ddf
distributed data framework

Integrating DDF

Table of Contents

License

WARNING If integrating with a Highly Available Cluster of DDF, see High Availability Guidance.

DDF is structured to enable flexible integration with external clients and into larger component systems.

If integrating with an existing installation of DDF, continue to the following sections on endpoints and data/metadata management.

If a new installation of DDF is required, first see the Managing section for installation and configuration instructions, then return to this section for guidance on connecting external clients.

If you would like to set up a test or demo installation to use while developing an external client, see the Quick Start Tutorial for demo instructions.

For troubleshooting and known issues, see the Release Notes.

1. Endpoints

Federation with DDF is primarily accomplished through Endpoints accessible through http(s) requests and responses.

NOTE Not all installations will expose all available endpoints. Check with DDF administrator to confirm availability of these endpoints.

1.1. Ingest Endpoints

Ingest is the process of getting data and/or metadata into the DDF catalog framework.

These endpoints are provided by DDF to be used by integrators to ingest content or metadata.

Catalog REST Endpoint

Allows clients to perform operations on the Catalog using REST, a simple architectural style that performs communication using HTTP.

CSW Endpoint

Searches collections of descriptive information (metadata) about geospatial data and services.

FTP Endpoint

Ingests files directly into the DDF catalog using the FTP protocol.

1.2. CRUD Endpoints

To perform CRUD (Create, Read, Update, Delete) operations on data or metadata in the catalog, work with one of these endpoints.

Catalog REST Endpoint

Allows clients to perform operations on the Catalog using REST, a simple architectural style that performs communication using HTTP.

CSW Endpoint

Searches collections of descriptive information (metadata) about geospatial data and services.

1.3. Query Endpoints

Query data or metadata stored within an instance of DDF using one of these endpoints.

CSW Endpoint

Searches collections of descriptive information (metadata) about geospatial data and services.

OpenSearch Endpoint

Sends query parameters and receives search results.

1.4. Content Retrieval Endpoints

To retrieve content from an instance of DDF, use one of these endpoints.

Catalog REST Endpoint

Allows clients to perform operations on the Catalog using REST, a simple architectural style that performs communication using HTTP.

1.5. Pub-Sub Endpoints

These endpoints provide publication and subscription services to allow notifications when certain events happen within DDF.

CSW Endpoint

Searches collections of descriptive information (metadata) about geospatial data and services.

1.6. Other Endpoints

DDF also includes specialized endpoints for specific purposes.

KML Endpoint

Generates a view-based KML Query Results Network Link.

WPS Endpoint

Execute and monitor long running processes.

1.7. Endpoint Details

1.7.1. Catalog REST Endpoint

The Catalog REST Endpoint allows clients to perform operations on the Catalog using REST, a simple architectural style that performs communication using HTTP.

Catalog REST Endpoint Operations

The Catalog REST Endpoint provides the capability to query, create, update, and delete metacards and associated resources in the catalog provider.

Any web browser can be used to perform a REST read. Various other tools and libraries can be used to perform the other HTTP operations on the REST endpoint (e.g., soapUI, cURL, etc.)

Bulk operations are not supported: for all RESTful CRUD commands, only one metacard ID is supported in the URL.

Operations on the REST endpoint can be performed as follows:

Table 1. Catalog REST Endpoint Operations

Operation	HTTP Request Type	Details	Example URL
create	`POST`	HTTP request body contains the input to be ingested. `<input transformer>` is the name of the transformer to use when parsing metadata (optional).	`http://{FQDN}:{PORT}/services/catalog?transform=<input transformer>`

Operation	HTTP Request Type	Details	Example URL
update	`PUT`	The ID of the Metacard to be updated is appended to the end of the URL. The updated metadata is contained in the HTTP body. `<metacardId>` is the `Metacard.ID` of the metacard to be updated and `<input transformer>` is the name of the transformer to use when parsing an override metadata attribute (optional).	`http://{FQDN}:{PORT}/services/catalog<metacardId>?transform=<input transformer>`
delete	`DELETE`	The ID of the Metacard to be deleted is appended to the end of the URL. `<metacardId>` is the `Metacard.ID` of the metacard to be deleted.	`http://{FQDN}:{PORT}/services/catalog<metacardId>`

Operation	HTTP Request Type	Details	Example URL
read	`GET`	The ID of the Metacard to be retrieved is appended to the end of the URL. By default, the response body will include the XML representation of the Metacard. `<metacardId>` is the `Metacard.ID` of the metacard to be retrieved.	`http://{FQDN}:{PORT}/services/catalog<metacardId>`
federated read	`GET`	The SOURCE ID of a federated source is appended to the URL before the ID of the Metacard to be retrieved is appended to the end. `<sourceId>` is the `FEDERATED SOURCE ID` and `<metacardId>` is the `Metacard.ID` of the Metacard to be retrieved.	`http://{FQDN}:{PORT}/services/catalog/sources/<sourceId>/<metacardId>`
sources	`GET`	Retrieves information about federated sources, including `sourceId`, `availability`, `contentTypes`,and `version`.	`http://{FQDN}:{PORT}/services/catalog/sources/`

1.7.1.1. Catalog REST Endpoint Sample Operations

Create Operation Examples

The REST endpoint can be used to upload resources as attachments. The `create` and `update` methods both support the multipart mime format. If only a single attachment exists, it will be interpreted as a resource to be parsed, which will result in a metacard and resource being stored in the system.

If multiple attachments exist, then the REST endpoint will assume that 1 attachment is the actual resource (attachment should be named `parse.resource`) and the other attachments are overrides of metacard attributes (attachment names should follow metacard attribute names). In the case of the metadata attribute, it is possible to also have the system transform that metadata and use the results of that to override the metacard that would be generated from the resource (attachment should be named `parse.metadata`).

Create Request Example

```
POST /services/catalog?transform=xml HTTP/1.1
Host: <FQDN>:<PORT>
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW
Cache-Control: no-cache

------WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="parse.resource"; filename=""
Content-Type:


------WebKitFormBoundary7MA4YWxkTrZu0gW
Content-Disposition: form-data; name="parse.metadata"; filename=""
Content-Type:


------WebKitFormBoundary7MA4YWxkTrZu0gW--
```

Example Metacard

```xml
<?xml version="1.0" encoding="UTF-8"?>
<metacard xmlns="urn:catalog:metacard" xmlns:gml="http://www.opengis.net/gml"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:smil="http://www.w3.org/2001/SMIL20/"
xmlns:smillang="http://www.w3.org/2001/SMIL20/Language" gml:id=
"3a59483ba44e403a9f0044580343007e">
  <type>ddf.metacard</type>
  <string name="title">
    <value>Test REST Metacard</value>
  </string>
  <string name="description">
    <value>Vestibulum quis ipsum sit amet metus imperdiet vehicula. Nulla scelerisque
cursus mi.</value>
  </string>
</metacard>
```

Create Error Response Examples

If content or metadata is not ingested successfully, check for these error messages.

Malformed XML Response

If the XML being ingested is not well-formed, an HTTP 400 is returned and the following response body is returned and the specific error is logged in the error log.

```
<pre>Error while storing entry in catalog: </pre>
```

Request with Unknown Schema

If ingest is attempted with a schema that is unknown, unsupported, or not configured by the endpoint, DDF creates a generic resource metacard with the provided XML as content for the `metadata` XML field in the metacard.

Request with Missing XML Prologue

If a request with a missing XML prologue is sent to the Catalog ReST endpoint, the metacard is created successfully.

Request with Non-XML Data

If a request with non-XML data sent to the Catalog ReST endpoint, the metacard will be created and the content will stored in the `metadata` field.

Read Operation Examples

The `read` operation can be used to retrieve metadata in different formats.

1. Make a read request to the REST URL specifying the catalog id.
2. Add a transform query parameter to the end of the URL specifying the shortname of the transformer to be used (e.g., `transform=kml`).

NOTE Not all installations will have all transformers installed. Contact DDF administrator for available transformers.

Read Request Example

```
http://{FQDN}:{PORT}/services/catalog/<metacardId>
```

Metacard Transform Request Example

```
http://{FQDN}:{PORT}/services/catalog/<metacardId>?transform=<TRANSFORMER_ID>
```

TIP Transforms also work on read operations for metacards in federated sources. https://{FQDN}:{PORT}/services/catalog/sources/<sourceId>/<metacardId>?transform=<TRANSFORMER_ID>

See Metacard Transformers for details on metacard transformers.

Request with Invalid Transform

If a request specifies a transformer that is invalid, unsupported, or not configured, DDF will return an HTTP 400 and the following response body.

```
<pre>Error while storing entry in catalog: </pre>
```

Sources Operation Example

In the example below there is the local DDF distribution and a DDF OpenSearch federated source with id "DDF-OS".

Sources Response Example

```
[
   {
      "id" : "DDF-OS",
      "available" : true,
      "contentTypes" :
         [
         ],
      "version" : "2.0"
   },
   {
      "id" : "ddf.distribution",
      "available" : true,
      "contentTypes" :
         [
         ],
      "version" : "2.5.0-SNAPSHOT"
   }
]
```

1.7.2. CSW Endpoint

The CSW endpoint enables a client to search collections of descriptive information (metadata) about geospatial data and services.

Table 2. CSW Endpoint Operations

Operation	HTTP Request Type	Details	Example URL
ingest			`https://{FQDN}:{PORT}/services/csw`
query			`https://{FQDN}:{PORT}/services/csw?`
subscription			`https://{FQDN}:{PORT}/services/csw`
Get Capabilities			`https://{FQDN}:{PORT}/services/csw`

1.7.2.1. CSW Endpoint Sample Operations

NOTE *Sample Responses May Not Match Actual Responses*

Actual responses may vary from these samples, depending on your configuration. Send a GET or POST request to obtain an accurate response.

`GetCapabilities` Operation

The `GetCapabilities` operation is meant to describe the operations the catalog supports and the URLs used to access those operations. The CSW endpoint supports both `HTTP GET` and `HTTP POST` requests for the `GetCapabilities` operation. The response to either request will always be a `csw:Capabilities` XML document. This XML document is defined by the CSW-Discovery XML Schema.

