Open Geospatial Consortium

Submission Date: <yyyy-mm-dd>

Approval Date: <yyyy-mm-dd>

Publication Date: <yyyy-mm-dd>

External identifier of this OGC® document: http://www.opengis.net/doc/IS/xx/xx

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

Version: 1.0.0-SNAPSHOT (Editor's draft)

Latest Published Draft: n/a

Category: OGC® Implementation Specification

Editors: Pekka Latvala

OGC API - TJS - Part 1: Core

Document type: OGC® Standard

Document subtype: Interface

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

1. Scope
2. Conformance
3. References
4. Terms and Definitions
    4.1. attribute dataset
    4.2. spatial dataset
5. Conventions and background
    5.1. Identifiers
    5.2. Link relations
        5.2.1. Response Schema for the Link Object
    5.3. Exception messages
        5.3.1. Response Schema for the Exception Messages
6. Requirements Class "Core"
    6.1. Overview
    6.2. Operation Set: Discovery Operations
        6.2.1. API Landing Page
        6.2.2. API Definition
        6.2.3. Declaration of Conformance Classes
    6.3. Operation Set: Data Joining Operations
        6.3.1. Spatial Datasets
        6.3.2. Spatial Dataset Creation
        6.3.3. Spatial Dataset
        6.3.4. Spatial Dataset Update
        6.3.5. Spatial Dataset Delete
        6.3.6. Spatial Dataset Key Fields
        6.3.7. Spatial Dataset Key Field
        6.3.8. Spatial Dataset Key Field Key
        6.3.9. Joins
        6.3.10. Join Attribute Data from a File with a Spatial Dataset on the Server
        6.3.11. Join
        6.3.12. Join Update
        6.3.13. Join Delete
    6.4. Operation Set: Spatial Joining Operations
        6.4.1. Attribute Datasets
        6.4.2. Attribute Dataset Creation
        6.4.3. Attribute Dataset
        6.4.4. Attribute Dataset Update
        6.4.5. Attribute Dataset Delete
        6.4.6. Attribute Dataset Key Fields
        6.4.7. Attribute Dataset Key Field
        6.4.8. Attribute Dataset Key Field Key
        6.4.9. Join Attribute Dataset with a Spatial Dataset File
    6.5. Operation Set: File Joining Operations
        6.5.1. Data Joining from an Attribute Dataset File with a Spatial Dataset File
7. Media Types for any data encoding(s)
    7.1. Operation Sets: discovery operations and data joining operations
    7.2. Outputs for the Joined Ddta in Operation Set: data joining operations
    7.3. Operation Sets: spatial joining operations and file joining operations

Annex A: Abstract Test Suite (Normative)

A.1. Conformance Class "Core"
A.1.1. Landing Page {root}/
A.1.2. API Definiton path {root}/api
A.1.3. Conformance {root}/conformance
A.2. Conformance class: Core / Data Joining
A.2.1. Spatial Datasets {root}/spatialdatasets
A.2.2. Spatial Dataset Creation {root}/spatialdatasets
A.2.3. Spatial Dataset {root}/spatialdatasets/{spatialdatasetid}
A.2.4. Spatial Dataset Update {root}/spatialdatasets/{spatialdatasetid}
A.2.5. Spatial Dataset Delete {root}/spatialdatasets/{spatialdatasetid}
A.2.6. Spatial Dataset keys {root}/spatialdatasets/{spatialdatasetid}/keys
A.2.7. Spatial Dataset key field {root}/spatialdatasets/{spatialdatasetid}/keys/{keyname}
A.2.8. Spatial Dataset key field key {root}/spatialdatasets/{spatialdatasetid}/keys/{keyname}/{key}
A.2.9. Joins {root}/joins
A.2.10. Spatial Dataset joining with Attribute Dataset File {root}/joins
A.2.11. Join {root}/joins/{joinid}
A.2.12. Join Update {root}/joins/{joinid}
A.2.13. Join Delete {root}/joins/{joinid}
A.3. Conformance class: Core / Spatial Joining
A.3.1. Attribute Datasets {root}/attributedatasets
A.3.2. Attribute Dataset Creation {root}/attributedatasets
A.3.3. Attribute Dataset {root}/attributedatasets/{attributedatasetid}
A.3.4. Attribute Dataset Update {root}/attributedatasets/{attributedatasetid}
A.3.5. Attribute Dataset Delete {root}/attributedatasets/{attributedatasetid}
A.3.6. Attribute Dataset keys {root}/attributedatasets/{attributedatasetid}/keys
A.3.7. Attribute Dataset key field {root}/attributedatasets/{attributedatasetid}/keys/{keyname}
A.3.8. Attribute Dataset key field key {root}/attributedatasets/{attributedatasetid}/keys/{keyname}/{key}
A.3.9. Attribute Dataset Joining To Spatial Dataset File {root}//joinattributedataset
A.4. Conformance class: Core / File Joining
A.4.1. File Joining {root}//joinfiles
Annex B: Revision History
Annex C: Bibliography

i. Abstract

This document is the specification for the core module of the OGC API - TJS standard. The core module specifies a service interface that allows non-spatial attribute data to be joined with spatial datasets via common identifiers that are available in both datasets. The TJS core module supports also operations for viewing metadata on attribute and spatial datasets and their keys that are available on the server and operations for adding, updating and deleting the attribute and the spatial datasets. It contains also operations for accessing, updating and deleting the created joins, operation for joining attribute dataset from the server with inputted spatial data files and an operation for joining inputted attribute data files directly with inputted spatial data files.

CAUTION	This is a DRAFT version of the OGC API - TJS standard. This draft is not complete and there are open issues that are still under discussion.

ii. Keywords

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

ogcdoc, OGC document, standard, TJS, API, openapi

iii. Preface

This standard is the result of the work that was executed to renew the OGC implementation standard: OpenGIS® Georeferenced Table Joining Service (TJS) (document nr. 10-070r2), specified in 2010.

This document defines the core module of the OGC API - TJS standard. The specification is a multi-part document that can be extended by specifying extension modules to the core module.

This document does not suggest any updates to the OGC Abstract Specification

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):

- National Land Survey of Finland

v. Submitters

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

Name	Affiliation
Pekka Latvala *(editor)*	National Land Survey of Finland

1. Scope

This OpenGIS® standard defines the core module for the OGC API - TJS specification. The core module specifies a RESTful

service interface that contains 4 operation sets: *discovery operations*, *data joining operations*, *spatial joining operations* and *file joining operations*.

The operation set *discovery operations* contains operations for obtaining general information on the TJS implementation. It includes operations for accessing the API landing page, the API definition file and information on the service's conformance to the TJS standard.

The operation set *data joining operations* contains functionalities for retrieving metadata and key values on the spatial datasets that are available on the server and adding, updating and deleting them. It contains also operations for joining attribute data from attribute data files with these spatial datasets and accessing, updating and deleting the created joins.

The operation set *spatial joining operations* contains functionalities for retrieving metadata and key values on the attribute datasets that are available on the server and adding, updating and deleting them. It contains also an operation for joining the attribute datasets that are on the server with spatial data files.

The operation set *file joining operations* can be used for joining attribute data from inputted attribute data files directly with inputted spatial data files.

The core module contains support for attribute data files in the csv format and for spatial data files in the GeoJSON format. The support for other data formats can be defined in potential extension modules.

---

2. Conformance

This standard defines 1 requirement class: "core".

Requirements for 1 standardization target types are considered:

- Web services

This standard defines 4 operations classes *discovery operations*, *data joining operations*, *spatial joining operations* and *file joining operations*.

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

---

3. 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.

- Internet Engineering Task Force (IETF). RFC 2616, **Hypertext Transfer Protocol — HTTP/1.1** [online]. Edited by R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach and T. Berners-Lee. 1999 [viewed 2020-04-06]. Available at: https://tools.ietf.org/html/rfc2616
- Internet Engineering Task Force (IETF). RFC 3986, **Uniform Resource Identifier (URI): Generic Syntax** [online]. Edited by T. Berners-Lee, R. Fielding and L. Masinter. 2005 [viewed 2020-04-06]. Available at: https://tools.ietf.org/html/rfc3986
- Internet Engineering Task Force (IETF). RFC 6266 **Use of the Content-Disposition Header Field in the Hypertext**

Transfer Protocol (HTTP) [online]. Edited by J. Reschke. 2011 [viewed 2020-04-06]. Available at: https://tools.ietf.org/html/rfc6266

- Internet Engineering Task Force (IETF). RFC 7578 **Returning Values from Forms: multipart/form-data** [online]. Edited by L. Masinter. 2015 [viewed 2020-04-06]. Available at: https://tools.ietf.org/html/rfc7578
- Internet Engineering Task Force (IETF). RFC 7946 **The GeoJSON Format** [online]. Edited by H. Butler, M. Daly, A. Doyle, S. Gillies, S. Hagen, and T. Schaub. 2016 [viewed 2020-04-06]. Available at: https://tools.ietf.org/html/rfc7946
- Internet Engineering Task Force (IETF). RFC 8288 **Web Linking** [online]. Edited by M. Nottingham. 2017 [viewed 2020-04-06]. Available at: http://tools.ietf.org/rfc/rfc8288
- Open API Initiative (OAI): **The OpenAPI specification 3.0** [online]. 2020 [viewed 2020-04-06]. Available at: https://github.com/OAI/OpenAPI-Specification/blob/master/versions/

---

4. Terms and Definitions

This document uses the terms defined in Sub-clause 5.3 of [OGC 06-121r9], 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.

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

4.1. attribute dataset

Dataset that contains attribute information that can be joined with a spatial dataset through common identifiers.

4.2. spatial dataset

Dataset that contains geometry information.

---

5. Conventions and background

5.1. Identifiers

The normative provisions in this specification are denoted by the URI `http://www.opengis.net/spec/ogcapi-tjs/1.0`

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

5.2. Link relations

To express relationships between resources, RFC 8288 (Web Linking) is used.

The following registered link relation types [IANA] are used in this document.

- **alternate**: Refers to a substitute for this context.
- **self**: Conveys an identifier for the link’s context.
- **service-desc**: Identifies service description for the context that is primarily intended for consumption by machines.
  - API definitions are considered service descriptions.
- **service-doc**: Identifies service documentation for the context that is primarily intended for human consumption.

In addition the following link relation types are used for which no applicable registered link relation type could be identified.

- **attributedatasets**: Refers to a resource that is comprised of the metadata of the attribute datasets that are available on the server.
- **attributedataset**: Refers to a resource that is comprised of the metadata of the specific attribute dataset that is available on the server.
- **conformance**: Refers to a resource that identifies the specifications that the link's context conforms to.
- **keys**: Refers to a resource that is comprised of the metadata of the key fields that belong to a specific attribute or spatial dataset that is available on the server.
- **keyname**: Refers to a resource that is comprised of the metadata of the key values that belong to a specific key field of a specific attribute or spatial dataset that is available on the server.
- **key**: Refers to a resource that is comprised of a specific key value of a specific key field of a specific attribute or spatial dataset that is available on the server.
- **joins**: Refers to a resource that is comprised of the metadata of the created joins that are available on the server.
- **join**: Refers to a resource that is comprised of the metadata of the specific join that is available on the server.
- **spatialdatasets**: Refers to a resource that is comprised of the metadata of the spatial datasets that are available on the server.
- **spatialdataset**: Refers to a resource that is comprised of the metadata of the specific spatial dataset that is available on the server.

5.2.1. Response Schema for the Link Object

```
Link:
  required:
  - href
  type: object
  properties:
    href:
      type: string
    rel:
      type: string
    type:
      type: string
    hreflang:
      type: string
    title:
      type: string
```

5.3. Exception messages

The exception messages have the following structure:

Name	Description
status	The HTTP status code of the response.
message	Details of the exception.
locator	Name of the request parameter that caused the exception

5.3.1. Response Schema for the Exception Messages

```
schema:
  $ref: '#/components/schemas/ExceptionMessage'

ExceptionMessage:
  required:
  - message
  - status
  type: object
  properties:
    status:
      type: string
    message:
      type: string
    locator:
      type: string
```

---

6. Requirements Class "Core"

6.1. Overview

The requirement class “core” contains 4 operation sets: *discovery operations*, *data joining operations*, *spatial joining operations* and *file joining operations*. The Table 1 contains an overview of the operations specified in the core module.

The operation set *discovery operations* contains functionalities for accessing the API landing page, the API definition file and the information on the service’s conformance to the specification. The TJS implementations SHALL support all operations in the operation set *discovery operations*.

The TJS implementations SHALL support at least one of the operation sets: *data joining operations*, *spatial joining operations* and *file joining operations*. If a server supports a particular operation set it SHALL implement all mandatory operations that belong to it.

The operation set *data joining operations* contains functionalities for accessing metadata and key values on the spatial datasets that are available on the server and adding, updating and deleting them. It contains also functionalities for joining attribute data files with the spatial datasets and accessing, updating and deleting the joins on the server.

The operation set *spatial joining operations* contains functionalities for accessing metadata and key values on the attribute datasets that are available on the server and adding updating and deleting them. It contains also functionalities for joining the attribute datasets with inputted spatial dataset files.

The operation set *file joining operations* contains a functionality for joining attribute data files directly spatial data files.

The core module contains support for attribute data files in the csv format and spatial data files in the GeoJSON format.

Table 1. Overview of the operations in the OGC API - TJS core module

Path	HTTP method	Description
Discovery Operations		
/	GET	API landing page
/api	GET	API definition document
/conformance	GET	API conformance declaration
Data Joining Operations		
/spatialdatasets	GET	Returns metadata on all spatial datasets available on the server

Path	HTTP method	Description
/spatialdatasets	POST	Adds a new spatial dataset to the server
/spatialdatasets/{spatialdatasetid}	GET	Returns metadata on a specific spatial dataset
/spatialdatasets/{spatialdatasetid}	PUT	Updates fully a spatial dataset
/spatialdatasets/{spatialdatasetid}	DELETE	Deletes a spatial dataset
/spatialdatasets/{spatialdatasetid}/keys	GET	Returns a list of available key fields of a specific spatial dataset
/spatialdatasets/{spatialdatasetid}/keys/{keyname}	GET	Returns the key values of a specific key field of a specific spatial dataset
/spatialdatasets/{spatialdatasetid}/keys/{keyname}/{key}	GET	Returns a specific key value of a specific key field of a specific spatial dataset
/joins	GET	Returns a list of all joins available on the server
/joins	POST	Creates a new join by joining attribute data from a file with a specific spatial dataset
/joins/{joinid}	GET	Returns metadata on a specific join
/joins/{joinid}	PUT	Updates fully a specific join
/joins/{joinid}	DELETE	Deletes a specific join
Spatial Joining Operations		
/attributedatasets	GET	Returns metadata on all attribute datasets available on the server
/attributedatasets	POST	Adds a new attribute dataset to the server
/attributedatasets/{attributedatasetid}	GET	Returns metadata on a specific attribute dataset
/attributedatasets/{attributedatasetid}	PUT	Updates fully a specific attribute dataset
/attributedatasets/{attributedatasetid}	DELETE	Deletes an attribute dataset
/attributedatasets/{attributedatasetid}/keys	GET	Returns a list of available key fields of a specific attribute dataset
/attributedatasets/{attributedatasetid}/keys/{keyname}	GET	Returns the key values of a specific key field of a specific attribute dataset
/attributedatasets/{attributedatasetid}/keys/{keyname}/{key}	GET	Returns a specific key value of a specific key field of a specific attribute dataset
/joinattributedataset	POST	Joins attribute dataset from the server with a inputted spatial data file
File Joining Operations		

Path	HTTP method	Description
/joinfiles	POST	Joins attribute data from an inputted file to an inputted spatial dataset file

6.2. Operation Set: Discovery Operations

The operation set *discovery operations* contains operations that provide general information on the TJS implementation. The TJS implementations SHALL support all operations in this operation set.

