OGC® DOCUMENT: 25-014

External identifier of this OGC® document: http://www.opengis.net/doc/spec/ogcapi-edr-3/1.0

Open Geospatial Consortium

OGC API - ENVIRONMENTAL DATA RETRIEVAL STANDARD - PART 3: PROFILES

STANDARD
Implementation

DRAFT

Version: 1.0
Submission Date: 2025-08-21
Approval Date: 2029-03-30
Publication Date: 2029-03-30
Editor: Mark Burgoyne, Charles Heazel

License Agreement

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

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

Copyright notice

Note

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

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

CONTENTS

LIST OF TABLES

LIST OF RECOMMENDATIONS

I ABSTRACT

The OGC API-EDR Part 1: Core standard was designed to be flexible and straightforward to understand and implement for Web developers. As it is being widely implemented, various groups of users have identified the need to restrict some of the flexibility to improve interoperability between different implementations of both servers and clients within their domains of interest. A set of these stricter specifications for a specific domain of user is a Profile.

The aim of the OGC API-EDR Part 3: Service Profile Support standard is to ensure interoperability between API implementations by defining a standard approach to specifying a Profile of OGC API-EDR Part 1: Core.

To ensure consistency, providers must adopt a standardised approach when defining `Collections` and their `Instances`. The OGC API-EDR Profile outlines a set of requirements that an EDR API implementation must meet to be considered compliant with the service profile.

II KEYWORDS

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

ogcdoc, OGC document, ModSpec, API, OpenAPI, html, Profile

III PREFACE

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

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

IV SECURITY CONSIDERATIONS

No security considerations have been made for this Standard.

V SUBMITTING ORGANIZATIONS

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

- UK Met Office
- Heazeltech

VI SUBMITTERS

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

Name	Affiliation
Mark Burgoyne	Met Office
Charles Heazel	Heazel Tech
Chris Little	Met Office

VII CONTRIBUTORS

Additional contributors to this Standard include the following:

Individual name(s), Organization

PREFACE

PREFACE

NOTE: The aim of the OGC API-EDR Part 3: Service Profile Support standard is to ensure interoperability between EDR API implementations by defining a standard approach to specifying a Profile of OGC API-EDR Part 1: Core. To achieve this, it is essential that service providers use a consistent approach when defining `Collections` and `Instances` of `Collections`. An OGC API-EDR Profile will specify a set of Requirements that an EDR API implementation must support to be a profile-Markcompliant implementation.

This standard specifies Requirements and Recommendations for a Profile definition, and also conforms to the OGC Mod Spec Standard.

It is envisaged that this approach may be useful for other OGC API Standards.

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

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

1 SCOPE

1 SCOPE

NOTE: This Standard defines how to specify a Profile of the OGC API-EDR Part 1: Core Standard. It only defines restrictive profiles, not profiles that extend the EDR API Standard with new functionality, which may not maintain backward compatibility with the EDR API.

Some parts of the specification could be used by other OGC APIs.

The restrictions are defined by using JSON Schema fragments, which can be formally tested.

2

CONFORMANCE

2 CONFORMANCE

Conformance to the OGC API-EDR-Part 3 Standard (this document) by a profile of the OGC API — Environmental Data Retrieval Standard can be tested by inspection. The test suite is provided in Annex A.

This Standard contains normative language and thus places requirements on conformance, or mechanism for adoption, of candidate standards to which this Standard applies. In particular:

- OGC API-EDR Requirements Class: Core specifies the core requirements which shall be met by all standards claiming conformance to this Standard.

Annex B provides guidance on how to build a profile of an ISO Standard. While not normative, following these practices increases the likelihood that the suite of OGC API-EDR Standards and profiles will form an interoperable whole.

3

NORMATIVE REFERENCES

3 NORMATIVE REFERENCES

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

ISO: ISO 19106, *Geographic information — Profiles*. International Organization for Standardization, Geneva https://www.iso.org/standard/26011.html.

Mark Burgoyne, Dave Blodgett, Chuck Heazel, Chris Little: OGC 19-086r4, *OGC API — Environmental Data Retrieval Standard*. Open Geospatial Consortium (2021). http://www.opengis.net/doc/IS/ogcapi-edr-1/1.0.0.

https://docs.ogc.org/is/17-069r4/17-069r4.html, OGC APIFeatures — Part 1: Core, Open Geospatial Consortium (2022).

https://docs.ogc.org/is/19-072/19-072.html, OGC API — Common — Part 1: Core, Open Geospatial Consortium (2023).

http://docs.ogc.org/DRAFTS/20-024.html, OGC API — Common — Part 2: Geospatial Data (Draft), Open Geospatial Consortium

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

OpenAPI Initiative (OAI). **OpenAPI Specification 3.0** [online]. 2024 [viewed 2025-01-03]. The latest patch version at the time of publication of this standard was 3.0.4, available at https://spec.openapis.org/oas/v3.0.4

OpenAPI Initiative (OAI). **OpenAPI Specification 3.1** [online]. 2024 [viewed 2025-01-03]. The latest patch version at the time of publication of this standard was 3.1.1, available at https://spec.openapis.org/oas/v3.1.1

4

TERMS AND DEFINITIONS

4 TERMS AND DEFINITIONS

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

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

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

4.1. Collection

Body of resources that belong or are used together. An aggregate, set, or group of related resources.

[**SOURCE:** OGC 20-024]

4.2. Profile

An implementation case of a more general standard or set of standards.

4.3. Conformance Module; Conformance Test Module

A set of related conformance classes and their associated components.

Note 1 to entry: 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.

[**SOURCE:** OGC 08-131r5]

4.4. Conformance Class; Conformance Test Class

A set of conformance tests that must be passed to receive a single certificate of conformance.

Note 1 to entry: When no ambiguity is possible, the word *test* may be left out, so conformance test class maybe called a conformance class.

[**SOURCE:** OGC 08-131r5]

4.5. Conformance Test

A test, abstract or real, of one or more requirements contained within a standard, or set of standards.

[**SOURCE:** OGC 08-131r5]

4.6. Requirement

Expression in the content of a standard conveying criteria to be fulfilled if compliance with the standard is to be claimed and from which no deviation is permitted.

[**SOURCE:** OGC 08-131r5]

4.7. Requirements Class

An aggregate of requirements with a single standardization target type that must all be satisfied to pass a conformance test Class.

[**SOURCE:** OGC 08-131r5]

4.8. Requirements Module

A set of related requirements classes and their associated components.

[**SOURCE:** OGC 08-131r5]

4.9. Standardization Goal

A concise statement of the problem that the standard helps address and the strategy envisioned for achieving a solution. This strategy typically identifies real-world entities that need to be modified or constrained. At the abstract level, those entities are the Standardization Target Types.

[**SOURCE:** OGC 08-131r5]

4.10. Standardization Target

Entity to which some requirements of a standard apply.

Note 1 to entry: The standardization target is the entity which may receive a certificate of conformance for a requirements class.

[**SOURCE:** OGC 08-131r5]

4.11. Standardization Target Type

Type of entity or set of entities to which the requirement of a standard apply.