`GetCapabilities` *HTTP GET*

The `HTTP GET` form of `GetCapabilities` uses query parameters via the following URL:

`GetCapabilities` *KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/csw?service=CSW&version=2.0.2&request=GetCapabilities
```

`GetCapabilities` *HTTP POST*

The `HTTP POST` form of `GetCapabilities` operates on the root CSW endpoint URL (https://{FQDN}:{PORT}/services/csw) with an XML message body that is defined by the `GetCapabilities` element of the CSW-Discovery XML Schema.

`GetCapabilities` *XML Request*

```
<?xml version="1.0" ?>
<csw:GetCapabilities
  xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
  service="CSW"
  version="2.0.2" >
</csw:GetCapabilities>
```

`GetCapabilities` *Sample Response* (`application/xml`)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Capabilities xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml=
"http://www.opengis.net/gml" xmlns:csw="http://www.opengis.net/cat/csw/2.0.2" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" version="2.0.2"
ns10:schemaLocation="http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <ows:ServiceIdentification>
        <ows:Title>Catalog Service for the Web</ows:Title>
        <ows:Abstract>DDF CSW Endpoint</ows:Abstract>
        <ows:ServiceType>CSW</ows:ServiceType>
        <ows:ServiceTypeVersion>2.0.2</ows:ServiceTypeVersion>
    </ows:ServiceIdentification>
    <ows:ServiceProvider>
        <ows:ProviderName>DDF</ows:ProviderName>
        <ows:ProviderSite/>
        <ows:ServiceContact/>
    </ows:ServiceProvider>
    <ows:OperationsMetadata>
        <ows:Operation name="GetCapabilities">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get ns2:href="https://{FQDN}:{PORT}/services/csw"/>
                    <ows:Post ns2:href="https://{FQDN}:{PORT}/services/csw">
                        <ows:Constraint name="PostEncoding">
                            <ows:Value>XML</ows:Value>
                        </ows:Constraint>
                    </ows:Post>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="sections">
                <ows:Value>ServiceIdentification</ows:Value>
                <ows:Value>ServiceProvider</ows:Value>
                <ows:Value>OperationsMetadata</ows:Value>
                <ows:Value>Filter_Capabilities</ows:Value>
            </ows:Parameter>
        </ows:Operation>
        <ows:Operation name="DescribeRecord">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get ns2:href="https://{FQDN}:{PORT}/services/csw"/>
                    <ows:Post ns2:href="https://{FQDN}:{PORT}/services/csw">
                        <ows:Constraint name="PostEncoding">
                            <ows:Value>XML</ows:Value>
                        </ows:Constraint>
                    </ows:Post>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="typeName">
                <ows:Value>csw:Record</ows:Value>
                <ows:Value>gmd:MD_Metadata</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="OutputFormat">
                <ows:Value>application/xml</ows:Value>
                <ows:Value>application/json</ows:Value>
                <ows:Value>application/atom+xml</ows:Value>
                <ows:Value>text/xml</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="schemaLanguage">
                <ows:Value>http://www.w3.org/XMLSchema</ows:Value>
                <ows:Value>http://www.w3.org/XML/Schema</ows:Value>
                <ows:Value>http://www.w3.org/2001/XMLSchema</ows:Value>
                <ows:Value>http://www.w3.org/TR/xmlschema-1/</ows:Value>
            </ows:Parameter>
        </ows:Operation>
        <ows:Operation name="GetRecords">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get ns2:href="https://{FQDN}:{PORT}/services/csw"/>
                    <ows:Post ns2:href="https://{FQDN}:{PORT}/services/csw">
                        <ows:Constraint name="PostEncoding">
                            <ows:Value>XML</ows:Value>
                        </ows:Constraint>
                    </ows:Post>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="ResultType">
                <ows:Value>hits</ows:Value>
                <ows:Value>results</ows:Value>
                <ows:Value>validate</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="OutputFormat">
                <ows:Value>application/xml</ows:Value>
                <ows:Value>application/json</ows:Value>
                <ows:Value>application/atom+xml</ows:Value>
                <ows:Value>text/xml</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="OutputSchema">
                <ows:Value>urn:catalog:metacard</ows:Value>
                <ows:Value>http://www.isotc211.org/2005/gmd</ows:Value>
                <ows:Value>http://www.opengis.net/cat/csw/2.0.2</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="typeNames">
```

```
                <ows:Value>csw:Record</ows:Value>
                <ows:Value>gmd:MD_Metadata</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="ConstraintLanguage">
                <ows:Value>Filter</ows:Value>
                <ows:Value>CQL_Text</ows:Value>
            </ows:Parameter>
            <ows:Constraint name="FederatedCatalogs">
                <ows:Value>Source1</ows:Value>
                <ows:Value>Source2</ows:Value>
            </ows:Constraint>
        </ows:Operation>
        <ows:Operation name="GetRecordById">
            <ows:DCP>
                <ows:HTTP>
                    <ows:Get ns2:href="https://{FQDN}:{PORT}/services/csw"/>
                    <ows:Post ns2:href="https://{FQDN}:{PORT}/services/csw">
                        <ows:Constraint name="PostEncoding">
                            <ows:Value>XML</ows:Value>
                        </ows:Constraint>
                    </ows:Post>
                </ows:HTTP>
            </ows:DCP>
            <ows:Parameter name="OutputSchema">
                <ows:Value>urn:catalog:metacard</ows:Value>
                <ows:Value>http://www.isotc211.org/2005/gmd</ows:Value>
                <ows:Value>http://www.opengis.net/cat/csw/2.0.2</ows:Value>
                <ows:Value>http://www.iana.org/assignments/media-types/application/octet-
stream</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="OutputFormat">
                <ows:Value>application/xml</ows:Value>
                <ows:Value>application/json</ows:Value>
                <ows:Value>application/atom+xml</ows:Value>
                <ows:Value>text/xml</ows:Value>
                <ows:Value>application/octet-stream</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="ResultType">
                <ows:Value>hits</ows:Value>
                <ows:Value>results</ows:Value>
                <ows:Value>validate</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="ElementSetName">
                <ows:Value>brief</ows:Value>
                <ows:Value>summary</ows:Value>
                <ows:Value>full</ows:Value>
            </ows:Parameter>
        </ows:Operation>
```

```xml
            <ows:Operation name="Transaction">
                <ows:DCP>
                    <ows:HTTP>
                        <ows:Post ns2:href="https://{FQDN}:{PORT}/services/csw">
                            <ows:Constraint name="PostEncoding">
                                <ows:Value>XML</ows:Value>
                            </ows:Constraint>
                        </ows:Post>
                    </ows:HTTP>
                </ows:DCP>
                <ows:Parameter name="typeNames">
                    <ows:Value>xml</ows:Value>
                    <ows:Value>appxml</ows:Value>
                    <ows:Value>csw:Record</ows:Value>
                    <ows:Value>gmd:MD_Metadata</ows:Value>
                    <ows:Value>tika</ows:Value>
                </ows:Parameter>
                <ows:Parameter name="ConstraintLanguage">
                    <ows:Value>Filter</ows:Value>
                    <ows:Value>CQL_Text</ows:Value>
                </ows:Parameter>
            </ows:Operation>
            <ows:Parameter name="service">
                <ows:Value>CSW</ows:Value>
            </ows:Parameter>
            <ows:Parameter name="version">
                <ows:Value>2.0.2</ows:Value>
            </ows:Parameter>
        </ows:OperationsMetadata>
        <ogc:Filter_Capabilities>
            <ogc:Spatial_Capabilities>
                <ogc:GeometryOperands>
                    <ogc:GeometryOperand>gml:Point</ogc:GeometryOperand>
                    <ogc:GeometryOperand>gml:LineString</ogc:GeometryOperand>
                    <ogc:GeometryOperand>gml:Polygon</ogc:GeometryOperand>
                </ogc:GeometryOperands>
                <ogc:SpatialOperators>
                    <ogc:SpatialOperator name="BBOX"/>
                    <ogc:SpatialOperator name="Beyond"/>
                    <ogc:SpatialOperator name="Contains"/>
                    <ogc:SpatialOperator name="Crosses"/>
                    <ogc:SpatialOperator name="Disjoint"/>
                    <ogc:SpatialOperator name="DWithin"/>
                    <ogc:SpatialOperator name="Intersects"/>
                    <ogc:SpatialOperator name="Overlaps"/>
                    <ogc:SpatialOperator name="Touches"/>
                    <ogc:SpatialOperator name="Within"/>
                </ogc:SpatialOperators>
            </ogc:Spatial_Capabilities>
            <ogc:Scalar_Capabilities>
                <ogc:LogicalOperators/>
                <ogc:ComparisonOperators>
                    <ogc:ComparisonOperator>Between</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>NullCheck</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>Like</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>EqualTo</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>GreaterThan</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>GreaterThanEqualTo</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>LessThan</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>LessThanEqualTo</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>EqualTo</ogc:ComparisonOperator>
                    <ogc:ComparisonOperator>NotEqualTo</ogc:ComparisonOperator>
                </ogc:ComparisonOperators>
            </ogc:Scalar_Capabilities>
            <ogc:Id_Capabilities>
                <ogc:EID/>
            </ogc:Id_Capabilities>
        </ogc:Filter_Capabilities>
    </csw:Capabilities>
```

`DescribeRecord` *Operation*

The `describeRecord` operation retrieves the type definition used by metadata of one or more registered resource types. There are two request types one for `GET` and one for `POST`. Each request has the following common data parameters:

Namespace

In `POST` operations, namespaces are defined in the xml. In `GET` operations, namespaces are defined in a comma separated list of the form: `xmlns([prefix=]namespace-url)(,xmlns([prefix=]namespace-url))*`

Service

The service being used, in this case it is fixed at CSW.

Version

The version of the service being used (2.0.2).

OutputFormat

The requester wants the response to be in this intended output. Currently, only one format is supported (application/xml). If this parameter is supplied, it is validated against the known type. If this parameter is not supported, it passes through and returns the XML response upon success.

SchemaLanguage

The schema language from the request. This is validated against the known list of schema languages supported (refer to http://www.w3.org/XML/Schema).

`DescribeRecord` *HTTP GET*

The `HTTP GET` request differs from the `POST` request in that the typeName is a comma-separated list of namespace prefix qualified types as strings (e.g., csw:Record,xyz:MyType). These prefixes are then matched against the prefix qualified namespaces in the request. This is converted to a list of QName(s). In this way, it behaves exactly as the post request that uses a list of QName(s) in the first place.