6.2.1. API Landing Page

The HTTP GET operation at service root path `/` returns the API landing page document. The API landing page document contains links to the API definition document, conformance information and to the metadata on the attribute datasets, spatial datasets and joins that are available on the server.

6.2.1.1. Request

Requirement 1	/req/core/root-op
A	The server SHALL support the HTTP GET operation at the path `/`.

6.2.1.2. Response

Requirement 2	/req/core/root-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response is an API landing page document that is based on the landing page schema. The response document SHALL contain links to the following resources: • `self` (link rel property value: 'self') • `/api` (link rel property value: 'service-desc' or 'service-doc') • `/conformance` (link rel property value: 'conformance') If the TJS implementation supports the *data joining operations* operation set, the landing page SHALL contain also links to the following resources: • `/spatialdatasets` (link rel property value: 'spatialdatasets') • `/joins` (link rel property value: 'joins') If the TJS implementation supports the *spatial joining operations* operation set, the landing page SHALL contain also links to the following resources: • `/attributedatasets` (link rel property value: 'attributedatasets') The links SHALL contain the property values 'href', 'rel', 'title' and 'type'.

6.2.1.3. Errors

Requirement 3	/req/core/root-error
A	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.2.1.4. Schema for the Landing Page

```
schema:
  $ref: '#/components/schemas/LandingPageResponseObject'

LandingPageResponseObject:
  required:
  - links
  type: object
  properties:
    title:
      type: string
    description:
      type: string
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
```

6.2.2. API Definition

The HTTP GET operation at path `/api` returns the service's API definition document.

6.2.2.1. Request

Requirement 4	/req/core/api-definition-op
A	The URIs of all API definitions referenced from the landing page SHALL support the HTTP GET method.

6.2.2.2. Response

Requirement 5	/req/core/api-definition-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`. The server SHALL return an API definition document. The recommended format is OpenAPI version 3.0.

6.2.2.3. Errors

Requirement 6	/req/core/api-definition-error
A	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.2.3. Declaration of Conformance Classes

The HTTP GET operation at path `/conformance` returns a list of TJS conformance classes that the server supports.

The conformance class values, defined in the core module are:

- Core
  - http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/core
- Data Joining from Attribute Data Files with Spatial Datasets on the Server
  - http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/core/data-joining
- File joining Between Attribute Data Files and Spatial Data Files
  - http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/core/file-joining
- Data Joining From Attribute Datasets on the Server with Spatial Data Files
  - http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/core/spatial-joining
- Support for CSV files
  - http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/input/csv
- Support for GeoJSON files
  - http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/input/geojson
- Support for Fetching Input Files via HTTP
  - http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/input/http-ref
- Support for Input Files in the HTTP request body
  - http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/input/inline

6.2.3.1. Request

Requirement 7	**/req/core/conformance-op**
A	The server SHALL support the HTTP GET operation at the path `/conformance`.

6.2.3.2. Response

Requirement 8	**/req/core/conformance-success**
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response contains a list of conformance classes that the service supports. The response document is based on the conformance schema.

6.2.3.3. Errors

Requirement 9	**/req/core/conformance-error**
A	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`.

Requirement 9	/req/core/conformance-error
	The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.2.3.4. Response schema for the Conformance

```
schema:
  $ref: '#/components/schemas/ConformanceResponseObject'

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

6.3. Operation Set: Data Joining Operations

The operation set *data joining operations* contains operations for:

- Retrieving metadata and key values of the spatial datasets that are available on the server
- Adding spatial datasets to the server
- Updating the spatial datasets on the server
- Deleting the spatial datasets from the server.
- Joining attribute data from inputted files with the spatial datasets on the server
- Accessing, updating and deleting the created joins.

The TJS core module contains support for attribute data files in the csv format and spatial data files in the GeoJSON format.

The data joins are executed through common keys that are shared between the spatial dataset and the attribute dataset.

6.3.1. Spatial Datasets

The HTTP GET operation at path `/spatialdatasets` returns metadata on all spatial datasets that are available on the server.

6.3.1.1. Request

Requirement 10	/req/core/spatialdatasets-get-op
A	If the server implements the *data joining operations* operation set it SHALL support the HTTP GET operation at path `/spatialdatasets`. The server implementations SHALL support the following query parameters: **Query parameters:**

Name	Type	Description
title	String	Filters the spatial datasets whose title contains the value of the parameter
startDate	String	Filters the spatial datasets by start date.

Name	Type	Description
		Format: yyyy-mm-dd
endDate	String	Filters the spatial datasets by end date. Format: yyyy-mm-dd

6.3.1.2. Response

Requirement 11	/req/core/spatialdatasets-get-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the spatialdatasets schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type. **Description of properties of the spatialDatasets property:**

Name	Description
date	Date that applies to the spatial dataset. Format: yyyy-mm-dd
description	Description of the spatial dataset
documentation	Link to the spatial dataset´s documentation
links	Links to this spatial dataset´s different representations. The links object SHALL include the properties href, rel and type. The value of the rel property SHALL be 'spatialdataset'
spatialDatasetId	Unique identifier for the spatial dataset. The identifier is used in other operations for indicating the spatial dataset in question
title	Title of the spatial dataset

6.3.1.3. Errors

Requirement 12	/req/core/spatialdatasets-get-error
A	If an incorrect request is made to the server, it SHALL be reported as a response

Requirement 12	/req/core/spatialdatasets-get-error
	with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If spatial datasets are not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Spatial datasets not found'.
C	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.1.4. Response Schema for the Spatial Datasets

```
schema:
  $ref: '#/components/schemas/SpatialDatasetsResponseObject'

SpatialDatasetsResponseObject:
  required:
  - links
  - spatialDatasets
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    spatialDatasets:
      type: array
      items:
        $ref: '#/components/schemas/SpatialDatasetsObject'

SpatialDatasetsObject:
  required:
  - date
  - links
  - spatialDatasetId
  - title
  type: object
  properties:
    date:
      type: string
    description:
      type: string
    documentation:
      type: string
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    spatialDatasetId:
      type: integer
    title:
      type: string
```

6.3.2. Spatial Dataset Creation

The HTTP POST operation at path `/spatialdatasets` adds a new spatial dataset to the server.

The TJS core module contains support for the GeoJSON format.

6.3.2.1. Request

Requirement 13	/req/core/spatialdatasets-post-op
A	If the server implements the *data joining operations* operation set it MAY support the HTTP POST operation at path `/spatialdatasets`. The spatial dataset file can be either uploaded to the server with the *spatialDatasetFile* parameter or provided through URL link with the *spatialDatasetURL* parameter. The request SHALL contain the header: • Content-Type: multipart/form-data; If the spatial dataset file is provided by upload, it SHALL contain the header: • Content-Disposition: form-data; filename="[spatial dataset file´s name]"; name="spatialDatasetFile"; **Request´s form data parameters:**

Name	Description	Type and values	Required
spatialDatasetFormat	The format of the spatial dataset	String	Mandatory[a]
spatialDatasetFile	The spatial dataset file (uploaded file)	File	Optional[b]
spatialDatasetURL	A URL link to the spatial dataset file	URL type	Optional[b]
spatialDatasetDefaultKey	The path to the key field name in the spatial dataset file that contains key values. This value will be used as a default key field in the data joins. Example: 'features.properties.kunta'	String	Mandatory
spatialDatasetDefaultKeyLanguage	ISO 639-1 code of the language of the default key field values (if applicable).	String	Optional
spatialDatasetKeyTitle	The path to the field name in the spatial dataset file that contains the titles for the key field. Example: 'features.properties.name'	String	Optional
spatialDatasetAdditionalKeys	A comma-separated list of additional paths to the key field names in the spatial dataset file that contain key values. Example: 'features.properties.nimi,features.properties.name'.	String, (separated by commas)	Optional
spatialDatasetAdditionalKeyLanguages	A comma-separated list of ISO 639-1 codes of the languages of the additional key field values (if applicable). The languages SHALL be in the same order than the values in the *spatialDatasetAdditionalKeys* parameter	String, (separated by commas)	Optional
spatialDatasetTitle	Title of the spatial dataset.	String	Mandatory
spatialDatasetDescription	Description of the spatial dataset.	String	Optional

Name	Description	Type and values	Required
spatialDatasetDate	Date of the spatial dataset (format: yyyy-mm-dd). If date is not provided in the request, servers SHALL use the value of the current date.	String	Optional
spatialDatasetDocumentationLink	URL link that contains documentation for the spatial dataset.	String	Optional
[a] The TJS core module contains support for the format: 'geojson'.			
[b] One of the parameters: *spatialDatasetFile* or *spatialDatasetURL* is mandatory to be used with the operation. The *spatialDatasetFile* parameter can be used for uploading a GeoJSON file to the server. The *spatialDatasetURL* parameter can be used for providing the GeoJSON file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateSpatialDatasetFileInput'.			

6.3.2.2. Response

Requirement 14	/req/core/spatialdatasets-post-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `201`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the spatialdatasets post schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type. The response property: spatialDataset SHALL contain metadata on the created spatial dataset. **Description of properties of the spatialDataset property:**

Name	Description
date	Date that applies to the spatial dataset. Format: (yyyy-mm-dd).
description	Description of the spatial dataset.
documentation	Link to the spatial dataset´s documentation.
keys	Links to the different representations of this spatial dataset´s key fields. The links SHALL include the properties href, rel and type. The value of the rel property SHALL be 'keys'.

Name	Description
spatialDatasetId	Unique identifier for the spatial dataset.
title	Title of the spatial dataset.

6.3.2.3. Errors

Requirement 15	/req/core/spatialdatasets-post-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.2.4. Response Schema for the Spatial Dataset Creation

```
schema:
  $ref: '#/components/schemas/SpatialDatasetResponseObject'

SpatialDatasetResponseObject:
  required:
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    spatialDataset:
      $ref: '#/components/schemas/SpatialDatasetObject'

SpatialDatasetObject:
  required:
  - date
  - keys
  - spatialDatasetId
  - title
  type: object
  properties:
    date:
      type: string
    description:
      type: string
    documentation:
      type: string
    keys:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    spatialDatasetId:
      type: integer
    title:
      type: string
```

6.3.3. Spatial Dataset

The HTTP GET operation at path `/spatialdatasets/{spatialdatasetid}` returns metadata on a specific spatial dataset

available on the server.

6.3.3.1. Request

Requirement 16	/req/core/spatialdatasets-spatialdatasetid-get-op
A	If the server implements the *data joining operations* operation set it SHALL support the HTTP GET operation at path `/spatialdatasets/{spatialdatasetid}`.

6.3.3.2. Response

Requirement 17	/req/core/spatialdatasets-spatialdatasetid-get-success
A	A successful execution of the operation shall be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the spatialdataset schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type. The response property: spatialDataset SHALL contain metadata on the requested spatial dataset. **Description of properties of the spatialDataset property:**

Name	Description
date	Date that applies to the spatial dataset. Format: yyyy-mm-dd
description	Description of the spatial dataset
documentation	Link to the spatial dataset´s documentation
keys	Links to the different representations of this spatial dataset´s key fields. The links SHALL include the properties href, rel and type. The value of the rel property SHALL be 'keys'.
spatialDatasetId	Unique identifier for the spatial dataset
title	Title of the spatial dataset

6.3.3.3. Errors

Requirement 18	/req/core/spatialdatasets-spatialdatasetid-get-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the spatial dataset is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Spatial dataset {spatialdatasetid} not found'.
C	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.3.4. Response Schema for the Spatial Dataset

```
schema:
  $ref: '#/components/schemas/SpatialDatasetResponseObject'

SpatialDatasetResponseObject:
  required:
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    spatialDataset:
      $ref: '#/components/schemas/SpatialDatasetObject'

SpatialDatasetObject:
  required:
  - date
  - keys
  - spatialDatasetId
  - title
  type: object
  properties:
    date:
      type: string
    description:
      type: string
    documentation:
      type: string
    keys:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    spatialDatasetId:
      type: integer
    title:
      type: string
```

6.3.4. Spatial Dataset Update

The HTTP PUT operation at path `/spatialdatasets/{spatialdatasetid}` updates fully the specific spatial dataset available on the server.

The TJS core module contains support for the GeoJSON format for updating the spatial datasets.

6.3.4.1. Request

Requirement 19	**/req/core/spatialdatasets-spatialdataset-put-op**
A	If the server implements the *data joining operations* operation set, it MAY support the HTTP PUT operation at path `/spatialdatasets/{spatialdatasetid}`. The spatial dataset file can be either uploaded to the server with the *spatialDatasetFile* parameter or provided through URL link with the *spatialDatasetURL* parameter. The request SHALL contain the header: • Content-Type: multipart/form-data; If the spatial dataset file is provided by upload, it SHALL contain the header: • Content-Disposition: form-data; filename="[spatial dataset file´s name]"; name="spatialDatasetFile"; **Request´s form data parameters:**

Name	**Description**	**Type and values**	**Required**
spatialDatasetFormat	The format of the spatial dataset	String	Mandatory[a]
spatialDatasetFile	The spatial dataset file (uploaded file)	File	Optional[b]
spatialDatasetURL	A URL link to the spatial dataset file	URL type	Optional[b]
spatialDatasetDefaultKey	The path to the key field name in the spatial dataset file that contains key values. This value will be used as a default key field in the data joins. Example: 'features.properties.kunta'	String	Mandatory
spatialDatasetDefaultKeyLanguage	ISO 639-1 code of the language of the default key field values (if applicable).	String	Optional
spatialDatasetKeyTitle	The path to the field name in the spatial dataset file that contains the titles for key the field. Example: 'features.properties.name'	String	Optional
spatialDatasetAdditionalKeys	A comma-separated list of additional paths to the key field names in the spatial dataset file that contain key values. Example: 'features.properties.nimi,features.properties.name'.	String, (separated by commas)	Optional
spatialDatasetAdditionalKeyLanguages	A comma-separated list of ISO 639-1 codes of the languages of the additional key field values (if applicable). The languages SHALL be in the same order than the values in the *spatialDatasetAdditionalKeys* parameter	String, (separated by commas)	Optional

Name	Description	Type and values	Required
spatialDatasetTitle	Title of the spatial dataset.	String	Mandatory
spatialDatasetDescription	Description of the spatial dataset.	String	Optional
spatialDatasetDate	Date of the spatial dataset (format: yyyy-mm-dd). If date is not provided in the request, servers SHALL use the value of the current date.	String	Optional
spatialDatasetDocumentationLink	URL link that contains documentation for the spatial dataset.	String	Optional

[a] The TJS core module contains support for the format: 'geojson'.

[b] One of the parameters: *spatialDatasetFile* or *spatialDatasetURL* is mandatory to be used with the operation. The *spatialDatasetFile* parameter can be used for uploading a GeoJSON file to the server. The *spatialDatasetURL* parameter can be used for providing the GeoJSON file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateSpatialDatasetFileInput'.

6.3.4.2. Response

Requirement 20	/req/core/spatialdatasets-spatialdatasetid-put-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the spatialdatasets_spatialdatasetid_put_schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type. The response property: spatialDataset SHALL contain metadata on the updated spatial dataset. **Description of properties of the spatialDataset property:**

Name	Description
date	Date that applies to the spatial dataset. Format: (yyyy-mm-dd).
description	Description of the spatial dataset.
documentation	Link to the spatial dataset´s documentation.

Name	Description
keys	Links to the different representations of this spatial dataset´s key fields. The links SHALL include the properties href, rel and type. The value of the rel property SHALL be 'keys'.
spatialDatasetId	Unique identifier for the spatial dataset.
title	Title of the spatial dataset.

6.3.4.3. Errors

Requirement 21	/req/core/spatialdatasets-spatialdatasetid-put-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the spatial dataset is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Spatial dataset {spatialdatasetid} not found'.
C	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.4.4. Response Schema for the Spatial Dataset Update

