validate that a document was returned with a status code 400.
Abstract Test 71	**/conf/trajectory**
Test Purpose	Validate that an error is returned by a `Trajectory` query when the coords query parameter does not contain a valid LINESTRINGZ Well Known Text value.
Requirement	/req/queries/trajectory
Test Method	1. Check coords query parameter with time parameter is a valid Well Known Text LINESTRINGZ value, the `Z` coordinate must be a within the range of vertical levels advertised in the Collection metadata 2. Validate that a document was returned with a status code 400.
Abstract Test 72	**/conf/trajectory**

Test Purpose	Validate that an error is returned by a `Trajectory` query when the `coords` query parameter contains invalid time coordinates
Requirement	/req/queries/trajectory
Test Method	1. If a time values are specified in the `coords` query parameter check that they are within the range of time values defined in the Collection metadata 2. Validate that a document was returned with a status code 400.
Abstract Test 73	**/conf/trajectory**
Test Purpose	Validate that resources can be identified and extracted from a Collection with a `Trajectory` query using query parameters.
Requirement	/req/queries/trajectory
Test Method	1. Test with valid query parameters 2. Validate that a document was returned with a status code 200. Repeat these tests using the following parameter tests: **Coordinates** • Parameter /conf/collections/rc-coords-definition • Response /conf/collections/rc-coords-response **VerticalLevel** • Parameter /conf/collections/rc-z-definition • Response /conf/collections/rc-z-response **Parameters** * Parameter /conf/collections/rc-parametername-definition * Response /conf/collections/rc-parametername-response Execute requests with combinations of the "coords", "parametername","z","crs" and "f" query parameters and verify that only information that matches the selection criteria is returned.

Abstract Test 74	**/conf/edr/rc-coords-definition**
Test Purpose	Validate that the coords query parameters are constructed correctly.

Requirement	/req/edr/rc-coords-definition

Test Method	Verify that the `coords` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: coords`
	`in: query`
	`required: true`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`
	
	Use a coords value in all requests:
	
	• A valid Well-known text representation of geometry string

Abstract Test 75	**/conf/edr/rc-coords-response**
Test Purpose	Validate that the `coords` query parameters are processed corrrectly.
Requirement	/req/edr/rc-coords-response
Test Method	1. Verify that only resources that have a spatial geometry that intersects the coordinates are returned as part of the result set.
Test Method	1. Verify `coords` values are valid for the specified coordinate reference system 2. Verify that the coordinate reference system of the geometries are valid for the parameter defined by `crs`. If the `crs` parameter is not defined the geometries must be valid for WGS 84 longitude/latitude (`http://www.opengis.net/def/crs/OGC/1.3/CRS84` or `http://www.opengis.net/def/crs/OGC/0/CRS84h`).

Abstract Test 76	**/conf/collections/rc-parametername-definition**
Test Purpose	Validate that the parametername query parameters are constructed correctly.

Requirement	/req/edr/rc-parametername-definition

Test Method	Verify that the `parametername` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: parametername`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 77	**/conf/edr/rc-parametername-response**
Test Purpose	Validate that the parametername query parameters are processed correctly.
Requirement	/req/edr/rc-parametername-response
Test Method	1. Verify that only resources for the requested parameters were included in the result set 2. Validate that the parametername parameter complies with the syntax described in /req/edr/rc-parameters-response.

Abstract Test 78	**/conf/edr/rc-outputCRS-definition**
Test Purpose	Validate that the crs query parameters are constructed correctly.
Requirement	/req/edr/rc-outputCRS-definition

Test Method	Verify that the `crs` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: crs`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 79	**/conf/edr/rc-outputCRS-response**
Test Purpose	Validate that the crs query parameters are processed correctly.
Requirement	/req/edr/rc-outputCRS-response
Test Method	1. Verify that the geometry of the resources returned are valid for the requested coordinate reference system
Test Method	1. Verify that all crs values defined in the collections metadata are supported by the collection 2. Validate that the crs parameter complies with the syntax described in /req/collections/edr/rc-crs-response.

Abstract Test 80	**/conf/edr/rc-outputformat-definition**
Test Purpose	Validate that the `f` query parameter is constructed correctly.
Requirement	/req/edr/rc-outputformat-definition

Test Method	Verify that the `f` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: f`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 81	**/conf/collections/rc-outputformat-response**
Test Purpose	Validate that the `f` query parameters are processed correctly.
Requirement	/req/collections/rc-outputformat-response
Test Method	1. Verify that the response is returned in the requested data format

Test Method	1. Verify that all output format values defined in the collections metadata are supported by the collection 2. Validate that the `f` parameter complies with the syntax described in /req/collections/edr/rc-output-response.

B.10.2. Collections and Instances

Feature Collections {root}/collections

Abstract Test 82	**/conf/collections/rc-md-op**
Test Purpose	Validate that information about the Collections can be retrieved from the expected location.
Requirement	/req/collections/rc-md-op
Test Method	1. Issue an HTTP GET request to the URL {root}/collections 2. Validate that a document was returned with a status code 200 3. Validate the contents of the returned document using test /conf/collections/rc-md-success.

Abstract Test 83	**/conf/rc-md-success**
Test Purpose	Validate that the Collections content complies with the required structure and contents.
Requirement	/req/collections/rc-md-success, /req/edr/rc-crs
Test Method	1. Validate that all response documents comply with /conf/collections/rc-collection-info-links 2. In case the response includes a "crs" property, validate that it includes a valid Well Known Text definition 3. Validate the collections content for all supported media types using the resources and tests identified in Table 11

The Collections content may be retrieved in a number of different formats. The following table identifies the applicable schema document for each format and the test to be used to validate the against that schema. All supported formats should be exercised.

Table 13. Schema and Tests for Collections content

Format	Schema Document	Test ID
HTML	collections.json	/conf/html/content
JSON	collections.json	/conf/geojson/content

Feature Collection {root}/collections/{collectionId}

Abstract Test 84	/conf/collections/src-md-op
Test Purpose	Validate that the Collection content can be retrieved from the expected location.
Requirement	/req/collections/src-md-op
Test Method	For every Feature Collection described in the Collections content, issue an HTTP GET request to the URL `/collections/{collectionId}` where `{collectionId}` is the `id` property for the collection. . Validate that a Collection was returned with a status code 200 . Validate the contents of the returned document using test /conf/collections/src-md-success.