`DescribeRecord` *KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/csw?service=CSW&version=2.0.2&request=DescribeRecord&NAMES
PACE=xmlns(http://www.opengis.net/cat/csw/2.0.2)&outputFormat=application/xml&schemaLangu
age=http://www.w3.org/XML/Schema
```

`DescribeRecord` *HTTP POST*

The HTTP POST request `DescribeRecordType` has the `typeName` as a List of QName(s). The QNames are matched against the namespaces by prefix, if prefixes exist.

`DescribeRecord` *XML Request*

```
<?xml version="1.0" ?>
  <DescribeRecord
    version="2.0.2"
    service="CSW"
    outputFormat="application/xml"
    schemaLanguage="http://www.w3.org/XML/Schema"
    xmlns="http://www.opengis.net/cat/csw/2.0.2">
  </DescribeRecord>
```

`DescribeRecord` *Sample Response (*`application/xml`*)*

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:DescribeRecordResponse xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml=
"http://www.opengis.net/gml" xmlns:csw="http://www.opengis.net/cat/csw/2.0.2" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" ns10:schemaLocation=
"http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <csw:SchemaComponent targetNamespace="http://www.opengis.net/cat/csw/2.0.2"
schemaLanguage="http://www.w3.org/XML/Schema">
        <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault=
"qualified" id="csw-record" targetNamespace="http://www.opengis.net/cat/csw/2.0.2"
version="2.0.2">
            <xsd:annotation>
                <xsd:appinfo>
                    <dc:identifier>
http://schemas.opengis.net/csw/2.0.2/record.xsd</dc:identifier>
```

```
                </xsd:appinfo>
                <xsd:documentation xml:lang="en">
         This schema defines the basic record types that must be supported
         by all CSW implementations. These correspond to full, summary, and
         brief views based on DCMI metadata terms.
      </xsd:documentation>

            </xsd:annotation>
            <xsd:import namespace="http://purl.org/dc/terms/" schemaLocation="rec-
dcterms.xsd"/>
            <xsd:import namespace="http://purl.org/dc/elements/1.1/" schemaLocation="rec-
dcmes.xsd"/>
            <xsd:import namespace="http://www.opengis.net/ows" schemaLocation=
"../../ows/1.0.0/owsAll.xsd"/>
            <xsd:element abstract="true" id="AbstractRecord" name="AbstractRecord" type=
"csw:AbstractRecordType"/>
            <xsd:complexType abstract="true" id="AbstractRecordType" name=
"AbstractRecordType"/>
            <xsd:element name="DCMIRecord" substitutionGroup="csw:AbstractRecord" type=
"csw:DCMIRecordType"/>
            <xsd:complexType name="DCMIRecordType">
                <xsd:annotation>
                    <xsd:documentation xml:lang="en">
            This type encapsulates all of the standard DCMI metadata terms,
            including the Dublin Core refinements; these terms may be mapped
            to the profile-specific information model.
         </xsd:documentation>

                </xsd:annotation>
                <xsd:complexContent>
                    <xsd:extension base="csw:AbstractRecordType">
                        <xsd:sequence>
                            <xsd:group ref="dct:DCMI-terms"/>

                        </xsd:sequence>

                    </xsd:extension>

                </xsd:complexContent>

            </xsd:complexType>
            <xsd:element name="BriefRecord" substitutionGroup="csw:AbstractRecord" type=
"csw:BriefRecordType"/>
            <xsd:complexType final="#all" name="BriefRecordType">
                <xsd:annotation>
                    <xsd:documentation xml:lang="en">
            This type defines a brief representation of the common record
            format.  It extends AbstractRecordType to include only the
             dc:identifier and dc:type properties.
         </xsd:documentation>

                </xsd:annotation>
                <xsd:complexContent>
                    <xsd:extension base="csw:AbstractRecordType">
                        <xsd:sequence>
                            <xsd:element maxOccurs="unbounded" minOccurs="1" ref=
"dc:identifier"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="1" ref=
"dc:title"/>
                            <xsd:element minOccurs="0" ref="dc:type"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"ows:BoundingBox"/>

                        </xsd:sequence>

                    </xsd:extension>

                </xsd:complexContent>

            </xsd:complexType>
            <xsd:element name="SummaryRecord" substitutionGroup="csw:AbstractRecord"
type="csw:SummaryRecordType"/>
            <xsd:complexType final="#all" name="SummaryRecordType">
                <xsd:annotation>
                    <xsd:documentation xml:lang="en">
            This type defines a summary representation of the common record
            format.  It extends AbstractRecordType to include the core
            properties.
         </xsd:documentation>

                </xsd:annotation>
                <xsd:complexContent>
                    <xsd:extension base="csw:AbstractRecordType">
                        <xsd:sequence>
                            <xsd:element maxOccurs="unbounded" minOccurs="1" ref=
"dc:identifier"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="1" ref=
"dc:title"/>
                            <xsd:element minOccurs="0" ref="dc:type"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dc:subject"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dc:format"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dc:relation"/>
```

```
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dct:modified"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dct:abstract"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dct:spatial"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"ows:BoundingBox"/>

                        </xsd:sequence>

                    </xsd:extension>

                </xsd:complexContent>

            </xsd:complexType>
            <xsd:element name="Record" substitutionGroup="csw:AbstractRecord" type=
"csw:RecordType"/>
            <xsd:complexType final="#all" name="RecordType">
                <xsd:annotation>
                    <xsd:documentation xml:lang="en">
            This type extends DCMIRecordType to add ows:BoundingBox;
            it may be used to specify a spatial envelope for the
            catalogued resource.
         </xsd:documentation>

                </xsd:annotation>
                <xsd:complexContent>
                    <xsd:extension base="csw:DCMIRecordType">
                        <xsd:sequence>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" name=
"AnyText" type="csw:EmptyType"/>
                            <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"ows:BoundingBox"/>

                        </xsd:sequence>

                    </xsd:extension>

                </xsd:complexContent>

            </xsd:complexType>
            <xsd:complexType name="EmptyType"/>
        </xsd:schema>
    </csw:SchemaComponent>
    <csw:SchemaComponent targetNamespace="http://www.isotc211.org/2005/gmd"
schemaLanguage="http://www.w3.org/XML/Schema">
        <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:gco=
"http://www.isotc211.org/2005/gco" xmlns:gmd="http://www.isotc211.org/2005/gmd"
xmlns:xlink="http://www.w3.org/1999/xlink" elementFormDefault="qualified"
targetNamespace="http://www.isotc211.org/2005/gmd" version="2012-07-13">
            <xs:annotation>
                <xs:documentation>
            Geographic MetaData (GMD) extensible markup language is a component of the
XML Schema Implementation of Geographic Information Metadata documented in ISO/TS
19139:2007. GMD includes all the definitions of http://www.isotc211.org/2005/gmd
namespace. The root document of this namespace is the file gmd.xsd. This
identification.xsd schema implements the UML conceptual schema defined in A.2.2 of ISO
19115:2003. It contains the implementation of the following classes: MD_Identification,
MD_BrowseGraphic, MD_DataIdentification, MD_ServiceIdentification,
MD_RepresentativeFraction, MD_Usage, MD_Keywords, DS_Association,
MD_AggregateInformation, MD_CharacterSetCode, MD_SpatialRepresentationTypeCode,
MD_TopicCategoryCode, MD_ProgressCode, MD_KeywordTypeCode, DS_AssociationTypeCode,
DS_InitiativeTypeCode, MD_ResolutionType.
        </xs:documentation>

            </xs:annotation>
            <xs:import namespace="http://www.isotc211.org/2005/gco" schemaLocation=
"http://schemas.opengis.net/iso/19139/20070417/gco/gco.xsd"/>
            <xs:include schemaLocation="gmd.xsd"/>
            <xs:include schemaLocation="constraints.xsd"/>
            <xs:include schemaLocation="distribution.xsd"/>
            <xs:include schemaLocation="maintenance.xsd"/>
            <xs:complexType abstract="true" name="AbstractMD_Identification_Type">
                <xs:annotation>
                    <xs:documentation>Basic information about data</xs:documentation>

                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="gco:AbstractObject_Type">
                        <xs:sequence>
                            <xs:element name="citation" type=
"gmd:CI_Citation_PropertyType"/>
                            <xs:element name="abstract" type=
"gco:CharacterString_PropertyType"/>
                            <xs:element minOccurs="0" name="purpose" type=
"gco:CharacterString_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name="credit"
type="gco:CharacterString_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name="status"
type="gmd:MD_ProgressCode_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"pointOfContact" type="gmd:CI_ResponsibleParty_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"resourceMaintenance" type="gmd:MD_MaintenanceInformation_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"graphicOverview" type="gmd:MD_BrowseGraphic_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"resourceFormat" type="gmd:MD_Format_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"descriptiveKeywords" type="gmd:MD_Keywords_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"resourceSpecificUsage" type="gmd:MD_Usage_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"resourceConstraints" type="gmd:MD_Constraints_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"aggregationInfo" type="gmd:MD_AggregateInformation_PropertyType"/>

                        </xs:sequence>

                    </xs:extension>

                </xs:complexContent>

            </xs:complexType>
            <xs:element abstract="true" name="AbstractMD_Identification" type=
"gmd:AbstractMD_Identification_Type"/>
            <xs:complexType name="MD_Identification_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:AbstractMD_Identification"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="MD_BrowseGraphic_Type">
                <xs:annotation>
                    <xs:documentation>
                Graphic that provides an illustration of the dataset (should include a
legend for the graphic)
            </xs:documentation>

                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="gco:AbstractObject_Type">
                        <xs:sequence>
                            <xs:element name="fileName" type=
"gco:CharacterString_PropertyType"/>
                            <xs:element minOccurs="0" name="fileDescription" type=
"gco:CharacterString_PropertyType"/>
                            <xs:element minOccurs="0" name="fileType" type=
"gco:CharacterString_PropertyType"/>
```

```
                        </xs:sequence>

                    </xs:extension>

                </xs:complexContent>

            </xs:complexType>
            <xs:element name="MD_BrowseGraphic" type="gmd:MD_BrowseGraphic_Type"/>
            <xs:complexType name="MD_BrowseGraphic_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_BrowseGraphic"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="MD_DataIdentification_Type">
                <xs:complexContent>
                    <xs:extension base="gmd:AbstractMD_Identification_Type">
                        <xs:sequence>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"spatialRepresentationType" type="gmd:MD_SpatialRepresentationTypeCode_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"spatialResolution" type="gmd:MD_Resolution_PropertyType"/>
                            <xs:element maxOccurs="unbounded" name="language" type=
"gco:CharacterString_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"characterSet" type="gmd:MD_CharacterSetCode_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name=
"topicCategory" type="gmd:MD_TopicCategoryCode_PropertyType"/>
                            <xs:element minOccurs="0" name="environmentDescription" type
="gco:CharacterString_PropertyType"/>
                            <xs:element maxOccurs="unbounded" minOccurs="0" name="extent"
type="gmd:EX_Extent_PropertyType"/>
                            <xs:element minOccurs="0" name="supplementalInformation"
type="gco:CharacterString_PropertyType"/>

                        </xs:sequence>

                    </xs:extension>

                </xs:complexContent>

            </xs:complexType>
            <xs:element name="MD_DataIdentification" substitutionGroup=
"gmd:AbstractMD_Identification" type="gmd:MD_DataIdentification_Type"/>
            <xs:complexType name="MD_DataIdentification_PropertyType">
```

```
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_DataIdentification"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="MD_ServiceIdentification_Type">
                <xs:annotation>
                    <xs:documentation>See 19119 for further info</xs:documentation>

                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="gmd:AbstractMD_Identification_Type"/>

                </xs:complexContent>

            </xs:complexType>
            <xs:element name="MD_ServiceIdentification" substitutionGroup=
"gmd:AbstractMD_Identification" type="gmd:MD_ServiceIdentification_Type"/>
            <xs:complexType name="MD_ServiceIdentification_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_ServiceIdentification"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="MD_RepresentativeFraction_Type">
                <xs:complexContent>
                    <xs:extension base="gco:AbstractObject_Type">
                        <xs:sequence>
                            <xs:element name="denominator" type="
gco:Integer_PropertyType"/>

                        </xs:sequence>

                    </xs:extension>

                </xs:complexContent>

            </xs:complexType>
            <xs:element name="MD_RepresentativeFraction" type=
"gmd:MD_RepresentativeFraction_Type"/>
            <xs:complexType name="MD_RepresentativeFraction_PropertyType">
                <xs:sequence minOccurs="0">
```

```
                    <xs:element ref="gmd:MD_RepresentativeFraction"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="MD_Usage_Type">
                <xs:annotation>
                    <xs:documentation>
                Brief description of ways in which the dataset is currently used.
            </xs:documentation>

                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="gco:AbstractObject_Type">
                        <xs:sequence>
                            <xs:element name="specificUsage" type=
"gco:CharacterString_PropertyType"/>
                            <xs:element minOccurs="0" name="usageDateTime" type=
"gco:DateTime_PropertyType"/>
                            <xs:element minOccurs="0" name="userDeterminedLimitations"
type="gco:CharacterString_PropertyType"/>
                            <xs:element maxOccurs="unbounded" name="userContactInfo"
type="gmd:CI_ResponsibleParty_PropertyType"/>

                        </xs:sequence>

                    </xs:extension>

                </xs:complexContent>

            </xs:complexType>
            <xs:element name="MD_Usage" type="gmd:MD_Usage_Type"/>
            <xs:complexType name="MD_Usage_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_Usage"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="MD_Keywords_Type">
                <xs:annotation>
                    <xs:documentation>Keywords, their type and reference
source</xs:documentation>
```

```
                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="gco:AbstractObject_Type">
                        <xs:sequence>
                            <xs:element maxOccurs="unbounded" name="keyword" type=
"gco:CharacterString_PropertyType"/>
                            <xs:element minOccurs="0" name="type" type=
"gmd:MD_KeywordTypeCode_PropertyType"/>
                            <xs:element minOccurs="0" name="thesaurusName" type=
"gmd:CI_Citation_PropertyType"/>

                        </xs:sequence>

                    </xs:extension>

                </xs:complexContent>

            </xs:complexType>
            <xs:element name="MD_Keywords" type="gmd:MD_Keywords_Type"/>
            <xs:complexType name="MD_Keywords_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_Keywords"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="DS_Association_Type">
                <xs:complexContent>
                    <xs:extension base="gco:AbstractObject_Type">
                        <xs:sequence/>

                    </xs:extension>

                </xs:complexContent>

            </xs:complexType>
            <xs:element name="DS_Association" type="gmd:DS_Association_Type"/>
            <xs:complexType name="DS_Association_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:DS_Association"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="MD_AggregateInformation_Type">
                <xs:annotation>
                    <xs:documentation>Encapsulates the dataset aggregation
information</xs:documentation>

                </xs:annotation>
                <xs:complexContent>
                    <xs:extension base="gco:AbstractObject_Type">
                        <xs:sequence>
                            <xs:element minOccurs="0" name="aggregateDataSetName" type=
"gmd:CI_Citation_PropertyType"/>
                            <xs:element minOccurs="0" name="aggregateDataSetIdentifier"
type="gmd:MD_Identifier_PropertyType"/>
                            <xs:element name="associationType" type=
"gmd:DS_AssociationTypeCode_PropertyType"/>
                            <xs:element minOccurs="0" name="initiativeType" type=
"gmd:DS_InitiativeTypeCode_PropertyType"/>

                        </xs:sequence>

                    </xs:extension>

                </xs:complexContent>

            </xs:complexType>
            <xs:element name="MD_AggregateInformation" type=
"gmd:MD_AggregateInformation_Type"/>
            <xs:complexType name="MD_AggregateInformation_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_AggregateInformation"/>

                </xs:sequence>
                <xs:attributeGroup ref="gco:ObjectReference"/>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:complexType name="MD_Resolution_Type">
                <xs:choice>
                    <xs:element name="equivalentScale" type=
"gmd:MD_RepresentativeFraction_PropertyType"/>
                    <xs:element name="distance" type="gco:Distance_PropertyType"/>

                </xs:choice>

            </xs:complexType>
            <xs:element name="MD_Resolution" type="gmd:MD_Resolution_Type"/>
            <xs:complexType name="MD_Resolution_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_Resolution"/>

                </xs:sequence>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:simpleType name="MD_TopicCategoryCode_Type">
                <xs:annotation>
                    <xs:documentation>
                High-level geospatial data thematic classification to assist in the
grouping and search of available geospatial datasets
            </xs:documentation>

                </xs:annotation>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="farming"/>
                    <xs:enumeration value="biota"/>
                    <xs:enumeration value="boundaries"/>
                    <xs:enumeration value="climatologyMeteorologyAtmosphere"/>
                    <xs:enumeration value="economy"/>
                    <xs:enumeration value="elevation"/>
                    <xs:enumeration value="environment"/>
                    <xs:enumeration value="geoscientificInformation"/>
                    <xs:enumeration value="health"/>
                    <xs:enumeration value="imageryBaseMapsEarthCover"/>
                    <xs:enumeration value="inlandWaters"/>
                    <xs:enumeration value="intelligenceMilitary"/>
                    <xs:enumeration value="location"/>
                    <xs:enumeration value="oceans"/>
                    <xs:enumeration value="planningCadastre"/>
                    <xs:enumeration value="society"/>
                    <xs:enumeration value="structure"/>
                    <xs:enumeration value="transportation"/>
                    <xs:enumeration value="utilitiesCommunication"/>

                </xs:restriction>

            </xs:simpleType>
            <xs:element name="MD_TopicCategoryCode" substitutionGroup=
"gco:CharacterString" type="gmd:MD_TopicCategoryCode_Type"/>
            <xs:complexType name="MD_TopicCategoryCode_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_TopicCategoryCode"/>

                </xs:sequence>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
```

```
            <xs:element name="MD_CharacterSetCode" substitutionGroup="
gco:CharacterString" type="gco:CodeListValue_Type"/>
            <xs:complexType name="MD_CharacterSetCode_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_CharacterSetCode"/>

                </xs:sequence>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:element name="MD_SpatialRepresentationTypeCode" substitutionGroup=
"gco:CharacterString" type="gco:CodeListValue_Type"/>
            <xs:complexType name="MD_SpatialRepresentationTypeCode_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_SpatialRepresentationTypeCode"/>

                </xs:sequence>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:element name="MD_ProgressCode" substitutionGroup="gco:CharacterString"
type="gco:CodeListValue_Type"/>
            <xs:complexType name="MD_ProgressCode_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_ProgressCode"/>

                </xs:sequence>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:element name="MD_KeywordTypeCode" substitutionGroup="gco:CharacterString"
type="gco:CodeListValue_Type"/>
            <xs:complexType name="MD_KeywordTypeCode_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:MD_KeywordTypeCode"/>

                </xs:sequence>
                <xs:attribute ref="gco:nilReason"/>

            </xs:complexType>
            <xs:element name="DS_AssociationTypeCode" substitutionGroup=
"gco:CharacterString" type="gco:CodeListValue_Type"/>
            <xs:complexType name="DS_AssociationTypeCode_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:DS_AssociationTypeCode"/>
                </xs:sequence>
                <xs:attribute ref="gco:nilReason"/>
            </xs:complexType>
            <xs:element name="DS_InitiativeTypeCode" substitutionGroup=
"gco:CharacterString" type="gco:CodeListValue_Type"/>
            <xs:complexType name="DS_InitiativeTypeCode_PropertyType">
                <xs:sequence minOccurs="0">
                    <xs:element ref="gmd:DS_InitiativeTypeCode"/>
                </xs:sequence>
                <xs:attribute ref="gco:nilReason"/>
            </xs:complexType>
        </xs:schema>
    </csw:SchemaComponent>
</csw:DescribeRecordResponse>
```