Note 1 to entry: For example, the standardization target type for The OGC API – Features Standard are Web APIs. The standardization target type for the CDB Standard is "datastore". It is important to understand that a standard's root standardization target type can have sub-types, and that there can be a hierarchy of target types. For example, a Web API can have sub types of client, server, security, and so forth. As such, each requirements class can have a standardization target type that is a sub-type of the root.

[**SOURCE:** OGC 08-131r5]

5

CONVENTIONS

5 CONVENTIONS

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

5.1. Identifiers

The normative provisions in this standard are denoted by the URI

`http://www.opengis.net/doc/spec/ogcapi-edr-3/1.0`

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

5.1.1. Shortcuts

In the interest of readability, the following terms will be used as shorthand for more complex text:

- Profile: A Profile is a standard or specification which restricts and/or extends an existing standard. This standard defines the rules for creating a profile of the OGC API — Environmental Data Retrieval Standard. The term “Profile” will be used in this document as shorthand for “profile of the OGC API — Environmental Data Retrieval Standard”.
- OGC API-EDR: The term OGC API-EDR will be used in this document as shorthand for the term “OGC API — Environmental Data Retrieval Standard”

6 CONTEXT

6 CONTEXT

6.1. Standardization Goal

The goal of this Standard is to ensure interoperabilty between implementations of the OGC API — Environmental Data Retrieval Standard (OGC API-EDR).

The OGC API-EDR Standard does not try to address every possible application domain. Rather, it provides a foundation which can be tailored for a specific domain. The result of this tailoring is a domain specific “profile” of the EDR API Standard.

A significant risk to this approach is that, in the act of profiling, interoperabilty will be compromized. This risk can be mitigated by establishing rules for how the OGC API-EDR Standard can be profiled. The goal of this Standard is to define a set of rules sufficient to ensure interoperability while retaining the adaptability needed to support domain-specific requirements.

6.2. Standardization Target Type

The Standardization Target Type for this Standard is the set of standards and specifications which profile the OGC API — Environmental Data Retrieval Standard.

It is important to understand that:

- This Standard is a standard for writing standards. It does not address the EDR API implementation.
- This Standard is a profile of the OGC ModSpec Model — Part 1: Core — A Standard for Designing and Writing Modular Standards (ModSpec).
- Implementations of this Standard are Profiles of the OGC API — Environmental Data Retrieval Standard
- The profiling model used is defined in ISO 19106:2004 Geographic information — Profiles

6.3. Profiles

ISO 19106:2004 Geographic information — Profiles is the ISO TC211 Standard for developing profiles of ISO TC211 Standards. This standard defines two conformance classes. These conformance classes can be thought of as two classes of profile.

- A Class 1 profile is a pure subset of the ISO geographic information standards.
- A Class 2 profile has the same basis as Class 1 but includes extensions within the contexts permitted in the base standard. Additionally, a Class 2 profile permits the profiling of non-ISO geographic information standards as part of the profile.

In other words, a Class 1 profile restricts the base standard while a Class 2 profile both restricts and extends the base standard.

Both Class 1 and Class 2 Profiles of the OGC API-EDR Standard are allowed.

Detailed guidance on how to create a valid Class 1 and Class 2 profile are provided in Annex B.

7

REQUIREMENTS CLASS CORE

7 REQUIREMENTS CLASS CORE

REQUIREMENTS CLASS 1: REQUIREMENTS CLASS 'CORE'	
IDENTIFIER	http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core
CONFORMANCE CLASS	Conformance class A.1: http://www.opengis.net/spec/ogcapi-edr-3/1.0/conf/conf-class-core
TARGET-TYPE	OGC API-EDR Profile Standard
NORMATIVE STATEMENTS	Requirement 1: /req/core/modspec Requirement 2: /req/core/edr-conformant Requirement 3: /req/core/parameter-names Requirement 4: /req/core/root Requirement 5: /req/core/root-description Requirement 7: /req/core/root-keywords Requirement 8: /req/core/root-provider Requirement 9: /req/core/root-contact Requirement 10: /req/core/root-links Requirement 11: /req/core/publishing Requirement 12: /req/core/openapi Requirement 13: /req/core/api Requirement 14: /req/core/requirements-set Requirement 15: /req/core/collectionid Requirement 16: /req/core/extent Requirement 17: /req/core/extent-spatial Requirement 18: /req/core/extent-temporal Requirement 19: /req/core/extent-vertical Requirement 20: /req/core/extent-custom Requirement 21: /req/core/data-query Requirement 22: /req/core/output-format Requirement 23: /req/core/data-query-area Requirement 24: /req/core/data-query-corridor Requirement 25: /req/core/data-query-cube Requirement 27: /req/core/data-query-instances Requirement 1-26: /req/core/instanceid Requirement 28: /req/core/paging-support Requirement 26: /req/core/data-query-locations Requirement 29: /req/core/data-query-position Requirement 30: /req/core/data-query-radius Requirement 31: /req/core/data-query-trajectory Requirement 32: /req/core/status-codes Requirement 33: /req/core/links Requirement 34: /req/core/asynchronous

REQUIREMENTS CLASS 1: REQUIREMENTS CLASS 'CORE'
Requirement 35: `/req/core/pubsub`

7.1. Profiling Requirements

Profile is conformant with the ModSpec

REQUIREMENT 1	
IDENTIFIER	`/req/core/modspec`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A profile of the OGC API — Environmental Data Retrieval Standard *SHALL* be conformant to the OGC Modular Specification.

Implementations of the Profile are conformant with OGC API-EDR Part 1

REQUIREMENT 2	
IDENTIFIER	`/req/core/edr-conformant`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require that a conformant implementation (standardization target) of that profile demonstrate conformance to the OGC API — Environmental Data Retrieval Standard.

Profiles often focus on restricting the values of Path parameters. A Profile should clearly specify the requirements for these restrictions

REQUIREMENT 3	
IDENTIFIER	`/req/core/parameter-names`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* the valid values and definitions of parameter_names, then,

REQUIREMENT 3

A	Requirements *SHALL* be defined which specify the parameter_names and their definitions.
B	The parameter_names requirement definitions *SHALL* specify the required parameter_names objects in full including: • name, • unit, • data type and • measurement duration

STATEMENT	for example:
	`"parameter_names": {`
	`"prmsl": {`
	`"type": "Parameter",`
	`"description": "Air pressure at sea level",`
	`"unit": {`
	`"label": "Pascals",`
	`"symbol": {`
	`"value": "Pa",`
	`"type": "https://qudt.org/vocab/unit/PA"`
	`}`
	`},`
	`"observedProperty": {`
	`"id": "http://codes.wmo.int/grib2/codeflag/4.2/_0-3-1",`
	`"label": "MSL Pressure"`
	`}`
	`},`
	`"t2m": {`
	`"type": "Parameter",`
	`"description": "Air temperature at 2m",`
	`"unit": {`
	`"label": "Kelvin",`
	`"symbol": {`
	`"value": "K",`
	`"type": "https://qudt.org/vocab/unit/K"`
	`}`
	`},`
	`"observedProperty": {`
	`"id": "http://codes.wmo.int/grib2/codeflag/4.2/0-0-0",`
	`"label": "Air temperature at 2m"`
	`}`
	`},`
	`"dd": {`
	`"type": "Parameter",`
	`"description": "Wind Direction",`
	`"unit": {`
	`"label": "degree true",`
	`"symbol": {`
	`"value": "°",`
	`"type": "https://qudt.org/vocab/unit/DEG"`
	`}`
	`},`
	`"observedProperty": {`
	`"id": "http://codes.wmo.int/grib2/codeflag/4.2/0-2-0",`
	`"label": "Wind Direction"`
	`},`
	`"measurementType": {`
	`"method": "mean",`
	`"duration": "-PT10M"`
	`}`
	`},`
	`"ff": {`
	`"type": "Parameter",`

