Community Standard Justification: 21-040r2

TITLE: Justification for CoverageJSON as an OGC Community Standard

CONTRIBUTOR

Name: Chris Little

Organization: Met Office

Address: FitzRoy Road, Exeter, Devon, EX1 3PB, UK

Phone: +44(0)330 135 0442

Email: chris.little@metoffice.gov.uk

DATE: 2021-05-27

# Introduction

This document provides a justification to the OGC Technical Committee (TC) for consideration of CoverageJSON as a Community Standard. This justification, along with the submitted candidate Community standard, will form the basis for TC review and vote to approve the start of a Work Item as the first step in the Community Standard process for this standard.

The submitters agree to abide by the TC Policies and Procedures and OGC Intellectual Property Rights Policy ([http://www.opengeospatial.org/ogc/policies)](http://www.opengeospatial.org/ogc/policies%29) during the processing of this submission.

Once approved, the Community Standard Work Item defined by this document is valid for six (6) months.

# Overview of proposed submission

## Primary Benefits

CoverageJSON is a format for publishing geo-temporal data to the Web. It is used for encoding coverage data such as multi-dimensional grids, time series, and vertical profiles, distinguished by the geometry of their spatio-temporal domain. The format supports the efficient download of useful quantities of data to lightweight clients, such as browsers and mobile applications, from data stores, enabling local manipulation of the data in a format familiar to, and popular with, web developers, and readily usable, for example, by science researchers. It uses linked-data (JSON-LD) to reduce data payload volumes. Simplicity, machine and human readability of the format were the primary design goals.

It was developed in 2015 at the University of Reading, as part of MELODIES (Maximizing the Exploitation of Linked Open Data in Enterprise and Science), a EU Framework 7 project [1], inspired by a demonstration by Joan Masó of CREAF.

During the project, it was observed that there was a gap in the capabilities available to the developers of websites and apps who wish to consume scientific data. Existing data formats (e.g. NetCDF, HDF, GRIB, XML) were either highly complex or too inefficient to be used in these environments. Therefore the CoverageJSON format was developed for encoding many kinds of scientific and Earth Observation data in a manner that is friendly for web and app developers. The format was based on the concepts and standards from ISO and OGC and the specification is published openly on the web [2]. The specification is supported by a Cookbook [3], plus a suite of open source tools [4] for producing and consuming the format. Great interest in the format has already been shown by the user community, who are also encouraged to contribute to its development. CoverageJSON was also discussed as a possible future standard for INSPIRE compliance, as it was perceived to be simpler than existing mechanisms.

## Additional Benefits

2.2.1 The CoverageJSON format also supports an n-dimensional tiled structure for data, where this is considered useful, allowing large coverages to be partitioned among separate JSON documents.

2.2.2. To a limited extent, a CoverageJSON document can be converted into RDF through the use of a JSON-LD context header, thus enabling conversions between domain specific, well-established, binary formats like NetCDF and the tools of the semantic web. This approach makes CoverageJSON objects easily discoverable on the Web. Conversion to RDF was not a primary goal, under the assumption that few web developers would require a pure RDF representation of the data, which would be extremely verbose. Also, enabling a full conversion to RDF was hindered by the limitations of JSON-LD V1.0.

2.2.3 CoverageJSON makes frequent use of URIs to denote key concepts, such as Units of Measure, observed properties, Coordinate Reference Systems, domain types and links to other CoverageJSON documents. Clients can make use of these to detect these concepts unambiguously, whether or not they perform a translation to RDF. This approach is one strongly recommended by the OGC/W3C Spatial Data on the Web Best Practices [5] and its underlying Data on the Web Best Practices [6]

2.2.4 Extension Points. Data providers can extend the format in a controlled manner. The possible extensions that can be defined by users include:

* Custom properties (e.g. high-level metadata such as licence information)
* Custom domain types
* Custom data types for axes
* Custom referencing system types (e.g. [HEALPix](http://healpix.jpl.nasa.gov) grids of NASA JPL)
* Different grammars for encoding unit symbols (e.g. UDUNITS)
* Alternative encodings for range values

In each case it is recommended that URIs are used to denote these extensions (and to point to definitions), to avoid the possibility of clashes between extensions.