Abstract Test 85	/conf/collections/src-md-success
Test Purpose	Validate that the Collection content complies with the required structure and contents.
Requirement	/req/collections/src-md-success
Test Method	Verify that the content of the response is consistent with the content for this Resource Collection in the `/collections` response. That is, the values for `id`, `title`, `description` and `extent` are identical.

Features {root}/collections/{collectionId}/items

NOTE	This test is too Feature centric. Will need to be greatly reduced in scope.

Abstract Test 86	/conf/core/rc-op
Test Purpose	Validate that resources can be identified and extracted from a Collection using query parameters.
Requirement	/req/collections/rc-op

Test Method	1. For every resource collection identified in Collections, issue an HTTP GET request to the URL `/collections/{collectionId}/items` where `{collectionId}` is the `id` property for a Collection described in the Collections content. 2. Validate that a document was returned with a status code 200. Repeat these tests using the following parameter tests: **Bounding Box:** • Parameter /conf/core/rc-bbox-definition • Response /conf/core/rc-bbox-response **DateTime:** • Parameter /conf/core/rc-time-definition • Response /conf/core/rc-time-response Execute requests with combinations of the "bbox" and "datetime" query parameters and verify that only features are returned that match both selection criteria.

Abstract Test 87	**/conf/core/bbox-definition**
Test Purpose	Validate that the bounding box query parameters are constructed correctly.
Requirement	/req/core/bbox-definition

Test Method	Verify that the `bbox` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: bbox`
	`in: query`
	`required: false`
	`schema:`
	`  type: array`
	`  minItems: 4`
	`  maxItems: 6`
	`  items:`
	`  type: number`
	`style: form`
	`explode: false`
	
	Use a bounding box with four numbers in all requests:
	
	• Lower left corner, WGS 84 longitude
	• Lower left corner, WGS 84 latitude
	• Upper right corner, WGS 84 longitude
	• Upper right corner, WGS 84 latitude

Abstract Test 88	**/conf/core/rc-bbox-response**
Test Purpose	Validate that the bounding box query parameters are processed corrrectly.
Requirement	/req/core/rc-bbox-response
Test Method	1. Verify that only resources that have a spatial geometry that intersects the bounding box are returned as part of the result set. 2. Verify that the `bbox` parameter matched all resources in the collection that were not associated with a spatial geometry (this is only applicable for datasets that include resources without a spatial geometry). 3. Verify that the coordinate reference system of the geometries is WGS 84 longitude/latitude (`http://www.opengis.net/def/crs/OGC/1.3/CRS84` or `http://www.opengis.net/def/crs/OGC/0/CRS84h`) since no parameter `bbox-crs` was specified in the request.

Abstract Test 89	**/conf/core/rc-time-definition**
Test Purpose	Validate that the dateTime query parameters are constructed correctly.
Requirement	/req/core/rc-time-definition

Test Method	Verify that the `time` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: time`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 90	**/conf/core/rc-time-response**
Test Purpose	Validate that the dataTime query parameters are processed correctly.
Requirement	/req/core/rc-time-response
Test Method	1. Verify that only resources that have a temporal geometry that intersects the temporal information in the `datetime` parameter were included in the result set 2. Verify that all resources in the collection that are not associated with a temporal geometry are included in the result set 3. Validate that the dateime parameter complies with the syntax described in /req/core/rc-time-response.

Abstract Test 91	**/conf/collections/rc-response**
Test Purpose	Validate that the Resource Collection complies with the require structure and contents.
Requirement	/req/collections/rc-response
Test Method	The test method is specific to the resource type returned.

Instances {root}/collections/{collectionId}/instances

Abstract Test 92	**/conf/instances/rc-md-op**
Test Purpose	Validate that information about the instances of a Collection can be retrieved from the expected location.
Requirement	/req/instances/rc-md-op
Test Method	1. Issue an HTTP GET request to the URL {root}/collections/{collectionId}/instances 2. Validate that a document was returned with a status code 200 3. Validate the contents of the returned document using test /conf/collections/rc-md-success.

Abstract Test 93	**/conf/instances_rc-md-success**
Test Purpose	Validate that the instances of the Collection content complies with the required structure and contents.
Requirement	/req/instances/rc-md-success, /req/core/crs84
Test Method	1. Validate that all response documents comply with /conf/collections/rc-md-links 2. In case the response includes a "crs" property, validate that the first value is either "http://www.opengis.net/def/crs/OGC/1.3/CRS84" or "http://www.opengis.net/def/crs/OGC/0/CRS84h" 3. Validate the collections content for all supported media types using the resources and tests identified in Table 11

The Instances content, unlike the Collections content, may only be retrieved in the same formats as specified for the single parent collection.

Table 14. Schema and Tests for Collections content

Format	**Schema Document**	**Test ID**
HTML	collections.json	/conf/html/content
JSON	collections.json	/conf/geojson/content

Instance {root}/collections/{collectionId}/instances/instanceId

Abstract Test 94	**/conf/instances/src-md-op**

Test Purpose	Validate that the Instances of the Collection content can be retrieved from the expected location.
Requirement	/req/collections/src-md-op
Test Method	For every Instance of a Collection described in the Collections content, issue an HTTP GET request to the URL `/collections/{collectionId}/instances/{instanceId}` where `{collectionId}` is the `id` property for the collection and `{instanceId}` is the `id` property for the instance. . Validate that an Instance of a Collection was returned with a status code 200 . Validate the contents of the returned document using test /conf/collections/src-md-success.

Abstract Test 95	**/conf/instances/src-md-success**
Test Purpose	Validate that the Instance of a Collection content complies with the required structure and contents.
Requirement	/req/collections/src-md-success
Test Method	Verify that the content of the response is consistent with the content for this Resource Collection in the `/collections` response. That is, the values for `id`, `title`, `description` and `extent` are identical.

B.10.3. Second Tier Tests

These tests are invoked by other tests.

Items

Abstract Test 96	**/conf/edr/rc-collection-info**
Test Purpose	Validate that each collection provided by the server is described in the Collections Metadata.
Requirement	/req/edr/rc-collection-info

Test Method	1. Verify that all collections listed in the `collections` array of the Collections Metadata exist. 2. Verify that each collection entry includes an identifier. 3. Verify that each collection entry includes links in accordance with /core/rc-collection-info-links. 4. Verify that if the collection entry includes an extent property, the property complies with /core/rc-extent 5. Verify that if the collection entry includes an crs property, the property complies with /edr/rc-crs 6. Verify that if the collection entry includes an parameters property, the property complies with /edr/rc-parameters 7. Validate each collection entry for all supported media types using the resources and tests identified in Table 12

The collection entries may be encoded in a number of different formats. The following table identifies the applicable schema document for each format and the test to be used to validate the against that schema. All supported formats should be exercised.