```
schema:
  $ref: '#/components/schemas/SpatialDatasetResponseObject'

SpatialDatasetResponseObject:
  required:
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    spatialDataset:
      $ref: '#/components/schemas/SpatialDatasetObject'

SpatialDatasetObject:
  required:
  - date
  - keys
  - spatialDatasetId
  - title
  type: object
  properties:
    date:
      type: string
    description:
      type: string
    documentation:
      type: string
    keys:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    spatialDatasetId:
      type: integer
    title:
      type: string
```

6.3.5. Spatial Dataset Delete

The HTTP DELETE operation at path `/spatialdatasets/{spatialdatasetid}` deletes the specific spatial dataset from the server.

6.3.5.1. Request

Requirement 22	**/req/core/spatialdatasets-spatialdatasetid-delete-op**
A	If the server implements the *data joining operations* operation set, it MAY support the HTTP DELETE operation at the path `/spatialdatasets/{spatialdatasetid}`.

6.3.5.2. Response

Requirement 23	**/req/core/spatialdatasets-spatialdatasetid-delete-success**
A	A successful execution of the operation shall be reported as a response with a HTTP status code `204`. The response body SHALL be empty.

6.3.5.3. Errors

Requirement 24	**/req/core/spatialdatasets-spatialdatasetid-delete-error**
A	If the spatial dataset is not found on the server, it shall be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Spatial dataset {spatialdatasetid} not found'.

Requirement 24	/req/core/spatialdatasets-spatialdatasetid-delete-error
B	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.5.4. Response Schema for the Spatial Dataset Delete

The response of the operation is empty.

6.3.6. Spatial Dataset Key Fields

The HTTP GET operation at path `/spatialdatasets/{spatialdatasetid}/keys` returns a list of key fields of a specific spatial dataset.

6.3.6.1. Request

Requirement 25	/req/core/spatialdatasets-spatialdatasetid-keys-op
A	If the server implements the *data joining operations* operation set, it SHALL support the HTTP GET operation at the path `/spatialdatasets/{spatialdatasetid}/keys`.

6.3.6.2. Response

Requirement 26	/req/core/spatialdatasets-spatialdatasetid-keys-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the spatialdataset keys schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type. **Description of properties of the keys property:**

Name	Description
isDefault	Indicates if the key is used as a default key field in the data joins with this spatial dataset. Only one object in the response SHALL have the value 'true'.
keyLanguage	ISO 639-1 code of the language of the key field values (if applicable).
keyName	Name of the key field.

Name	Description
links	Links to the different representations of this key value. The links SHALL have the properties href, rel and type. The value of the rel property SHALL be 'keyname'.

6.3.6.3. Errors

Requirement 27	/req/core/spatialdataset-spatialdatasetid-keys-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the spatial dataset keys are not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Spatial dataset {spatialdatasetid} keys not found'.
B	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.6.4. Response Schema for the Spatial Dataset Key Fields

```
schema:
  $ref: '#/components/schemas/SpatialDatasetKeysResponseObject'

SpatialDatasetKeysResponseObject:
  required:
  - keys
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    keys:
      type: array
      items:
        $ref: '#/components/schemas/SpatialDatasetKeysObject'

SpatialDatasetKeysObject:
  required:
  - isDefault
  - keyName
  - links
  type: object
  properties:
    isDefault:
      type: boolean
    keyLanguage:
      type: string
    keyName:
      type: string
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
```

6.3.7. Spatial Dataset Key Field

The HTTP GET operation at path `/spatialdatasets/{spatialdatasetid}/keys/{keyname}` returns a list of key values from a specific key field of a specific spatial dataset.

6.3.7.1. Request

Requirement 28	**/req/core/spatialdatasets-spatialdatasetid-keys-keyname-op**
A	If the server implements the *data joining operations* operation set, it SHALL support the HTTP GET operation at the path `/spatialdatasets/{spatialdatasetid}/keys/{keyname}`.

6.3.7.2. Response

Requirement 29	**/req/core/spatialdatasets-spatialdatasetid-keys-keyname-success**
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the spatialdataset key field schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type.

Description of properties of the keys property:

Name	Description
key	Spatial dataset´s key field´s key value.
links	Links to the different representations of this key value. The links SHALL have the properties href, rel and type. The value of the rel property SHALL be 'key'.
title	Human-readable description of the key value.

6.3.7.3. Errors

Requirement 30	/req/core/spatialdatasets-spatialdatasetid-keys-keyname-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the spatial dataset´s key field´s key values are not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Spatial dataset {spatialdatasetid} key field {keyname} not found'.
C	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.7.4. Response Schema for the Spatial Dataset Key Field

```
schema:
  $ref: '#/components/schemas/SpatialDatasetKeysKeynameResponseObject'

SpatialDatasetKeysKeynameResponseObject:
  required:
  - keys
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    keys:
      type: array
      items:
        $ref: '#/components/schemas/SpatialDatasetKeysKeynameObject'

SpatialDatasetKeysKeynameObject:
  required:
  - key
  - links
  type: object
  properties:
    key:
      type: string
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    title:
      type: string
```

6.3.8. Spatial Dataset Key Field Key

The HTTP GET operation at path `/spatialdatasets/{spatialdatasetid}/keys/{keyname}/{key}` returns a specific key value, from a specific key field from a specific spatial dataset.

6.3.8.1. Request

Requirement 31	**/req/core/spatialdatasets-spatialdatasetid-keys-keyname-key-op**
A	If the server implements the *data joining operations* operation set it SHALL support the HTTP GET operation at the path `/spatialdatasets/{spatialdatasetid}/keys/{keyname}/{key}`.

6.3.8.2. Response

Requirement 32	**/req/core/spatialdatasets-spatialdatasetid-keys-keyname-key-success**
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the spatialdataset key field key schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type **Description of properties of the key property:**

Name	Description
key	Spatial dataset´s key field´s key value.
title	Human-readable description of the key value.

6.3.8.3. Errors

Requirement 33	/req/core/spatialdatasets-spatialdatasetid-keys-keyname-key-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the spatial dataset key value is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Spatial dataset {spatialdatasetid} key field {keyname} key {key} not found'.
C	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.8.4. Response Schema for the Spatial Dataset Key Field Key

```
schema:
  $ref: '#/components/schemas/SpatialDatasetKeysKeynameKeyResponseObject'

SpatialDatasetKeysKeynameKeyResponseObject:
  required:
  - key
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    key:
      $ref: '#/components/schemas/SpatialDatasetKeysKeynameKeyObject'

SpatialDatasetKeysKeynameKeyObject:
  required:
  - key
  type: object
  properties:
    key:
      type: string
    title:
      type: string
```

6.3.9. Joins

The HTTP GET operation at path `/joins` returns list of all joins that are available on the server.

6.3.9.1. Request

Requirement 34	**/req/core/joins-get-op**
A	If the server implements the *data joining operations* operation set it SHALL support the HTTP GET operation at the path `/joins`.

6.3.9.2. Response

Requirement 35	**/req/core/joins-get-success**
A	A successful execution of the operation shall be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported. The server SHALL return metadata on all joins that are available on the server.
C	The response document is based on the joins schema. The response document SHALL include the following links: • a link to this document (link rel: 'self') • links to this document in other supported media types (link rel: 'alternate') **Description of elements of the joins property:** **Name** / **Description** joinId — Unique identifier for the join. The identifier is used in other operations for indicating the join in question. joinedLayerName — Name of the created layer for the join. joinTimestamp — Timestamp when the join has been executed. links — Links to the different representations of the join. The links SHALL have the properties href, rel and type. The value of the rel property SHALL be 'join'.

6.3.9.3. Errors

Requirement 36	**/req/core/joins-get-error**
A	If joins are not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON

Requirement 36	/req/core/joins-get-error
	format that is based on exception message schema. The value of the message property SHALL be 'Joins not found'.
B	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.9.4. Response schema for the Joins

```
schema:
  $ref: '#/components/schemas/JoinsResponseObject'

JoinsResponseObject:
  required:
  - joins
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    joins:
      type: array
      items:
        $ref: '#/components/schemas/JoinsObject'

JoinsObject:
  required:
  - joinId
  - joinTimestamp
  - links
  type: object
  properties:
    joinId:
      type: integer
    joinedLayerName:
      type: string
    joinTimestamp:
      type: string
      format: date-time
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
```

6.3.10. Join Attribute Data from a File with a Spatial Dataset on the Server

The HTTP POST operation at path `/joins` joins attribute data from an attribute data file with a spatial dataset available on the server. The joins are executed by using shared key values in the two datasets.

The TJS core module supports data joining from csv files.

6.3.10.1. Request

Requirement 37	/req/core/joins-post-op
A	If the server implements the *data joining operations* operation set, it SHALL support the HTTP POST operation at path `/joins`. The attribute dataset file can be either uploaded to the server with the *attributeDatasetFile* parameter or provided through URL link with the *attributeDatasetURL* parameter. The request SHALL contain the header:

- Content-Type: multipart/form-data;

If the attribute dataset file is provided by upload, it SHALL contain the header

- Content-Disposition: form-data; filename="[attribute dataset file´s name]"; name="attributeDatasetFile";

If the attribute dataset is provided as a csv file and it contains multiple rows with the same key value, the value is used in the join operation from the first row where the key is encountered.

The servers MAY support an optional direct GeoJSON response for the joined data. In this case, the server returns the joined data directly to the client in the GeoJSON format instead of the join response document. The direct GeoJSON output can be requested with the *outputFormats* parameter value: 'geojson-direct'.

Request´s form data parameters:

Name	**Description**	**Type and values**	**Required**
attributeDatasetFormat	The format of the attribute dataset.	String	Mandatory[a]
attributeDatasetFile	The attribute dataset file (uploaded file).	File	Optional[b]
attributeDatasetURL	The attribute dataset URL.	URL	Optional[b]
attributeDatasetKey	The path to the key field in the attribute dataset that contains the key values. For csv format, this value is the column number that contains the key values (counting starts from 0).	String	Mandatory
attributeDatasetDataValueList	Paths to fields in the attribute dataset that contain the attribute values that are to be	String, separated by commas	Mandatory

Name	Description	Type and values	Required
	joined with the spatial dataset. For csv format, the values are the column numbers that contain the attribute values (counting starts from 0).		
csvFileDelimiter	The delimiter character used in the csv file.	String	Optional[c]
csvFileContainsHeaderRow	Information on the existence of header row in the csv file.	Boolean	Optional[d]
includeJoinMetadata	Includes the joinInformation element to the response document that contains information on the successfulness of the join operation.	Boolean	Optional[e]
outputFormats	List of outputs that will be included to the response document.	String, separated by commas	Optional[f]
spatialDatasetId	The id number of the spatial dataset on the server that will be used in the join operation.	Integer	Mandatory
spatialDatasetKey	The key field of the spatial dataset that will be used in the join operation.	String	Optional[g]
[a] The TJS core module contains support for the format: 'csv'.			

Name	Description	Type and Values	Required
[b] One of the parameters: *attributeDatasetFile* or *attributeDatasetURL* is mandatory to be used with the operation. The *attributeDatasetFile* parameter can be used for uploading a attribute dataset file to the server. The *attributeDatasetURL* parameter can be used for providing the attribute dataset file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateAttributeDatasetFileInput'.			
[c] The *csvFileDelimiter* paramater is mandatory to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules.			
[d] The *csvFileContainsHeaderRow* paramater is optional to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules. Values: (*true*, *false*). If parameter is not provided in the request, default value *false* is used. If parameter has value *true*, csv file's header is assumed to be on the first row of the csv file and data values are assumed to start from the second row. If parameter has value *false*, the data values are assumed to start from csv file's first row.			
[e] Values: (*true*, *false*). If parameter is not provided in the request, a default value *false* is used.			
[f] Comma-separated list of the outputs that will be included to the response document. The output formats that the server implementation supports SHALL be listed in the operation's *outputFormats* parameter description in the API description document. If the parameter value is not provided in the request, a default value 'geojson' is used. If the server supports the direct geojson response for the join it SHALL support the parameter value 'geojson-direct'.			
[g] If spatialDatasetKey parameter is not provided in the request, a default key field of the spatial dataset will be used in the join operation.			

6.3.10.2. Response

Requirement 38	/req/core/joins-success
A	If the client has requested the direct GeoJSON output with *outputFormats* parameter value: 'geojson-direct', the successful execution of the opeation SHALL be reported as a response with HTTP status code `200`. The response SHALL contain the joined data in the GeoJSON format.
B	For other output formats than 'geojson-direct', a successful execution of the operation SHALL be reported as a response with a HTTP status code `201`.
C	For other output formats than 'geojson-direct', the server implementations SHALL support the JSON output format. Any other output formats MAY also be supported. The response document contains links to the created outputs for the joined data and it MAY contain metadata on the successfulness of the data joining operation. The TJS implementations SHALL support the GeoJSON output format for the joined data and MAY support any other output formats. Other recommended output

formats to be supported are WFS, WMS, OGC API Features and OGC API Maps.

D

For other output formats than 'geojson-direct', the response document is based on the data joining from am attribute dataset file with a spatial dataset schema.

The response document SHALL include the following links:

- a link to this document (link rel: 'self')
- links to this document in other supported media types (link rel: 'alternate')

Description of properties of the join property:

Name	Description
inputs	Join operation inputs
joinId	Unique identifier for the join
joinInformation	Information on the execution of the data join operation
outputs	Links to the created outputs
timestamp	Timestamp when the join has been executed

Description of properties of the inputs property:

Name	Description
attributeDataset	Name or URL of the joined csv file
spatialDataset	Link objects that contain links to spatialdataset´s different representations. Properties href, rel and type are mandatory.

Description of properties of the joinInformation property:

Name	Description
additionalAttributeKeys	List of additional keys in the csv file that were not available in the spatial dataset keys
duplicateAttributeKeys	List of duplicate keys in the csv file
matchedSpatialDatasetKeys	List of spatial dataset keys that were successfully matched with attribute data
numberOfAdditionalAttributeKeys	The number of additional attribute key values in the attribute dataset that were not available in the spatial dataset

Name	Description
numberOfDuplicateAttributeKeys	The number of attribute keys that had duplicate entries
numberOfMatchedSpatialDatasetKeys	The number of spatial dataset keys, to which attribute data was joined successfully
numberOfUnmatchedSpatialDatasetKeys	The number of spatial dataset keys, to which attribute data couldn´t be joined
unmatchedSpatialDatasetKeys	List of spatial dataset keys, to which attribute data couldn´t be joined

Description of properties of the outputs property:

Name	Description
format	Name of the output format
layerName	Name of the joined data layer (For WMS and WFS outputs)
link	Link to the output
styleName	Name of the joined data layer style

6.3.10.3. Errors

Requirement 39	/req/core/joins-post-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The possible values for the message property are 'InvalidParameterValue', 'MissingParameterValue' and 'DuplicateAttributeDatasetFileInput'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the spatialdataset is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Spatial dataset {spatialdatasetid} not found'.
C	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON

Requirement 39	format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'. **/req/core/joins-post-error**

6.3.10.4. Response schema for the Attribute Data Join from a File with a Spatial Dataset on the Server