`DescribeRecord` *HTTP POST With TypeNames*

The HTTP POST request `DescribeRecordType` has the `typeName` as a List of QName(s). The QNames are matched against the namespaces by prefix, if prefixes exist. .`DescribeRecord` XML Request

```
<?xml version="1.0" ?>
  <DescribeRecord
    version="2.0.2"
    service="CSW"
    schemaLanguage="http://www.w3.org/XML/Schema"
    xmlns="http://www.opengis.net/cat/csw/2.0.2"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2">
      <TypeName>csw:Record</TypeName>
  </DescribeRecord>
```

`DescribeRecord` *Sample Response (*`application/xml`*)*

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:DescribeRecordResponse xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml=
"http://www.opengis.net/gml" xmlns:csw="http://www.opengis.net/cat/csw/2.0.2" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" ns10:schemaLocation=
"http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <csw:SchemaComponent targetNamespace="http://www.opengis.net/cat/csw/2.0.2"
schemaLanguage="http://www.w3.org/XML/Schema">
        <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault=
"qualified" id="csw-record" targetNamespace="http://www.opengis.net/cat/csw/2.0.2"
version="2.0.2">
            <xsd:annotation>
                <xsd:appinfo>
                    <dc:identifier>
http://schemas.opengis.net/csw/2.0.2/record.xsd</dc:identifier>
```

```
                </xsd:appinfo>
                <xsd:documentation xml:lang="en">
         This schema defines the basic record types that must be supported
         by all CSW implementations. These correspond to full, summary, and
         brief views based on DCMI metadata terms.
      </xsd:documentation>