Abstract Test 97	**/conf/core/rc-collection-info-links**
Test Purpose	Validate that each Collection metadata entry in the Collections Metadata document includes all required links.
Requirement	/req/core/rc-collection-info-links
Test Method	1. Verify that each Collection item in the Collections Metadata document includes a `link` property for each supported encoding. 2. Verify that the `links` properties of the collection includes an item for each supported encoding with a link to the features resource (relation: `items`). 3. Verify that all links include the `rel` and `type` link parameters.

Locations

Abstract Test 98	**/conf/locations**
Test Purpose	Validate that a list of valid locations are returned by a `Locations` query if no query parameters are specified.
Requirement	/req/queries/locations

Test Method	1. No query parameters are specified 2. Validate that a GeoJSON document was returned with a status code 200 containing at least a list of features one for each location supported by the collection.
Abstract Test 99	**/conf/locations**
Test Purpose	Validate that an error is returned by a `Locations` query when the the locationId is invalid.
Requirement	/req/queries/locations
Test Method	1. Check that invalid `locationId` values return an error message 2. Validate that a document was returned with a status code 404.
Abstract Test 100	**/conf/locations**
Test Purpose	Validate that resources can be identified and extracted from a Collection with a `Locations` query using query parameters.
Requirement	/req/queries/locations
Test Method	1. Test with valid query parameters 2. Validate that a document was returned with a status code 200. Repeat these tests using the following parameter tests: **Parameters** * Parameter /conf/collections/rc-parametername-definition * Response /conf/collections/rc-parametername-response **DateTime** • Parameter /conf/collections/rc-time-definition • Response /conf/collections/rc-time-response Execute requests with combinations of the "time","parametername","crs" and "f" query parameters and verify that only information that matches the selection criteria is returned.

Abstract Test 101	**/conf/core/rc-time-definition**

Test Purpose	Validate that the dateTime query parameters are constructed correctly.
Requirement	/req/core/rc-time-definition

Test Method	Verify that the `time` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: time`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 102	**/conf/core/rc-time-response**
Test Purpose	Validate that the dataTime query parameters are processed correctly.
Requirement	/req/core/rc-time-response
Test Method	1. Verify that only resources that have a temporal geometry that intersects the temporal information in the `datetime` parameter were included in the result set 2. Verify that all resources in the collection that are not associated with a temporal geometry are included in the result set 3. Validate that the dateime parameter complies with the syntax described in /req/core/rc-time-response.

Abstract Test 103	**/conf/collections/rc-parametername-definition**
Test Purpose	Validate that the parametername query parameters are constructed correctly.
Requirement	/req/edr/rc-parametername-definition

Test Method	Verify that the `parametername` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: parametername`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 104	**/conf/edr/rc-parametername-response**
Test Purpose	Validate that the parametername query parameters are processed correctly.
Requirement	/req/edr/rc-parametername-response
Test Method	1. Verify that only resources for the requested parameters were included in the result set 2. Validate that the parametername parameter complies with the syntax described in /req/edr/rc-parameters-response.

Abstract Test 105	**/conf/edr/rc-outputCRS-definition**
Test Purpose	Validate that the crs query parameters are constructed correctly.
Requirement	/req/edr/rc-outputCRS-definition

Test Method	Verify that the `crs` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	
	`name: crs`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 106	**/conf/edr/rc-outputCRS-response**
Test Purpose	Validate that the crs query parameters are processed correctly.
Requirement	/req/edr/rc-outputCRS-response
Test Method	1. Verify that the geometry of the resources returned are valid for the requested coordinate reference system
Test Method	1. Verify that all crs values defined in the collections metadata are supported by the collection 2. Validate that the crs parameter complies with the syntax described in /req/collections/edr/rc-crs-response.

Abstract Test 107	**/conf/edr/rc-outputformat-definition**
Test Purpose	Validate that the `f` query parameter is constructed correctly.
Requirement	/req/edr/rc-outputformat-definition

Test Method	Verify that the `f` query parameter complies with the following definition (using an OpenAPI Specification 3.0 fragment):
	`name: f`
	`in: query`
	`required: false`
	`schema:`
	`  type: string`
	`style: form`
	`explode: false`

Abstract Test 108	**/conf/collections/rc-outputformat-response**
Test Purpose	Validate that the `f` query parameters are processed correctly.
Requirement	/req/collections/rc-outputformat-response
Test Method	1. Verify that the response is returned in the requested data format

Test Method	1. Verify that all output format values defined in the collections metadata are supported by the collection 2. Validate that the `f` parameter complies with the syntax described in /req/collections/edr/rc-output-response.

Annex C: Examples (Informative)

C.1. OpenAPI definition

OpenAPI YAML

OpenAPI JSON

C.2. Example Landing Pages

Example 43. JSON Landing Page


```
{
  "links": [
    { "href": "http://data.example.org/",
      "rel": "self", "type": "application/json", "title": "this document" },
    { "href": "http://data.example.org/api",
      "rel": "service", "type": "application/openapi+json;version=3.0", "title":
"the API definition" },
    { "href": "http://data.example.org/conformance",
      "rel": "conformance", "type": "application/json", "title": "OGC conformance
classes implemented by this API" },
    { "href": "http://data.example.org/collections",
      "rel": "data", "type": "application/json", "title": "Metadata about the
resource collections" }
  ]
}
```


C.3. API Description Examples

NOTE include::examples/tbd.adoc[]

C.4. Conformance Examples

Example 44. Conformance Response

This example response in JSON is for an OGC API Features that supports OpenAPI 3.0 for the API definition and HTML and GeoJSON as encodings for resources.

```
{
  "conformsTo": [
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/core",
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/oas30",
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/html",
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/geojson"
  ]
}
```

C.5. Collections Metadata Examples

Example 45. Collections metadata response document

The example below shows a service with two collections, one for observations and another for forecast data. The forecast data is regenerated every hour so the collection provides access to multiple instances of the collection via an instances endpoint.

There are links to the responses of the collections (link relation type: "self").

Representations of these resource in other formats are referenced using link relation type "alternate".

The data queries that are supported by each collection are referenced using link relation type "data".

