

OGC WPS 2.0 REST/JSON Binding Extension

Table of Contents

1. Scope	6
2. Conformance	7
3. References	8
4. Terms and Definitions	9
4.1. Process	9
4.2. Process description	9
4.3. Process input	9
4.4. Process output	9
4.5. Process profile	9
4.6. WPS Server	10
4.7. Process offering	10
4.8. Process execution	10
4.9. Job	10
4.10. Service profiles for WPS	10
4.11. REST or RESTful	10
4.12. JSON	11
5. Conventions	12
5.1. Identifiers	12
5.2. Abbreviated Terms	12
5.3. Use of the Term "Process"	12
5.4. Namespace Conventions	13
6. Requirements Class "API"	14
6.1. API landing page	14
6.1.1. Operation	14
6.1.2. Response	14
6.1.3. Error situations	15
6.2. API definition	15
6.2.1. Operation	15
6.2.2. Response	15
6.2.3. Error situations	16
7. Requirements Class "Process Collection"	17
7.1. Operation	17
7.2. Response	17
7.3. Error situations	18
8. Requirements Class "Process"	19
8.1. Operation	19
8.2. Response	19
8.3. Error situations	20
9. Requirements Class "Job Collection"	21

9.1. Operation	21
9.2. Response	21
9.3. Error situations	22
10. Requirements Class "Job"	23
10.1. Operation	23
10.2. Response	23
10.3. Error situations	24
Annex A: Conformance Class Abstract Test Suite (Normative)	25
A.1. Conformance Class A	25
A.1.1. Requirement 1	25
A.1.2. Requirement 2	25
Annex B: Revision History	26
Annex C: Bibliography	27

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/wps-rest/1.0>

Internal reference number of this OGC® document: 18-062

Version: 1.0-draft

Category: OGC® Implementation Specification

Editor: Benjamin Pross

OGC WPS 2.0 REST/JSON Binding Extension

Copyright notice

Copyright © 2018 Open Geospatial Consortium

To obtain additional rights of use, visit <http://www.opengeospatial.org/legal/>

Warning

This document is not an OGC Standard. This document is distributed for review and comment. This document is subject to change without notice and may not be referred to as an OGC Standard.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

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.

i. Abstract

In many cases geospatial or location data, including data from sensors, must be processed before the information can be used effectively. The OGC Web Processing Service (WPS) Interface Standard provides a standard interface that simplifies the task of making simple or complex computational processing services accessible via web services. Such services include well-known processes found in GIS software as well as specialized processes for spatio-temporal modeling and simulation. While the OGC WPS standard was designed with spatial processing in mind, it can also be used to readily insert non-spatial processing tasks into a web services environment. The WPS standard provides a robust, interoperable, and versatile protocol for process execution on web services. It supports both immediate processing for computational tasks that take little time and asynchronous processing for more complex and time consuming tasks. Moreover, the WPS standard defines a general process model that is designed to provide an interoperable description of processing functions. It is intended to support process cataloguing and discovery in a distributed environment. The OGC WPS REST/JSON Binding extension builds on the WPS 2.0 standard and defines the processing standards to communicate over a RESTful protocol using JSON encodings. This binding definition will be a newer and more modern way of programming and interacting with resources over the web while allowing better integration into existing software packages

ii. Keywords

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

geoprocessing, ogcdoc, OGC document, processes, WPS, REST, JSON

iii. Preface

This extension is a continuation of WPS 2.0, a standard for web-based processing of geospatial data. It defines how the interfaces for WPS 2.0 operations should be constructed and interpreted using a REST based protocol with JSON encoding. Within the current version of WPS 2.0, bindings are defined for HTTP/POST using XML encodings and HTTP/GET using KVP encodings. Also in the current WPS 2.0 standard, a core conceptual model is provided that may be used to specify a WPS in different architectures such as REST or SOAP. Therefore, this extension is a natural fit to what is already defined in the standard.

iv. Submitting organizations

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

52°North Intergraph Corporation (Hexagon Geospatial)

v. Submitters

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

Name	Representing	OGC Member
Benjamin Pross	52°North	Yes
Stan Tillman	Intergraph Corporation (Hexagon Geospatial)	Yes

Chapter 1. Scope

This document specifies the interface to a general-purpose Web Processing Service (WPS) using a RESTful protocol transporting JSON encoded requests and responses. A WPS is a web service that enables the execution of computing processes and the retrieval of metadata describing their purpose and functionality. Typically, these processes combine raster, vector, and/or coverage data with well-defined algorithms to produce new raster, vector, and/or coverage information. The document is an extension to the OGC WPS 2.0 Interface Standard [14-065]. It should be considered as another binding extension to HTTP/POST + XML and HTTP/GET + KVP as defined in Section 10 (Binding Extensions for WPS Operations) of the WPS 2.0 standard.