```
schema:
  $ref: '#/components/schemas/JoinDataResponseObject'

JoinDataResponseObject:
  required:
  - join
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    join:
      $ref: '#/components/schemas/JoinDataObject'

JoinDataObject:
  required:
  - inputs
  - joinId
  - outputs
  - timestamp
  type: object
  properties:
    joinId:
      type: integer
    timestamp:
      type: string
      format: date-time
    inputs:
      $ref: '#/components/schemas/JoinInputsObject'
    outputs:
      type: array
      items:
        $ref: '#/components/schemas/OutputObject'
    joinInformation:
      $ref: '#/components/schemas/JoinInformationObject'

JoinInputsObject:
  required:
  - attributeDataset
  - spatialDataset
  type: object
  properties:
    attributeDataset:
      type: string
    spatialDataset:
      type: array
      items:
        $ref: '#/components/schemas/Link'

OutputObject:
  required:
  - format
  - link
  type: object
  properties:
    format:
      type: string
    layerName:
      type: string
    link:
      type: string
    styleName:
      type: string

JoinInformationObject:
  type: object
  properties:
    numberOfMatchedSpatialDatasetKeys:
      type: integer
    numberOfUnmatchedSpatialDatasetKeys:
      type: integer
    numberOfAdditionalAttributeKeys:
      type: integer
    matchedSpatialDatasetKeys:
      type: array
      items:
        type: string
    unmatchedSpatialDatasetKeys:
      type: array
```

```
    items:
      type: string
  additionalAttributeKeys:
    type: array
    items:
      type: string
  duplicateAttributeKeys:
    type: array
    items:
      type: string
  numberOfDuplicateAttributeKeys:
    type: integer
```

6.3.11. Join

The HTTP GET operation at path `/joins/{joinid}` returns metadata on a specific join that is available on the server.

6.3.11.1. Request

Requirement 40	**/req/core/joins-joinid-get-op**
A	If the server implements the *data joining operations* operation set it SHALL support the HTTP GET operation at the path `/joins/{joinid}`.

6.3.11.2. Response

Requirement 41	**/req/core/joins-joinid-get-success**
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the join schema. The response document SHALL include the following links: • a link to this document (link rel: 'self') • links to this document in other supported media types (link rel: 'alternate') **Description of properties of the join property:** **Name** / **Description** inputs — Join operation inputs joinId — Unique identifier for the join joinInformation — Information on the execution of the data join operation outputs — Links to the created outputs timestamp — Timestamp when the join has been executed **Description of properties of the inputs property:**

Name	Description
attributeDataset	Name or URL of the joined csv file
spatialDataset	Link objects that contain links to spatialdataset´s different representations. Properties href, rel and type are mandatory.

Description of properties of the joinInformation property:

Name	Description
additionalAttributeKeys	List of additional keys in the csv file that were not available in the spatial dataset keys
duplicateAttributeKeys	List of duplicate keys in the csv file
matchedSpatialDatasetKeys	List of spatial dataset keys that were successfully matched with attribute data
numberOfAdditionalAttributeKeys	The number of additional attribute key values in the attribute dataset that were not available in the spatial dataset
numberOfDuplicateAttributeKeys	The number of attribute keys that had duplicate entries
numberOfMatchedSpatialDatasetKeys	The number of spatial dataset keys, to which attribute data was joined successfully
numberOfUnmatchedSpatialDatasetKeys	The number of spatial dataset keys, to which attribute data couldn´t be joined
unmatchedSpatialDatasetKeys	List of spatial dataset keys, to which attribute data couldn´t be joined

Description of properties of the outputs property:

Name	Description
format	Name of the output format
layerName	Name of the joined data layer (For WMS and WFS outputs)
link	link Link to the output
styleName	Name of the joined data layer style

6.3.11.3. Errors

Requirement 42	/req/core/joins-joinid-get-error
A	If spatial datasets are not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Join {joinid} not found'.
B	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.11.4. Response schema for the Join

```
schema:
  $ref: '#/components/schemas/JoinResponseObject'

JoinResponseObject:
  required:
  - join
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    join:
      $ref: '#/components/schemas/JoinDataObject'

JoinDataObject:
  required:
  - inputs
  - joinId
  - outputs
  - timestamp
  type: object
  properties:
    joinId:
      type: integer
    timestamp:
      type: string
      format: date-time
    inputs:
      $ref: '#/components/schemas/JoinInputsObject'
    outputs:
      type: array
      items:
        $ref: '#/components/schemas/OutputObject'
    joinInformation:
      $ref: '#/components/schemas/JoinInformationObject'

JoinInputsObject:
  required:
  - attributeDataset
  - spatialDataset
  type: object
  properties:
    attributeDataset:
      type: string
    spatialDataset:
      type: array
      items:
        $ref: '#/components/schemas/Link'

OutputObject:
  required:
  - format
  - link
  type: object
```

```
  properties:
    format:
      type: string
    layerName:
      type: string
    link:
      type: string
    styleName:
      type: string

JoinInformationObject:
  type: object
  properties:
    numberOfMatchedSpatialDatasetKeys:
      type: integer
    numberOfUnmatchedSpatialDatasetKeys:
      type: integer
    numberOfAdditionalAttributeKeys:
      type: integer
    matchedSpatialDatasetKeys:
      type: array
      items:
        type: string
    unmatchedSpatialDatasetKeys:
      type: array
      items:
        type: string
    additionalAttributeKeys:
      type: array
      items:
        type: string
    duplicateAttributeKeys:
      type: array
      items:
        type: string
    numberOfDuplicateAttributeKeys:
      type: integer
```

6.3.12. Join Update

The HTTP PUT operation at path `/joins/{joinid}` updates fully the specific join with data from an attribute dataset file.

The TJS core module scontains upport for join updating from csv files.

6.3.12.1. Request

Requirement 43	**/req/core/joins-joinid-put-op**
A	If the server implements the *data joining operations* operation set it MAY support the HTTP PUT operation at the path `/joins/{joinid}`. The attribute dataset file can be either uploaded to the server with the *attributeDatasetFile* parameter or provided through URL link with the *attributeDatasetURL* parameter. The request SHALL contain the header: • Content-Type: multipart/form-data; If attribute dataset file is provided by upload, it SHALL contain the header • Content-Disposition: form-data; filename="[attribute dtaset file´s name]"; name="attributeDatasetFile"; If the attribute dataset is provided as a csv file and it contains multiple rows with the same key value, the value is used in the join operation from the first row where the key is encountered. **Request´s form data parameters:**

Name	Description	Type and values	Required
attributeDatasetFormat	The format of the attribute dataset.	String	Mandatory[a]
attributeDatasetFile	The attribute dataset file (uploaded file).	File	Optional[b]
attributeDatasetURL	The attribute dataset URL.	URL	Optional[b]
attributeDatasetKey	The path to the key field in the attribute dataset that contains the key values. For csv format, this value is the column number that contains the key values (counting starts from 0).	String	Mandatory
attributeDatasetDataValueList	Paths to fields in the attribute dataset that contain the attribute values that are to be joined with the spatial dataset. For csv format, the values are the column numbers that contain the attribute values (counting starts from 0).	String, separated by commas	Mandatory
csvFileDelimiter	The delimiter character used in the csv file.	String	Optional[c]
csvFileContainsHeaderRow	Information on the existence	Boolean	Optional[d]

Name	Description	Type and Values	Required
	of header row in the csv file.		
includeJoinMetadata	Includes the joinInformation element to the response document that contains information on the successfulness of the join operation.	Boolean	Optional[e]
outputFormats	List of outputs that will be included to the response document.	String, separated by commas	Optional[f]
spatialDatasetId	The id number of the spatial dataset on the server that will be used in the join operation.	Integer	Mandatory
spatialDatasetKey	The key field of the spatial dataset that will be used in the join operation.	String	Optional[g]

[a] The TJS core module contains support for the format: 'csv'.

[b] One of the parameters: *attributeDatasetFile* or *attributeDatasetURL* is mandatory to be used with the operation. The *attributeDatasetFile* parameter can be used for uploading a attribute dataset file to the server. The *attributeDatasetURL* parameter can be used for providing the attribute dataset file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateAttributeDatasetFileInput'.

[c] The *csvFileDelimiter* paramater is mandatory to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules.

[d] The *csvFileContainsHeaderRow* paramater is optional to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules. Values: (*true*, *false*). If parameter is not provided in the request, default value *false* is used. If parameter has value *true*, csv file's header is assumed to be on the first row of the csv file and data values are assumed to start from the second row. If parameter has value *false*, the data values are assumed to start from csv file's first row.

Name	Description	Type and values	Required
[e] Values: (*true*, *false*). If parameter is not provided in the request, a default value *false* is used.			
[f] Comma-separated list of the outputs that will be included to the response document. The output formats that the server implementation supports SHALL be listed in the operation's *outputFormats* parameter description in the API description document. If the parameter value is not provided in the request, a default value 'geojson' is used.			
[g] If spatialDatasetKey parameter is not provided in the request, a default key field of the spatial dataset will be used in the join operation.			

6.3.12.2. Response

Requirement 44	/req/core/joins-joinid-put-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported. The response document contains links to the created outputs for the joined data and it MAY contain metadata on the successfulness of the data joining operation. The TJS implementations SHALL support the GeoJSON output format for the joined data and MAY support any other output formats. Other recommended output formats to be supported are WFS, WMS, OGC API Features and OGC API Maps.
C	The response document is based on the join updating with attribute dataset file schema. The response document SHALL include the following links: • a link to this document (link rel: 'self') • links to this document in other supported media types (link rel: 'alternate') **Description of properties of the join property:** **Name** / **Description** inputs — Join operation inputs joinId — Unique identifier for the join joinInformation — Information on the execution of the data join operation outputs — Links to the created outputs timestamp — Timestamp when the join has been executed

Description of properties of the inputs property:

Name	Description
attributeDataset	Name or URL of the joined csv file
spatialDataset	Link objects that contain links to spatialdataset´s different representations. Properties href, rel and type are mandatory.

Description of properties of the joinInformation property:

Name	Description
additionalAttributeKeys	List of additional keys in the csv file that were not available in the spatial dataset keys
duplicateAttributeKeys	List of duplicate keys in the csv file
matchedSpatialDatasetKeys	List of spatial dataset keys that were successfully matched with attribute data
numberOfAdditionalAttributeKeys	The number of additional attribute key values in the attribute dataset that were not available in the spatial dataset
numberOfDuplicateAttributeKeys	The number of attribute keys that had duplicate entries
numberOfMatchedSpatialDatasetKeys	The number of spatial dataset keys, to which attribute data was joined successfully
numberOfUnmatchedSpatialDatasetKeys	The number of spatial dataset keys, to which attribute data couldn´t be joined
unmatchedSpatialDatasetKeys	List of spatial dataset keys, to which attribute data couldn´t be joined

Description of properties of the outputs property:

Name	Description
format	Name of the output format
layerName	Name of the joined data layer (For WMS and WFS outputs)
link	link Link to the output

Name	Description
styleName	Name of the joined data layer style

6.3.12.3. Errors

Requirement 45	/req/core/joins-joinid-put-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The possible values for the message property are 'InvalidParameterValue', 'MissingParameterValue', 'DuplicateAttributeDatasetFileInput' and 'DuplicateStylingFileInput'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the join is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Join {joinid} not found'.
C	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.12.4. Response schema for the Join Update

```
schema:
  $ref: '#/components/schemas/JoinDataResponseObject'

JoinDataResponseObject:
  required:
  - join
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    join:
      $ref: '#/components/schemas/JoinDataObject'

JoinDataObject:
  required:
  - inputs
  - joinId
  - outputs
  - timestamp
  type: object
  properties:
    joinId:
      type: integer
    timestamp:
      type: string
      format: date-time
    inputs:
      $ref: '#/components/schemas/JoinInputsObject'
    outputs:
      type: array
      items:
```

```
        $ref: '#/components/schemas/OutputObject'
    joinInformation:
      $ref: '#/components/schemas/JoinInformationObject'

JoinInputsObject:
  required:
  - attributeDataset
  - spatialDataset
  type: object
  properties:
    attributeDataset:
      type: string
    spatialDataset:
      type: array
      items:
        $ref: '#/components/schemas/Link'

OutputObject:
  required:
  - format
  - link
  type: object
  properties:
    format:
      type: string
    layerName:
      type: string
    link:
      type: string
    styleName:
      type: string