REQUIREMENT 3

```
                    "description": "10m Wind Speed",
                    "unit": {
                        "label": "m/s",
                        "symbol": {
                          "value": "ms-1",
                          "type": "https://qudt.org/vocab/unit/M/s"
                        }
                    },
                    "observedProperty": {
                        "id": "http://codes.wmo.int/grib2/codeflag/4.2/0-2-1",
                        "label": "10m Wind Speed"
                    },
                    "measurementType": {
                        "method": "mean",
                        "duration": "-PT10M"
                    }
                }
            }
```

A Profile must be interoperable with other OGC API-EDR data providers. Any valid OGC API-EDR document should be valid under the profile. That means:

- If a data element is valid for OGC API-EDR, then it should not be prohibited under the profile
- Custom query elements defined by a Service Profile *SHALL* be optional, not mandatory, for data queries.
- It is valid for a profile to prohibit the production and population of OGC API-EDR optional elements by data providers within the profile's domain.

7.2. Platform Resources

OGC 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 spatio-temporal data resources.

Table 1 — Platform Resource Paths

PATH TEMPLATE	METHOD	RESOURCE
{root}/	GET	Landing page
{root}/api	GET	API Description (optional)
{root}/conformance	GET	Conformance Classes

Where: {root} = Base URI for the API server

7.2.1. API Landing Page

Path = {root}/

Dependencies

- OGC API — Common — Part 1: Core
- OGC API — Environmental Data Retrieval Standard

The landing page provides links that support exploration of the resources offered via the API. The most important component of a landing page is a list of links. The Landing Page resource is initially defined in the Core conformance class of the OGC API — Common — Part 1 Standard. The OGC API — Environmental Data Retrieval Standard does not make any changes to this definition.

The normative JSON Schema for an EDR Landing Page is defined in the LandingPage.yaml document. While this schema provides a rich body of information about the API, only the Links property is required.

Profiles of the OGC API — Environmental Data Retrieval Standard are expected to provide a richer description of the API. The additional content that Profiles should mandate is defined in the following requirements.

REQUIREMENT 4	
IDENTIFIER	`/req/core/root`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	The OGC API-EDR Landing Page schema only requires the `links` property, however, a Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require the following additional properties and content:
A	The `Title` property *SHALL* be required and populated
B	The `Links` property *SHALL* define the links that SHALL be included in the Root response and *SHALL* be populated with href and rel properties.

RECOMMENDATION 1	
IDENTIFIER	`/rec/core/root`

RECOMMENDATION 1

STATEMENT	The OGC API-EDR Landing Page schema only requires the `links` property, however, a Profile of the OGC API — Environmental Data Retrieval Standard *SHOULD* require the following additional properties:
A	The `Description` property *SHOULD* be required
B	The `Keywords` property *SHOULD* be required
C	The `Provider` property *SHOULD* be required and populated with the name and url properties
D	The `Contact` property *SHOULD* be required and populated with the with the email properties

REQUIREMENT 5

IDENTIFIER	`/req/core/root-description`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require that when an EDR Landing Page includes the `Description` property, that property *SHALL* be populated.

REQUIREMENT 6

IDENTIFIER	`/req/core/root-attribution`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require that when an EDR Landing Page includes the `Attribution` property, that property *SHALL* be populated.

REQUIREMENT 7

IDENTIFIER	`/req/core/root-keywords`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require that when an EDR Landing Page includes the `Keywords` property, that property *SHALL* be populated with at least one keyword entry.

REQUIREMENT 8

IDENTIFIER	`/req/core/root-provider`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require that when an EDR Landing Page includes the `Provider` property.
A	A `Provider` property *SHALL* be populated with a `name` property.
B	A `Provider` property *MAY* be populated with a `url` property.

REQUIREMENT 9

IDENTIFIER	`/req/core/root-contact`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require that when an EDR Landing Page includes the `Contact` property,
A	A `Contact` property *SHALL* be populated with the `email` property.
B	A `Contact` property *MAY* be populated with the `name` property.
C	A `Contact` property *MAY* be populated with the `url` property.

REQUIREMENT 10

IDENTIFIER	`/req/core/root-links`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require that when an EDR Landing Page includes the `Links` property:
A	The `Links` property *SHALL* define the links that *SHALL* be included in the Root response
B	The `Links` property *SHALL* be populated with href and rel properties
C	The `Links` property *MAY* be populated with type, hreflang and length properties

RECOMMENDATION 2	
IDENTIFIER	`/rec/core/root-links`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHOULD* require that when an EDR Landing Page includes the `Links` property, the title property of each link *SHALL* be populated.

7.2.2. API Definition

Path = {root}/api

Dependencies

- OGC API — Common — Part 1: Core
- OGC API — Environmental Data Retrieval Standard

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. The API Definition resource is initially defined in the Core conformance class of the OGC API — Common — Part 1 Standard. The OGC API — Environmental Data Retrieval Standard Standard does not make any changes to this definition.

Profiles of the OGC API — Environmental Data Retrieval Standard are required to provide an OpenAPI 3.1 document. This document extends the API definition provided by the OGC API-EDR Standard. These extensions reflect the additional requirements added by the Profile. Implementors of the profile will then build on that document to produce the API definition document for their implementation.

NOTE: At this time only OpenAPI 3.0 and OpenAPI 3.1 documents are supported by OGC Web API Standards.

REQUIREMENT 11	
IDENTIFIER	`/req/core/publishing`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	An EDR profile *SHALL* be published as an OpenAPI JSON document.
A	The rules described in the requirements *SHALL* be encoded using the OpenAPI 3.1 specification.
B	The requirement rules *SHALL* be encoded in either the OpenAPI Path Item or in the Response object schema sections of the document.