            </xsd:annotation>
            <xsd:import namespace="http://purl.org/dc/terms/" schemaLocation="rec-
dcterms.xsd"/>
            <xsd:import namespace="http://purl.org/dc/elements/1.1/" schemaLocation="rec-
dcmes.xsd"/>
            <xsd:import namespace="http://www.opengis.net/ows" schemaLocation=
"../../ows/1.0.0/owsAll.xsd"/>
            <xsd:element abstract="true" id="AbstractRecord" name="AbstractRecord" type=
"csw:AbstractRecordType"/>
            <xsd:complexType abstract="true" id="AbstractRecordType" name=
"AbstractRecordType"/>
            <xsd:element name="DCMIRecord" substitutionGroup="csw:AbstractRecord" type=
"csw:DCMIRecordType"/>
            <xsd:complexType name="DCMIRecordType">
                <xsd:annotation>
                    <xsd:documentation xml:lang="en">
            This type encapsulates all of the standard DCMI metadata terms,
            including the Dublin Core refinements; these terms may be mapped
            to the profile-specific information model.
         </xsd:documentation>

                </xsd:annotation>
                <xsd:complexContent>
                    <xsd:extension base="csw:AbstractRecordType">
                        <xsd:sequence>
                            <xsd:group ref="dct:DCMI-terms"/>

                        </xsd:sequence>

                    </xsd:extension>

                </xsd:complexContent>

            </xsd:complexType>
            <xsd:element name="BriefRecord" substitutionGroup="csw:AbstractRecord" type=
"csw:BriefRecordType"/>
            <xsd:complexType final="#all" name="BriefRecordType">
                <xsd:annotation>
                    <xsd:documentation xml:lang="en">
            This type defines a brief representation of the common record
            format.  It extends AbstractRecordType to include only the
            dc:identifier and dc:type properties.
        </xsd:documentation>

               </xsd:annotation>
               <xsd:complexContent>
                   <xsd:extension base="csw:AbstractRecordType">
                       <xsd:sequence>
                           <xsd:element maxOccurs="unbounded" minOccurs="1" ref=
"dc:identifier"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="1" ref=
"dc:title"/>
                           <xsd:element minOccurs="0" ref="dc:type"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"ows:BoundingBox"/>

                       </xsd:sequence>

                   </xsd:extension>

               </xsd:complexContent>

           </xsd:complexType>
           <xsd:element name="SummaryRecord" substitutionGroup="csw:AbstractRecord"
type="csw:SummaryRecordType"/>
           <xsd:complexType final="#all" name="SummaryRecordType">
               <xsd:annotation>
                   <xsd:documentation xml:lang="en">
           This type defines a summary representation of the common record
           format.  It extends AbstractRecordType to include the core
           properties.
        </xsd:documentation>

               </xsd:annotation>
               <xsd:complexContent>
                   <xsd:extension base="csw:AbstractRecordType">
                       <xsd:sequence>
                           <xsd:element maxOccurs="unbounded" minOccurs="1" ref=
"dc:identifier"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="1" ref=
"dc:title"/>
                           <xsd:element minOccurs="0" ref="dc:type"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dc:subject"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dc:format"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dc:relation"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dct:modified"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dct:abstract"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"dct:spatial"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"ows:BoundingBox"/>

                       </xsd:sequence>

                   </xsd:extension>

               </xsd:complexContent>

           </xsd:complexType>
           <xsd:element name="Record" substitutionGroup="csw:AbstractRecord" type=
"csw:RecordType"/>
           <xsd:complexType final="#all" name="RecordType">
               <xsd:annotation>
                   <xsd:documentation xml:lang="en">
           This type extends DCMIRecordType to add ows:BoundingBox;
           it may be used to specify a spatial envelope for the
           catalogued resource.
        </xsd:documentation>

               </xsd:annotation>
               <xsd:complexContent>
                   <xsd:extension base="csw:DCMIRecordType">
                       <xsd:sequence>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" name=
"AnyText" type="csw:EmptyType"/>
                           <xsd:element maxOccurs="unbounded" minOccurs="0" ref=
"ows:BoundingBox"/>

                       </xsd:sequence>

                   </xsd:extension>

               </xsd:complexContent>

           </xsd:complexType>
           <xsd:complexType name="EmptyType"/>
       </xsd:schema>
   </csw:SchemaComponent>
</csw:DescribeRecordResponse>
```

`GetRecords` *Operation*

The `GetRecords` operation is the principal means of searching the catalog. The matching entries may be included with the response. The client may assign a `requestId` (absolute URI). A distributed search is performed if the `DistributedSearch` element is present and the catalog is a member of a federation. Profiles may allow alternative query expressions. There are two types of request types: one for `GET` and one for `POST`. Each request has the following common data parameters:

Namespace

In POST operations, namespaces are defined in the XML. In GET operations, namespaces are defined in a comma-separated list of the form xmlns([prefix=]namespace-url)(,xmlns([pref::=]namespace-url))*.

Service

The service being used, in this case it is fixed at CSW.

Version

The version of the service being used (2.0.2).

OutputFormat

The requester wants the response to be in this intended output. Currently, only one format is supported (application/xml). If this parameter is supplied, it is validated against the known type. If this parameter is not supported, it passes through and returns the XML response upon success.

OutputSchema

The OutputSchema indicates which schema shall be used to generate the response to the GetRecords operation. The supported output schemas are listed in the GetCapabilities response.

ElementSetName

CodeList with allowed values of “brief”, “summary”, or “full”. The default value is "summary". The predefined set names of “brief”, “summary”, and “full” represent different levels of detail for the source record. "Brief" represents the least amount of detail, and "full" represents all the metadata record elements.

IMPORTANT

```
The CSW Endpoint expects all geospatial filters using the EPSG:4326
CRS to use "longitude then latitude" coordinate ordering.
Similarly, unless the output schema explicitly states otherwise, the
GetRecordsResponse will use the same coordinate ordering.
```

`GetRecords HTTP GET`

The `HTTP GET` request differs from the `POST` request in that it has the "typeNames" as a comma-separated list of namespace prefix qualified types as strings. For example `csw:Record,xyz:MyType`. These prefixes are then matched against the prefix qualified namespaces in the request. This is converted to a list QName(s). In this way, it behaves exactly as the post request that uses a list of QName(s) in the first place.

`GetRecords` *KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/csw?service=CSW&version=2.0.2&request=GetRecords&outputFor
mat=application/xml&outputSchema=http://www.opengis.net/cat/csw/2.0.2&NAMESPACE=xmlns(csw
=http://www.opengis.net/cat/csw/2.0.2)&resultType=results&typeNames=csw:Record&ElementSet
Name=brief&ConstraintLanguage=CQL_TEXT&constraint=AnyText Like '%25'
```

`GetRecords` *HTTP POST*

The `HTTP POST` request GetRecords has the `typeNames` as a List of QName(s). The QNames are matched against the namespaces by prefix, if prefixes exist.

`GetRecords` *XML Request*

```
<?xml version="1.0" ?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
        xmlns:ogc="http://www.opengis.net/ogc"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        service="CSW"
        version="2.0.2"
        maxRecords="4"
        startPosition="1"
        resultType="results"
        outputFormat="application/xml"
        outputSchema="http://www.opengis.net/cat/csw/2.0.2"
        xsi:schemaLocation="http://www.opengis.net/cat/csw/2.0.2 ../../../csw/2.0.2/CSW-
discovery.xsd">
    <Query typeNames="Record">
        <ElementSetName>summary</ElementSetName>
        <Constraint version="1.1.0">
            <ogc:Filter>
                <ogc:PropertyIsLike wildCard="%" singleChar="_" escapeChar="\">
                    <ogc:PropertyName>AnyText</ogc:PropertyName>
                    <ogc:Literal>%</ogc:Literal>
                </ogc:PropertyIsLike>
            </ogc:Filter>
        </Constraint>
    </Query>
</GetRecords>
```

`GetRecords Specific Source`

It is possible to query a `Specific Source` by specifying a query for that source-id. The valid `source-id`'s will be listed in the `FederatedCatalogs` section of the `GetCapabilities` Response. The example below shows how to query for a specifc source.

NOTE The DistributedSearch element must be specific with a hopCount greater than 1 to identify it as a federated query, otherwise the source-id's will be ignored.

GetRecords *XML Request*

```xml
<?xml version="1.0" ?>
<csw:GetRecords resultType="results"
    outputFormat="application/xml"
    outputSchema="urn:catalog:metacard"
    startPosition="1"
    maxRecords="10"
    service="CSW"
    version="2.0.2"
    xmlns:ns2="http://www.opengis.net/ogc" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2" xmlns:ns4="http://www.w3.org/1999/xlink"
xmlns:ns3="http://www.opengis.net/gml" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns5="http://www.opengis.net/ows" xmlns:ns6="http://purl.org/dc/elements/1.1/"
xmlns:ns7="http://purl.org/dc/terms/" xmlns:ns8="http://www.w3.org/2001/SMIL20/">
  <csw:DistributedSearch hopCount="2" />
    <ns10:Query typeNames="csw:Record" xmlns="" xmlns:ns10=
"http://www.opengis.net/cat/csw/2.0.2">
        <ns10:ElementSetName>full</ns10:ElementSetName>
        <ns10:Constraint version="1.1.0">
            <ns2:Filter>
              <ns2:And>
                <ns2:PropertyIsEqualTo wildCard="*" singleChar="#" escapeChar="!">
                  <ns2:PropertyName>source-id</ns2:PropertyName>
                  <ns2:Literal>Source1</ns2:Literal>
                </ns2:PropertyIsEqualTo>
                <ns2:PropertyIsLike wildCard="*" singleChar="#" escapeChar="!">
                  <ns2:PropertyName>title</ns2:PropertyName>
                    <ns2:Literal>*</ns2:Literal>
                </ns2:PropertyIsLike>
              </ns2:And>
            </ns2:Filter>
        </ns10:Constraint>
    </ns10:Query>
</csw:GetRecords>
```