There are also links to the license information for the observation and forecast data (using:https://www.iana.org/assignments/link-relations/link-relations.xhtml[link relation type] "license") and also the terms and conditions of service (using:https://www.iana.org/assignments/link-relations/link-relations.xhtml[link relation type] "restrictions") .


```
{
    "links": [
        {
            "href": "http://www.example.org/edr/collections/",
            "hreflang": "en",
            "rel": "self",
            "type": "application/json"
        },
        {
            "href": "http://www.example.org/edr/collections/",
            "hreflang": "en",
            "rel": "alternate",
            "type": "text/html"
        },
        {
            "href": "http://www.example.org/edr/collections/",
            "hreflang": "en",
            "rel": "alternate",
            "type": "application/xml"
        }
    ],
    "collections": [
        {
            "id": "hourly observations",
            "title": "Hourly Site Specific observations",
            "description": "Observation data for UK observing sites",
            "keywords": [
                "Wind Direction",
                "Wind Speed",
                "Wind Gust",
```

```
                "Air Temperature",
                "Weather",
                "Relative Humidity",
                "Dew point",
                "Pressure",
                "Pressure Tendancy",
                "Visibility"
            ],
            "links": [
                {
                    "href": "http://www.example.org/uk-hourly-site-specific-
observations",
                    "hreflang": "en",
                    "rel": "service-doc",
                    "type": "text/html",
                    "title": ""
                },
                {
                    "href": "http://www.example.org/terms-and-conditions---
datapoint#datalicence",
                    "hreflang": "en",
                    "rel": "license",
                    "type": "text/html",
                    "title": ""
                },
                {
                    "href":
"https://www.metoffice.gov.uk/services/data/datapoint/terms-and-conditions---
datapoint#termsofservice",
                    "hreflang": "en",
                    "rel": "restrictions",
                    "type": "text/html",
                    "title": ""
                },
                {
                    "href":
"http://www.example.org/edr/collections/hrly_obs/position",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "position",
                    "title": ""
                },
                {
                    "href":
"http://www.example.org/edr/collections/hrly_obs/radius",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "radius",
                    "title": ""
                },
                {
```

```
                    "href":
"http://www.example.org/edr/collections/hrly_obs/area",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "area",
                    "title": ""
                },
                {
                    "href":
"http://www.example.org/edr/collections/hrly_obs/locations",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "location",
                    "title": ""
                }
            ],
            "extent": {
                "spatial": {
                    "bbox": [
                        -15.0,
                        48.0,
                        5.0,
                        62.0
                    ],
                    "crs": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                },
                "temporal": {
                    "interval": [
                        "2020-04-19T11:00:00Z/2020-06-30T09:00:00Z"
                    ],
                    "trs": "TIMECRS[\"DateTime\",TDATUM[\"Gregorian
Calendar\"],CS[TemporalDateTime,1],AXIS[\"Time (T)\",future]"
                }
            },
            "crs": [
                {
                    "name": "CRS84",
                    "wkt": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                }
            ],
            "distanceunits": [
                "km",
                "miles"
            ],
            "outputformat": [
```

```
                {
                    "name": "GeoJSON",
                    "data_schema":
"http://www.example.org/edr/static/json/dp_schema.json"
                },
                {
                    "name": "CoverageJSON"
                }
            ],
            "parameters": {
                "Wind Direction": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "degree true"
                        },
                        "symbol": {
                            "value": "°",
                            "type":
"http://www.example.org/edr/metadata/units/degree"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_windDirection",
                        "label": {
                            "en": "Wind Direction"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Speed": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://www.example.org/edr/metadata/units/mph"
                        }
```

```
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_windSpeed",
                        "label": {
                            "en": "Wind Speed"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Gust": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_maximumWindGustSpeed",
                        "label": {
                            "en": "Wind Gust"
                        }
                    },
                    "measurementType": {
                        "method": "maximum",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Air Temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
```

```
                            "type":
"http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Air Temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Weather": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "weather"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://www.example.org/edr/metadata/lookup/mo_dp_weather"
                        }
                    },
                    "observedProperty": {
                        "id":
"http://codes.wmo.int/wmdr/ObservedVariableAtmosphere/_266",
                        "label": {
                            "en": "Weather"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Relative Humidity": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
```

```
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/bufr4/b/13/_009",
                        "label": {
                            "en": "Relative Humidity"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Dew point": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_dewPointTemperature",
                        "label": {
                            "en": "Dew point"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Pressure": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
```

```
                        "label": {
                            "en": "hPa"
                        },
                        "symbol": {
                            "value": "hPa",
                            "type":
"http://www.example.org/edr/metadata/units/hPa"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/bufr4/b/10/_051",
                        "label": {
                            "en": "Pressure"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Pressure Tendancy": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "tendency"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://www.example.org/edr/metadata/units/hPa"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_pressureTendency",
                        "label": {
                            "en": "Pressure Tendancy"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Visibility": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
```

```
                    },
                    "unit": {
                        "label": {
                            "en": "m"
                        },
                        "symbol": {
                            "value": "m",
                            "type": "http://www.example.org/edr/metadata/units/m"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_horizontalVisibility",
                        "label": {
                            "en": "Visibility"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                }
            }
        },
        {
            "id": "UK 3 hourly forecast",
            "title": "UK 3 Hourly Site Specific Forecast",
            "description": "Five day site specific forecast for 6000 UK
locations",
            "keywords": [
                "Wind Direction",
                "Wind Speed",
                "Wind Gust",
                "Air Temperature",
                "Weather",
                "Relative Humidity",
                "Feels like temperature",
                "UV index",
                "Probabilty of precipitation",
                "Visibility"
            ],
            "links": [
                {
                    "href": "https://http://www.example.org/uk-3-hourly-site-
specific-forecast",
                    "hreflang": "en",
                    "rel": "service-doc",
                    "type": "text/html",
                    "title": ""
                },
                {
```

```
                    "href": "https://http://www.example.org/terms-and-
conditions---datapoint#datalicence",
                    "hreflang": "en",
                    "rel": "licence",
                    "type": "text/html",
                    "title": ""
                },
                {
                    "href": "https://http://www.example.org/terms-and-
conditions---datapoint#termsofservice",
                    "hreflang": "en",
                    "rel": "restrictions",
                    "type": "text/html",
                    "title": ""
                },
                {
                    "href":
"http://www.example.org/edr/collections/3_hrly_fcst/instances",
                    "hreflang": "en",
                    "rel": "collection",
                    "type": "instances",
                    "title": ""
                }
            ],
            "extent": {
                "spatial": {
                    "bbox": [
                        -15.0,
                        48.0,
                        5.0,
                        62.0
                    ],
                    "crs": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                },
                "temporal": {
                    "interval": [
                        "2020-06-23T18:00:00Z/2020-07-04T21:00:00Z"
                    ],
                    "trs": "TIMECRS[\"DateTime\",TDATUM[\"Gregorian
Calendar\"],CS[TemporalDateTime,1],AXIS[\"Time (T)\",future]"
                }
            },
            "crs": [
                {
                    "name": "CRS84",
                    "wkt": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
```

```
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                }
            ],
            "distanceunits": [
                "km",
                "miles"
            ],
            "outputformat": [
                {
                    "name": "GeoJSON",
                    "data_schema":
"http://www.example.org/edr/static/json/dp_schema.json"
                },
                {
                    "name": "CoverageJSON"
                }
            ],
            "parameters": {
                "Wind Direction": {
                    "type": "Parameter",
                    "description": {
                        "en": "Direction wind is from"
                    },
                    "unit": {
                        "label": {
                            "en": "degree true"
                        },
                        "symbol": {
                            "value": "°",
                            "type":
"http://www.example.org/edr/metadata/units/degree"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-0",
                        "label": {
                            "en": "Wind Direction"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Speed": {
                    "type": "Parameter",
                    "description": {
                        "en": "Average wind speed"
                    },
                    "unit": {
                        "label": {
```

```
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Speed"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Gust": {
                    "type": "Parameter",
                    "description": {
                        "en": "Wind gusts are a rapid increase in strength of the
wind relative to the wind speed."
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Gust"
                        }
                    },
                    "measurementType": {
                        "method": "maximum",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Air Temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": "2m air temperature in the shade and out of the
wind"
```

```
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Air Temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Weather": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "weather"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://www.example.org/edr/metadata/lookup/mo_dp_weather"
                        }
                    },
                    "observedProperty": {
                        "id":
"http://codes.wmo.int/wmdr/ObservedVariableAtmosphere/_266",
                        "label": {
                            "en": "Weather"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Relative Humidity": {
```

```
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Relative Humidity"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Feels like temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Feels like temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
```

```
                },
                "UV index": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "UV_index"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://www.example.org/edr/metadata/lookup/mo_dp_uv"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-4-51",
                        "label": {
                            "en": "UV index"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Probabilty of precipitation": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Probabilty of precipitation"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
```

```
                    }
                },
                "Visibility": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "quality"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://www.example.org/edr/metadata/lookup/mo_dp_visibility"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_horizontalVisibility",
                        "label": {
                            "en": "Visibility"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                }
            }
        }
    ]
}
```