REQUIREMENT 12	
IDENTIFIER	`/req/core/openapi`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	The profile OpenAPI document *SHALL* describe the profile EDR API as follows:
A	The servers attributes of the OpenAPI root object *SHALL* be blank (the profile is not linked to specific implementations)
B	The Extent requirement rules *SHALL* be encoded in the JSON schema defined in the 200 responses for the /collections and /collections/{collectionId} Paths object
C	The data_query type requirement rules *SHALL* be encoded in the JSON schema defined in the 200 responses for the /collections and /collections/{collectionId} Paths object
D	The conformance classes *SHALL* be encoded in the JSON schema defined in the 200 responses for the /conformance Paths object.
E	The data_query types *SHALL* be encoded as Paths objects in the OpenAPI document, where appropriate the output_format, default_output_format, crs, within_units, width-units, height-units and limit (paging) requirements SHALL be encoded in the child Parameter objects of the Paths object.
F	The output_format requirement rules *SHALL* be encoded in the 200 responses of the data_query type Paths objects
G	The Parameter_names requirements *SHALL* be encoded in the JSON schema defined in the 200 responses for the /collections and /collections/{collectionId} Paths object.
H	Where a `Collection` has `Instances`, Parameter_names requirements *SHALL* be encoded in the JSON schema defined in the 200 responses for the /collections/{collectionId}/instances and /collections/{collectionId}/instances/{instanceId} Paths object.
I	An EDR API *SHALL* advertise the location of the profile OpenAPI document it complies with in the links section of the API root with a link relation value of 'profile'

7.2.3. Declaration of Conformance Classes

Path = {root}/conformance

Dependencies

- OGC API — Common — Part 1: Core
- OGC API — Environmental Data Retrieval Standard

To support "generic" clients that want to access implementations of 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. The Conformance Classes resource is initially defined in

the Core conformance class of the OGC API — Common — Part 1 Standard. The OGC API — Environmental Data Retrieval Standard does not make any changes to this definition.

Profiles of the OGC API — Environmental Data Retrieval Standard have additional requirements governing which Conformance Classes and identifiers must be included in this resource.

REQUIREMENT 13	
IDENTIFIER	`/req/core/api`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* specify the versions of OpenAPI that an implementation *SHALL* support.

NOTE 1: OpenAPI 3.0 and OpenAPI 3.1 are two distinct Conformance Classes in the OGC API-EDR Standard. This requirement can be addressed in a Profile by including the appropriate conformance classes at {root}/conformance.

NOTE 2: Get guidance from the OGC Naming Authority on valid URIs for Profiles.

7.3. Spatio-temporal and Information Resources

Table 2 — Spatialtemporal and Information Resource Paths

PATH TEMPLATE	METHOD	RESOURCE
{root}/collections	GET	Metadata describing the Collections of data available from this API.
{root}/collections/{collectionId}	GET	Metadata describing the `Collection` of data which has the unique identifier {collectionId}

Where:

- {root} = Base URI for the API server
- {collectionId} = an identifier for a specific `Collection` of data

7.3.1. Collections

OGC API implementations typically organize their geospatial resources into collections. Information about those Collections is accessed through the /collections path and the http://www.opengis.net/def/rel/ogc/1.0/data link relation.

Path = {root}/collections

Dependencies

- OGC API — Common — Part 2: Geospatial Data
- OGC API — Environmental Data Retrieval Standard

The Collections resource is initially defined in the `Collections` conformance class of the OGC API — Common — Part 2 Standard. The OGC API — Environmental Data Retrieval Standard Standard does not make any changes to this definition.

An API may support multiple collections. Additional requirements address how the Profile should document requirements at the per-collection level as well as on the landing page (where appropriate)

NOTE: A service may consist of multiple `Collections`. While there may be common rules for all `Collections`, a profile should be able to support different rules depending on the `Collection`.

REQUIREMENT 14	
IDENTIFIER	`/req/core/requirements-set`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
A	The profile SHALL clearly define which collection a requirement applies to.
B	The profile SHALL consist of a set of requirements for a `Collection` and (if the `Collection` supports `Instances`) the `Instances` of the `Collection`. For each of the attributes listed, if it is in the `Collection` (or instance), there SHALL be a requirement to define it.
C	A profile MAY include requirements for the landing page.
D	A profile MAY include requirements for multiple collections.

7.3.2. Collection Description

Each resource `Collection` is described by a set of metadata. That metadata can be accessed directly using the /collections/{collectionId} path and as an entry in the Collections property of the /collections response.

Path:

- {root}/collections (returns metadata for every `Collection`)
- {root}/collections/{collectionId} (returns metadata for the specified `Collection`)

Dependencies

- OGC API — Common — Part 2: Geospatial Data
- OGC API — Environmental Data Retrieval Standard

7.3.2.1. Collection ID parameter restictions

REQUIREMENT 15	
IDENTIFIER	`/req/core/collectionid`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* the valid values of the `collectionId` parameter, then:
A	The Profile *SHALL* specify the rules that the `collectionId` values must follow.
B	Those rules *SHALL* include a brief description explaining how the `collectionId` is generated.
C	Those rules *SHALL* be specified using either: • identifier string or • Regular expression defining valid string patterns.

7.3.2.2. Extent property restrictions

The `Collection` metadata includes an Extent property which defines a spatial-temporal envelope that encompasses the geospatial data in the `Collection`.

REQUIREMENT 16

IDENTIFIER	`/req/core/extent`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* define a requirement defining a `bbox` which specifies the minimum spatial bounds that *SHALL* be supported

RECOMMENDATION 3

IDENTIFIER	`/rec/core/extent`
STATEMENT	A requirement *SHOULD* be defined specifying the rules for defining the `Collection` extent.

REQUIREMENT 17

IDENTIFIER	`/req/core/extent-spatial`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
NOTE	Regular expressions could be used to restrict reference system definitions to WKT2 or EPSG values
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard supports Extents with spatial dimensions, then:
A	The Profile *SHALL* specify the rules for the spatial Coordinate Reference System (CRS).
B	Those rules *SHALL* be specified using either: • Enumerated list of valid CRS values • Regular expression defining valid CRS string patterns.

PERMISSION 1

IDENTIFIER	`/per/core/extent-spatial`
STATEMENT	Regular expressions *MAY* be used to restrict reference system definitions to WKT2 or EPSG values

REQUIREMENT 18

IDENTIFIER	`/req/core/extent-temporal`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard supports Extents with a temporal dimension, then:
A	The Profile *SHALL* specify the rules for expressing the Temporal Reference System (TRS).
B	Those rules *SHALL* be specified using either: • Enumerated list of valid TRS values • Regular expression defining valid TRS string patterns.

RECOMMENDATION 4

IDENTIFIER	`/rec/core/extent-temporal`
STATEMENT	A requirement *SHOULD* be defined specifying the minimum temporal bounds that *SHALL* be supported

REQUIREMENT 19

IDENTIFIER	`/req/core/extent-vertical`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard supports Extents with a vertical dimension, then:
A	The Profile *SHALL* specify the rules for expressing the Vertical Reference System (VRS).
B	Those rules *SHALL* be specified using either: • Enumerated list of valid VRS values • Regular expression defining valid VRS string patterns.