GetRecords *Sample Response (*application/xml*)*

```xml
<csw:GetRecordsResponse version="2.0.2" xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:dct="http://purl.org/dc/terms/" xmlns:ows="http://www.opengis.net/ows" xmlns:xs=
"http://www.w3.org/2001/XMLSchema"  xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <csw:SearchStatus timestamp="2014-02-19T15:33:44.602-05:00"/>
```

```
    <csw:SearchResults numberOfRecordsMatched="41" numberOfRecordsReturned="4"
nextRecord="5" recordSchema="http://www.opengis.net/cat/csw/2.0.2" elementSet="summary">
      <csw:SummaryRecord>
        <dc:identifier>182fb33103414e5cbb06f8693b526239</dc:identifier>
        <dc:title>Product10</dc:title>
        <dc:type>pdf</dc:type>
        <dct:modified>2014-02-19T15:22:51.563-05:00</dct:modified>
        <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
          <ows:LowerCorner>20.0 10.0</ows:LowerCorner>
          <ows:UpperCorner>20.0 10.0</ows:UpperCorner>
        </ows:BoundingBox>
      </csw:SummaryRecord>
      <csw:SummaryRecord>
        <dc:identifier>c607440db9b0407e92000d9260d35444</dc:identifier>
        <dc:title>Product03</dc:title>
        <dc:type>pdf</dc:type>
        <dct:modified>2014-02-19T15:22:51.563-05:00</dct:modified>
        <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
          <ows:LowerCorner>6.0 3.0</ows:LowerCorner>
          <ows:UpperCorner>6.0 3.0</ows:UpperCorner>
        </ows:BoundingBox>
      </csw:SummaryRecord>
      <csw:SummaryRecord>
        <dc:identifier>034cc757abd645f0abe6acaccfe194de</dc:identifier>
        <dc:title>Product03</dc:title>
        <dc:type>pdf</dc:type>
        <dct:modified>2014-02-19T15:22:51.563-05:00</dct:modified>
        <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
          <ows:LowerCorner>6.0 3.0</ows:LowerCorner>
          <ows:UpperCorner>6.0 3.0</ows:UpperCorner>
        </ows:BoundingBox>
      </csw:SummaryRecord>
      <csw:SummaryRecord>
        <dc:identifier>5d6e987bd6084bd4919d06b63b77a007</dc:identifier>
        <dc:title>Product01</dc:title>
        <dc:type>pdf</dc:type>
        <dct:modified>2014-02-19T15:22:51.563-05:00</dct:modified>
        <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
          <ows:LowerCorner>2.0 1.0</ows:LowerCorner>
          <ows:UpperCorner>2.0 1.0</ows:UpperCorner>
        </ows:BoundingBox>
      </csw:SummaryRecord>
    </csw:SearchResults>
  </csw:GetRecordsResponse>
```

`GetRecords` *GMD OutputSchema*

It is possible to receive a response to a `GetRecords` query that conforms to the GMD specification.

`GetRecords` *XML Request*

```
<?xml version="1.0" ?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
        xmlns:ogc="http://www.opengis.net/ogc"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:gmd="http://www.isotc211.org/2005/gmd"
        xmlns:gml="http://www.opengis.net/gml"
        service="CSW"
        version="2.0.2"
        maxRecords="8"
        startPosition="1"
        resultType="results"
        outputFormat="application/xml"
        outputSchema="http://www.isotc211.org/2005/gmd"
        xsi:schemaLocation="http://www.opengis.net/cat/csw/2.0.2 ../../../csw/2.0.2/CSW-
discovery.xsd">
    <Query typeNames="gmd:MD_Metadata">
        <ElementSetName>summary</ElementSetName>
        <Constraint version="1.1.0">
            <ogc:Filter>
                <ogc:PropertyIsLike wildCard="%" singleChar="_" escapeChar="\">
                    <ogc:PropertyName>apiso:Title</ogc:PropertyName>
                    <ogc:Literal>prod%</ogc:Literal>
                </ogc:PropertyIsLike>
            </ogc:Filter>
        </Constraint>
    </Query>
</GetRecords>
```

`GetRecords` *Sample Response (*`application/xml`*)*

```
<?xml version='1.0' encoding='UTF-8'?>
<csw:GetRecordsResponse xmlns:dct="http://purl.org/dc/terms/" xmlns:xml=
"http://www.w3.org/XML/1998/namespace" xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
xmlns:ows="http://www.opengis.net/ows" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:dc=
"http://purl.org/dc/elements/1.1/" version="2.0.2">
    <csw:SearchStatus timestamp="2016-03-23T11:31:34.531-06:00"/>
    <csw:SearchResults numberOfRecordsMatched="7" numberOfRecordsReturned="1" nextRecord
="2" recordSchema="http://www.isotc211.org/2005/gmd" elementSet="summary">
        <MD_Metadata xmlns="http://www.isotc211.org/2005/gmd" xmlns:gco=
"http://www.isotc211.org/2005/gco">
            <fileIdentifier>
                <gco:CharacterString>
d5f6acd5ccf34d18af5192c38a276b12</gco:CharacterString>
            </fileIdentifier>
```

```xml
            <hierarchyLevel>
                <MD_ScopeCode codeListValue="nitf" codeList="urn:catalog:metacard"/>
            </hierarchyLevel>
            <contact/>
            <dateStamp>
                <gco:DateTime>2015-03-04T17:23:42.332-07:00</gco:DateTime>
            </dateStamp>
            <identificationInfo>
                <MD_DataIdentification>
                    <citation>
                        <CI_Citation>
                            <title>
                                <gco:CharacterString>product.ntf</gco:CharacterString>
                            </title>
                            <date>
                                <CI_Date>
                                    <date>
                                        <gco:DateTime>2015-03-04T17:23:42.332-
07:00</gco:DateTime>
                                    </date>
                                    <dateType>
                                        <CI_DateTypeCode codeList="urn:catalog:metacard"
codeListValue="created"/>
                                    </dateType>
                                </CI_Date>
                            </date>
                        </CI_Citation>
                    </citation>
                    <abstract>
                        <gco:CharacterString></gco:CharacterString>
                    </abstract>
                    <pointOfContact>
                        <CI_ResponsibleParty>
                            <organisationName>
                                <gco:CharacterString></gco:CharacterString>
                            </organisationName>
                            <role/>
                        </CI_ResponsibleParty>
                    </pointOfContact>
                    <language>
                        <gco:CharacterString>en</gco:CharacterString>
                    </language>
                    <extent>
                        <EX_Extent>
                            <geographicElement>
                                <EX_GeographicBoundingBox>
                                    <westBoundLongitude>
                                        <gco:Decimal>32.975277</gco:Decimal>
                                    </westBoundLongitude>
                                    <eastBoundLongitude>
                                        <gco:Decimal>32.996944</gco:Decimal>
                                    </eastBoundLongitude>
                                    <southBoundLatitude>
                                        <gco:Decimal>32.305</gco:Decimal>
                                    </southBoundLatitude>
                                    <northBoundLatitude>
                                        <gco:Decimal>32.323333</gco:Decimal>
                                    </northBoundLatitude>
                                </EX_GeographicBoundingBox>
                            </geographicElement>
                        </EX_Extent>
                    </extent>
                </MD_DataIdentification>
            </identificationInfo>
            <distributionInfo>
                <MD_Distribution>
                    <distributor>
                        <MD_Distributor>
                            <distributorContact/>
                            <distributorTransferOptions>
                                <MD_DigitalTransferOptions>
                                    <onLine>
                                        <CI_OnlineResource>
                                            <linkage>
                                                <URL>http://example.com</URL>
                                            </linkage>
                                        </CI_OnlineResource>
                                    </onLine>
                                </MD_DigitalTransferOptions>
                            </distributorTransferOptions>
                        </MD_Distributor>
                    </distributor>
                </MD_Distribution>
            </distributionInfo>
        </MD_Metadata>
    </csw:SearchResults>
</csw:GetRecordsResponse>
```

`GetRecords` *XML Request using UTM Coordinates*

UTM coordinates can be used when making a CSW GetRecords request using an `ogc:Filter`. UTM coordinates should use EPSG:326XX as the srsName where XX is the zone within the northern hemisphere. UTM coordinates should use EPSG:327XX as the srsName where XX is the zone within the southern hemisphere.

Note: UTM coordinates are only supported with requests providing an `ogc:Filter`, but not with CQL as there isn’t a way to specify the UTM srsName in CQL.

`GetRecords` *XML Request - UTM Northern Hemisphere Zone 36*

```
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
        xmlns:ogc="http://www.opengis.net/ogc"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:gml="http://www.opengis.net/gml"
        service="CSW"
        version="2.0.2"
        maxRecords="4"
        startPosition="1"
        resultType="results"
        outputFormat="application/xml"
        outputSchema="http://www.opengis.net/cat/csw/2.0.2"
        xsi:schemaLocation="http://www.opengis.net/cat/csw/2.0.2 ../../../csw/2.0.2/CSW-
discovery.xsd">
    <Query typeNames="Record">
        <ElementSetName>summary</ElementSetName>
        <Constraint version="1.1.0">
            <ogc:Filter>
                <ogc:Intersects>
                    <ogc:PropertyName>ows:BoundingBox</ogc:PropertyName>
                    <gml:Envelope srsName="EPSG:32636">
                        <gml:lowerCorner>171070 1106907</gml:lowerCorner>
                        <gml:upperCorner>225928 1106910</gml:upperCorner>
                    </gml:Envelope>
                </ogc:Intersects>
            </ogc:Filter>
        </Constraint>
    </Query>
</GetRecords>
```

`GetRecordById` *Operation*

The `GetRecordById` operation request retrieves the default representation of catalog records using their identifier. This operation presumes that a previous query has been performed in order to obtain the identifiers that may be used with this operation. For example, records returned by a `GetRecords` operation may contain references to other records in the catalog that may be retrieved using the `GetRecordById` operation. This operation is also a subset of the `GetRecords` operation and is included as a convenient short form for retrieving and linking to records in a catalog.

Clients can also retrieve products from the catalog using the `GetRecordById` operation. The client sets the output schema to `http://www.iana.org/assignments/media-types/application/octet-stream` and the output format to `application/octet-stream` within the request. The endpoint will do the following: check that only one Id is provided, otherwise an error will occur as multiple products cannot be retrieved. If both output format and output schema are set to values mentioned above, the catalog framework will retrieve the resource for that Id. The HTTP content type is then set to the resource's MIME type and the data is sent out. The endpoint also supports the resumption of partial downloads. This would typically occur at the request of a browser when a download was prematurely terminated.

There are two request types: one for `GET` and one for `POST`. Each request has the following common data parameters:

Namespace

In POST operations, namespaces are defined in the XML. In GET operations namespaces are defined in a comma separated list of the form: xmlns([prefix=]namespace-url)(,xmlns([prefix=]namespace-url))*.

Service

The service being used, in this case it is fixed at "CSW".

Version

The version of the service being used (2.0.2).

