**Open Geospatial Consortium**

Submission Date: <yyyy-dd-mm>

Approval Date:   <yyyy-dd-mm>

Publication Date:   <yyyy-dd-mm>

External identifier of this OGC® document: <<http://www.opengis.net/def/doc-type/standard/1.0>>

Internal reference number of this OGC® document:    13-057

Version: 2.0.0

Category: OGC® Standard

Editor:   Peter Baumann

OGC Web Coverage Service Interface Standard – Transaction Extension

**Copyright notice**

Copyright © <year> 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.

Document type:    OGC® Standard

Document stage:    Draft

Document language:  English

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.

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.

Contents

1. Scope 6

2. Conformance 6

3. References 7

4. Terms and Definitions 7

5. Conventions 8

5.1 UML notation 8

5.2 Data dictionary tables 8

5.3 Namespace prefix conventions 8

5.4 Multiple representations 9

6. *Insert+delete* requirements class 9

6.1 Overview 9

6.2 Modifications to *GetCapabilities* 9

6.3 Modifications to DescribeCoverage 9

6.4 Modifications to *GetCoverage* 9

6.5 InsertCoverage request 9

6.5.1 InsertCoverage request 9

6.5.2 InsertCoverage response 11

6.6 DeleteCoverage request 12

6.6.1 DeleteCoverage request 12

6.6.2 DeleteCoverage response 13

6.7 Atomicity and isolation 13

6.8 Encodings 14

6.8.1 Overview 14

6.8.2 GET/KVP Encoding 14

6.8.3 XML/POST Encoding 15

6.8.4 SOAP Encoding 16

6.9 Exceptions 16

7. *Update* requirements class 17

7.1 Overview 17

7.2 Modifications to *GetCapabilities* 17

7.3 Modifications to DescribeCoverage 17

7.4 Modifications to *GetCoverage* 17

7.5 Modifications to *InsertCoverage* 17

7.6 Modifications to *DeleteCoverage* 17

7.7 UpdateCoverage request 17

7.7.1 Overview 17

7.7.2 UpdateCoverage request 18

7.7.3 *UpdateCoverage* response 21

7.8 Encodings 22

7.8.1 Overview 22

7.8.2 GET/KVP Encoding 23

7.8.3 XML/POST Encoding 24

7.8.4 SOAP Encoding 25

7.9 Exceptions 25

A.1 Conformance class: *insert+delete* 26

A.2 Conformance class: *update* 26

* Abstract

This OGC *Web Coverage Service (WCS) – Transaction Extension* (in short: *Transaction Extension*) defines an extension to the WCS Core [OGC 09-110] for updating coverage offerings on a server.

To this end, a set of three new requests is defined:

* *InsertCoverage* for adding a coverage provided as parameter to the WCS server’s coverage offering. After successful completion of this request, this coverage will be accessible through all WCS operations.
* *DeleteCoverage* for entirely removing a coverage, identified by its coverage id passed in the request, from the WCS server’s coverage offering. After successful completion of this request, this coverage will not be accessible through any WCS operation. However, subsequently a new coverage may be created using the same identifier; such a coverage will bear no relation to the one previously deleted.
* *UpdateCoverage* for modifying parts of a coverage existing in a WCS server’s coverage offering, identified by its coverage id passed in the request. All updates must maintain internal consistency of the coverage, as per GMLCOV [OGC 09-146].

All requests defined in this Transaction Extension adhere to the ACID (atomicity, consistency, isolation, durability) concepts of transactions in databases.

Note: the extension name, Transaction, traces back to the database concept of transactions, which has been adopted here.

* Keywords

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

ogcdoc, OGC document, <tags separated by commas>

* Preface

This WCS Transaction Extension is an OGC Interface Standard which relies on WCS Core [OGC 09-110] and the GML Application Schema for Coverages [OGC 09-146]. This Standard adds three request types to the OGC Web Coverage Service (WCS): *InsertCoverage*, *DeleteCoverage*, and *UpdateCoverage*, respectively. This allows clients to modify a server’s offerings by adding, deleting, and updating coverages, respectively.

Suggested additions, changes, and comments on this document are welcome and encouraged. Such suggestions may be submitted by email message or by making suggested changes in an edited copy of this document

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

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

* Submitting organizations

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

Jacobs University Bremen

* Submitters

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

|  |  |
| --- | --- |
| 1. Name
 | 1. Affiliation
 |
| Peter Baumann | Jacobs University Bremen, rasdaman GmbH |

# Scope

This OGC WCS Transaction Extension – in short: *Transaction Extension* or *WCS-T* – defines how to modify a WCS server’s coverage offering.