RECOMMENDATION 5

IDENTIFIER	`/rec/core/extent-vertical`
STATEMENT	A requirement *SHOULD* be defined specifying the minimum vertical bounds that *SHALL* be supported

REQUIREMENT 20	
IDENTIFIER	`/req/core/extent-custom`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard supports Extents with a custom dimension, then:
A	The Profile *SHALL* specify the rules for expressing the custom dimension.
B	Those rules *SHALL* be specified using either: • A custom dimension id • A custom dimension reference value • A custom interval value • An enumerated list of valid custom dimension values

RECOMMENDATION 6	
IDENTIFIER	`/rec/core/extent-custom`
STATEMENT	A requirement *SHOULD* be defined specifying the minimum bounds of custom extents that *SHALL* be supported

7.4. Query Resources

Table 3 — Query Resource Paths

PATH TEMPLATE	METHOD	RESOURCE
{root}/collections/{collectionId}/{queryType}	GET, POST (Optional)	Retrieve data according to the query pattern from a `Collection` with the unique identifier {collectionId}
{root}/collections/{collectionId}/instances	GET	Retrieve metadata about Instances of a `Collection`
{root}/collections/{collectionId}/instances/{instanceId}	GET	Retrieve metadata from a specific `Instance` of a `Collection` with the unique identifiers{collectionId} and {instanceId}
{root}/collections/{collectionId}/instances/{instanceId}/{queryType}	GET, POST (Optional)	Retrieve data according to the query pattern from a specific `Instance` of a `Collection` with the unique identifiers {collectionId} and {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

Path = {root}/collections/{collectionId}/{queryType}

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 21	
IDENTIFIER	`/req/core/data-query`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard Standard *SHALL* require definition of the supported data queries.
A	The data_queries definitions *SHALL* specify which data queries a service supports. This can be defined as follows: • Enumerated list of query types
B	Each data_query type listed *SHALL* have a requirement definition.

7.4.1. Parameters

The following parameters are supported by all OGC API-EDR queries.

7.4.1.1. Output Format parameter

Also known as the -f parameter.

Data format for the output data (available options are listed in the Collections response).

REQUIREMENT 22	
IDENTIFIER	`/req/core/output-format`

REQUIREMENT 22	
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	For every output_format specified in any of the data_query enumerated lists, there *SHALL* be a requirement which defines the schema or structure of the data (depending on the format).

RECOMMENDATION 7	
IDENTIFIER	`/rec/core/output-format`
STATEMENT	The recommended definition approaches are as follows: • JSON – Link to a JSON Schema definition • XML – Link to a XML Schema definition • CSV, TSV, PSV, SSV – Link to a definition based on the CSV on the web recommendations available from the CSV on the Web Working Group. • Other types (e.g. binary file types) – Link to a description of the format

7.4.2. Area Query

The Area query returns data within the polygon defined by the coords parameter. Logic for identifying the best match for the requested area will depend on the Collection and is at the discretion of the query service implementer.

Path = {root}/collections/{collectionId}/area

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 23	
IDENTIFIER	`/req/core/data-query-area`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* data queries by making the Area query mandatory, then:
A	The Profile *SHALL* include a requirement mandating the `Area` query.
B	The `Area` query requirement *SHALL* specify the following: • Enumerated list of output_format types • The default_output_format

REQUIREMENT 23	
	• Enumerated list of crs_details values • Enumerated list of the operations that the query supports (i.e. GET, POST)
C	The `Area` requirement *MAY* specify which OGC API-EDR `Area` query parameters are mandatory for compliance with the profile.

7.4.3. Corridor Query

The Corridor query returns data along and around the path defined by the coords parameter. Logic for identifying the best match for the requested corridor will depend on the `Collection` and is at the discretion of the query service implementer.

Path = {root}/collections/{collectionId}/corridor

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 24	
IDENTIFIER	`/req/core/data-query-corridor`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* data queries by making the `Corridor` query mandatory, then:
A	The Profile *SHALL* include a requirement mandating the `Corridor` query.
B	The `Corridor` requirement *SHALL* specify the following: • Enumerated list of output_format types • The default_output_format • Enumerated list of crs_details values • Enumerated list of width-units values • Enumerated list of height-units values • Enumerated list of the operations that the query supports (i.e. GET, POST)
C	The `Corridor` requirement *MAY* specify which OGC API-EDR `Corridor` query parameters are mandatory for compliance with the profile.

7.4.4. Cube Query

The Cube query returns a data cube defined by the bbox and z parameters.

Path = {root}/collections/{collectionId}/cube

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 25	
IDENTIFIER	`/req/core/data-query-cube`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* data queries by making the `Cube` query mandatory, then:
A	The Profile *SHALL* include a requirement mandating the `Cube` query.
B	The `Cube` query requirement *SHALL* specify the following: • Enumerated list of output_format types • The default_output_format • Enumerated list of crs_details values • Enumerated list of the operations that the query supports (i.e. GET, POST)
C	The `Cube` requirement *MAY* specify which OGC API-EDR `Cube` query parameters are mandatory for compliance with the profile.

7.4.5. Locations Query

The Locations query returns data for the named location. Logic for identifying the best match for the coordinate will depend on the `Collection` and is at the discretion of the query service implementer. If a location id is not defined the API SHALL return a GeoJSON features array of valid location identifiers, the schema of the GeoJSON response SHOULD be defined in the OpenAPI definition of the EDR service.

Path = {root}/collections/{collectionId}/locations

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 26	
IDENTIFIER	`/req/core/data-query-locations`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* data queries by making the Locations query mandatory, then:

REQUIREMENT 26	
A	The Profile *SHALL* include a requirement mandating the `Locations` query.
B	The `Locations` query requirement *SHALL* specify the following: • Enumerated list of output_format types • The default_output_format • Enumerated list of crs_details values • Enumerated list of the operations that the query supports (i.e. GET, POST)
C	The `Locations` requirement *MAY* specify which OGC API-EDR `Locations` query parameters are mandatory for compliance with the profile.
D	The `Locations` query requirement *MAY* specify a list of required `locationId` values.

7.4.6. Instances Query

Having multiple versions or `Instances` of the same `Collection`, where the same information is reprocessed or regenerated is not unusal. Although these versions could be described as new Collections the `Instance` query type allows this data to be described as different views of the same `Collection`.

Path = {root}/collections/{collectionId}/instances

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 27	
IDENTIFIER	`/req/core/data-query-instances`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *extends* data queries by making the Instances within a `Collection` queryable, then:
A	`Instances` *SHALL* be defined in the data_queries enumerated list.
B	A *NULL* value *SHALL* be used to indicate that no child instances can be queried.
C	The Profile *SHALL* specify the rules that the `instanceId` values must follow.
D	Those rules *SHALL* include a brief description explaining how the `instanceId` is generated.
E	Those rules *SHALL* be specified using either: • identifier string

REQUIREMENT 27	
	• Regular expression defining valid string patterns.
F	The Profile *MAY* define a `InstanceId` string which represents either a default or the latest value of `InstanceId`.

7.4.6.1. Parameter queryType

Path — Instance Query {root}/collections/{collectionId}/instances/{instanceId}/{queryType}

7.4.7. Items Query

Paths: * {root}/collections/{collectionId}/items * {root}/collections/{collectionId}/items/{itemid}

Dependencies

- GC API — Features — Part 1: Core
- OGC API — Environmental Data Retrieval Standard

The EDR API Items query is like an OGC API — Features Part 1: Core endpoint. It offers some basic compatibility with API-Features for convenience. It may be used to catalog pre-existing EDR sampling features. The pre-existence of an EDR sampling feature may be because a particular query has been cached for later use, such as a monitoring location. Or there may be a catalog of spatio-temporal sampling features such as domains of anomalies in a dataset. A GeoJSON-compatible JSON-Schema is specified to document an EDR API query endpoint and valid query parameters including time range, parameters, and spatial characteristics. A service can define a custom GeoJSON schema in the OpenAPI definition for the service, with the default being the edr-geojson schema if no alternative is documented.