OutputFormat

The requester wants the response to be in this intended output. Currently, two output formats are supported: `application/xml` for retrieving records, and `application/octet-stream` for retrieving a product. If this parameter is supplied, it is validated against the known type. If this parameter is not supported, it passes through and returns the XML response upon success.

OutputSchema

The OutputSchema indicates which schema shall be used to generate the response to the GetRecordById operation. The supported output schemas are listed in the GetCapabilities response.

ElementSetName

CodeList with allowed values of “brief”, “summary”, or “full”. The default value is "summary". The predefined set names of “brief”, “summary”, and “full” represent different levels of detail for the source record. "Brief" represents the least amount of detail, and "full" represents all the metadata record elements.

Id

The Id parameter is a comma-separated list of record identifiers for the records that CSW returns to the client. In the XML encoding, one or more <Id> elements may be used to specify the record identifier to be retrieved.

`GetRecordById` *HTTP GET KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/csw?service=CSW&version=2.0.2&request=GetRecordById&NAMESP
ACE=xmlns="http://www.opengis.net/cat/csw/2.0.2"&ElementSetName=full&outputFormat=applica
tion/xml&outputSchema=http://www.opengis.net/cat/csw/2.0.2&id=fd7ff1535dfe47db8793b550d41
70424,ba908634c0eb439b84b5d9c42af1f871
```

`GetRecordById` *HTTP POST*

```
<GetRecordById xmlns="http://www.opengis.net/cat/csw/2.0.2"
  xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  service="CSW"
  version="2.0.2"
  outputFormat="application/xml"
  outputSchema="http://www.opengis.net/cat/csw/2.0.2"
  xsi:schemaLocation="http://www.opengis.net/cat/csw/2.0.2
../../../csw/2.0.2/CSW-discovery.xsd">
 <ElementSetName>full</ElementSetName>
 <Id>182fb33103414e5cbb06f8693b526239</Id>
 <Id>c607440db9b0407e92000d9260d35444</Id>
</GetRecordById>
```

`GetRecordByIdResponse` *Sample Response* (`application/xml`)

```
<csw:GetRecordByIdResponse xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:dct="http://purl.org/dc/terms/" xmlns:ows="http://www.opengis.net/ows"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <csw:Record>
      <dc:identifier>182fb33103414e5cbb06f8693b526239</dc:identifier>
<dct:bibliographicCitation>182fb33103414e5cbb06f8693b526239</dct:bibliographicCitation>
      <dc:title>Product10</dc:title>
      <dct:alternative>Product10</dct:alternative>
      <dc:type>pdf</dc:type>
      <dc:date>2014-02-19T15:22:51.563-05:00</dc:date>
      <dct:modified>2014-02-19T15:22:51.563-05:00</dct:modified>
      <dct:created>2014-02-19T15:22:51.563-05:00</dct:created>
      <dct:dateAccepted>2014-02-19T15:22:51.563-05:00</dct:dateAccepted>
      <dct:dateCopyrighted>2014-02-19T15:22:51.563-05:00</dct:dateCopyrighted>
      <dct:dateSubmitted>2014-02-19T15:22:51.563-05:00</dct:dateSubmitted>
      <dct:issued>2014-02-19T15:22:51.563-05:00</dct:issued>
      <dc:source>ddf.distribution</dc:source>
      <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
         <ows:LowerCorner>20.0 10.0</ows:LowerCorner>
         <ows:UpperCorner>20.0 10.0</ows:UpperCorner>
      </ows:BoundingBox>
   </csw:Record>
   <csw:Record>
      <dc:identifier>c607440db9b0407e92000d9260d35444</dc:identifier>
<dct:bibliographicCitation>c607440db9b0407e92000d9260d35444</dct:bibliographicCitation>
      <dc:title>Product03</dc:title>
      <dct:alternative>Product03</dct:alternative>
      <dc:type>pdf</dc:type>
      <dc:date>2014-02-19T15:22:51.563-05:00</dc:date>
      <dct:modified>2014-02-19T15:22:51.563-05:00</dct:modified>
      <dct:created>2014-02-19T15:22:51.563-05:00</dct:created>
      <dct:dateAccepted>2014-02-19T15:22:51.563-05:00</dct:dateAccepted>
      <dct:dateCopyrighted>2014-02-19T15:22:51.563-05:00</dct:dateCopyrighted>
      <dct:dateSubmitted>2014-02-19T15:22:51.563-05:00</dct:dateSubmitted>
      <dct:issued>2014-02-19T15:22:51.563-05:00</dct:issued>
      <dc:source>ddf.distribution</dc:source>
      <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
         <ows:LowerCorner>6.0 3.0</ows:LowerCorner>
         <ows:UpperCorner>6.0 3.0</ows:UpperCorner>
      </ows:BoundingBox>
   </csw:Record>
</csw:GetRecordByIdResponse>
```

Table 3. CSW Record to Metacard Mapping

CSW Record Field	Metacard Field	Brief Record	Summary Record	Record
`dc:title`	`title`	1-n	1-n	0-n
`dc:creator`				0-n
`dc:subject`			0-n	0-n
`dc:description`				0-n
`dc:publisher`				0-n
`dc:contributor`				0-n
`dc:date`	`modified`			0-n
`dc:type`	`metadata-content-type`	0-1	0-1	0-n
`dc:format`			0-n	0-n
`dc:identifier`	`id`	1-n	1-n	0-n
`dc:source`	`source-id`			0-n
`dc:language`				0-n
`dc:relation`			0-n	0-n
`dc:coverage`				0-n
`dc:rights`				0-n
`dct:abstract`	`description`		0-n	0-n
`dct:accessRights`				0-n
`dct:alternative`	`title`			0-n
`dct:audience`				0-n
`dct:available`				0-n
`dct:bibliographicCitation`	`id`			0-n
`dct:conformsTo`				0-n
`dct:created`	`created`			0-n
`dct:dateAccepted`	`effective`			0-n
`dct:Copyrighted`	`effective`			0-n
`dct:dateSubmitted`	`modified`			0-n
`dct:educationLevel`				0-n
`dct:extent`				0-n
`dct:hasFormat`				0-n
`dct:hasPart`				0-n

CSW Record Field	Metacard Field	Brief Record	Summary Record	Record
`dct:hasVersion`				0-n
`dct:isFormatOf`				0-n
`dct:isPartOf`				0-n
`dct:isReferencedBy`				0-n
`dct:isReplacedBy`				0-n
`dct:isRequiredBy`				0-n
`dct:issued`	`modified`			0-n
`dct:isVersionOf`				0-n
`dct:license`				0-n
`dct:mediator`				0-n
`dct:medium`				0-n
`dct:modified`	`modified`		0-n	0-n
`dct:provenance`				0-n
`dct:references`				0-n
`dct:replaces`				0-n
`dct:requires`				0-n
`dct:rightsHolder`				0-n
`dct:spatial`	`location`		0-n	0-n
`dct:tableOfContents`				0-n
`dct:temporal`	`effective + " - " + expiration`			0-n
`dct:valid`	`expiration`			0-n
`ows:BoundingBox`		0-n	0-n	0-n

1.7.2.2. Transaction Operations

Transactions define the operations for creating, modifying, and deleting catalog records. The supported sub-operations for the Transaction operation are Insert, Update, and Delete.

The CSW Transactions endpoint only supports `HTTP POST` requests since there are no KVP (Key-Value Pairs) operations.

1.7.2.3. Transaction Insert Sub-Operation `HTTP POST`

The Insert sub-operation is a method for one or more records to be inserted into the catalog. The schema of the record needs to conform to the schema of the information model that the catalog supports as described using the `DescribeRecord` operation.

The following example shows a request for a record to be inserted.

Sample XML Transaction `Insert` Request

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction
    service="CSW"
    version="2.0.2"
    verboseResponse="true"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2">
    <csw:Insert typeName="csw:Record">
        <csw:Record
            xmlns:ows="http://www.opengis.net/ows"
            xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:dct="http://purl.org/dc/terms/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <dc:identifier></dc:identifier>
            <dc:title>Aliquam fermentum purus quis arcu</dc:title>
            <dc:type>http://purl.org/dc/dcmitype/Text</dc:type>
            <dc:subject>Hydrography--Dictionaries</dc:subject>
            <dc:format>application/pdf</dc:format>
            <dc:date>2006-05-12</dc:date>
            <dct:abstract>Vestibulum quis ipsum sit amet metus imperdiet vehicula. Nulla
scelerisque cursus mi.</dct:abstract>
            <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
                <ows:LowerCorner>44.792 -6.171</ows:LowerCorner>
                <ows:UpperCorner>51.126 -2.228</ows:UpperCorner>
            </ows:BoundingBox>
        </csw:Record>
    </csw:Insert>
</csw:Transaction>
```

NOTE The `typeName` attribute in the `csw:Insert` element can be used to specify the document type that’s being inserted and to select the appropriate input transformer.

Sample XML transformer insert

```
<csw:Transaction service="CSW" version="2.0.2" verboseResponse="true" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2">
  <csw:Insert typeName="xml">
    <metacard xmlns="urn:catalog:metacard" xmlns:ns2="http://www.opengis.net/gml"
          xmlns:ns3="http://www.w3.org/1999/xlink" xmlns:ns4=
"http://www.w3.org/2001/SMIL20/"
          xmlns:ns5="http://www.w3.org/2001/SMIL20/Language">
        <type>ddf.metacard</type>
        <string name="title">
            <value>PlainXml near</value>
        </string>
    </metacard>
  </csw:Insert>
</csw:Transaction>
```

1.7.2.4. Transaction Insert Response

The following is an example of an `application/xml` response to the Transaction Insert sub-operation:

Note that you will only receive the `InsertResult` element if you specify `verboseResponse="true"`.