Chapter 2. Conformance

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

In order to conform to this OGC® interface standard, a software implementation shall choose to implement:

- Any one of the conformance levels specified in Annex B (normative).
- Any one of the Distributed Computing Platform profiles specified in Annexes TBD through TBD (normative).

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

Chapter 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.

OGC 14-065, OGC WPS 2.0 Interface Standard, version 2.0.1

OGC 06-121r9, OGC Web Service Common Specification, version 2.0

OGC 08-131r3 – The Specification Model – A Standard for Modular Specifications

IETF RFC 4646: Tags for Identifying Languages

IETF RFC 3986: Uniform Resource Identifier (URI): Generic Syntax

ISO 8601:2004, Data elements and interchange formats – Information interchange – Representation of dates and times

XML Schema Part 2: Datatypes Second Edition, W3C Recommendation 28 October 2004.

Chapter 4. Terms and Definitions

This document uses the terms defined in Sub-clause 5.3 of [OGC 06-121r8], 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. Process

A process p is a function that for each input returns a corresponding output

$$p: X \rightarrow Y$$

where X denotes the domain of arguments x and Y denotes the co-domain of values y . Within this specification, process arguments are referred to as process inputs and result values are referred to as process outputs. Processes that have no process inputs represent value generators that deliver constant or random process outputs.

4.2. Process description

A process description is an information model that specifies the interface of a process. A process description is used for a machine-readable description of the process itself but also provides some basic information about the process inputs and outputs.

4.3. Process input

Process inputs are the arguments of a process and refer to data provided to a process. Each process input is an identifiable item.

4.4. Process output

Process outputs are the results of a process and refer to data returned by a process. Each process output is an identifiable item.

4.5. Process profile

A process profile is a description of a process on an interface level. Process profiles may have different levels of abstraction and cover several aspects. On a generic level, a process

profile may only refer to the provided functionality of a process, i.e. by giving a verbal or formal definition how the outputs are derived from the inputs. On a concrete level a process profile may completely define inputs and outputs including data type definitions and formats.

4.6. WPS Server

A WPS Server is a web server that provides access to simple or complex computational processing services

4.7. Process offering

A process offering is an identifiable process that may be executed on a particular service instance. A process offering contains a process description as well as service-specific information about the supported execution protocols (e.g. synchronous and asynchronous execution).

4.8. Process execution

The execution of a process is an action that calculates the outputs of a given process for a given set of data inputs.

4.9. Job

The (processing) job is a server-side object created by a processing service for a particular process execution. A job may be latent in the case of synchronous execution or explicit in the case of asynchronous execution. Since the client has only oblique access to a processing job, a Job ID is used to monitor and control a job.

4.10. Service profiles for WPS

A service profile for WPS is a conformance class that defines the general capabilities of a WPS server, by (1) specifying the supported service operations, (2) the process model, (3) the supported process execution modes, (4) the supported operation binding(s).

4.11. REST or RESTful

Representational state transfer. REST-compliant Web services allow requesting systems to access and manipulate textual representations of Web resources using a uniform and predefined set of stateless operations.

4.12. JSON

JavaScript Object Notation is a lightweight data-interchange format. It is easy for humans to read and write and it is easy for machines to parse and generate.

Chapter 5. Conventions

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

5.1. Identifiers

The normative provisions in this specification are denoted by the URI

<http://www.opengis.net/spec/wps-rest/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. Abbreviated Terms

Abbreviated Term	Meaning
CRS	Coordinate Reference System
GML	Geography Markup Language
HTTP	Hypertext Transfer Protocol
ISO	International Organization for Standardization
KVP	Keyword Value Pair
MIME	Multipurpose Internet Mail Extensions
OGC	Open Geospatial Consortium
URI	Universal Resource Identifier
URL	Uniform Resource Locator
WPS	Web Processing Service
XML	Extensible Markup Language
REST	Representational State Transfer
JSON	JavaScript Object Notation

5.3. Use of the Term "Process"

The term process is one of the most used terms both in the information and geosciences domain. If not stated otherwise, this specification uses the term process as an umbrella term for any algorithm, calculation or model that either generates new data or transforms some input data into output data as defined in section 4.1 of the WPS 2.0 standard.