7.4.7.1. ItemID parameter

If an itemId is not specified, the query will return a list of the available itemId's. This behavior is specified in OGC API — Features.

7.4.7.2. Limit parameter

Paging restrictions (limit parameter provided in the request, multi-page response).

REQUIREMENT 28	
IDENTIFIER	`/req/core/paging-support`

REQUIREMENT 28	
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard is *extended* to support paging, then:
A	A requirement *SHALL* be created for each combination of query pattern and output format that must support paging.
B	Each paging requirement *SHALL* specify the default number of items to return per page request.

7.4.8. 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` and is at the discretion of the query service implementer.

Path = {root}/collections/{collectionId}/positions

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 29	
IDENTIFIER	`/req/core/data-query-position`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* data queries by making the `Position` query mandatory, then:
A	The Profile *SHALL* include a requirement mandating the `Position` query.
B	The `Position` query requirement *SHALL* specify the following: • Enumerated list of output_format types • The default_output_format • Enumerated list of crs_details values • Enumerated list of the operations that the query supports (i.e. GET, POST)
C	The `Position` query requirement *SHALL* also specify the logic used in selecting the data returned by the response, i.e. exact, nearest neighbour, most representative or interpolated.
D	The `Position` requirement *MAY* specify which OGC API-EDR `Position` query parameters are mandatory for compliance with the profile.

7.4.9. Radius Query

The Radius query returns data within the defined radius of the requested coordinate.

Path = {root}/collections/{collectionId}/radius

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 30	
IDENTIFIER	`/req/core/data-query-radius`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* data queries by making the `Radius` query mandatory, then:
A	The Profile *SHALL* include a requirement mandating the `Radius` query.
B	The `Radius` query requirement *SHALL* specify the following: • Enumerated list of output_format types • The default_output_format • Enumerated list of crs_details values • Enumerated list of within_units values • Enumerated list of the operations that the query supports (i.e. GET, POST)
C	The `Radius` requirement *MAY* specify which OGC API-EDR `Radius` query parameters are mandatory for compliance with the profile.

7.4.10. Trajectory Query

The Trajectory query returns data along the path defined by the coords parameter. Logic for identifying the best matches for the requested trajectory will depend on the `Collection` and is at the discretion of the query service implementer.

Path = {root}/collections/{collectionId}/trajectory

Dependencies: OGC API — Environmental Data Retrieval Standard

REQUIREMENT 31	
IDENTIFIER	`/req/core/data-query-trajectory`

REQUIREMENT 31	
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* data queries by making the `Trajectory` query mandatory, then:
A	The Profile *SHALL* include a requirement mandating the `Trajectory` query.
B	The `Trajectory` query requirement *SHALL* specify the following: • Enumerated list of output_format types • The default_output_format • Enumerated list of crs_details values • Enumerated list of the operations that the query supports (i.e. GET, POST)
C	The `Trajectory` requirement *MAY* specify which OGC API-EDR `Trajectory` query parameters are mandatory for compliance with the profile.

7.5. General Requirements

7.5.1. HTTP Status Codes

HTTP response

- Response status codes

REQUIREMENT 32	
IDENTIFIER	`/req/core/status-codes`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	A Profile of the OGC API — Environmental Data Retrieval Standard *SHALL* require that the definitions of all http status codes SHALL be provided.
A	These definitions *SHALL* provide the following: • A description of the cause of the response. • A JSON schema for the message body structure

7.5.2. Links

- Response links

REQUIREMENT 33	
IDENTIFIER	`/req/core/links`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *restricts* valid responses to only those which include links, then:
A	The Profile *SHALL* require that link objects are included in a response.
B	The Profile *SHALL* define the required link objects in full.
C	The link objects *SHALL* require that the `href`, `rel` and `type` attributes are populated.

STATEMENT	for example:
	`{`
	`"href": "https://creativecommons.org/licenses/by-nc/4.0/",`
	`"rel": "licence",`
	`"type": "text/html"`
	`},`
	`{`
	`"href": "https://docs.ogc.org/cs/21-069r2/21-069r2.html",`
	`"rel": "service-doc",`
	`"title": "CoverageJSON Community Standard v1.0"`
	`"type": "text/html"`
	`}`

7.5.3. Asynchronous Queries

While Web protocols typically use request-response operations, there is also support for asychonous operations.

HTTP Asynchrouous — This requirement address the use of HTTP asynchronous operations such as Webhooks and Callbacks.

REQUIREMENT 34	
IDENTIFIER	`/req/core/asynchronous`

REQUIREMENT 34	
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile *extends* the OGC API — Environmental Data Retrieval Standard with support for asynchronous opperations, then:
A	Requirements *SHALL* be defined for each query type that is asynchronous
B	Each asynchronous query type requirement *SHALL* define the HTTP Status Code and provide a message schema and text used to inform the user that the response is asynchronous.
C	Each asynchronous query type requirement *SHALL* document the mechanism for delivering the result of the asynchronous query.

PERMISSION 2	
IDENTIFIER	`/per/core/asynchronous`
STATEMENT	The documentation of the mechanism for delivering the result of the asynchronous query *MAY* be provided through a link to an external document.

Publish-Subscribe — This requirement addresses the use of Publish-Subscribe protocols. These are protocols supported in addition to HTTP.

REQUIREMENT 35	
IDENTIFIER	`/req/core/pubsub`
INCLUDED IN	Requirements class 1: `http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core`
STATEMENT	If a Profile of the OGC API — Environmental Data Retrieval Standard *extends* the supported operations to include Publish-Subscribe operations, then:
A	Support for the OGC API — Environmental Data Retrieval — Part 2: Publish-Subscribe workflow Standard *SHALL* be required.
B	The pubsub requirement *SHALL* specify the channels that *SHALL* be supported
C	The pubsub requirement *SHALL* specify the payloads that a pubsub channel *SHALL* support

8

MEDIA TYPES FOR ANY DATA ENCODING(S)

8 MEDIA TYPES FOR ANY DATA ENCODING(S)

A section describing the MIME-types to be used is mandatory for any standard involving data encodings. If no suitable MIME type exists in http://www.iana.org/assignments/media-types/index.html then this section may be used to define a new MIME type for registration with IANA.