C.6. Instance Metadata Examples

Example 46. Collection instance metadata response document

This is an example of the metadata returned by the instances query

(link relation type: "items").

There is a link to the instance response itself (link relation type: "self").

Representations of this resource in other formats are referenced using link relation type "alternate".

The data queries that are supported by each instance are referenced using link relation type "data".

There are also links to the license information for the observation and forecast data (using:https://www.iana.org/assignments/link-relations/link-relations.xhtml[link relation type] "license") and also the terms and conditions of service (using:https://www.iana.org/assignments/link-relations/link-relations.xhtml[link relation type] "restrictions") .


```
{
    "links": [
        {
            "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/",
            "hreflang": "en",
            "rel": "self",
            "type": "application/json"
        },
        {
            "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/",
            "hreflang": "en",
            "rel": "alternate",
            "type": "text/html"
        },
        {
            "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/",
            "hreflang": "en",
            "rel": "alternate",
            "type": "application/xml"
        },
        {
            "href": "https://http://www.example.org/terms-and-conditions---
datapoint#termsofservice",
            "hreflang": "en",
            "rel": "restrictions",
            "type": "text/html",
            "title": ""
```

```
        },
        {
            "href": "https://http://www.example.org/terms-and-conditions---
datapoint#datalicence",
            "hreflang": "en",
            "rel": "license",
            "type": "text/html",
            "title": ""
        },
        {
            "href": "https://http://www.example.org/uk-3-hourly-site-specific-
forecast",
            "hreflang": "en",
            "rel": "service-doc",
            "type": "text/html",
            "title": ""
        }
    ],
    "instances": [
        {
            "id": "2020-06-30T10:00:00Z",
            "title": "3 hrly fcst",
            "description": "Five day site specific forecast for 6000 UK locations
3 hrly fcst",
            "keywords": [
                "Wind Direction",
                "Wind Speed",
                "Wind Gust",
                "Air Temperature",
                "Weather",
                "Relative Humidity",
                "Feels like temperature",
                "UV index",
                "Probabilty of precipitation",
                "Visibility"
            ],
            "links": [
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T10:00:00Z/position",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "position",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T10:00:00Z/radius",
                    "hreflang": "en",
```

```
                    "rel": "data",
                    "type": "radius",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T10:00:00Z/area",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "area",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T10:00:00Z/locations",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "location",
                    "title": ""
                }
            ],
            "extent": {
                "spatial": {
                    "bbox": [
                        -15.0,
                        48.0,
                        5.0,
                        62.0
                    ],
                    "crs": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                },
                "temporal": {
                    "interval": [
                        "2020-06-30T06:00:00Z/2020-07-04T21:00:00Z"
                    ],
                    "trs": "TIMECRS[\"DateTime\",TDATUM[\"Gregorian
Calendar\"],CS[TemporalDateTime,1],AXIS[\"Time (T)\",future]"
                }
            },
            "crs": [
                {
                    "name": "CRS84",
                    "wkt": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
```

```
                }
            ],
            "distanceunits": [
                "km",
                "miles"
            ],
            "outputformat": [
                {
                    "name": "GeoJSON",
                    "data_schema":
"http://http://www.example.org/edr/static/json/dp_schema.json"
                },
                {
                    "name": "CoverageJSON"
                }
            ],
            "parameters": {
                "Wind Direction": {
                    "type": "Parameter",
                    "description": {
                        "en": "Direction wind is from"
                    },
                    "unit": {
                        "label": {
                            "en": "degree true"
                        },
                        "symbol": {
                            "value": "°",
                            "type":
"http://http://www.example.org/edr/metadata/units/degree"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-0",
                        "label": {
                            "en": "Wind Direction"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Speed": {
                    "type": "Parameter",
                    "description": {
                        "en": "Average wind speed"
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
```

```
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Speed"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Gust": {
                    "type": "Parameter",
                    "description": {
                        "en": "Wind gusts are a rapid increase in strength of the
wind relative to the wind speed."
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Gust"
                        }
                    },
                    "measurementType": {
                        "method": "maximum",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Air Temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": "2m air temperature in the shade and out of the
wind"
                    },
```

```
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Air Temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Weather": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "weather"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_weather"
                        }
                    },
                    "observedProperty": {
                        "id":
"http://codes.wmo.int/wmdr/ObservedVariableAtmosphere/_266",
                        "label": {
                            "en": "Weather"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Relative Humidity": {
                    "type": "Parameter",
```

```
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Relative Humidity"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Feels like temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Feels like temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
```

```
                "UV index": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "UV_index"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_uv"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-4-51",
                        "label": {
                            "en": "UV index"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Probabilty of precipitation": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Probabilty of precipitation"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
```

```
                },
                "Visibility": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "quality"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_visibility"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_horizontalVisibility",
                        "label": {
                            "en": "Visibility"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                }
            }
        },
        {
            "id": "2020-06-30T09:00:00Z",
            "title": "3 hrly fcst",
            "description": "Five day site specific forecast for 6000 UK locations
3 hrly fcst",
            "keywords": [
                "Wind Direction",
                "Wind Speed",
                "Wind Gust",
                "Air Temperature",
                "Weather",
                "Relative Humidity",
                "Feels like temperature",
                "UV index",
                "Probabilty of precipitation",
                "Visibility"
            ],
            "links": [
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
```

```
30T09:00:00Z/position",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "position",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T09:00:00Z/radius",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "radius",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T09:00:00Z/area",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "area",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T09:00:00Z/locations",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "location",
                    "title": ""
                }
            ],
            "extent": {
                "spatial": {
                    "bbox": [
                        -15.0,
                        48.0,