# Conformance

This document establishes the following requirements and conformance classes:

* *insert+delete*, of URI [http://www.opengis.net/spec/WCS\_service-extension\_transaction/2.0/req/insert+delete](http://www.opengis.net/spec/WCS_service-extension_transaction/2.0/req/insert%2Bdelete); the corresponding conformance class is *insert+delete*, with URI [http://www.opengis.net/spec/WCS\_service-extension\_transaction/2.0/conf/insert+delete](http://www.opengis.net/spec/WCS_service-extension_transaction/2.0/conf/insert%2Bdelete).

This is the mandatory core conformance class of this extension.

* *update*, of URI <http://www.opengis.net/spec/WCS_service-extension_transaction/2.0/req/update>; the corresponding conformance class is *update*, with URI <http://www.opengis.net/spec/WCS_service-extension_transaction/2.0/conf/update>.

Standardization target of all requirements and conformance classes are WCS implementations (currently: servers).

Requirements URIs defined in this document are relative to <http://www.opengis.net/spec/WCS_service-extension_transaction/2.0/req>,
conformance test URIs are relative to <http://www.opengis.net/spec/WCS_service-extension_transaction/2.0/conf>.

Annex A lists the conformance tests which shall be exercised on any software artifact claiming to implement WCS.

# References

This *OGC WCS Transaction Extension* specification consists of the present document and an XML Schema. The complete specification is identified by OGC URI <http://www.opengis.net/spec/WCS_service-extension_transaction/2.0>, the document has OGC URI <http://www.opengis.net/doc/ISx/WCS_service-extension_transaction/2.0>.

The complete specification is available for download from [http://www.opengeospatial.org/standards/wcs](http://www.opengeospatial.org/stan%1Fdards/wcs); additionally, the XML Schema is posted online at <http://schemas.opengis.net/wcs/transaction/2.0> as part of the OGC schema repository. In the event of a discrepancy between bundled and schema repository versions of the XML Schema files, the schema repository shall be considered authoritative.

The normative documents listed in Table 1 contain provisions that, through reference in this text, constitute provisions of this specification. 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.

Table 1 — Conformance class dependencies

|  |  |  |
| --- | --- | --- |
| **Transaction conformance class** | **Dependency document** | **Dependency conformance class** |
| *insert+delete* | OGC 09-146, *GML 3.2.1 Application Schema for Coverages*, version 1.0OGC 09-110, *OGC®**Web Coverage Service 2.0 Interface Standard - Core*, version 2.0 | *gml-coverage* *core* |
| *update* | This document | *insert+delete* |

# 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.

Input coverage

Coverage sent to the server through the WCS-T request on hand.

Updated coverage

Coverage to be updated through the WCS-T request on hand.

# Conventions

## UML notation

Unified Modeling Language (UML) static structure diagrams appearing in this specification are used as described in Subclause 5.2 of OGC Web Services Common [OGC 06-121r9].

## Data dictionary tables

The UML model data dictionary is specified herein in a series of tables. The contents of the columns in these tables are described in Subclause 5.5 of [OGC 06-121r9]. The contents of these data dictionary tables are normative, including any table footnotes.

## Namespace prefix conventions

The following namespaces are used in this document. The prefix abbreviations used constitute conventions used here, but are **not** normative. The namespaces to which the prefixes refer are normative, however.

Table 2 — Namespace mappings

|  |  |  |
| --- | --- | --- |
| **Prefix** | **Namespace URI** | **Description** |
| xsd | <http://www.w3.org/2001/XMLSchema>  | XML Schema namespace |
| gml | <http://www.opengis.net/gml/3.2>  | GML 3.2.1 |
| gmlcov | <http://www.opengis.net/gmlcov/1.0> | GML Application Schema for Coverages 1.0 |
| wcs | <http://www.opengis.net/wcs/2.0> | WCS 2.0 Core |
| wcst | <http://www.opengis.net/wcs/transaction/2.0> | WCS 2.0 Transaction Extension |

## Multiple representations

When multiple representations of the same information are given in a specification document these are consistent. Should this not be the case then this is considered an error, and the XML schema shall take precedence

# *Insert+delete* requirements class

## Overview

This Clause 1 establishes the mandatory Transaction Extension core requirements class, *insert+delete*. Clients and servers supporting this *insert+delete* requirements class shall support insertion into and deletion from a WCS server’s coverage offerings through two dedicated request types, *InsertCoverage* and *DeleteCoverage*.

## Modifications to *GetCapabilities*

A server announces support of the *insert+delete* requirements class to a client by adding the URL identifying this extension to the list of supported extensions delivered in the Capabilities document.