A

ANNEX A (INFORMATIVE) CONFORMANCE CLASS ABSTRACT TEST SUITE (NORMATIVE)

A ANNEX A (INFORMATIVE) CONFORMANCE CLASS ABSTRACT TEST SUITE (NORMATIVE)

A.1. Conformance Class Core

CONFORMANCE CLASS A.1: CONFORMANCE CLASS 'CORE'	
IDENTIFIER	http://www.opengis.net/spec/ogcapi-edr-3/1.0/conf/conf-class-core
REQUIREMENTS CLASS	Requirements class 1: http://www.opengis.net/spec/ogcapi-edr-3/1.0/req/req-class-core
CONFORMANCE TESTS	Abstract test A.1: /conf/core/modspec Abstract test A.2: /conf/core/publishing Abstract test A.3: /conf/core/api Abstract test A.4: /conf/core/edr-conformant Abstract test A.5: /conf/core/root Abstract test A.6: /conf/core/requirements-set Abstract test A.7: /conf/core/parameter-names Abstract test A.8: /conf/core/collectionid Abstract test A.9: /conf/core/extent Abstract test A.10: /conf/core/output-format Abstract test A.11: /conf/core/paging-support Abstract test A.12: /conf/core/status-codes Abstract test A.13: /conf/core/links Abstract test A.14: /conf/core/data-query Abstract test A.15: /conf/core/data-query-area Abstract test A.16: /conf/core/data-query-corridor Abstract test A.17: /conf/core/data-query-cube Abstract test A.18: /conf/core/data-query-instances Abstract test A.1-19: /conf/core/data-query-locations Abstract test A.19: /conf/core/data-query-position Abstract test A.20: /conf/core/data-query-radius Abstract test A.21: /conf/core/data-query-trajectory Abstract test A.22: /conf/core/asynchronous

CONFORMANCE CLASS A.1: CONFORMANCE CLASS 'CORE'

Abstract test A.23: `/conf/core/pubsub`

ABSTRACT TEST A.1

IDENTIFIER	`/conf/core/modspec`
REQUIREMENT	Requirement 1: `/req/core/modspec`
TEST PURPOSE	Validate that the profile is compliant with the OGC Modular Specification.
TEST METHOD	
STEP	Verify that the profile is compliant with the OGC Modular Specification.

ABSTRACT TEST A.2

IDENTIFIER	`/conf/core/publishing`
REQUIREMENT	Requirement 11: `/req/core/publishing`
TEST PURPOSE	Validate that an OpenAPI description of the profile is available
TEST METHOD	
STEP	Verify that an OpenAPI document describing the profile exists Verify that the OpenAPI document is compliant with the OpenAPI 3.1 specification Verify that the OpenAPI document includes schemas for the `Collection` and `Instance` query responses Verify that the `Collection` and `Instance` schemas include all of the queries defined in the data-query requirement Verify that the `Collection` and `Instance` schema include Path objects for each of the data_queries defined in the profile. Verify that the Path Object define Items for the HTTP operations defined by the data query requirements in the profile Verify that the Path Item definitions include the enumerated lists defined by the data query requirements in the profile

ABSTRACT TEST A.3

IDENTIFIER	`/conf/core/api`
REQUIREMENT	Requirement 13: `/req/core/api`
TEST PURPOSE	Validate that the profile includes a requirement specifying the OpenAPI versions that implementations shall support.
TEST METHOD	
STEP	Verify that the profile includes a requirement defining the OpenAPI versions that implementations shall support. Verify that the profile includes a conformance class testing that OpenAPI versions are supported.

ABSTRACT TEST A.4

IDENTIFIER	`/conf/core/edr-conformant`
REQUIREMENT	Requirement 2: `/req/core/edr-conformant`
TEST PURPOSE	Validate that the Profile Standard requires that all implementations demonstrate conformance with the OGC API-EDR Standard.
TEST METHOD	
STEP	Verify that the profile specifies that regardless of any profile specifc requirements, implementations shall function as OGC API-EDR Part 1 — Core compliant API's.

NOTE: this "purpose" requires more specificity.

ABSTRACT TEST A.5

IDENTIFIER	`/conf/core/root`
REQUIREMENT	Requirement 4: `/req/core/root`
TEST PURPOSE	Validate that the profile defines the service landing page
TEST METHOD	
STEP	Verify that the profile defines a `Title` for the service

ABSTRACT TEST A.5

	Verify that the profile defines the `Links` required for the service
	Verify that each link defined for the service has a `href` and `rel` attribute.
	Verify that any `Description` attributes in the profile have a defined value.
	Verify that any `Attribution` attributes in the profile have a defined value.
	Verify that any `Keywords` attributes in the profile have defined values.
	Verify that any `Provider` attributes in the profile have `name` and `url` attributes.
	Verify that the providers `name` and `url` attributes have defined values.
	Verify that any `Contact` attributes in the profile have an `email` attributes.
	Verify that contact `email` attributes has a defined value.

ABSTRACT TEST A.6

IDENTIFIER	`/conf/core/requirements-set`
REQUIREMENT	Requirement 14: `/req/core/requirements-set`
TEST PURPOSE	Validate that the profile defines at least one requirement for a collection
TEST METHOD	
STEP	Verify that the profile contains at least one requirement which applies to a collection. Verify that collection requirements in the profile clearly identify which collection a requirement applies to.

ABSTRACT TEST A.7

IDENTIFIER	`/conf/core/parameter-names`
REQUIREMENT	Requirement 3: `/req/core/parameter-names`
TEST PURPOSE	Validate that the parameter_names requirement is correctly defined
TEST METHOD	

ABSTRACT TEST A.7	
STEP	Verify that the parameter_names requirement defines a set of parameter objects
	Verify that each parameter object in the set has a unique key
	Verify that each parameter object in the set has a type value of “Parameter”
	Verify that each parameter object in the set has a description attribute
	Verify that each parameter object in the set has a unit attribute
	Verify that each unit object a label attribute
	Verify that each unit object a symbol attribute
	Verify that each symbol object a value attribute
	Verify that each symbol object a type attribute
	Verify that each parameter object in the set has an observedProperty attribute
	Verify that each observedProperty object has an id attribute
	Verify that each observedProperty object has a label attribute

ABSTRACT TEST A.8	
IDENTIFIER	`/conf/core/collectionid`
REQUIREMENT	Requirement 15: `/req/core/collectionid`
TEST PURPOSE	Validate that a `collectionId` requirement is correctly defined.
TEST METHOD	
STEP	Verify that a `collectionId` requirement includes an explanation of how a `collectionId` is derived.
	Verify that a `collectionId` requirement specifies either an identifier string or a Regular expression.

ABSTRACT TEST A.9

IDENTIFIER	`/conf/core/extent`
REQUIREMENT	Requirement 16: `/req/core/extent`
TEST PURPOSE	Validate the profile extent requirements are defined correctly.
TEST METHOD	
STEP	Verfiy that the profile has a requirement specifying the spatial extent of the `Collection`. Verfiy that the profile specifies either an enumerated list of CRS values or a Regular expression definition for valid CRS values. Verify that Temporal extent requirements specify either an enumerated list of TRS values or a Regular expression definition for valid TRS values. Verify that Vertical extent requirements specify either an enumerated list of VRS values or a Regular expression definition for valid VRS values. Verify that Custom extent requirements specify the id of the custom dimension. Verify that Custom extent requirements specify the custom dimension reference value. Verify that Custom extent requirements specify the custom dimension interval value. Verify that Custom extent requirements specify an enumerated list of the valid custom dimension values.