                        5.0,
                        62.0
                    ],
                    "crs": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                },
                "temporal": {
                    "interval": [
                        "2020-06-30T06:00:00Z/2020-07-04T21:00:00Z"
                    ],
```

```
                    "trs": "TIMECRS[\"DateTime\",TDATUM[\"Gregorian
Calendar\"],CS[TemporalDateTime,1],AXIS[\"Time (T)\",future]"
                }
            },
            "crs": [
                {
                    "name": "CRS84",
                    "wkt": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                }
            ],
            "distanceunits": [
                "km",
                "miles"
            ],
            "outputformat": [
                {
                    "name": "GeoJSON",
                    "data_schema":
"http://http://www.example.org/edr/static/json/dp_schema.json"
                },
                {
                    "name": "CoverageJSON"
                }
            ],
            "parameters": {
                "Wind Direction": {
                    "type": "Parameter",
                    "description": {
                        "en": "Direction wind is from"
                    },
                    "unit": {
                        "label": {
                            "en": "degree true"
                        },
                        "symbol": {
                            "value": "°",
                            "type":
"http://http://www.example.org/edr/metadata/units/degree"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-0",
                        "label": {
                            "en": "Wind Direction"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
```

```
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Speed": {
                    "type": "Parameter",
                    "description": {
                        "en": "Average wind speed"
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Speed"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Gust": {
                    "type": "Parameter",
                    "description": {
                        "en": "Wind gusts are a rapid increase in strength of the
wind relative to the wind speed."
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Gust"
                        }
                    },
```

```
                    "measurementType": {
                        "method": "maximum",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Air Temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": "2m air temperature in the shade and out of the
wind"
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Air Temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Weather": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "weather"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_weather"
                        }
                    },
                    "observedProperty": {
                        "id":
"http://codes.wmo.int/wmdr/ObservedVariableAtmosphere/_266",
```

```
                        "label": {
                            "en": "Weather"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Relative Humidity": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Relative Humidity"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Feels like temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
```

```
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Feels like temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "UV index": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "UV_index"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_uv"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-4-51",
                        "label": {
                            "en": "UV index"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Probabilty of precipitation": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://http://www.example.org/edr/metadata/units/percent"
                        }
```

```
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Probabilty of precipitation"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Visibility": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "quality"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_visibility"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_horizontalVisibility",
                        "label": {
                            "en": "Visibility"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                }
            }
        },
        {
            "id": "2020-06-30T08:00:00Z",
            "title": "3 hrly fcst",
            "description": "Five day site specific forecast for 6000 UK locations
3 hrly fcst",
            "keywords": [
                "Wind Direction",
                "Wind Speed",
                "Wind Gust",
                "Air Temperature",
```

```
                "Weather",
                "Relative Humidity",
                "Feels like temperature",
                "UV index",
                "Probabilty of precipitation",
                "Visibility"
            ],
            "links": [
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T08:00:00Z/position",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "position",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T08:00:00Z/radius",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "radius",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T08:00:00Z/area",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "area",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T08:00:00Z/locations",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "location",
                    "title": ""
                }
            ],
            "extent": {
                "spatial": {
                    "bbox": [
                        -15.0,
                        48.0,
                        5.0,
```

```
                        62.0
                    ],
                    "crs": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                },
                "temporal": {
                    "interval": [
                        "2020-06-30T03:00:00Z/2020-07-04T21:00:00Z"
                    ],
                    "trs": "TIMECRS[\"DateTime\",TDATUM[\"Gregorian
Calendar\"],CS[TemporalDateTime,1],AXIS[\"Time (T)\",future]"
                }
            },
            "crs": [
                {
                    "name": "CRS84",
                    "wkt": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                }
            ],
            "distanceunits": [
                "km",
                "miles"
            ],
            "outputformat": [
                {
                    "name": "GeoJSON",
                    "data_schema":
"http://http://www.example.org/edr/static/json/dp_schema.json"
                },
                {
                    "name": "CoverageJSON"
                }
            ],
            "parameters": {
                "Wind Direction": {
                    "type": "Parameter",
                    "description": {
                        "en": "Direction wind is from"
                    },
                    "unit": {
                        "label": {
                            "en": "degree true"
                        },
                        "symbol": {
                            "value": "°",
                            "type":
```

```
"http://http://www.example.org/edr/metadata/units/degree"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-0",
                        "label": {
                            "en": "Wind Direction"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Speed": {
                    "type": "Parameter",
                    "description": {
                        "en": "Average wind speed"
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Speed"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Gust": {
                    "type": "Parameter",
                    "description": {
                        "en": "Wind gusts are a rapid increase in strength of the
wind relative to the wind speed."
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
```

```
                            "value": "mph",
                            "type":
"http://http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Gust"
                        }
                    },
                    "measurementType": {
                        "method": "maximum",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Air Temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": "2m air temperature in the shade and out of the
wind"
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Air Temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Weather": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
```

```
                            "en": "weather"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_weather"
                        }
                    },
                    "observedProperty": {
                        "id":
"http://codes.wmo.int/wmdr/ObservedVariableAtmosphere/_266",
                        "label": {
                            "en": "Weather"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Relative Humidity": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Relative Humidity"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Feels like temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