JoinInformationObject:
  type: object
  properties:
    numberOfMatchedSpatialDatasetKeys:
      type: integer
    numberOfUnmatchedSpatialDatasetKeys:
      type: integer
    numberOfAdditionalAttributeKeys:
      type: integer
    matchedSpatialDatasetKeys:
      type: array
      items:
        type: string
    unmatchedSpatialDatasetKeys:
      type: array
      items:
        type: string
    additionalAttributeKeys:
      type: array
      items:
        type: string
    duplicateAttributeKeys:
      type: array
      items:
        type: string
    numberOfDuplicateAttributeKeys:
      type: integer
```

6.3.13. Join Delete

The HTTP DELETE operation at path `/joins/{joinid}` deletes the specific join from the server.

6.3.13.1. Request

Requirement 46	**/req/core/joins-joinid-delete-op**
A	If the server implements the *data joining operations* operation set, it MAY support the HTTP DELETE operation at the path `/joins/{joinid}`.

6.3.13.2. Response

Requirement 47	**/req/core/joins-joinid-delete-success**
A	A successful execution of the operation shall be reported as a response with a HTTP status code `204`.

	The response body SHALL be empty.

6.3.13.3. Errors

Requirement 48	/req/core/joins-joinid-delete-error
A	If the join is not found on the server, it shall be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Join {joinid} not found'.
B	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.3.13.4. Response schema for the Join Delete

The response of the operation is empty.

6.4. Operation Set: Spatial Joining Operations

The operation set *spatial joining operations* contains operations for:

- Retrieving metadata and key values of the attribute datasets that are available on the server
- Adding attribute datasets to the server
- Updating the attribute datasets on the server
- Deleting the attribute datasets from the server
- Joining attribute datasets from the server with inputted GeoJSON files

The core module contains support for attribute data files in the csv format and for spatial data files in the GeoJSON format.

The data joins are executed through common keys that are shared between the attribute dataset and the spatial dataset file.

6.4.1. Attribute Datasets

The HTTP GET operation at path `/attributedatasets` returns metadata on all attribute datasets that are available on the server.

6.4.1.1. Request

Requirement 49	/req/core/attributedatasets-get-op
A	If the server implements the *spatial joining operations* operation set it SHALL support the HTTP GET operation at path `/attributedatasets`. The server implementations SHALL support the following query parameters: **Query parameters:**

Name	Type	Description
title	String	Filters the attribute datasets whose title contains the value of the parameter
startDate	String	Filters the attribute datasets by start date. Format: yyyy-mm-dd
endDate	String	Filters the attribute datasets by end date. Format: yyyy-mm-dd

6.4.1.2. Response

Requirement 50	/req/core/attributedatasets-get-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the attributedatasets schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type. **Description of properties of the attributeDatasets property:**

Name	Description
attributeDatasetId	Unique identifier for the attribute dataset. The identifier is used in other operations for indicating the attribute dataset in question
date	Date that applies to the attribute dataset. Format: yyyy-mm-dd
description	Description of the attribute dataset
documentation	Link to the attribute dataset´s documentation
links	Links to this attribute dataset´s different representations. The links object SHALL include the properties href, rel and type. The value of the rel property SHALL be 'attributedataset'
title	Title of the attribute dataset

6.4.1.3. Errors

Requirement 51	**/req/core/attributedatasets-get-error**
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If attribute datasets are not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Attribute datasets not found'.
C	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.4.1.4. Response Schema for the Attribute Datasets

```
schema:
  $ref: '#/components/schemas/AttributeDatasetsResponseObject'

AttributeDatasetsResponseObject:
  required:
  - attributeDatasets
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    attributeDatasets:
      type: array
      items:
        $ref: '#/components/schemas/AttributeDatasetsObject'

AttributeDatasetsObject:
  required:
  - attributeDatasetId
  - date
  - links
  - title
  type: object
  properties:
    date:
      type: string
    description:
      type: string
    documentation:
      type: string
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    attributeDatasetId:
      type: integer
    title:
      type: string
```

6.4.2. Attribute Dataset Creation

The HTTP POST operation at path `/attributedatasets` adds a new attribute dataset to the server.

The TJS core module contains support for the csv format.

6.4.2.1. Request

Requirement 52	**/req/core/attributedatasets-post-op**
A	If the server implements the *spatial joining operations* operation set it MAY support the HTTP POST operation at path `/attributedatasets`. The attribute dataset file can be either uploaded to the server with the *attributeDatasetFile* parameter or provided through URL link with the *attributeDatasetURL* parameter. The request SHALL contain the header: • Content-Type: multipart/form-data; If the attribute dataset file is provided by upload, it SHALL contain the header: • Content-Disposition: form-data; filename="[attribute dataset file´s name]"; name="attributeDatasetFile"; If the attribute dataset is provided as a csv file and it contains multiple rows with the same key value, the value is used from the first row where the key is encountered.

Request´s form data parameters:

Name	Description	Type and values	Required
attributeDatasetFormat	The format of the attribute dataset	String	Mandatory[a]
attributeDatasetFile	The attribute dataset file (uploaded file)	File	Optional[b]
attributeDatasetURL	A URL link to the attribute dataset file	URL type	Optional[b]
attributeDatasetDefaultKey	The path to the key field in the attribute dataset file that contains key values. This value will be used as a default key field for the attribute dataset.	String	Mandatory[c]
attributeDatasetDefaultKeyLanguage	ISO 639-1 code of the language of the default key field values (if applicable).	String	Optional
attributeDatasetKeyTitle	The path to the field name in the attribute dataset file that contains the titles for the key field.	String	Optional[d]
attributeDatasetAdditionalKeys	A comma-separated list of paths to the additional key fields in the attribute dataset file that contain key values.	String (separated by commas).	Optional[e]
attributeDatasetAdditionalKeyLanguages	A comma-separated list of ISO 639-1 codes of the languages of the additional key field values (if applicable). The languages SHALL be in the same order than the values in the *attributeDatasetAdditionalKeys* parameter	String, (separated by commas)	Optional
attributeDatasetDataValueList	Paths to fields in the attribute dataset that contain the attribute values.	String, separated by commas	Mandatory[f]
csvFileDelimiter	The delimiter character used in the csv file.	String	Optional[g]
csvFileContainsHeaderRow	Information on the existence of header row in the csv file.	Boolean	Optional[h]
attributeDatasetTitle	Title of the attribute dataset.	String	Mandatory
attributeDatasetDescription	Description of the attribute dataset.	String	Optional

Name	Description	Type and values	Required
attributeDatasetDate	Date of the attribute dataset (format: yyyy-mm-dd). If date is not provided in the request, servers SHALL use the value of the current date.	String	Optional
attributeDatasetDocumentationLink	URL link that contains documentation for the attribute dataset.	String	Optional

[a] The TJS core module contains support for the format: 'csv'.

[b] One of the parameters: *attributeDatasetFile* or *attributeDatasetURL* is mandatory to be used with the operation. The *attributeDatasetFile* parameter can be used for uploading a csv file to the server. The *attributeDatasetURL* parameter can be used for providing the csv file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateAttributeDatasetFileInput'.

[c] For *attributeDatasetFormat* parameter value: 'csv', the value of *attributeDatasetDefaultKey* parameter SHALL be the number of the csv file's column that contains key values (counting starts from 0).

[d] For *attributeDatasetFormat* parameter value: 'csv', the value of *attributeDatasetKeyTitle* parameter SHALL be the number of the csv file's column that contains the title values for the key field (counting starts from 0).

[e] For *attributeDatasetFormat* parameter value: 'csv', the value of *attributeDatasetAdditionalKeys* parameter SHALL be a comma-separated list of csv file's column numbers that contain the key values (counting starts from 0).

[f] For *attributeDatasetFormat* parameter value: 'csv', the value of *attributeDatasetDataValueList* parameter SHALL be a comma-separated list of csv file's column numbers that contain the data values that will be joined with the spatial dataset file (counting starts from 0).

[g] The *csvFileDelimiter* paramater is mandatory to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules.

[h] The *csvFileContainsHeaderRow* paramater is optional to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules. Values: (*true*, *false*). If parameter is not provided in the request, default value *false* is used. If parameter has value *true*, csv file's header is assumed to be on the first row of the csv file and data values are assumed to start from the second row. If parameter has value *false*, the data values are assumed to start from csv file's first row.

6.4.2.2. Response

Requirement 53	/req/core/attributedatasets-post-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `201`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.

C	The response document is based on the attributedatasets post schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type. The response property: attributeDataset SHALL contain metadata on the created attribute dataset. **Description of properties of the attributeDataset property:** **Name** / **Description** attributeDatasetId — Unique identifier for the attribute dataset. attributes — A list of attributes available in the attribute dataset date — Date that applies to the attribute dataset. Format: (yyyy-mm-dd). description — Description of the attribute dataset. documentation — Link to the attribute dataset´s documentation. keys — Links to the different representations of this attribute dataset´s key fields. The links SHALL include the properties href, rel and type. The value of the rel property SHALL be 'keys'. title — Title of the attribute dataset.

6.4.2.3. Errors

Requirement 54	/req/core/attributedatasets-post-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.4.2.4. Response Schema for the Attribute Dataset Creation

```
schema:
  $ref: '#/components/schemas/AttributeDatasetResponseObject'

AttributeDatasetResponseObject:
  required:
  - attributeDataset
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    attributeDataset:
      $ref: '#/components/schemas/AttributeDatasetObject'

AttributeDatasetObject:
  required:
  - attributeDatasetId
  - attributes
  - date
  - keys
  - title
  type: object
  properties:
    date:
      type: string
    description:
      type: string
    documentation:
      type: string
    keys:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    attributeDatasetId:
      type: integer
    title:
      type: string
    attributes:
      type: array
      items:
      type: string
```

6.4.3. Attribute Dataset

The HTTP GET operation at path `/attributedatasets/{attributedatasetid}` returns metadata on a specific attribute dataset available on the server.

6.4.3.1. Request

Requirement 55	**/req/core/attributedatasets-attributedatasetid-get-op**
A	If the server implements the *spatial joining operations* operation set it SHALL support the HTTP GET operation at path `/attributedatasets/{attributedatasetid}`.

6.4.3.2. Response

Requirement 56	**/req/core/attributedatasets-attributedatasetid-get-success**
A	A successful execution of the operation shall be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the attributedataset schema. The response document SHALL include the following links:

- A link to this document (link rel: 'self')
- Links to this document in other supported media types (link rel: 'alternate')

The links SHALL contain the parameters href, rel, title and type.

The response property: attributeDataset SHALL contain metadata on the requested attribute dataset.

Description of properties of the attributeDataset property:

Name	Description
attributeDatasetId	Unique identifier for the attribute dataset.
attributes	A list of attributes available in the attribute dataset.
date	Date that applies to the attribute dataset. Format: yyyy-mm-dd.
description	Description of the attribute dataset.
documentation	Link to the attribute dataset´s documentation.
keys	Links to the different representations of this attribute dataset´s key fields. The links SHALL include the properties href, rel and type. The value of the rel property SHALL be 'keys'.
title	Title of the attribute dataset.

6.4.3.3. Errors

Requirement 57	/req/core/attributedatasets-attributedatasetid-get-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the attribute dataset is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Attribute dataset {attributedatasetid} not found'.
C	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`.

Requirement 57	/req/core/attributedatasets-attributedatasetid-get-error
	The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.4.3.4. Response Schema for the Attribute Dataset

```
schema:
  $ref: '#/components/schemas/AttributeDatasetResponseObject'

AttributeDatasetResponseObject:
  required:
  - attributeDataset
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    attributeDataset:
      $ref: '#/components/schemas/AttributeDatasetObject'

AttributeDatasetObject:
  required:
  - attributeDatasetId
  - attributes
  - date
  - keys
  - title
  type: object
  properties:
    date:
      type: string
    description:
      type: string
    documentation:
      type: string
    keys:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    attributeDatasetId:
      type: integer
    title:
      type: string
    attributes:
      type: array
      items:
      type: string
```

6.4.4. Attribute Dataset Update

The HTTP PUT operation at path `/attributedatasets/{attributedatasetid}` updates fully the specific attribute dataset available on the server.

The TJS core module contains support for updating the attribute dataset from csv files.

6.4.4.1. Request

Requirement 58	/req/core/attributedatasets-attributedataset-put-op
A	If the server implements the *spatial joining operations* operation set, it MAY support the HTTP PUT operation at path `/attributedatasets/{attributedatasetid}`. The attribute dataset file can be either uploaded to the server with the *attributeDatasetFile* parameter or provided through URL link with the *attributeDatasetURL* parameter. The request SHALL contain the header: • Content-Type: multipart/form-data;

If the attribute dataset file is provided by upload, it SHALL contain the header:

- Content-Disposition: form-data; filename="[attribute dataset file´s name]"; name="attributeDatasetFile";

If the attribute dataset is provided as a csv file and it contains multiples rows with same key values the value is used from the first row where the key is encountered.

Request´s form data parameters:

Name	Description	Type and values	Required
attributeDatasetFormat	The format of the attribute dataset	String	Mandatory[a]
attributeDatasetFile	The attribute dataset file (uploaded file)	File	Optional[b]
attributeDatasetURL	A URL link to the attribute dataset file	URL type	Optional[b]
attributeDatasetDefaultKey	The path to the key field in the attribute dataset file that contains key values. This value will be used as a default key field for the attribute dataset.	String	Mandatory[c]
attributeDatasetDefaultKeyLanguage	ISO 639-1 code of the language of the default key field values (if applicable).	String	Optional
attributeDatasetKeyTitle	The path to the field name in the attribute dataset file that contains the titles for the key field.	String	Optional[d]
attributeDatasetAdditionalKeys	A comma-separated list of paths to the additional key fields in the attribute dataset file that contain key values.	String (separated by commas).	Optional[e]
attributeDatasetAdditionalKeyLanguages	A comma-separated list of ISO 639-1 codes of the languages of the additional key field values (if applicable). The languages SHALL be in the same order than the values in the *attributeDatasetAdditionalKeys* parameter	String, (separated by commas)	Optional
attributeDatasetDataValueList	Paths to fields in the attribute dataset that contain the attribute values.	String, separated by commas	Mandatory[f]
csvFileDelimiter	The delimiter character used in the	String	Optional[g]

csv file.

Name	Description	Type and values	Required
csvFileContainsHeaderRow	Information on the existence of header row in the csv file.	Boolean	Optional[h]
attributeDatasetTitle	Title of the attribute dataset.	String	Mandatory
attributeDatasetDescription	Description of the attribute dataset.	String	Optional
attributeDatasetDate	Date of the attribute dataset (format: yyyy-mm-dd). If date is not provided in the request, servers SHALL use the value of the current date.	String	Optional
attributeDatasetDocumentationLink	URL link that contains documentation for the attribute dataset.	String	Optional

[a] The TJS core module contains support for the format: 'csv'.

[b] One of the parameters: *attributeDatasetFile* or *attributeDatasetURL* is mandatory to be used with the operation. The *attributeDatasetFile* parameter can be used for uploading a csv file to the server. The *attributeDatasetURL* parameter can be used for providing the csv file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateAttributeDatasetFileInput'.

[c] For *attributeDatasetFormat* parameter value: 'csv', the value of *attributeDatasetDefaultKey* parameter SHALL be the number of the csv file's column that contains key values (counting starts from 0).

[d] For *attributeDatasetFormat* parameter value: 'csv', the value of *attributeDatasetKeyTitle* parameter SHALL be the number of the csv file's column that contains the title values for the key field (counting starts from 0).

[e] For *attributeDatasetFormat* parameter value: 'csv', the value of *attributeDatasetAdditionalKeys* parameter SHALL be a comma-separated list of csv file's column numbers that contain the key values (counting starts from 0).

[f] For *attributeDatasetFormat* parameter value: 'csv', the value of *attributeDatasetDataValueList* parameter SHALL be a comma-separated list of csv file's column numbers that contain the data values that will be joined with the spatial dataset file (counting starts from 0).

[g] The *csvFileDelimiter* paramater is mandatory to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules.

[h] The *csvFileContainsHeaderRow* paramater is optional to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules. Values: (*true*, *false*). If parameter is not provided in the request, default value *false* is used. If parameter has value *true*, csv file's header is assumed to be on the first row of the csv file and data values are assumed to start from the second row. If parameter has value *false*, the data values are assumed to start from csv file's first row.

6.4.4.2. Response

<table>
<tr><th>Requirement 59</th><th>/req/core/attributedatasets-attributedatasetid-put-success</th></tr>
<tr><td>A</td><td>A successful execution of the operation SHALL be reported as a response with a HTTP status code <code>200</code>.</td></tr>
<tr><td>B</td><td>The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.</td></tr>
<tr><td>C</td><td>The response document is based on the attributedatasets attributedatasetid put schema.

The response document SHALL include the following links:

• A link to this document (link rel: 'self')
• Links to this document in other supported media types (link rel: 'alternate')

The links SHALL contain the parameters href, rel, title and type.

The response property: attributeDataset SHALL contain metadata on the updated attribute dataset.

Description of properties of the attributeDataset property:

<table>
<tr><th>Name</th><th>Description</th></tr>
<tr><td>attributeDatasetId</td><td>Unique identifier for the attribute dataset.</td></tr>
<tr><td>attributes</td><td>A list of attributes available in the attribute dataset</td></tr>
<tr><td>date</td><td>Date that applies to the attribute dataset. Format: (yyyy-mm-dd).</td></tr>
<tr><td>description</td><td>Description of the attribute dataset.</td></tr>
<tr><td>documentation</td><td>Link to the attribute dataset´s documentation.</td></tr>
<tr><td>keys</td><td>Links to the different representations of this attribute dataset´s key fields. The links SHALL include the properties href, rel and type. The value of the rel property SHALL be 'keys'.</td></tr>
<tr><td>title</td><td>Title of the attribute dataset.</td></tr>
</table>
</td></tr>
</table>

6.4.4.3. Errors

<table>
<tr><th>Requirement 60</th><th>/req/core/attributedatasets-attributedatasetid-put-error</th></tr>
<tr><td>A</td><td>If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code <code>400</code>.

The response body SHALL contain an exception report message in the JSON</td></tr>
</table>

Requirement 60	format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
	/req/core/attributedatasets-attributedatasetid-put-error
B	If the attribute dataset is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Attribute dataset {attributedatasetid} not found'.
C	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.4.4.4. Response Schema for the Attribute Dataset Update