5.4. Namespace Conventions

Prefix	Namespace URI	Description
ows	http://www.opengis.net/ows/2.0	OWS Common 2.0 XML Schema
xlink	http://www.w3.org/1999/xlink	Definitions for XLINK
xml	http://www.w3.org/XML/1998/namespace	XML (required for xml:lang)
xs	http://www.w3.org/2001/XMLSchema	XML Schema

Chapter 6. Requirements Class "API"

6.1. API landing page

6.1.1. Operation

Requirement 1	/req/core/root-op The server SHALL support the HTTP GET operation at the path <code>/</code> .
---------------	---

6.1.2. Response

Requirement 2	/req/core/root-success A successful execution of the operation SHALL be reported as a response with a HTTP status code <code>200</code> . The content of that response SHALL be based upon the OpenAPI 3.0 schema <code>root.yaml</code> and include at least links to the following resources: * <code>/api</code> (relation type 'service') * <code>/conformance</code> (relation type 'conformance') * <code>/collections</code> (relation type 'data')
---------------	---

Schema for the landing page

```
type: object
required:
  - links
properties:
  links:
    type: array
    items:
      $ref:
        https://raw.githubusercontent.com/opengeospatial/WFS_FES/master/core/openapi/schemas/link.yaml
```

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

6.1.3. Error situations

6.2. API definition

6.2.1. Operation

Every WFS provides an API definition that describes the capabilities of the server and which can be used by developers to understand the API, by software clients to connect to the server, or by development tools to support the implementation of servers and clients.

Unresolved directive in clause_6_api.adoc - include::requirements/core/REQ_api-definition-op.adoc[]

6.2.2. Response

Unresolved directive in clause_6_api.adoc - include::requirements/core/REQ_api-definition-success.adoc[]

Unresolved directive in clause_6_api.adoc - include::requirements/core/REC_api-definition-oas.adoc[]

If multiple API definition formats are supported by a server, use content negotiation to select the desired representation.

The API definition document describes the API. In other words, there is no need to include the `/api` operation in the API definition itself.

The idea is that any WFS can be used by developers that are familiar with the API definition language(s) supported by the server. For example, if an OpenAPI definition is used, it should be possible to create a working client using the OpenAPI definition. The developer may need to learn a little bit about geometry data types, etc., but it should not be required to read this standard to access the data via the API.

6.2.3. Error situations

See [\[http_status_codes\]](#) for general guidance.

See [\[http_status_codes\]](#) for general guidance.

Requirement

Operation

Response

URL-Example

Schema

Example

Errors

Chapter 7. Requirements Class "Process Collection"

In the following sections, the resources and endpoint-URLs of the WPS 2.0 REST/JSON binding are described.

7.1. Operation

Requirement 3	/req/op/process-collection The server SHALL support the HTTP GET operation at the path <i>/processes</i> .
---------------	---

7.2. Response

Requirement 4	/req/op/process-collection The server SHALL support the HTTP GET operation at the path <i>/processes</i> .
---------------	---

Schema for the process collection

```
type: object
required:
  - processes
properties:
  processes:
    type: array
    items:
      $ref: '#/components/schemas/processSummary'
```

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

7.3. Error situations

Example of HTTP GET request for retrieving the list of offered processes encoded as JSON.

```
http://hostname.org/wps-rest/processes
```

Example of Process list encoded as JSON.

```
{
  "ProcessSummaries": [
    {
      "identifier": "ConvexHullAlgorithm",
      "title": "Convex Hull Algorithm",
      "processVersion": "1.0.0",
      "jobControlOptions": "sync-execute async-execute",
      "processDescription": "http://hostname.org/wps-
rest/processes/ConvexHullAlgorithm"
    }, ...
  ]
}
```

Chapter 8. Requirements Class "Process"

In the following sections, the resources and endpoint-URLs of the WPS 2.0 REST/JSON binding are described.