```

```
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Feels like temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "UV index": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "UV_index"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_uv"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-4-51",
                        "label": {
                            "en": "UV index"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Probabilty of precipitation": {
                    "type": "Parameter",
                    "description": {
```

```
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Probabilty of precipitation"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Visibility": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "quality"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_visibility"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_horizontalVisibility",
                        "label": {
                            "en": "Visibility"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                }
            }
```

```
        },
        {
            "id": "2020-06-30T07:00:00Z",
            "title": "3 hrly fcst",
            "description": "Five day site specific forecast for 6000 UK locations
3 hrly fcst",
            "keywords": [
                "Wind Direction",
                "Wind Speed",
                "Wind Gust",
                "Air Temperature",
                "Weather",
                "Relative Humidity",
                "Feels like temperature",
                "UV index",
                "Probabilty of precipitation",
                "Visibility"
            ],
            "links": [
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T07:00:00Z/position",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "position",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T07:00:00Z/radius",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "radius",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T07:00:00Z/area",
                    "hreflang": "en",
                    "rel": "data",
                    "type": "area",
                    "title": ""
                },
                {
                    "href":
"http://http://www.example.org/edr/collections/3_hrly_fcst/instances/2020-06-
30T07:00:00Z/locations",
                    "hreflang": "en",
```

```
                    "rel": "data",
                    "type": "location",
                    "title": ""
                }
            ],
            "extent": {
                "spatial": {
                    "bbox": [
                        -15.0,
                        48.0,
                        5.0,
                        62.0
                    ],
                    "crs": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                },
                "temporal": {
                    "interval": [
                        "2020-06-30T03:00:00Z/2020-07-04T21:00:00Z"
                    ],
                    "trs": "TIMECRS[\"DateTime\",TDATUM[\"Gregorian
Calendar\"],CS[TemporalDateTime,1],AXIS[\"Time (T)\",future]"
                }
            },
            "crs": [
                {
                    "name": "CRS84",
                    "wkt": "GEOGCS[\"WGS 84\",DATUM[\"WGS_1984\",SPHEROID[\"WGS
84\",6378137,298.257223563,AUTHORITY[\"EPSG\",\"7030\"]],AUTHORITY[\"EPSG\",\"6326
\"]],PRIMEM[\"Greenwich\",0,AUTHORITY[\"EPSG\",\"8901\"]],UNIT[\"degree\",0.017453
29251994328,AUTHORITY[\"EPSG\",\"9122\"]],AUTHORITY[\"EPSG\",\"4326\"]]"
                }
            ],
            "distanceunits": [
                "km",
                "miles"
            ],
            "outputformat": [
                {
                    "name": "GeoJSON",
                    "data_schema":
"http://http://www.example.org/edr/static/json/dp_schema.json"
                },
                {
                    "name": "CoverageJSON"
                }
            ],
            "parameters": {
                "Wind Direction": {
```

```
                    "type": "Parameter",
                    "description": {
                        "en": "Direction wind is from"
                    },
                    "unit": {
                        "label": {
                            "en": "degree true"
                        },
                        "symbol": {
                            "value": "°",
                            "type":
"http://http://www.example.org/edr/metadata/units/degree"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-0",
                        "label": {
                            "en": "Wind Direction"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Wind Speed": {
                    "type": "Parameter",
                    "description": {
                        "en": "Average wind speed"
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Speed"
                        }
                    },
                    "measurementType": {
                        "method": "mean",
                        "period": "-PT10M/PT0M"
                    }
                },
```

```
                "Wind Gust": {
                    "type": "Parameter",
                    "description": {
                        "en": "Wind gusts are a rapid increase in strength of the
wind relative to the wind speed."
                    },
                    "unit": {
                        "label": {
                            "en": "mph"
                        },
                        "symbol": {
                            "value": "mph",
                            "type":
"http://http://www.example.org/edr/metadata/units/mph"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-2-1",
                        "label": {
                            "en": "Wind Gust"
                        }
                    },
                    "measurementType": {
                        "method": "maximum",
                        "period": "-PT10M/PT0M"
                    }
                },
                "Air Temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": "2m air temperature in the shade and out of the
wind"
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Air Temperature"
                        }
                    },
                    "measurementType": {
```

```
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Weather": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "weather"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_weather"
                        }
                    },
                    "observedProperty": {
                        "id":
"http://codes.wmo.int/wmdr/ObservedVariableAtmosphere/_266",
                        "label": {
                            "en": "Weather"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Relative Humidity": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Relative Humidity"
                        }
```

```
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Feels like temperature": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "degC"
                        },
                        "symbol": {
                            "value": "°C",
                            "type":
"http://http://www.example.org/edr/metadata/units/degC"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
kind/_airTemperature",
                        "label": {
                            "en": "Feels like temperature"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "UV index": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "UV_index"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_uv"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-4-51",
                        "label": {
```

```
                            "en": "UV index"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Probabilty of precipitation": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "percent"
                        },
                        "symbol": {
                            "value": "%",
                            "type":
"http://http://www.example.org/edr/metadata/units/percent"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-1-1",
                        "label": {
                            "en": "Probabilty of precipitation"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                },
                "Visibility": {
                    "type": "Parameter",
                    "description": {
                        "en": ""
                    },
                    "unit": {
                        "label": {
                            "en": "quality"
                        },
                        "symbol": {
                            "value": "",
                            "type":
"http://http://www.example.org/edr/metadata/lookup/mo_dp_visibility"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/common/quantity-
```

```
kind/_horizontalVisibility",
                        "label": {
                            "en": "Visibility"
                        }
                    },
                    "measurementType": {
                        "method": "instantaneous",
                        "period": "PT0M"
                    }
                }
            }
        }
    ]
}
```