1. A WCS service implementing requirements class *insert+delete* of this Transaction Extension **shall** include the following URI in the Profile element of the ServiceIdentification in a *GetCapabilities* response:
http://www.opengis.net/spec/WCS\_service-extension\_transaction/2.0/conf/insert+delete

## Modifications to DescribeCoverage

None.

## Modifications to *GetCoverage*

None.

## InsertCoverage request

### InsertCoverage request

This request adds the coverage passed into the server’s offering. The coverage can be passed directly or via http reference. In any case, it has to be a coverage as per GMLCOV [OGC 09-146] in some encoding supported by the server. Where the GML coverage schema permits xlink references inside the coverage, these may be utilized to reference the corresponding parts of a coverage instead of providing them verbatim. All references must be resolvable by the server.

Note A *GetCapabilities* request sent to a server retrieves information about the encoding formats supported, among other details.

By default, the identifier of the new coverage on the server is the one indicated in the input coverage. As coverage identifiers have to be unique within a WCS offering, sometimes it is not easy for a client to determine an unused name for a new coverage. By setting a flag, useId=new, a client can request that the server shall generate a unique identifier and assign it to the newly inserted coverage. In this case, the identifier value provided with the input coverage is ignored.

Note The server may use some name generation scheme, such as consecutive numbering or an ASCII encoding of timestamps, coverage type, geographic extension, etc. However, such naming is pure informal convention and not utilized in any way by WCS.

By default, coverages inserted cannot be modified in their domain set (i.e., their spatial/temporal extent). By passing the optional parameter isExtensible, the coverage is marked as having an extensible domain set (while all other constituents remain fixed, such as the Native CRS – hence, it is not possible, for example, to change the number of dimensions a coverage has).

Note See requirements class *update* for further details.

1. An *InsertCoverage* request **shall** adhere to **Error! Reference source not found.**, **Error! Reference source not found.**, and the XML schema defined for this Transaction Extension.



Figure 1 — InsertCoverage request UML diagram

Table 3 — Components of WCST::InsertCoverage request structure

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Definition** | **Data type** | **Multiplicity** |
| coverage | Coverage to be inserted into the WCS offering | Abstract­Coverage | zero or one(optional) |
| coverageRef | Reference to the coverage to be inserted into the WCS offering | anyURI | zero or one(optional) |
| generateId | Flag indicating that the server shall assign a newly generated coverage id.Default if not present: use id of input coverage | (empty) | zero or one(optional) |
| isExtensible | Flag indicating that the domain set of the coverage created can be altered lateron through *UpdateCoverages*.Default: domain set is fixed | (empty) | zero or one(optional) |

1. An *InsertCoverage* request **shall** contain either a coverage or a coverageRef parameter.
2. The coverage parameter in an *InsertCoverage* request, if present, **shall** contain a coverage document as per GMLCOV [OGC 09-146].
3. The coverageRef parameter in an *InsertCoverage* request, if present, **shall** be a URL resolving to a coverage document as per GMLCOV [OGC 09-146].

### InsertCoverage response

The response to a successful *InsertCoverage* request is the identifier of the newly created coverage.

1. The response to a successful *InsertCoverage* request **shall** adhere to Figure 2, Table 4, and the XML schema defined for this Transaction Extension.



Figure 2 — InsertCoverage response UML diagram

Table 4 — Components of WCST::InsertCoverage response structure

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Definition** | **Data type** | **Multiplicity** |
| coverageId | Identifier of the coverage inserted into the WCS offering | NCName | one(mandatory) |

If a name was indicated in the coverage this is the name of the new coverage returned; otherwise, the server provides a self-selected name which is unique within this server’s offering.

1. The response to a successful *InsertCoverage* request **shall** be
- some coverage identifier previously not existing in the server’s offering if the request contained a generateId pararmeter,
- the identifier of the input coverage if the request contained no generateId pararmeter.

Note No requirement is placed on the effect of an isExtensible parameter; see Subclause 2.7 (*UpdateCoverage*) for the effect of this parameter.

Successful requests follow the “durability” aspect of ACID transactions.

1. After completion of a successful *InsertCoverage* request, the identifier of the coverage established in the server’s offering **shall** be listed as an existing coverage in this WCS service’s Capabilities document.

Successful requests follow the “consistency” aspect of ACID transactions.

1. After completion of a successful *InsertCoverage* request, a subsequent *GetCoverage* request accessing the coverage using the coverage identifier returned by this *InsertCoverage* **shall** deliver a valid coverage identical to the one submitted in the *InsertCoverage* request (with the coverage identifier properly substituted if generated by the server).