```
schema:
  $ref: '#/components/schemas/AttributeDatasetResponseObject'

AttributeDatasetResponseObject:
  required:
  - attributeDataset
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    attributeDataset:
      $ref: '#/components/schemas/AttributeDatasetObject'

AttributeDatasetObject:
  required:
  - attributeDatasetId
  - attributes
  - date
  - keys
  - title
  type: object
  properties:
    date:
      type: string
    description:
      type: string
    documentation:
      type: string
    keys:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    attributeDatasetId:
      type: integer
    title:
      type: string
    attributes:
      type: array
      items:
      type: string
```

6.4.5. Attribute Dataset Delete

The HTTP DELETE operation at path `/attributedatasets/{attributedatasetid}` deletes the specific attribute dataset from the server.

6.4.5.1. Request

Requirement 61	/req/core/attributedatasets-attributedatasetid-delete-op
A	If the server implements the *spatial joining operations* operation set, it MAY support the HTTP DELETE operation at the path `/attributedatasets/{attributedatasetid}`.

6.4.5.2. Response

Requirement 62	/req/core/attributedatasets-attributedatasetid-delete-success
A	A successful execution of the operation shall be reported as a response with a HTTP status code `204`. The response body SHALL be empty.

6.4.5.3. Errors

Requirement 63	/req/core/attributedatasets-attributedatasetid-delete-error
A	If the attribute dataset is not found on the server, it shall be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Attribute dataset {attributedatasetid} not found'.
B	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.4.5.4. Response Schema for the Attribute Dataset Delete

The response of the operation is empty.

6.4.6. Attribute Dataset Key Fields

The HTTP GET operation at path `/attributedatasets/{attributedatasetid}/keys` returns a list of key fields that belong to a specific attribute dataset.

6.4.6.1. Request

Requirement 64	/req/core/attributedatasets-attributedatasetid-keys-op
A	If the server implements the *spatial joining operations* operation set, it SHALL support the HTTP GET operation at the path `/attributedatasets/{attributedatasetid}/keys`.

6.4.6.2. Response

Requirement 65	/req/core/attributedatasets-attributedatasetid-keys-success
A	A successful execution of the operation SHALL be reported as a response with a

	HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the attributedataset keys schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type. **Description of properties of the keys property:** **Name** / **Description** isDefault: Indicates if the key is used as a default key field in the data joins with this attribute dataset. Only one object in the response SHALL have the value 'true'. keyLanguage: ISO 639-1 code of the language of the key field values (if applicable). keyName: Name of the key field. links: Links to the different representations of this key value. The links SHALL have the properties href, rel and type. The value of the rel property SHALL be 'keyname'.

6.4.6.3. Errors

Requirement 66	**/req/core/attributedataset-attributedatasetid-keys-error**
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the attribute dataset keys are not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Attribute dataset {attributedatasetid} keys not found'.
B	If there is an error in the server during the processing of the request, it SHALL be

Requirement 66	/req/core/attributedataset-attributedatasetid-keys-error
	The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

reported as a response with a HTTP status code `500`.

6.4.6.4. Response Schema for the Attribute Dataset Key Fields

```
schema:
  $ref: '#/components/schemas/AttributeDatasetKeysResponseObject'

AttributeDatasetKeysResponseObject:
  required:
  - keys
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    keys:
      type: array
      items:
        $ref: '#/components/schemas/AttributeDatasetKeysObject'

AttributeDatasetKeysObject:
  required:
  - isDefault
  - keyName
  - links
  type: object
  properties:
    isDefault:
      type: boolean
    keyLanguage:
      type: string
    keyName:
      type: string
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
```

6.4.7. Attribute Dataset Key Field

The HTTP GET operation at path `/attributedatasets/{attributedatasetid}/keys/{keyname}` returns a list of key values from a specific key field of a specific attribute dataset.

6.4.7.1. Request

Requirement 67	/req/core/attributedatasets-attributedatasetid-keys-keyname-op
A	If the server implements the *spatial joining operations* operation set, it SHALL support the HTTP GET operation at the path `/attributedatasets/{attributedatasetid}/keys/{keyname}`.

6.4.7.2. Response

Requirement 68	/req/core/attributedatasets-attributedatasetid-keys-keyname-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the attributedataset key field schema.

The response document SHALL include the following links:

- A link to this document (link rel: 'self')
- Links to this document in other supported media types (link rel: 'alternate')

The links SHALL contain the parameters href, rel, title and type.

Description of properties of the keys property:

Name	Description
key	Attribute dataset´s key field´s key value.
links	Links to the different representations of this key value. The links SHALL have the properties href, rel and type. The value of the rel property SHALL be 'key'.
title	Human-readable description of the key value.

6.4.7.3. Errors

Requirement 69	/req/core/attributedatasets-attributedatasetid-keys-keyname-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the attribute dataset´s key field´s key values are not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Attribute dataset {attributedatasetid} key field {keyname} not found'.
C	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.4.7.4. Response Schema for the Attribute Dataset Key Field

```
schema:
  $ref: '#/components/schemas/AttributeDatasetKeysKeynameResponseObject'

AttributeDatasetKeysKeynameResponseObject:
  required:
  - keys
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    keys:
      type: array
      items:
        $ref: '#/components/schemas/AttributeDatasetKeysKeynameObject'

AttributeDatasetKeysKeynameObject:
  required:
  - key
  - links
  type: object
  properties:
    key:
      type: string
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    title:
      type: string
```

6.4.8. Attribute Dataset Key Field Key

The HTTP GET operation at path `/attributedatasets/{attributedatasetid}/keys/{keyname}/{key}` returns a specific key value, from a specific key field from a specific attribute dataset.

6.4.8.1. Request

Requirement 70	**/req/core/attributedatasets-attributedatasetid-keys-keyname-key-op**
A	If the server implements the *spatial joining operations* operation set it SHALL support the HTTP GET operation at the path `/attributedatasets/{attributedatasetid}/keys/{keyname}/{key}`.

6.4.8.2. Response

Requirement 71	**/req/core/attributedatasets-attributedatasetid-keys-keyname-key-success**
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`.
B	The server implementations SHALL support the JSON output format. Any other output formats MAY also be supported.
C	The response document is based on the attributedataset key field key schema. The response document SHALL include the following links: • A link to this document (link rel: 'self') • Links to this document in other supported media types (link rel: 'alternate') The links SHALL contain the parameters href, rel, title and type **Description of properties of the key response property:**

Name	Description
key	Attribute dataset´s key field´s key value
title	Human-readable description of the key value

6.4.8.3. Errors

Requirement 72	/req/core/attributedatasets-attributedatasetid-keys-keyname-key-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'InvalidParameterValue'. The locator property SHALL contain the name of the request parameter that caused the exception.
B	If the attribute dataset key value is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Attribute dataset {attributedatasetid} key field {keyname} key {key} not found'.
C	If there is an error in the server during the processing of the request, it SHALL be reported as a response with a HTTP status code `500`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.4.8.4. Response Schema for the Attribute Dataset Key Field Key

```
schema:
  $ref: '#/components/schemas/AttributeDatasetKeysKeynameKeyResponseObject'

AttributeDatasetKeysKeynameKeyResponseObject:
  required:
  - key
  - links
  type: object
  properties:
    links:
      type: array
      items:
        $ref: '#/components/schemas/Link'
    key:
      $ref: '#/components/schemas/AttributeDatasetKeysKeynameKeyObject'