8.1. Operation

Requirement 5	/req/op/process-collection The server SHALL support the HTTP GET operation at the path /processes .
---------------	--

8.2. Response

Requirement 6	/req/op/process-collection The server SHALL support the HTTP GET operation at the path /processes .
---------------	--

Schema for the process collection

```
type: object
required:
  - processes
properties:
  processes:
    type: array
    items:
      $ref: '#/components/schemas/processSummary'
```

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

8.3. Error situations

Example of HTTP GET request for retrieving the list of offered processes encoded as JSON.

```
http://hostname.org/wps-rest/processes
```

Example of Process list encoded as JSON.

```
{
  "ProcessSummaries": [
    {
      "identifier": "ConvexHullAlgorithm",
      "title": "Convex Hull Algorithm",
      "processVersion": "1.0.0",
      "jobControlOptions": "sync-execute async-execute",
      "processDescription": "http://hostname.org/wps-
rest/processes/ConvexHullAlgorithm"
    }, ...
  ]
}
```

Chapter 9. Requirements Class "Job Collection"

In the following sections, the resources and endpoint-URLs of the WPS 2.0 REST/JSON binding are described.

9.1. Operation

Requirement 7	/req/op/process-collection The server SHALL support the HTTP GET operation at the path <i>/processes</i> .
---------------	---

9.2. Response

Requirement 8	/req/op/process-collection The server SHALL support the HTTP GET operation at the path <i>/processes</i> .
---------------	---

Schema for the process collection

```
type: object
required:
  - processes
properties:
  processes:
    type: array
    items:
      $ref: '#/components/schemas/processSummary'
```



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

9.3. Error situations

Example of HTTP GET request for retrieving the list of offered processes encoded as JSON.

```
http://hostname.org/wps-rest/processes
```

Example of Process list encoded as JSON.

```
{
  "ProcessSummaries": [
    {
      "identifier": "ConvexHullAlgorithm",
      "title": "Convex Hull Algorithm",
      "processVersion": "1.0.0",
      "jobControlOptions": "sync-execute async-execute",
      "processDescription": "http://hostname.org/wps-
rest/processes/ConvexHullAlgorithm"
    }, ...
  ]
}
```

Chapter 10. Requirements Class "Job"

In the following sections, the resources and endpoint-URLs of the WPS 2.0 REST/JSON binding are described.

10.1. Operation

Requirement 9	/req/op/process-collection The server SHALL support the HTTP GET operation at the path /processes .
---------------	--

10.2. Response

Requirement 10	/req/op/process-collection The server SHALL support the HTTP GET operation at the path /processes .
----------------	--

Schema for the process collection

```
type: object
required:
  - processes
properties:
  processes:
    type: array
    items:
      $ref: '#/components/schemas/processSummary'
```

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

10.3. Error situations

Example of HTTP GET request for retrieving the list of offered processes encoded as JSON.

```
http://hostname.org/wps-rest/processes
```

Example of Process list encoded as JSON.

```
{
  "ProcessSummaries": [
    {
      "identifier": "ConvexHullAlgorithm",
      "title": "Convex Hull Algorithm",
      "processVersion": "1.0.0",
      "jobControlOptions": "sync-execute async-execute",
      "processDescription": "http://hostname.org/wps-
rest/processes/ConvexHullAlgorithm"
    }, ...
  ]
}
```

Unresolved directive in er.adoc - include::clause_11process-output.adoc[]

Annex A: Conformance Class Abstract Test Suite (Normative)

NOTE

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

A.1. Conformance Class A

A.1.1. Requirement 1

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

A.1.2. Requirement 2

Annex B: Revision History

Date	Release	Editor	Primary clauses modified	Description
2017-03-09	0.1	Benjamin Pross	all	initial version
2017-xx-xx	0.2	Benjamin Pross	6	Update REST/JSON section
2017-10-16	0.3	Stan Tillman	1-5	Update section 1-5

Annex C: Bibliography

Example Bibliography (Delete this note).

The TC has approved Springer LNCS as the official document citation type.

Springer LNCS is widely used in technical and computer science journals and other publications

NOTE

- For citations in the text please use square brackets and consecutive numbers: [1], [2], [3]

– Actual References:

[n] Journal: Author Surname, A.: Title. Publication Title. Volume number, Issue number, Pages Used (Year Published)

[n] Web: Author Surname, A.: Title, <http://Website-Url>

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