## DeleteCoverage request

### DeleteCoverage request

1. A *DeleteCoverage* request **shall** adhere to Figure 3, Table 5, and the XML schema defined for this Transaction Extension.



Figure 3 — DeleteCoverage request UML diagram

Table 5 — Components of WCST::DeleteCoverage request structure

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Definition** | **Data type** | **Multiplicity** |
| coverageId | Identifiers of coverages to be deleted from the WCS offering | NCName | one or more(mandatory) |

1. Each coverageId submitted in a *DeleteCoverage* request **shall** identify a coverage existing in the coverage offering of the WCS server addressed.

Note Multiple occurrences of the same identifier are not harmful.

### DeleteCoverage response

1. The response to a successful *DeleteCoverage* request **shall** be empty.
2. A *DeleteCoverage* request **shall** succeed if and only if all coverages addressed in the request have been deleted successfully.

## Atomicity and isolation

Requests follow the “atomicity” aspect of ACID transactions.

1. No effect of an *InsertCoverage* or *DeleteCoverage* request **shall** be visible in the WCS server’s behavior prior to successful completion of this request.
2. No effect of an unsuccessful *InsertCoverage* or *DeleteCoverage* request **shall** be visible in the WCS server’s future behavior.

All requests follow the “isolation” aspect of ACID transactions.

1. During processing of an *InsertCoverage* or *DeleteCoverage* request in a WCS server, no intermediate state of processing **shall** be visible to other, concurrent requests to this WCS server, but only the complete, final result of the operation.

## Encodings

### Overview

This Subclause specifies, for each WCS protocol binding that a client and server support, encoding of an *InsertCoverage* and *DeleteCoverage* operation.

### GET/KVP Encoding

1. In an *InsertCoverage* request using the GET/KVP protocol, a CoverageRef parameter with http URL url **shall** be represented by an http key/value pair as follows:
 COVERAGEREF=url

Note Passing a coverage directly in the request is not supported by the GET/KVP protocol binding.

1. In an *InsertCoverage* request using the GET/KVP protocol, a generateId parameter **shall** be represented as
 GENERATEID=x
where x is any valid parameter string.

Note As the value will be ignored anyway it is recommended to use an empty string or some string that suggests a Boolean *true*:
 GENERATEID=true

1. In an *InsertCoverage* request using the GET/KVP protocol, an isExtensible parameter **shall** be represented as
 ISEXTENSIBLE=x
where x is any valid parameter string.

Example The following is a complete *InsertCoverage* request in GET/KVP notation, resulting in a coverage named NewLittleCoverage offered by the service, superseding any coverage identifier the coverage referenced in the request may have:

http://www.acme.com/ows?
 SERVICE=WCS &
 VERSION=2.0 &
 REQUEST=InsertCoverage &
 COVERAGEREF=http://www.acme.com/mycoverage.gml &
 GENERATEID=true &
 ISEXTENSIBLE=true

Note The COVERAGEREF URL in the above example needs to be escaped properly, as per OWS Common; this has been omitted for the reader’s convenience. Further, blanks have been introduced for the same purpose.

1. In a *DeleteCoverage* request with n>0 coverage identifiers id1,…,idn using the GET/KVP protocol, a CoverageId parameter **shall** be represented by an http key/value pair as follows:
 COVERAGEID=id1,…,idn

Example The following is a complete *DeleteCoverage* request in GET/KVP notation:

http://www.acme.com/ows?
 SERVICE=WCS &
 VERSION=2.0 &
 REQUEST=DeleteCoverage&
 COVERAGEID=GoodByeCoverage

### XML/POST Encoding

1. An *InsertCoverage* request using the XML/POST protocol **shall** be encoded as an wcst:InsertCoverage element.

Example The following is a complete *InsertCoverage* request plus a response (assuming success) in XML/POST encoding:

<?xml version="1.0" encoding="UTF-8"?>
<wcst:InsertCoverage xmlns:wcs="http://www.opengis.net/wcs/2.0"
 xmlns:wcst="http://www.opengis.net/wcs\_service-extension\_transaction/2.0"
 xmlns:gml="http://www.opengis.net/gml/3.2"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/wcst/2.0 http://schemas.opengis.net/wcs/2.0/wcsTransaction.xsd"
 service="WCS" version="2.0.1">
 <wcst:coverageRef>
 http://www.acme.com/mycoverage.gml
 </wcst:coverageRef>
 <wcst:generateId/>
 <wcst:isExtensible/>
</wcst:InsertCoverage>