AttributeDatasetKeysKeynameKeyObject:
  required:
  - key
  type: object
  properties:
    key:
      type: string
    title:
      type: string
```

6.4.9. Join Attribute Dataset with a Spatial Dataset File

The HTTP POST operation at path `/joinattributedataset` joins a specific attribute dataset available on the server with an inputted spatial dataset file.

The TJS core module contains support for attribute dataset joining with GeoJSON files.

6.4.9.1. Request

<table>
<tr><th>Requirement 73</th><th>/req/core/joinattributedataset-op</th></tr>
<tr><td>A</td><td>
If the server supports the spatial joining operations operation set it SHALL support the HTTP POST operation at the path <code>/joinattributedataset</code>.

The spatial dataset file can be either uploaded to the server with the spatialDatasetFile parameter or provided through URL link with the spatialDatasetURL parameter.

The request SHALL contain the header:

• Content-Type: multipart/form-data;

If the spatial dataset file is provided by upload, it SHALL contain the header:

• Content-Disposition: form-data; filename="[spatial dataset file´s name]"; name="spatialDatasetFile";

Request´s form data parameters:
<table>
<tr><th>Name</th><th>Description</th><th>Type and values</th><th>Required</th></tr>
<tr><td>spatialDatasetFormat</td><td>The format of the spatial dataset.</td><td>String</td><td>Mandatory[a]</td></tr>
<tr><td>spatialDatasetFile</td><td>The spatial dataset file (uploaded file)</td><td>File</td><td>Optional[b]</td></tr>
<tr><td>spatialDatasetURL</td><td>A URL link to the spatial dataset file</td><td>URL type</td><td>Optional[b]</td></tr>
<tr><td>spatialDatasetKey</td><td>The path to the key field in the spatial dataset file that contains key values. Example: 'features.properties.kunta'</td><td>String</td><td>Mandatory</td></tr>
<tr><td>attributeDatasetId</td><td>The id number of the attribute dataset</td><td>Integer</td><td>Mandatory</td></tr>
<tr><td>attributeDatasetKey</td><td>The key field of the attribute dataset that will be used in the join operation.</td><td>String</td><td>Optional[c]</td></tr>
</table>
</td></tr>
</table>

Name	Description	Type and values	Required
attributes	Comma-separated list of attributes that will be joined with the spatial dataset file.	String	Optional[d]

[a] The TJS core module contains support for the format: 'geojson'.

[b] One of the parameters: *spatialDatasetFile* or *spatialDatasetURL* is mandatory to be used with the operation. The *spatialDatasetFile* parameter can be used for uploading the spatial dataset file to the server. The *spatialDatasetURL* parameter can be used for providing the spatial dataset file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateSpatialDatasetFileInput'.

[c] If *attributeDatasetKey* parameter is not provided in the request, a default key field of the attribute dataset will be used in the join operation.

[d] If *attributes* parameter is not provided or it is empty, all attributes of the attribute dataset will be joined.

6.4.9.2. Response

Requirement 74	/req/core/joinattributedataset-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`. The server SHALL return the spatial dataset file that includes the joined attributes from the attribute dataset.

6.4.9.3. Errors

Requirement 75	/req/core/joinattributedataset-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The Possible values for the message property are 'InvalidParameterValue', 'MissingParameterValue', 'DuplicateSpatialDatasetFileInput'.
B	If the attribute dataset is not found on the server, it SHALL be reported as a response with a HTTP status code `404`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Attribute dataset {attributedatasetid} not found'.
C	If there is an error in the server during the processing of the request, it shall be reported as a response with a HTTP status code `500`.

Requirement 75	/req/core/joinattributedataset-error
	The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.4.9.4. Response schema for the Attribute Dataset Join With Spatial Dataset File

The response of the operation SHALL be the spatial dataset file that contains the joined attributes from the attribute dataset.

6.5. Operation Set: File Joining Operations

The operation set *file joining operations* contains an operation for joining attribute data from an inputted attribute dataset file with an inputted spatial dataset file.

The TJS core module contains support for attribute dataset files in the csv format and spatial dataset files in the GeoJSON format.

6.5.1. Data Joining from an Attribute Dataset File with a Spatial Dataset File

The HTTP POST operation at path `/joinfiles` joins data from an inputted attribute dataset file to an inputted spatial dataset file.

6.5.1.1. Request

Requirement 76	/req/core/joinfiles-op
A	If the server supports the *file joining operations* operation set it SHALL support the HTTP POST operation at the path `/joinfiles`. The spatial dataset file can be either uploaded to the server with the *spatialDatasetFile* parameter or provided through URL link with the *spatialDatasetURL* parameter. The attribute dataset file can be either uploaded to the server with the *attributeDatasetFile* parameter or provided through URL link with the *attributeDatasetURL* parameter. The request SHALL contain the header: • Content-Type: multipart/form-data; If the spatial dataset file is provided by upload, it SHALL contain the header: • Content-Disposition: form-data; filename="[spatial dataset file´s name]"; name="spatialDatasetFile"; If the attribute dataset file is provided by upload, it SHALL contain the header: • Content-Disposition: form-data; filename="[attribute dataset file´s name]"; name="attributeDatasetFile"; If the attribute dataset is provided as a csv file and it contains multiple rows with the same key value, the value is used in the join operation from the first row where the key is encountered. **Request´s form data parameters:**

Name	Description	Type and values	Required
attributeDatasetFormat	The format of the attribute dataset.	String	Mandatory[a]
attributeDatasetFile	The attribute dataset file (uploaded file).	File	Optional[b]
attributeDatasetURL	The attribute dataset URL.	URL	Optional[b]
attributeDatasetKey	The path to the key field in the attribute dataset that contains the key values. For csv format, this value is the column number that contains the key values (counting starts from 0).	String	Mandatory
attributeDatasetDataValueList	Paths to fields in the attribute dataset that contain the attribute values that are to be joined with the spatial dataset. For csv format, the values are the column numbers that contain the attribute values (counting starts from 0).	String, separated by commas	Mandatory
csvFileDelimiter	The delimiter character used in the csv file.	String	Optional[c]
csvFileContainsHeaderRow	Information on the existence of header row in the csv file.	Boolean	Optional[d]
spatialDatasetFormat	The format of the spatial dataset.	String	Mandatory[e]
spatialDatasetFile	The spatial dataset file (uploaded file)	File	Optional[f]
spatialDatasetURL	A URL link to the spatial dataset file	URL type	Optional[f]
spatialDatasetKey	The path to the key field in the spatial dataset file that contains key values. Example: 'features.properties.kunta'	String	Mandatory

Name	Description	Type and values	Required
[a] The TJS core module contains support for the format: 'csv'. [b] One of the parameters: *attributeDatasetFile* or *attributeDatasetURL* is mandatory to be used with the operation. The *attributeDatasetFile* parameter can be used for uploading a attribute dataset file to the server. The *attributeDatasetURL* parameter can be used for providing the attribute dataset file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateAttributeDatasetFileInput'. [c] The *csvFileDelimiter* paramater is mandatory to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules. [d] The *csvFileContainsHeaderRow* paramater is optional to be used with the *attributeDatasetFormat* parameter value: 'csv'. The parameter is not required for other formats that may be defined in the extension modules. Values: (*true*, *false*). If parameter is not provided in the request, default value *false* is used. If parameter has value *true*, csv file's header is assumed to be on the first row of the csv file and data values are assumed to start from the second row. If parameter has value *false*, the data values are assumed to start from csv file's first row. [e] The TJS core module contains support for the format: 'geojson'. [f] One of the parameters: *spatialDatasetFile* or *spatialDatasetURL* is mandatory to be used with the operation. The *spatialDatasetFile* parameter can be used for uploading the spatial dataset file to the server. The *spatialDatasetURL* parameter can be used for providing the spatial dataset file through URL link. If both parameters are provided in the query, the server SHALL send an exception with message 'DuplicateSpatialDatasetFileInput'.			

6.5.1.2. Response

Requirement 77	/req/core/joinfiles-success
A	A successful execution of the operation SHALL be reported as a response with a HTTP status code `200`. The server SHALL return the spatial dataset file that includes the joined attributes from the attribute dataset file.

6.5.1.3. Errors

Requirement 78	/req/core/joinfiles-error
A	If an incorrect request is made to the server, it SHALL be reported as a response with a HTTP status code `400`. The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The Possible values for the message property are 'InvalidParameterValue', 'MissingParameterValue', 'DuplicateAttributeDatasetFileInput' and 'DuplicateSpatialDatasetFileInput'.
B	If there is an error in the server during the processing of the request, it shall be

Requirement 78	reported as a response with a HTTP status code `500`. **/req/core/joinfiles-error**
	The response body SHALL contain an exception report message in the JSON format that is based on exception message schema. The value of the message property SHALL be 'Internal server error'.

6.5.1.4. Response Schema for the File Joining

The response of the operation SHALL be the spatial dataset file that contains the joined attributes from the attribute dataset file.

---

7. Media Types for any data encoding(s)

7.1. Operation Sets: discovery operations and data joining operations

Requirement 79	**/req/core/discovery-operations-and-data-joining-operations-outputs**
A	The server implementations SHALL support the JSON output format for all operations in the operation sets: *discovery operations* and *data joining operations*. Any other output formats MAY also be supported.

7.2. Outputs for the Joined Ddta in Operation Set: data joining operations

Requirement 80	**/req/core/data-joining-operations-joined-data-outputs**
A	The server implementations SHALL support the GeoJSON format for the joined data in the operation set *data joining operations*. Any other formats for the joined data MAY also be supported. Other recommended output formats to be supported are WFS and WMS, OGC API Features and OGC API Maps.

7.3. Operation Sets: spatial joining operations and file joining operations

Requirement 81	**/req/core/spatial-joining-operations-and-file-joining-operations-output**
A	The server implementations SHALL support the GeoJSON output format for the joined data in the operation sets: *spatial joining operations* and *file joining operations*.

Annex A: Abstract Test Suite (Normative)

A.1. Conformance Class "Core"

Conformance Class	
http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/core	
Target type	Web API
Requirements class	Requirements Class "Core"

A.1.1. Landing Page {root}/

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

Abstract Test 2	**/ats/core/root-success**
Test Purpose	Validate that a landing page complies with the required structure and contents.
Requirement	/req/core/root-success
Test Method	Validate the landing page for all supported media types using the landing page schema. • Validate that the landing page includes a 'service-desc' and/or 'service-doc' link to an API Definition • Validate that the landing page includes a 'conformance' link to the conformance class declaration For servers that implement the *data joining operations* operation set: • Validate that the landing page includes a 'spatialdatasets' link to the spatialdatasets metadata • Validate that the landing page includes a "joins' link to the joins

A.1.2. API Definiton path {root}/api

Abstract Test 3	**/ats/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 Method	1. Issue a HTTP GET request to the URL `{root}/api`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/api-definition-success

Abstract Test 4	**/ats/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.

A.1.3. Conformance {root}/conformance

Abstract Test 5	**/ats/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. Issue an HTTP GET request to the URL `{root}/conformance`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/conformance-success

Abstract Test 6	**/ats/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 conformance schema. 2. Validate that the document includes the conformance class 'http://www.opengis.net/spec/tjs/2.0/conf/core' 3. Validate that the document lists all other OGC API conformance classes the server implements

A.2. Conformance class: Core / Data Joining

Conformance Class	
http://www.opengis.net/spec/ogcapi-tjs/1.0/conf/core/data-joining	
Target type	Web API

Requirements class	Requirements Class "Core"

A.2.1. Spatial Datasets {root}/spatialdatasets

Abstract Test 7	**/ats/core/spatialdatasets-get-op**
Test Purpose	Validate that the information about spatial datasets can be retrieved from the expected location.
Requirement	/req/core/spatialdatasets-get-op
Test Method	1. Issue an HTTP GET request to the URL `{root}/spatialdatasets`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/spatialdatasets-get-success

Abstract Test 8	**/ats/core/spatialdatasets-get-success**
Test Purpose	Validate that the spatial datasets content complies with the required structure and contents.
Requirement	/req/core/spatialdatasets-get-success
Test Method	1. Validate that the response document complies with spatialdatasets schema.

A.2.2. Spatial Dataset Creation {root}/spatialdatasets

Abstract Test 9	**/ats/core/spatialdatasets-post-op**
Test Purpose	Validate that the spatial dataset can be added to the server from the expected location.
Requirement	/req/core/spatialdatasets-post-op
Test Method	1. Issue an HTTP POST request to the URL `{root}/spatialdatasets`. 2. Validate that a document was returned with a status code `201`. 3. Validate the contents of the returned document using test /ats/core/spatialdatasets-post-success

Abstract Test 10	**/ats/core/spatialdatasets-post-success**
Test Purpose	Validate that the spatial dataset's creation response document complies with the required structure and contents.
Requirement	/req/core/spatialdatasets-post-success
Test Method	1. Validate that the response document complies with spatialdatasets-spatialdatasetid schema.

A.2.3. Spatial Dataset {root}/spatialdatasets/{spatialdatasetid}

Abstract Test 11	/ats/core/spatialdatasets-spatialdatasetid-get-op
Test Purpose	Validate that a spatial dataset information can be retrieved from the expected location.
Requirement	/req/core/spatialdatasets-spatialdatasetid-get-op
Test Method	1. For a list of all spatial datasets (path `{root}/spatialdatasets`), issue an HTTP GET request to the URL `{root}/spatialdatasets/{spatialdatasetid}` where {spatialdatasetid} is the spatialDatasetId property of a spatial dataset. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/spatialdatasets-spatialdatasetid-get-success

Abstract Test 12	/ats/core/spatialdatasets-spatialdatasetid-get-success
Test Purpose	Validate that the spatial dataset complies with the required structure and contents.
Requirement	/req/core/spatialdatasets-spatialdatasetid-get-success
Test Method	1. Validate that the response document complies with spatialdataset schema.

A.2.4. Spatial Dataset Update {root}/spatialdatasets/{spatialdatasetid}

Abstract Test 13	/ats/core/spatialdatasets-spatialdatasetid-put-op
Test Purpose	Validate that the spatial dataset can be updated from the expected location.
Requirement	/req/core/spatialdatasets-spatialdatasetid-put-op
Test Method	1. Issue an HTTP PUT request to the URL `{root}/spatialdatasets/{spatialdatasetid}`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/spatialdatasets-spatialdatasetid-put-success

Abstract Test 14	/ats/core/spatialdatasets-spatialdatasetid-put-success
Test Purpose	Validate that the response document of the spatial dataset's update complies with the required structure and contents.
Requirement	/req/core/spatialdatasets-spatialdatasetid-put-success
Test Method	1. Validate that the response document complies with spatialdatasets-spatialdatasetid schema.

A.2.5. Spatial Dataset Delete {root}/spatialdatasets/{spatialdatasetid}

Abstract Test 15	/ats/core/spatialdatasets-spatialdatasetid-delete-op
Test Purpose	Validate that the spatial dataset can be deleted from the expected location.

Requirement	/req/core/spatialdatasets-spatialdatasetid-delete-op
Test Method	1. Issue an HTTP DELETE request to the URL `{root}/spatialdatasets/{spatialdatasetid}` where {spatialdatasetid} is the spatialDatasetId property of a spatial dataset (from query `{root}/spatialdatasets/{spatialdataset}`). 2. Validate that a document was returned with a status code `204`. 3. Validate the contents of the returned document using test /ats/core/spatialdatasets-spatialdatasetid-delete-success

Abstract Test 16	**/ats/core/spatialdatasets-spatialdatasetid-delete-success**
Test Purpose	Validate that the spatial dataset was deleted from the server.
Requirement	/req/core/spatialdatasets-spatialdatasetid-delete-success
Test Method	1. Validate that the spatial dataset has been deleted from the server by issuing an HTTP GET request to the URL `{root}/spatialdatasets/{spatialdatasetid}` where {spatialdatasetid} is the same spatialDatasetId property of the spatial dataset that was used in the delete request. 2. Validate that the server sent a response code `404`.

A.2.6. Spatial Dataset keys {root}/spatialdatasets/{spatialdatasetid}/keys

Abstract Test 17	**/ats/core/spatialdatasets-spatialdatasetid-keys-op**
Test Purpose	Validate that the information on spatial dataset key fields can be retrieved from the expected location.
Requirement	/req/core/spatialdatasets-spatialdatasetid-keys-op
Test Method	1. For a spatial dataset (path `{root}/spatialdatasets/{spatialdatasetid}`), issue an HTTP GET request to the URL `{root}/spatialdatasets/{spatialdatasetid}/keys` where {spatialdatasetid} is the spatialDatasetId property of a spatial dataset. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/spatialdatasets-spatialdatasetid-keys-success

Abstract Test 18	**/ats/core/spatialdatasets-spatialdatasetid-keys-success**