## Other Factors

2.3.1 The standard has been stable for several years, with a wide-spread international community in the environmental sciences. The current version is labelled ‘V0.2-draft’.

2.3.2 The source is open on GitHub [7].

2.3.3 There are numerous examples and a “Cookbook” [3] to guide developers, with a “Playground” or “sandbox” with working examples [8].

2.3.4 Future improvements have been identified, to use JSON-LD V1.1 which may give better compatibility with CIS.

# Relationship to other OGC standards

3.1 The CoverageJSON format is recommended for interoperability and performance reasons by the OGC API - EDR standard [12], but not normatively.

3.2 At the design stage of the CoverageJSON format, the OGC WCS and CIS standards were consulted for guidance, but not normatively, as JSON design principles were dominant.

3.3 The WCS SWG has agreed to a future work item with a goal to make CIS and CoverageJSON consistent and compatible.

# Alignment with OGC Standards Baseline

CoverageJSON has been demonstrated to be an effective, efficient format, friendly to web and application developers, and therefore consistent with the current OGC API developments.

It can be used to download coverages and collections of coverages, possibly tiled, to lightweight clients.

The overall structure of CoverageJSON is quite close to that of NetCDF [9], which is an OGC Standard and a supported response format for WCS2.1. It consists essentially of a set of orthogonal domain axes that can be combined in different ways. One major difference is that CoverageJSON has an explicit Domain object, whereas in NetCDF the domain is specified implicitly. One consequence of this is that NetCDF files can contain several domains and hence several Coverages. A NetCDF file could therefore be converted to a single Coverage or a Coverage Collection and either stored in CoverageJSON.

CoverageJSON supports the concept of mapping from a spatial, temporal or spatiotemporal domain to feature attribute values where feature attribute types are common to all geographical positions within the domain.

As CoverageJSON is a format for data, it cannot conform to the required operations of ISO19123:2005, though the CoverageJSON components seem sufficient to support any simple continuous or discrete coverages. So in this sense, CoverageJSON is consistent with the OGC Abstract Specification Topic 6 / ISO 19123 Coverage Model [10].

However, there are some differences from the OGC Coverage Implementation Schema (CIS) Standard [11] which is also a specialization of ISO19123.

The main points of difference seem to be:

* CIS uses a different set of rules for gridded and non-gridded data, whereas CoverageJSON uses a single consistent set.
* CIS allows each Coverage to have exactly one CRS, which may be Compound, whereas CoverageJSON allows different CRSs to be applied to different sets of coordinates in the domain (e.g. one CRS for x and y, and another CRS for z).
* The most recent version of CIS defines a JSON encoding that uses a near-literal translation of GML structures into JSON, making it harder to understand. CoverageJSON does not use GML as its starting point and uses structures more appropriate to JSON.

However, a possible evolutionary convergent path has been identified and the WCS SWG has offered to host the work, starting with a comparative review. It is possible that a subsequent CoverageJSON version could be backward compatible, perhaps as a separate conformance class of a full OGC standard, and better aligned with the CIS 1.1 standard. Hopefully, this would lead to a conceptually more consistent format without losing any performance advantages, and the communities of users would see benefit in migration to the later version.

# Evidence of implementation

The following implementations use the proposed Community standard, both servers, clients, and utilities. See [4].

**Implementation name**: Geoserver CoverageJSON output format

**Implementation description**: Geoserver community module to generate CoverageJSON

**Date of most recent version**: 2021-03-22

**Implementation URL**: <https://docs.geoserver.org/stable/en/user/community/cov-json/index.html>

**Is implementation complete**? X **Yes** [ ]  **No**

**Implementation name**: HYRAX CoverageJSON output format

**Implementation description**: OpenDAP server allowing users download data in CoverageJSON

**Date of most recent version**: 1.15.2, 2019-06-21

**Implementation URL**: <https://opendap.github.io/hyrax_guide/Master_Hyrax_Guide.html#Download_and_Install_Hyrax>

**Is implementation complete**? X **Yes** [ ]  **No**

**Implementation name**: THREDDS Data Server CoverageJSON output format

**Implementation description**: Unidata data server allowing users to choose to download data in CoverageJSON, using the ncWMS component below.