<?xml version="1.0" encoding="UTF-8"?>
<wcst:InsertCoverageResponse
 xmlns:wcst="http://www.opengis.net/wcs\_service-extension\_transaction/2.0"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/wcst/2.0 http://schemas.opengis.net/wcs/2.0/wcsTransaction.xsd" >
 NewLittleCoverage
</wcst:InsertCoverageResponse>

1. A *DeleteCoverage* request using the XML/POST protocol **shall** be encoded as a wcst:DeleteCoverage element.

Example The following is a complete *DeleteCoverage* request in XML/POST encoding:

<?xml version="1.0" encoding="UTF-8"?>
<wcst:DeleteCoverage xmlns:wcs="http://www.opengis.net/wcs/2.0"
 xmlns:wcst="http://www.opengis.net/wcs\_service-extension\_transaction/2.0"
 xmlns:gml="http://www.opengis.net/gml/3.2"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/wcst/2.0 http://schemas.opengis.net/wcs/2.0/wcsTransaction.xsd"
 service="WCS" version="2.0.1">
 <wcst:coverageId>GoodByeCoverage</wcst:coverageId>
</wcst:DeleteCoverage>

### SOAP Encoding

1. An *InsertCoverage* request using the SOAP protocol **shall** be encoded as an wcst:Insert­Coverage element.
2. A *DeleteCoverage* request using the SOAP protocol **shall** be encoded as a wcst:Delete­Coverage element.

## Exceptions

1. When a WCS server encounters an error while evaluating an *InsertCoverage* or *DeleteCoverage* operation it **shall** return an exception report message chosen as indicated in Table 6 with a locator parameter value as specified in the right column of Table 6 for each exception­Code listed.

Table 6 — Transaction extension exception codes

|  |  |  |  |
| --- | --- | --- | --- |
| **exceptionCode value** | **HTTP code** | **Meaning of exception code** | **locator value** |
| InvalidCoverage | 404 | Document uploaded is not a coverage  | Position of vio­lat­ing element / para­meter |
| CoverageNotFound | 404 | Server does not hold any coverage with the identifier provided | Identifier of the non-existing coverage |
| CannotStoreCoverage | 500 | Server cannot store coverage submitted for insertion, due to storage space or other constraints  | Coverage / cov­er­age reference caus­ing this ex­ception |
| CoverageTypeNotSupported | 501 | Server does not support the type of the coverage submitted for insertion | Coverage / cov­er­age reference caus­ing this ex­ception |

# *Update* requirements class

## Overview

This Clause 2 establishes an optional Transaction Extension requirements class, *update*. Clients and servers supporting this *update* requirements class shall support modification of coverages offered by a WCS server.

## Modifications to *GetCapabilities*

A server announces support of the *update* requirements class to a client by adding the URL identifying this extension to the list of supported extensions delivered in the Capabilities document.

1. A WCS service implementing requirements class *update* of this Transaction Extension **shall** include the following URI in the Profile element of the ServiceIdentification in a *GetCapabilities* response:
 http://www.opengis.net/spec/WCS\_service-extension\_transaction/2.0/conf/update

## Modifications to DescribeCoverage

None.

## Modifications to *GetCoverage*

None.

## Modifications to *InsertCoverage*

None.

## Modifications to *DeleteCoverage*

None.

## UpdateCoverage request

### Overview

The *UpdateCoverage* request type serves to modify some or all range values of a coverage existing in a coverage offering.

Note No other coverage components, beyond range set values, can be changed through *Update­Coverage*. In particular, it is not possible to change the domain set (i.e., the overall spatio-temporal extent) and the range type (such as nil values).

A coverage’s range set can be replaced completely or partially (so-called “partial update”). For a complete update, no further parameters are required; for a partial update, however, the set of target positions to be updated must be indicated by specifying

* The domain subset to be updated (unless the whole of the input coverage’s domain set is to be replaced), expressed in the stored coverage’s Native CRS.
* The range components to be updated (unless all range components are to be replaced).

Example An *UpdateCoverage* request may contain an RGB image of which only the red band is used for updating, according to the range component specification.

* Optionally, a mask indicating which direct positions of the input coverage are to be used for updating the coverage on the server. Only those direct positions are considered for the update where the mask has a direct position as well and where additionally the range value of the mask at this direct position has a value of 1.

By way of these indicators, a subset of the input coverage can be used for updating.