Test Purpose	Validate that the spatial datasets keys content complies with the required structure and contents.
Requirement	/req/core/spatialdatasets-spatialdatasetid-keys-success
Test Method	1. Validate that the response document complies with spatialdataset key fields schema.

A.2.7. Spatial Dataset key field {root}/spatialdatasets/{spatialdatasetid}/keys/{keyname}

Abstract Test 19	**/ats/core/spatialdatasets-spatialdatasetid-keys-keyname-op**
Test Purpose	Validate that the information on keys that belong to the spatial dataset´s key field can be retrieved from the expected location.
Requirement	/req/core/spatialdatasets-spatialdatasetid-keys-keyname-op
Test Method	1. For a spatial dataset (path `{root}/spatialdatasets/{spatialdatasetid}`), issue an HTTP GET request to the URL `{root}/spatialdatasets/{spatialdatasetid}/keys/{keyname}` where {spatialdatasetid} is the spatialDatasetId property of a spatial dataset and {keyname} is the keyName property of the spatial dataset key field. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/spatialdatasets-spatialdatasetid-keys-keyname-success

Abstract Test 20	**/ats/core/spatialdatasets-spatialdatasetid-keys-keyname-success**
Test Purpose	Validate that the spatial dataset key field´s contents comply with the required structure and contents.
Requirement	/req/core/spatialdatasets-spatialdatasetid-keys-keyname-success
Test Method	Validate that the response document complies with spatial dataset key field schema.

A.2.8. Spatial Dataset key field key {root}/spatialdatasets/{spatialdatasetid}/keys/{keyname}/{key}

Abstract Test 21	**/ats/core/spatialdatasets-spatialdatasetid-keys-keyname-key-op**
Test Purpose	Validate that the spatial dataset´s key field´s key information can be retrieved from the expected location.
Requirement	/req/core/spatialdatasets-spatialdatasetid-keys-keyname-key-op
Test Method	1. For a spatial dataset (path `{root}/spatialdatasets/{spatialdatasetid}`), issue an HTTP GET request to the URL `{root}/spatialdatasets/{spatialdatasetid}/keys/{keyname}/{key}` where {spatialdatasetid} is the spatialDatasetId property of a spatial dataset, {keyname} is the keyName property of the spatial dataset key field and {key} is the name of the key property of the spatial dataset key value. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/spatialdatasets-spatialdatasetid-keys-keyname-key-success

Abstract Test 22	**/ats/core/spatialdatasets-spatialdatasetid-keys-keyname-key-success**
Test Purpose	Validate that the spatial dataset´s key content complies with the required structure and contents.

Requirement	/req/core/spatialdatasets-spatialdatasetid-keys-keyname-key-success
Test Method	Validate that the response document complies with the spatial dataset key field key schema.

A.2.9. Joins {root}/joins

Abstract Test 23	**/ats/core/joins-op**
Test Purpose	Validate that the information about joins can be retrieved from the expected location.
Requirement	/req/core/joins-op
Test Method	1. Issue an HTTP GET request to the URL `{root}/joins`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/joins-success

Abstract Test 24	**/ats/core/joins-success**
Test Purpose	Validate that the joins content complies with the required structure and contents.
Requirement	/req/core/joins-success
Test Method	Validate that the response document complies with joins schema.

A.2.10. Spatial Dataset joining with Attribute Dataset File {root}/joins

Abstract Test 25	**/ats/core/joins-post-op**
Test Purpose	Validate that the data can be joined from a attribute dataset file with a specific spatial dataset from expected location.
Requirement	/req/core/joins-post-op
Test Method	1. Issue an HTTP POST request to the URL `{root}/joins` where form data parameter *spatialDatasetId* is the spatialDatasetId property of a spatial dataset (from query `{root}/spatialdatasets`). 2. Validate that a document was returned with a status code `201`. 3. Validate the contents of the returned document using test /ats/core/joins-post-success If the server supports the data joining with direct GeoJSON output: 1. Issue an HTTP POST request to the URL `{root}/joins` where form data parameter *spatialDatasetId* is the spatialDatasetId property of a spatial dataset (from query `{root}/spatialdatasets`) and outputFormats parameter has a value 'geojson-direct'. 2. Validate that a document was returned with a status code `200`.

	3. Validate the contents of the returned document using test /ats/core/joins-post-success

Abstract Test 26	**/ats/core/joins-post-success**
Test Purpose	Validate that the data join response document complies with the required structure and contents.
Requirement	/req/core/joins-post-success
Test Method	1. Validate that the response document complies with the joins post schema. 2. Validate that the response document contains the joined data in all the requested output formats that are supported by the service implementation. For validating direct geojson output: 1. Validate that the response is a valid GeoJSON document and it contains the joined attributes.

A.2.11. Join {root}/joins/{joinid}

Abstract Test 27	**/ats/core/joins-joinid-get-op**
Test Purpose	Validate that the information about a join can be retrieved from the expected location.
Requirement	/req/core/joins-joinid-get-op
Test Method	1. For a list of all joins (path `{root}/joins`), issue an HTTP GET request to the URL `{root}/joins/{joinid}` where {joinid} is the joinId property of a join. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/joins-joinid-get-success

Abstract Test 28	**/ats/core/joins-joinid-get-success**
Test Purpose	Validate that the join content complies with the required structure and contents.
Requirement	/req/core/joins-joinid-get-success
Test Method	Validate that the response document complies with join schema.

A.2.12. Join Update {root}/joins/{joinid}

Abstract Test 29	**/ats/core/joins-joinid-put-op**
Test Purpose	Validate that the join can be updated fully with a attribute dataset file data from expected location.
Requirement	/req/core/joins-joinid-put-op
Test Method	1. Issue an HTTP PUT request to the URL `{root}/joins/{joinid}` where

	{joinid} is the joinId property of a join (from HTTP GET query `{root}/joins/{joinid}`). 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/joins-joinid-put-success

Abstract Test 30	**/ats/core/joins-joinid-put-success**
Test Purpose	Validate that the data can be joined from an attribute dataset file with a specific spatial dataset.
Requirement	/req/core/joins-joinid-put-success
Test Method	1. Validate that the response document complies with join update schema. 2. Validate that the response document contains the joined data in all the requested output formats that are supported by the service implementation.

A.2.13. Join Delete {root}/joins/{joinid}

Abstract Test 31	**/ats/core/joins-joinid-delete-op**
Test Purpose	Validate that the join can be deleted from the expected location.
Requirement	/req/core/joins-joinid-delete-op
Test Method	1. Issue an HTTP DELETE request to the URL `{root}/joins/{joinid}` where {joinid} is the joinId property of a join (from query `{root}/joins/{joinid}`). 2. Validate that a document was returned with a status code `204`. 3. Validate the contents of the returned document using test /ats/core/joins-joinid-delete-success

Abstract Test 32	**/ats/core/joins-joinid-delete-success**
Test Purpose	Validate that the join was deleted from the server.
Requirement	/req/core/joins-joinid-delete-success
Test Method	1. Validate that the join has been deleted from the server by issuing an HTTP GET request to the URL `{root}/joins/{joinid}` where {joind} is the same joinId property of the join that was used in the delete request. 2. Validate that the server sent a response code `404`.

A.3. Conformance class: Core / Spatial Joining

Conformance Class	
http://www.opengis.net/spec/ogcapi-tjs/1.0/req/core/spatial-joining	
Target type	Web API

Requirements class	Requirements Class "Core"

A.3.1. Attribute Datasets {root}/attributedatasets

Abstract Test 33	**/ats/core/attributedatasets-get-op**
Test Purpose	Validate that the information about attribute datasets can be retrieved from the expected location.
Requirement	/req/core/attributedatasets-get-op
Test Method	1. Issue an HTTP GET request to the URL `{root}/attributedatasets`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/attributedatasets-get-success

Abstract Test 34	**/ats/core/attributedatasets-get-success**
Test Purpose	Validate that the attribute datasets content complies with the required structure and contents.
Requirement	/req/core/attributedatasets-get-success
Test Method	1. Validate that the response document complies with attributedatasets schema.

A.3.2. Attribute Dataset Creation {root}/attributedatasets

Abstract Test 35	**/ats/core/attributedatasets-post-op**
Test Purpose	Validate that the attribute dataset can be added to the server from the expected location.
Requirement	/req/core/attributedatasets-post-op
Test Method	1. Issue an HTTP POST request to the URL `{root}/attributedatasets`. 2. Validate that a document was returned with a status code `201`. 3. Validate the contents of the returned document using test /ats/core/attributedatasets-post-success

Abstract Test 36	**/ats/core/attributedatasets-post-success**
Test Purpose	Validate that the attribute dataset's creation response document complies with the required structure and contents.
Requirement	/req/core/attributedatasets-post-success
Test Method	1. Validate that the response document complies with attributedatasets-attributedatasetid schema.

A.3.3. Attribute Dataset {root}/attributedatasets/{attributedatasetid}

Abstract Test 37	/ats/core/attributedatasets-attributedatasetid-get-op
Test Purpose	Validate that a attribute dataset information can be retrieved from the expected location.
Requirement	/req/core/attributedatasets-attributedatasetid-get-op
Test Method	1. For a list of all attriute datasets (path `{root}/attributedatasets`), issue an HTTP GET request to the URL `{root}/attributedatasets/{attributedatasetid}` where {attributedatasetid} is the attributeDatasetId property of a attribute dataset. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/attributedatasets-attributedatasetid-get-success

Abstract Test 38	/ats/core/attributedatasets-attributedatasetid-get-success
Test Purpose	Validate that the attribute dataset complies with the required structure and contents.
Requirement	/req/core/attributedatasets-attributedatasetid-get-success
Test Method	1. Validate that the response document complies with attributedataset schema.

A.3.4. Attribute Dataset Update {root}/attributedatasets/{attributedatasetid}

Abstract Test 39	/ats/core/attributedatasets-attributedatasetid-put-op
Test Purpose	Validate that the attribute dataset can be updated from the expected location.
Requirement	/req/core/attributedatasets-attributedatasetid-put-op
Test Method	1. Issue an HTTP PUT request to the URL `{root}/attributedatasets/{attributedatasetid}`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/attributedatasets-attributedatasetid-put-success

Abstract Test 40	/ats/core/attributedatasets-attributedatasetid-put-success
Test Purpose	Validate that the response document of the attribute dataset's update complies with the required structure and contents.
Requirement	/req/core/attributedatasets-attributedatasetid-put-success
Test Method	1. Validate that the response document complies with attributedatasets-attributedatasetid schema.

A.3.5. Attribute Dataset Delete {root}/attributedatasets/{attributedatasetid}

Abstract Test 41	/ats/core/attributedatasets-attributedatasetid-delete-op

Test Purpose	Validate that the attribute dataset can be deleted from the expected location.
Requirement	/req/core/attributedatasets-attributedatasetid-delete-op
Test Method	1. Issue an HTTP DELETE request to the URL `{root}/attributedatasets/{attributedatasetid}` where {attributedatasetid} is the attributeDatasetId property of a attribute dataset (from query `{root}/attributedatasets/{attributedataset}`). 2. Validate that a document was returned with a status code `204`. 3. Validate the contents of the returned document using test /ats/core/attributedatasets-attributedatasetid-delete-success

Abstract Test 42	**/ats/core/attributedatasets-attributedatasetid-delete-success**
Test Purpose	Validate that the attribute dataset was deleted from the server.
Requirement	/req/core/attributedatasets-attributedatasetid-delete-success
Test Method	1. Validate that the attribute dataset has been deleted from the server by issuing an HTTP GET request to the URL `{root}/attributedatasets/{attributedatasetid}` where {attributedatasetid} is the same attributeDatasetId property of the attribute dataset that was used in the delete request. 2. Validate that the server sent a response code `404`.

A.3.6. Attribute Dataset keys {root}/attributedatasets/{attributedatasetid}/keys

Abstract Test 43	**/ats/core/attributedatasets-attributedatasetid-keys-op**
Test Purpose	Validate that the information on attribute dataset key fields can be retrieved from the expected location.
Requirement	/req/core/attributedatasets-attributedatasetid-keys-op
Test Method	1. For a attribute dataset (path `{root}/attributedatasets/{attributedatasetid}`), issue an HTTP GET request to the URL `{root}/attributedatasets/{attributedatasetid}/keys` where {attributedatasetid} is the attributeDatasetId property of a attribute dataset. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/attriutedatasets-attributedatasetid-keys-success

Abstract Test 44	**/ats/core/attributedatasets-attributedatasetid-keys-success**
Test Purpose	Validate that the attribute dataset's keys content complies with the required structure and contents.
Requirement	/req/core/attributedatasets-attributedatasetid-keys-success

Test Method	1. Validate that the response document complies with attributedataset key fields schema.

A.3.7. Attribute Dataset key field {root}/attributedatasets/{attributedatasetid}/keys/{keyname}

Abstract Test 45	**/ats/core/attributedatasets-attributedatasetid-keys-keyname-op**
Test Purpose	Validate that the information on keys that belong to the attribute dataset´s key field can be retrieved from the expected location.
Requirement	/req/core/attributedatasets-attributedatasetid-keys-keyname-op
Test Method	1. For a attribute dataset (path `{root}/attributedatasets/{attributedatasetid}`), issue an HTTP GET request to the URL `{root}/attributedatasets/{attributedatasetid}/keys/{keyname}` where {attributedatasetid} is the attributeDatasetId property of a attribute dataset and {keyname} is the keyName property of the attribute dataset key field. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/attributedatasets-attributedatasetid-keys-keyname-success

Abstract Test 46	**/ats/core/attributedatasets-attributedatasetid-keys-keyname-success**
Test Purpose	Validate that the attribute dataset key field´s contents comply with the required structure and contents.
Requirement	/req/core/attributedatasets-attributedatasetid-keys-keyname-success
Test Method	Validate that the response document complies with attribute dataset key field schema.

A.3.8. Attribute Dataset key field key {root}/attributedatasets/{attributedatasetid}/keys/{keyname}/{key}

Abstract Test 47	**/ats/core/attributedatasets-attributedatasetid-keys-keyname-key-op**
Test Purpose	Validate that the attribute dataset´s key field´s key information can be retrieved from the expected location.
Requirement	/req/core/attributedatasets-attribnutedatasetid-keys-keyname-key-op
Test Method	1. For a attribute dataset (path `{root}/attributedatasets/{attributedatasetid}`), issue an HTTP GET request to the URL `{root}/attributedatasets/{attributedatasetid}/keys/{keyname}/{key}` where {attributedatasetid} is the attributeDatasetId property of a spatial dataset, {keyname} is the keyName property of the attribute dataset key field and {key} is the name of the key property of the attribute dataset key value. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test

	/ats/core/attributedatasets-attributedatasetid-keys-keyname-key-success

Abstract Test 48	**/ats/core/attributedatasets-attributedatasetid-keys-keyname-key-success**
Test Purpose	Validate that the spatial dataset´s key content complies with the required structure and contents.
Requirement	/req/core/spatialdatasets-spatialdatasetid-keys-keyname-key-success
Test Method	Validate that the response document complies with the spatial dataset key field key schema.

A.3.9. Attribute Dataset Joining To Spatial Dataset File {root}//joinattributedataset

Abstract Test 49	**/ats/core/joinattributedataset-op**
Test Purpose	Validate that data from attribute dataset can be joined with spatial dataset file (GeoJSON file) from the expected location.
Requirement	/req/core/joinattributedataset-op
Test Method	1. Issue an HTTP POST request to the URL `{root}/joinattributedataset`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/joinattributedataset-success

Abstract Test 50	**/conf/core/joinattributedataset-success**
Test Purpose	Validate that the GeoJSON file contains the data that was joined from the attribute dataset.
Requirement	/req/core/joinattributedataset-success
Test Method	Validate that the GeoJSON file contains the attributes that were joined from the attribute dataset.

A.4. Conformance class: Core / File Joining

Conformance Class	
http://www.opengis.net/spec/ogcapi-tjs/1.0/req/core/file-joining	
Target type	Web API
Requirements class	Requirements Class "Core"

A.4.1. File Joining {root}//joinfiles

Abstract Test 51	**/ats/core/joinfiles-op**
Test Purpose	Validate that data from attribute dataset file can be joined with a spatial dataset file from the expected location.

Requirement	/req/core/joinfiles-op
Test Method	1. Issue an HTTP POST request to the URL `{root}/joinfiles`. 2. Validate that a document was returned with a status code `200`. 3. Validate the contents of the returned document using test /ats/core/joinfiles-success

Abstract Test 52	**/conf/core/joinfiles-success**
Test Purpose	Validate that the spatial dataset file contains the data that was joined from the attribute dataset file.
Requirement	/req/core/joinfiles-success
Test Method	Validate that the spatial dataset file contains the attributes that were joined from the attribute dataset file.

Annex B: Revision History

Date	Release	Editor	Primary clauses modified	Description
2020-08-10	1.0.0-SNAPSHOT	P. Latvala	6.3	Updated chapters 6.3.2, 6.3.4, 6.4.2 and 6.4.4
2020-07-10	1.0.0-SNAPSHOT	P. Latvala	all	added the *spatial joining operations* operation set, added spatial dataset creation and deleting, edited the whole document
2020-04-09	1.0.0-SNAPSHOT	P. Latvala	all	Document into asciidoc-based format

Annex C: Bibliography

- Internet Assigned Numbers Authority (IANA). **Link Relation Types** [online, viewed 2020-04-09], Available at https://www.iana.org/assignments/link-relations/link-relations.xml

Last updated 2020-09-02 08:01:44 +0300