ABSTRACT TEST A.10

IDENTIFIER	`/conf/core/output-format`
REQUIREMENT	Requirement 22: `/req/core/output-format`
TEST PURPOSE	Validate that the profile correctly defines output-formats
TEST METHOD	
STEP	Verify that the profile contains references to the schemas or format descriptions for all output-formats defined in the data queries requirements

ABSTRACT TEST A.11

IDENTIFIER	`/conf/core/paging-support`
REQUIREMENT	Requirement 28: `/req/core/paging-support`
TEST PURPOSE	Validate that paging support requirements are correctly defined
TEST METHOD	
STEP	Verify that a paging support requirement defines which query types will support output paging Verify that a paging support requirement identifies which output formats support paging in paging-capable queries Verify that a paging support requirement defines the default number of items to return per page request

ABSTRACT TEST A.12

IDENTIFIER	`/conf/core/status-codes`
REQUIREMENT	Requirement 32: `/req/core/status-codes`
TEST PURPOSE	Validate that the profile defines any HTTP status code responses required by a service.
TEST METHOD	
STEP	Verify that the profile contains HTTP status code definitions Verify that any HTTP status code definition has a description value. Verify that each HTTP status code definition includes a JSON schema for the response body.

ABSTRACT TEST A.13

IDENTIFIER	`/conf/core/links`
REQUIREMENT	Requirement 33: `/req/core/links`
TEST PURPOSE	Validate that link requirements are correctly defined

ABSTRACT TEST A.13

TEST METHOD	
STEP	Verify that link requirements define a `href` value
	Verify that link requirements define a `rel` value
	Verify that link requirements define a `type` value

ABSTRACT TEST A.14

IDENTIFIER	`/conf/core/data-query`
REQUIREMENT	Requirement 21: `/req/core/data-query`
TEST PURPOSE	Validate that a profile correctly defines the query types that a service shall support.
TEST METHOD	
STEP	Verify that the profile defines an enumerated list of the queries that a service shall support.
	Verify that for each definition in the enumerated list there is a corresponding data query requirement in the profile.

ABSTRACT TEST A.15

IDENTIFIER	`/conf/core/data-query-area`
REQUIREMENT	Requirement 23: `/req/core/data-query-area`
TEST PURPOSE	Verify that Area query requirements are defined correctly in a profile.
TEST METHOD	
STEP	Verify an Area query requirement defines an enumerated list of output_format types.
	Verify an Area query requirement defines the default output_format.
	Verify an Area query requirement defines an enumerated list of crs_details values.
	Verify an Area query requirement defines a list of supported HTTP operations.

ABSTRACT TEST A.16

IDENTIFIER	`/conf/core/data-query-corridor`
REQUIREMENT	Requirement 24: `/req/core/data-query-corridor`
TEST PURPOSE	Verify that Corridor query requirements are defined correctly in a profile.
TEST METHOD	
STEP	Verify that a Corridor requirement defines an enumerated list of output_format types. Verify that a Corridor requirement defines the default output_format. Verify that a Corridor requirement defines an enumerated list of crs_details values. Verify that a Corridor requirement defines an enumerated list of width-units values. Verify that a Corridor requirement defines an enumerated list of height-units values. Verify that a Corridor requirement defines a list of supported HTTP operations.

ABSTRACT TEST A.17

IDENTIFIER	`/conf/core/data-query-cube`
REQUIREMENT	Requirement 25: `/req/core/data-query-cube`
TEST PURPOSE	Verify that Cube query requirements are defined correctly in a profile.
TEST METHOD	
STEP	Verify a Cube query requirement defines an enumerated list of output_format types. Verify a Cube query requirement defines the default output_format. Verify a Cube query requirement defines an enumerated list of crs_details values. Verify a Cube query requirement defines a list of supported HTTP operations.

ABSTRACT TEST A.18

IDENTIFIER	`/conf/core/data-query-instances`

ABSTRACT TEST A.18	
REQUIREMENT	Requirement 27: `/req/core/data-query-instances`
TEST PURPOSE	Validate than a profile correctly defines support for `instances`
TEST METHOD	
STEP	Verify that the data queries requirement enumerated list includes an entry for `instances`. Verify that an `instanceId` requirement includes an explanation of how an `instanceId` is derived. Verify that an `instanceId` requirement contains either an identifier string or a Regular expression rule for defining the `instanceId`.

ABSTRACT TEST A.19	
IDENTIFIER	`/conf/core/data-query-position`
REQUIREMENT	Requirement 29: `/req/core/data-query-position`
TEST PURPOSE	Verify that Position query requirements are defined correctly in a profile.
TEST METHOD	
STEP	Verify a Position query requirement defines the logic in used selecting the appropriate data response. Verify a Position query requirement defines an enumerated list of output_format types. Verify a Position query requirement defines the default output_format. Verify a Position query requirement defines an enumerated list of crs_details values. Verify a Position query requirement defines a list of supported HTTP operations.

ABSTRACT TEST A.20	
IDENTIFIER	`/conf/core/data-query-radius`
REQUIREMENT	Requirement 30: `/req/core/data-query-radius`

ABSTRACT TEST A.20	
TEST PURPOSE	Verify that Radius query requirements are defined correctly in a profile.
TEST METHOD	
STEP	Verify a Radius query requirement defines an enumerated list of output_format types. Verify a Radius query requirement defines the default output_format. Verify a Radius query requirement defines an enumerated list of crs_details values. Verify a Radius query requirement defines an enumerated list of within_units values. Verify a Radius query requirement defines a list of supported HTTP operations.

ABSTRACT TEST A.21	
IDENTIFIER	`/conf/core/data-query-trajectory`
REQUIREMENT	Requirement 31: `/req/core/data-query-trajectory`
TEST PURPOSE	Verify that Trajectory query requirements are defined correctly in a profile.
TEST METHOD	
STEP	Verify a Trajectory query requirement defines an enumerated list of output_format types. Verify a Trajectory query requirement defines the default output_format. Verify a Trajectory query requirement defines an enumerated list of crs_details values. Verify a Trajectory query requirement defines a list of supported HTTP operations.

ABSTRACT TEST A.22	
IDENTIFIER	`/conf/core/asynchronous`
REQUIREMENT	Requirement 34: `/req/core/asynchronous`
TEST PURPOSE	Validate that an asynchronous query support requirement in correctly defined
TEST METHOD	

ABSTRACT TEST A.22	
STEP	Verify that all asynchronous query requirements in a profile define a HTTP Status Code for the asynchronous response. Verify that all asynchronous query requirements in a profile define a message schema for the asynchronous response. Verify that all asynchronous query requirements in a profile define the text for the asynchronous response. Verify that all asynchronous query requirements in a profile define the mechanism for delivering the asynchronous query result.

ABSTRACT TEST A.23	
IDENTIFIER	`/conf/core/pubsub`
REQUIREMENT	Requirement 35: `/req/core/pubsub`
TEST PURPOSE	Validate Pub/Sub requirements
TEST METHOD	
STEP	Verify that the profile specifes the OGC API-EDR Part 2 conformance class Verify that the profile defines the channels services shall implement. Verify that the profile defines the message payloads required for each of the channels in the service. Verify the profile contains an AsyncAPI definition that describes the channels and their messages.