Example A 2-D lat/long input coverage may replace a rectangular part of a 3-D lat/long/t coverage (e.g., a satellite image timeseries), given by a 3-D bounding box [ lat1 : lat2, long1 : long2, t1 ] indicating a slice at time position t1 with the lat/long extent indicated. Cells outside of this domain will remain unaffected. Further, assuming a hyperspectral range type containing red, green, and blue components, the input coverage may substitute only these three bands, leaving all other bands unaffected. Finally, a mask may be provided indicating those areas to be updated by a value of 1.

### UpdateCoverage request

1. An *UpdateCoverage* request **shall** adhere to Figure 4, Table 8, and the XML schema defined for this WCS Transaction Extension.



Figure 4 — UpdateCoverage request UML diagram

Table 7 — Components of WCST::UpdateCoverage request structure

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Definition** | **Data type** | **Multiplicity** |
| coverageId | Identifier of the coverage to be updated | NCName | One(mandatory) |
| input­Coverage | Coverage providing cell values for replacement | Abstract­Coverage | Zero or one(mandatory) |
| input­CoverageRef | URI to coverage providing cell values for replacement | anyURI | Zero or one(optional) |
| mask | coverage indicating which cell values to update from input coverage | Abstract­Coverage | Zero or one(optional) |
| maskRef | URI to coverage indicating which cell values to update from input coverage | anyURI | Zero or one(optional) |
| subset | Trim or slice expression, one per updated coverage dimension | Dimension­Subset | Zero or more(optional) |
| range­Component | Name of range component to be updated, and corresponding band to be used input coverage | Pair of NCName | Zero or more(optional) |

Where URIs are provided these must point to valid coverages.

1. In an *UpdateCoverage* request containing a URI (as inputCoverageRef or maskRef), each such URI **shall** reference a valid coverage as per GMLCOV [OGC 09-146].

Several constraints must hold in order to ensure consistency of the resulting updated coverage.

In a complete replacement (i.e., where no domain subset, range component, or masking parameter have been indicated):

* Native CRS of input coverage = Native CRS of updated coverage
* Domain set of input coverage = domain set of updated coverage
* Range type of input coverage = range type of updated coverage

Note The updated coverage’s description (i.e., WCS::CoverageDescription) will stay the same after a complete replacement if the coverage was inserted non-extensible. If, during *InsertCoverage*, isExtensible=true was specified then the domain set (but no other constituent like Native CRS, dimension, etc.) may be changed through an *UpdateCoverage* request.

1. In an *UpdateCoverage* request containing neither a subset, nor a rangeComponent, nor a mask parameter, the following **shall** hold for an input coverage ci (referenced or passed directly) and an updated coverage cu (where “=” in case of XML elements means deep equality):
– ci/gml:boundedBy/gml:Envelope =
 cu/gml:boundedBy/gml:Envelope
– ci/gmlcov:rangeType/swe:Record/swe:field/@name =
 cu/gmlcov:rangeType/swe:Record/swe:field/@name

In a partial replacement where a **domain subset** is indicated the following must hold:

* Domain subsetting must use the axes present in the axis set of the updated overage’s Native CRS
* Only if the coverage has been created as not extensible (see Subclause 1.5) then the input coverage’s domain set must be a subset of the updated coverage’s domain set. If the coverage has been created as extensible then no such restriction holds.
1. In an *UpdateCoverage* request containing a subset parameter, the dimension item **shall** be one of the names contained in the gml:boundedBy/gml:Envelope/@axisLabels attribute of the coverage updated.

Example The following specification of the area to be replaced is valid wrt. axis labels if the updated coverage has axis labels Lat, Long, and H, assuming the Get/KVP encoding (see Subclause 2.8):

 SUBSET=Lat(5.0:10.0) & SUBSET=H(0.0)

1. In an *UpdateCoverage* request containing one or more subset parameter, all dimension names **shall** be distinct.

Example The following specification is **illegal**:

 SUBSET=Lat(5.0:10.0) & SUBSET=Lat(0.0)

In a partial replacement where **range components** are indicated the following must hold:

* All input coverage range components indicated must be present in input coverage
* All updated coverage range components indicated must be present in updated coverage
1. In an *UpdateCoverage* request containing a rangeComponent parameter, this parameter **shall** consist of an unordered list of string pairs (rci,rcu) where
– the first component rci is identical to the name attribute of the swe:field element of one of the range components of the input coverage
– the second component rcu is identical to the name attribute of the swe:field element of one of the range components of the updated coverage.

Example In the Get/KVP encoding (see Subclause 2.8) of a request updating bands 1, 2, and 5 from an RGB image the rangeComponent parameter can be written as

 RANGECOMPONENT=band1:red,band2:green,band5:blue

In a partial replacement with a mask all range values in the mask are either 0 or 1.