**Date of most recent version**: 4.6, 2018-11, **requires Tomcat 7+ and Java 8+**

**Implementation URL**: [[thredds.war](https://www.unidata.ucar.edu/downloads/tds/)] [[source on GitHub](https://github.com/Unidata/thredds)]

**Is implementation complete**? X **Yes** [ ]  **No**

**Implementation name**: covjson-reader

**Date of most recent version**: 2016-10-21

**Implementation description**: CoverageJSON reader written in JavaScript, supporting sub-setting and lazy loading

**Implementation URL**: <https://github.com/Reading-eScience-Centre/covjson-reader>

**Is implementation complete**? X **Yes** [ ]  **No**

**Implementation name**: leaflet-coverage

**Date of most recent version**: 2016-10-27

**Implementation description**: a Javascript Leaflet plugin for visualizing CoverageJSON and other coverage data

**Implementation URL**: <https://github.com/Reading-eScience-Centre/leaflet-coverage>

**Is implementation complete**? X **Yes** [ ]  **No**

**Implementation name**: pycovjson

**Date of most recent version**: 2017-07-17

**Implementation description**: transforms scientific data formats like netCDF into CoverageJSON

**Implementation URL**: <https://github.com/Reading-eScience-Centre/pycovjson>

**Is implementation complete**? **Yes** X **No**

**If not, what portions of the proposed Community standard are implemented?** Only NetCDF currently supported

**Implementation name**: ncWMS

**Date of most recent version**: V2, 2020-09-23

**Implementation description**: a WMS server, written in Java, with CoverageJSON as output format, for visualising and exploring environmental data in a browser. It can be run on a server to make data available over the web or locally for personal use. Uses EDAL library.

**Implementation URL**: <https://github.com/Reading-eScience-Centre/edal-java>

**Is implementation complete**? X **Yes** [ ]  **No**

**Implementation name**: pygeoapi

**Date of most recent version**: 2021-03-25

**Implementation description**: pygeoapi is a Python server implementation of the OGC API suite of standards. pygeoapi implements OGC API – Coverages and OGC API – Environmental Data Retrieval, and provides CoverageJSON as a possible output format.

**Implementation URL**: <https://pygeoapi.io>

**Is implementation complete**? X **Yes** [ ]  **No**

**Implementation name**: webworldwind-covjson

**Date of most recent version**: 2016-09-22

**Implementation description**: a Javascript plugin to NASA’s WebWorldWind for visualizing CoverageJSON and other coverage data

**Implementation URL**: <https://github.com/Reading-eScience-Centre/webworldwind-covjson>

**Is implementation complete**? X **Yes** [ ]  **No**

Several implementations of the OGC API-Environmental Data Retrieval standard generate or consume CoverageJSON. See [10].

# Public availability

Is the proposed Community standard currently publicly available? X **Yes** [ ]  **No**

URL: <https://covjson.org/spec>

# Supporting member(s)

Met Office (Technical Member)

Meteorological Service of Canada (Associate Member)

Unidata UCAR (Technical Member)

University of Reading (Associate/University Member)

US NWS/NOAA (Principal Member)

# Intellectual property rights

Will the contributor retain intellectual property rights? X **Yes** [ ]  **No**

If yes, the contributor will be required to work with OGC staff to properly attribute the submitter’s intellectual property rights. Currently the specification is released under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).

If no, the contributor will assign intellectual property rights to the OGC.

# References

[1] <https://cordis.europa.eu/project/id/603525/reporting>

[2] <https://covjson.org>

[3] <https://covjson.org/cookbook>

[4] <https://covjson.org/tools>

[5] <https://www.w3.org/TR/sdw-bp>

[6] <https://www.w3.org/TR/dwbp>

[7] <https://covjson.org/spec>

[8] <https://covjson.org/playground>

[9] <https://www.unidata.ucar.edu/software/netcdf>

[10] OGC07-011, Abstract Specification Topic 6 [Schema for coverage geometry and functions](https://portal.ogc.org/files/?artifact_id=19820)

[11] OGC09-146r6, Coverage Implementation Schema v1.1 <http://docs.opengeospatial.org/is/09-146r6/09-146r6.html>

[12] <https://github.com/opengeospatial/ogcapi-environmental-data-retrieval/blob/master/implementations.md>