C.7. Location Query Metadata Examples

Example 47. Collection instance metadata response document

A example of the Locations metadata from a collection that supports the `Location` query pattern.

```
(link:https://www.iana.org/assignments/link-relations/link-relations.xhtml[link
relation type]: "items").
```

There is a link to the collections response itself (link relation type: "self").

Representations of this resource in other formats are referenced using link relation type "alternate".

Finally there are also links to the license information for the data (using:https://www.iana.org/assignments/link-relations/link-relations.xhtml[link relation type] "license").


```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "id": 3002,
      "geometry": {
        "type": "Point",
        "coordinates": [
          -0.854,
          60.749
        ]
      },
      "properties": {
        "name": "BALTASOUND",
        "datetime": "2020-03-30T19:00:00Z/2020-04-20T07:00:00Z",
        "detail":
"https://oscar.wmo.int/surface/rest/api/stations/station/1745/stationReport",
        "WIGOS Station Identifier": "0-20000-0-03002"
      }
    },
    {
      "type": "Feature",
      "id": 3005,
      "geometry": {
        "type": "Point",
        "coordinates": [
          -1.183,
          60.139
        ]
      },
      "properties": {
```

```
        "name": "LERWICK (S. SCREEN)",
        "datetime": "2020-03-30T19:00:00Z/2020-04-20T07:00:00Z",
        "detail":
"https://oscar.wmo.int/surface/rest/api/stations/station/1746/stationReport",
        "WIGOS Station Identifier": "0-20000-0-03005"
      }
    },
    {
      "type": "Feature",
      "id": 3008,
      "geometry": {
        "type": "Point",
        "coordinates": [
          -1.628,
          59.527
        ]
      },
      "properties": {
        "name": "FAIR ISLE",
        "datetime": "2020-03-30T19:00:00Z/2020-04-20T07:00:00Z",
        "detail":
"https://oscar.wmo.int/surface/rest/api/stations/station/1747/stationReport",
        "WIGOS Station Identifier": "0-20000-0-03008"
      }
    },
    {
      "type": "Feature",
      "id": 3017,
      "geometry": {
        "type": "Point",
        "coordinates": [
          -2.9,
          58.954
        ]
      },
      "properties": {
        "name": "KIRKWALL",
        "datetime": "2020-03-30T19:00:00Z/2020-04-20T07:00:00Z",
        "detail":
"https://oscar.wmo.int/surface/rest/api/stations/station/1750/stationReport",
        "WIGOS Station Identifier": "0-20000-0-03017"
      }
    },
    {
      "type": "Feature",
      "id": 3023,
      "geometry": {
        "type": "Point",
        "coordinates": [
          -7.397,
          57.358
```

```
        ]
      },
      "properties": {
        "name": "SOUTH UIST RANGE",
        "datetime": "2020-03-30T19:00:00Z/2020-04-20T07:00:00Z",
        "detail":
"https://oscar.wmo.int/surface/rest/api/stations/station/1751/stationReport",
        "WIGOS Station Identifier": "0-20000-0-03023"
      }
    }
  ]
```

Annex D: Glossary

Abstract Test Suite (ATS)

A compendium of test assertions applicable to implementations of a standard. An ATS provides a basis for developing an Executable Test Suite to verify that the implementation under test conforms to all the relevant functional specifications.

Collection

body of resources that belong or are used together. An aggregate, set, or group of related resources. (OGC 20-024)

Conformance Module; Conformance Test Module

set of related tests, all within a single conformance test class (OGC 08-131)

NOTE	When no ambiguity is possible, the word `test` may be omitted. i.e. conformance test module is the same as conformance module. Conformance modules may be nested in a hierarchical way.

Conformance Class; Conformance Test Class

set of conformance test modules that must be applied to receive a single certificate of conformance (OGC 08-131)

NOTE	When no ambiguity is possible, the word *test* may be left out, so conformance test class maybe called a conformance class.

dataset

collection of data, published or curated by a single agent, and available for access or download in one or more formats (DCAT)

distribution

represents an accessible form of a **dataset** (DCAT)

NOTE	EXAMPLE: a downloadable file, an RSS feed or a web service that provides the data.

Executable Test Suite (ETS)

A set of code (e.g. Java and CTL) that provides runtime tests for the assertions defined by the ATS. Test data required to do the tests are part of the ETS (OGC 08-134)

Recommendation

expression in the content of a document conveying that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required, or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited (OGC 08-131)

Requirement

expression in the content of a document conveying criteria to be fulfilled if compliance with the

document is to be claimed and from which no deviation is permitted (OGC 08-131)

Requirements Class

aggregate of all requirement modules that must all be satisfied to satisfy a conformance test class (OGC 08-131)

Requirements Module

aggregate of requirements and recommendations of a specification against a single standardization target type (OGC 08-131)

Spatial Resource

resources usually considered as Geospatial Data. (OGC 19-072)

Standardization Target

entity to which some requirements of a standard apply (OGC 08-131)

NOTE	The standardization target is the entity which may receive a certificate of conformance for a requirements class.

Web API

API using an architectural style that is founded on the technologies of the Web. (W3C Data on the Web Best Practices)

Annex E: Revision History

Date	Release	Editor	Primary clauses modified	Description
2019-10-31	October 2019 snapshot	C. Heazel	all	Baseline update
2020-06-04	June 2020 master	Mark Burgoyne	all	Resolved Trajectory pattern
2020-06-05	June 2020 branch	Chris Little	all	Increase alignment with Common
2020-07-15	July 2020 branch	Chris Little	all	Editorial consistency
2020-07-22	Restructure branch	Chris Little	all	Restructure 7,8,9,10
2020-07-22	Issues 106, 107	Dave Blodgett	all	Fix broken links
2020-10-20	Oct 2020 Master	Chris Little	Definitions	Added missing Cube definition

Annex F: Bibliography

- Open Geospatial Consortium: The Specification Model — A Standard for Modular specifications, OGC 08-131
- W3C/OGC: Spatial Data on the Web Best Practices, W3C Working Group Note 28 September 2017, https://www.w3.org/TR/sdw-bp/
- W3C: Data on the Web Best Practices, W3C Recommendation 31 January 2017, https://www.w3.org/TR/dwbp/
- W3C: Data Catalog Vocabulary, W3C Recommendation 16 January 2014, https://www.w3.org/TR/vocab-dcat/
- IANA: Link Relation Types, https://www.iana.org/assignments/link-relations/link-relations.xml