1. In an *UpdateCoverage* request containing a maskGrid parameter, the range set of this mask­Grid **shall** contain only range set values of 0 and 1.

Note 0 and 1 are used as indicator values instead of Boolean *true* and *false* because many relevant formats (such as image encodings) do not support a Boolean data type.

### *UpdateCoverage* response

The response to a successful *UpdateCoverage* request is empty. On the server, the following side effects will hold.

Note These changes will be visible, e.g., in subsequent *GetCoverage* requests.

1. After a successful *UpdateCoverage* request the following **shall** hold:
– the domain set of the new updated coverage is unchanged, unless a subset parameter changes it;
– the range type of the new updated coverage is unchanged;
– the range set of the new updated coverage is identical to the range set of the input coverage, unless a subset, rangeComponent, or mask parameter changes it.
2. After a successful *UpdateCoverage* request with a subset parameter, the following **shall** hold:
– the domain set of the new updated coverage is the union of original updated and input coverage, in case of a gridded coverage: all direct positions of the smallest grid encompassing updated and input coverage;
– the range set of the new updated coverage consists of
 - the input coverage values, at direct positions in the domain set of the input
 coverage, for all range components to be updated;
 - the updated coverage values, at direct positions in the domain set of the original
 updated coverage, for all range components to be updated;
 - a value x, for all other direct positions of the new updated coverage, which is taken
 non-deterministically from the nil values of the updated coverage; or 0 for those
 range components where no nil value exists.
3. After a successful *UpdateCoverage* request with a rangeComponent parameter, the following **shall** hold:
– for each direct position p of the new updated coverage which is affected by the update, range component values are as follows: for each pair (rci,rcu) in the rangeComponent parameter, the range component value named rci of the input coverage at p is identical to the range component value named rcu of the new updated coverage, as per range type definitions of input and updated coverage.
4. After a successful *UpdateCoverage* request with a mask parameter, the following **shall** hold:
– for each direct position p of the new updated coverage, the range value is changed (according to the other request parameters) if and only if position p is contained in some direct position q of the mask coverage and the range value of the mask coverage at position q is 1; otherwise the value of the new updated coverage is equal to its original value.

## Encodings

### Overview

This Subclause specifies, for each WCS protocol binding that a client and server support, encoding of an *UpdateCoverage* operation.

### GET/KVP Encoding

1. In an *UpdateCoverage* request using the GET/KVP protocol, a coverageId parameter **shall** be represented as
 COVERAGEID=c
where c is an NCName.
2. In an *UpdateCoverage* request using the GET/KVP protocol, a subset parameter **shall** be represented as
 SUBSET=a(p1:p2)
or
 SUBSET=a(p)
where a is an NCName and p, p1, p2 are coordinates of direct positions.

Example In the Get/KVP encoding of a request using subsetting the following specification is valid wrt. axis labels if the updated coverage has axis labels Lat, Long, and H:

 SUBSET=Lat(5.0:10.0) & SUBSET=H(0.0)

1. In an *UpdateCoverage* request using the GET/KVP protocol, a rangeComponent parameter **shall** be represented as
 RANGECOMPONENT=cu1:ci1,…,cun:cin
where cui and cii are NCNames for 1≤i≤n∈**N**.

Example In the Get/KVP encoding of a request updating bands 1, 2, and 5 from an RGB image the rangeComponent parameter can be written as

 RANGECOMPONENT=band1:red,band2:green,band5:blue

1. In an *UpdateCoverage* request using the GET/KVP protocol, an inputCoverageRef parameter **shall** be represented as
 INPUTCOVERAGEREF=u
where u is some URI.

Note Passing a coverage directly in the request is not supported by the GET/KVP protocol binding.

1. In an *UpdateCoverage* request using the GET/KVP protocol, a maskRef parameter **shall** be represented as
 MASKREF=u
where u is some URI.

Example The following is a complete *UpdateCoverage* request in GET/KVP encoding:

http://www.acme.com/ows?
 SERVICE=WCS &
 VERSION=2.0 &
 REQUEST=UpdateCoverage &
 COVERAGEID=CoverageToBeUpdated &
 INPUTCOVERAGEREF=http://www.acme.com/update.gml &
 SUBSET=Lat(5.0:10.0) & SUBSET=H(0.0) &
 RANGECOMPONENT=band1:red,band2:green,band5:blue & MASKREF=http://www.acme.com/mask.gml

Note Both the INPUTCOVERAGEREF and MASKREF URLs in the above example need to be escaped properly, as per OWS Common; this has been omitted for the reader’s convenience. Further, blanks have been introduced for the same purpose.

### XML/POST Encoding

1. An *UpdateCoverage* request using the XML/POST protocol **shall** be encoded as a wcst:Up­dateCoverage element.

Example The following is a complete *UpdateCoverage* request in GET/KVP encoding:

<?xml version="1.0" encoding="UTF-8"?>
<wcs:UpdateCoverage
 xmlns:wcst="http://www.opengis.net/wcs\_service-extension\_transaction/2.0"
 xmlns:wcs="http://www.opengis.net/wcs/2.0"
 xmlns:gml="http://www.opengis.net/gml/3.2"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/wcst/2.0 http://schemas.opengis.net/wcs/2.0/wcsTransaction.xsd"
 service="WCS" version="2.0.1">
 <wcst:coverageId>CoverageToBeUpdated</wcst:coverageId>
 <wcst:inputCoverageRef>
 http://www.acme.com/update.gml
 </wcst:inputCoverageRef>
 <wcs:DimensionTrim>
 <wcs:Dimension>Lat</wcs:Dimension>
 <wcs:TrimLow>5.0</wcs:TrimLow>
 <wcs:TrimHigh>10.0</wcs:TrimHigh>
 <wcs:DimensionSlice>
 <wcs:Dimension>H</wcs:Dimension>
 <wcs:SlicePoint>0.0</wcs:SlicePoint>
 </wcs:DimensionSlice>
 <wcst:rangeComponent>
 <wcst:inputRangeComponent>
 band1
 </wcst:inputRangeComponent>
 <wcst:updatedRangeComponent>
 red
 </wcst:updatedRangeComponent>
 <wcst:maskRef>
 http://www.acme.com/mask.gml
 </wcst:maskRef>
</wcst:InsertCoverage>

### SOAP Encoding

1. An *UpdateCoverage* request using the SOAP protocol **shall** be encoded as a wcst:Up­date­Coverage element.

## Exceptions

1. When a WCS server encounters an error while evaluating an *UpdateCoverage* operation it **shall** return an exception report message chosen as indicated in Table 6 with a locator parameter value as specified in the right column of Table 6 for each exceptionCode listed.

Table 8 — Exception codes for *UpdateCoverage*

|  |  |  |  |
| --- | --- | --- | --- |
| **exceptionCode value** | **HTTP code** | **Meaning of exception code** | **locator value** |
| InvalidCoverage | 404 | Document uploaded is not a coverage  | Input coverage |
| CoverageNotFound | 404 | Server does not hold coverage with identifier provided | Coverage identifier |
| DomainSetMismatch | 404 | Domain subset specified is not sa subset of server coverage’s domain set | Domain subset |
| NoSuchRangeComponent | 404 | One or more of the range components listed is not existing in input or updated coverage | First violating range component name |
| MaskMismatch | 404 | Mask domain set is not equal to input coverage domain set (i.e., mask is not aligned with input coverage) | Mask parameter |
| IllegalMask | 404 | Mask contains range values other than 0 or 1 | Mask parameter |
| InconsistentChange | 404 | Update requested would make coverage inconsistent as per GMLCOV [OGC 09-146] | Violating input element |
| NotExtensible | 404 | Updated coverage does not allow extension, and update area is not completely inside updated coverage domain set | none |

Annex A: Conformance Class Abstract Test Suite (Normative)

A Transaction Extension implementation must satisfy the following system characteristics to be conformant with this specification.

Test identifiers below are relative to <http://www.opengis.net/spec/WCS/2.0/WCS_service-extension_transaction/2.0/conf>.

Conformance class: *insert+delete*

The OGC URI identifier of this conformance class is:
[http://www.opengis.net/spec/WCS/2.0/conf/WCS\_service-extension\_transaction/2.0/conf/insert+delete](http://www.opengis.net/spec/WCS/2.0/conf/WCS_service-extension_transaction/2.0/conf/insert%2Bdelete).

TBD

Conformance class: *update*

The OGC URI identifier of this conformance class is:
<http://www.opengis.net/spec/WCS/2.0/conf/WCS_service-extension_transaction/2.0/conf/update>.

TBD

Annex B: Revision history

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. Date
 | 1. Release
 | 1. Author
 | 1. Paragraph modified
 | 1. Description
 |
| 2014-07-24 | 2.0.0  | Peter Baumann | All | Created |
| 2014-08-19 | 2.0.0 | Peter Baumann | All | Document completed (up to ATS) |
| 2015-04-30 | 2.0.0 | Scott Simmons | All | Document migrated to latest OGC Standard template |

Annex C: Bibliography

1. none