Sample XML Transaction `Insert` Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:TransactionResponse xmlns:ogc="http://www.opengis.net/ogc"
                         xmlns:gml="http://www.opengis.net/gml"
                         xmlns:ns3="http://www.w3.org/1999/xlink"
                         xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
                         xmlns:ns5="http://www.w3.org/2001/SMIL20/"
                         xmlns:dc="http://purl.org/dc/elements/1.1/"
                         xmlns:ows="http://www.opengis.net/ows"
                         xmlns:dct="http://purl.org/dc/terms/"
                         xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
                         xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance"
                         version="2.0.2"
                         ns10:schemaLocation="http://www.opengis.net/csw
/ogc/csw/2.0.2/CSW-publication.xsd">
    <csw:TransactionSummary>
        <csw:totalInserted>1</csw:totalInserted>
        <csw:totalUpdated>0</csw:totalUpdated>
        <csw:totalDeleted>0</csw:totalDeleted>
    </csw:TransactionSummary>
    <csw:InsertResult>
        <csw:BriefRecord>
            <dc:identifier>2dbcfba3f3e24e3e8f68c50f5a98a4d1</dc:identifier>
            <dc:title>Aliquam fermentum purus quis arcu</dc:title>
            <dc:type>http://purl.org/dc/dcmitype/Text</dc:type>
            <ows:BoundingBox crs="EPSG:4326">
                <ows:LowerCorner>-6.171 44.792</ows:LowerCorner>
                <ows:UpperCorner>-2.228 51.126</ows:UpperCorner>
            </ows:BoundingBox>
        </csw:BriefRecord>
    </csw:InsertResult>
</csw:TransactionResponse>
```

1.7.2.5. Transaction Update Sub-Operation `HTTP POST`

The Update sub-operation is a method to specify values used to change existing information in the catalog. If individual record property values are specified in the `Update` element, using the `RecordProperty` element, then those individual property values of a catalog record are replaced. The `RecordProperty` contains a `Name` and `Value` element. The `Name` element is used to specify the name of the record property to be updated. The `Value` element contains the value that will be used to update the record in the catalog. The values in the `Update` will completely replace those that are already in the record.

Some properties are given default `` Value`s if no `Value `` is provided.

Table 4. `RecordProperty` Default Values

Property	Default Value
`metadata-content-type`	Resource
`created`	*current time*
`modified`	*current time*
`effective`	*current time*
`metadata-content-type-version`	*myVersion*
`metacard.created`	*current time*
`metacard.modified`	*current time*
`metacard-tags`	resource, VALID
`point-of-contact`	system@localhost
`title`	*current time*

Other properties are removed if the `RecordProperty` contains a `Name` but not a `Value`.

The number of records affected by an Update operation is determined by the contents of the `Constraint` element, which contains a filter for limiting the update to a specific record or group of records.

The following example shows how the newly inserted record could be updated to modify the date field. If your update request contains a `<csw:Record>` rather than a set of `<RecordProperty>` elements plus a `<Constraint>` , the existing record with the same ID will be replaced with the new record.

Sample XML Transaction `Update` *Request*

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction
    service="CSW"
    version="2.0.2"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2">
    <csw:Update>
        <csw:Record
            xmlns:ows="http://www.opengis.net/ows"
            xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:dct="http://purl.org/dc/terms/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <dc:identifier>2dbcfba3f3e24e3e8f68c50f5a98a4d1</dc:identifier>
            <dc:title>Aliquam fermentum purus quis arcu</dc:title>
            <dc:type>http://purl.org/dc/dcmitype/Text</dc:type>
            <dc:subject>Hydrography--Dictionaries</dc:subject>
            <dc:format>application/pdf</dc:format>
            <dc:date>2008-08-10</dc:date>
            <dct:abstract>Vestibulum quis ipsum sit amet metus imperdiet vehicula. Nulla
scelerisque cursus mi.</dct:abstract>
            <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
                <ows:LowerCorner>44.792 -6.171</ows:LowerCorner>
                <ows:UpperCorner>51.126 -2.228</ows:UpperCorner>
            </ows:BoundingBox>
        </csw:Record>
    </csw:Update>
</csw:Transaction>
```

The following example shows how the newly inserted record could be updated to modify the date field while using a filter constraint with title equal to `Aliquam fermentum purus quis arcu`.

Sample XML Transaction `Update` Request with filter constraint

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction
    service="CSW"
    version="2.0.2"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
    xmlns:ogc="http://www.opengis.net/ogc">
    <csw:Update>
        <csw:RecordProperty>
            <csw:Name>title</csw:Name>
            <csw:Value>Updated Title</csw:Value>
        </csw:RecordProperty>
        <csw:RecordProperty>
            <csw:Name>date</csw:Name>
            <csw:Value>2015-08-25</csw:Value>
        </csw:RecordProperty>
        <csw:RecordProperty>
            <csw:Name>format</csw:Name>
            <csw:Value></csw:Value>
        </csw:RecordProperty>
        <csw:Constraint version="2.0.0">
            <ogc:Filter>
                <ogc:PropertyIsEqualTo>
                    <ogc:PropertyName>title</ogc:PropertyName>
                    <ogc:Literal>Aliquam fermentum purus quis arcu</ogc:Literal>
                </ogc:PropertyIsEqualTo>
            </ogc:Filter>
        </csw:Constraint>
    </csw:Update>
</csw:Transaction>
```

The following example shows how the newly inserted record could be updated to modify the date field while using a CQL filter constraint with title equal to `Aliquam fermentum purus quis arcu`.

Sample XML Transaction `Update` *Request with CQL filter constraint*

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction
    service="CSW"
    version="2.0.2"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
    xmlns:ogc="http://www.opengis.net/ogc">
    <csw:Update>
        <csw:RecordProperty>
            <csw:Name>title</csw:Name>
            <csw:Value>Updated Title</csw:Value>
        </csw:RecordProperty>
        <csw:RecordProperty>
            <csw:Name>date</csw:Name>
            <csw:Value>2015-08-25</csw:Value>
        </csw:RecordProperty>
        <csw:RecordProperty>
            <csw:Name>format</csw:Name>
            <csw:Value></csw:Value>
        </csw:RecordProperty>
        <csw:Constraint version="2.0.0">
            <ogc:CqlText>
                title = 'Aliquam fermentum purus quis arcu'
            </ogc:CqlText>
        </csw:Constraint>
    </csw:Update>
</csw:Transaction>
```

Sample XML Transaction Update Response

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:TransactionResponse xmlns:ogc="http://www.opengis.net/ogc"
                         xmlns:gml="http://www.opengis.net/gml"
                         xmlns:ns3="http://www.w3.org/1999/xlink"
                         xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
                         xmlns:ns5="http://www.w3.org/2001/SMIL20/"
                         xmlns:dc="http://purl.org/dc/elements/1.1/"
                         xmlns:ows="http://www.opengis.net/ows"
                         xmlns:dct="http://purl.org/dc/terms/"
                         xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
                         xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance"
                         ns10:schemaLocation="http://www.opengis.net/csw
/ogc/csw/2.0.2/CSW-publication.xsd"
                         version="2.0.2">
    <csw:TransactionSummary>
        <csw:totalInserted>0</csw:totalInserted>
        <csw:totalUpdated>1</csw:totalUpdated>
        <csw:totalDeleted>0</csw:totalDeleted>
    </csw:TransactionSummary>
</csw:TransactionResponse>
```

1.7.2.5.1. Transaction `Delete` Sub-Operation `HTTP POST`

The Delete sub-operation is a method to identify a set of records to be deleted from the catalog.

The following example shows a delete request for all records with a SpatialReferenceSystem name equal to `WGS-84`.

Sample XML Transaction `Delete` Request with filter constraint

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction service="CSW" version="2.0.2"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:ogc="http://www.opengis.net/ogc">
    <csw:Delete typeName="csw:Record" handle="something">
        <csw:Constraint version="2.0.0">
            <ogc:Filter>
                <ogc:PropertyIsEqualTo>
                   <ogc:PropertyName>SpatialReferenceSystem</ogc:PropertyName>
                   <ogc:Literal>WGS-84</ogc:Literal>
                </ogc:PropertyIsEqualTo>
            </ogc:Filter>
        </csw:Constraint>
    </csw:Delete>
</csw:Transaction>
```

The following example shows a delete operation specifying a CQL constraint to delete all records with a title equal to `Aliquam fermentum purus quis arcu`

Sample XML Transaction `Delete` Request with CQL filter constraint

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction service="CSW" version="2.0.2"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:ogc="http://www.opengis.net/ogc">
    <csw:Delete typeName="csw:Record" handle="something">
        <csw:Constraint version="2.0.0">
            <ogc:CqlText>
                 title = 'Aliquam fermentum purus quis arcu'
            </ogc:CqlText>
        </csw:Constraint>
    </csw:Delete>
</csw:Transaction>
```

Sample XML Transaction Delete Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:TransactionResponse
                         xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
                         xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance"
                         ns10:schemaLocation="http://www.opengis.net/csw
/ogc/csw/2.0.2/CSW-publication.xsd"
                         version="2.0.2">
    <csw:TransactionSummary>
        <csw:totalInserted>0</csw:totalInserted>
        <csw:totalUpdated>0</csw:totalUpdated>
        <csw:totalDeleted>1</csw:totalDeleted>
    </csw:TransactionSummary>
</csw:TransactionResponse>
```

1.7.2.5.2. Subscription `GetRecords` Operation

The subscription `GetRecords` operation is very similar to the `GetRecords` operation used to search the catalog but it subscribes to a search and sends events to a `ResponseHandler` endpoint as metacards are ingested matching the GetRecords request used in the subscription. The `ResponseHandler` must use the https protocol and receive a HEAD request to poll for availability and POST/PUT/DELETE requests for creation, updates, and deletions. The response to a `GetRecords` request on the subscription url will be an acknowledgement containing the original GetRecords request and a requestId The client will be assigned a requestId (URN). A Subscription listens for events from federated sources if the `DistributedSearch` element is present and the catalog is a member of a federation.

1.7.2.5.3. Subscription `GetRecords` HTTP GET

`GetRecords` *KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/csw/subscription?service=CSW&version=2.0.2&request=GetReco
rds&outputFormat=application/xml&outputSchema=http://www.opengis.net/cat/csw/2.0.2&NAMESP
ACE=xmlns(csw=http://www.opengis.net/cat/csw/2.0.2)&resultType=results&typeNames=csw:Reco
rd&ElementSetName=brief&ResponseHandler=https%3A%2F%2Fsome.ddf%2Fservices%2Fcsw%2Fsubscri
ption%2Fevent&ConstraintLanguage=CQL_TEXT&constraint=Text Like '%25'
```

1.7.2.5.4. Subscription `GetRecords` HTTP POST

Subscription `GetRecords` *XML Request*

```
<?xml version="1.0" ?>
<GetRecords xmlns="http://www.opengis.net/cat/csw/2.0.2"
        xmlns:ogc="http://www.opengis.net/ogc"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        service="CSW"
        version="2.0.2"
        maxRecords="4"
        startPosition="1"
        resultType="results"
        outputFormat="application/xml"
        outputSchema="http://www.opengis.net/cat/csw/2.0.2"
        xsi:schemaLocation="http://www.opengis.net/cat/csw/2.0.2 ../../../csw/2.0.2/CSW-
discovery.xsd">
    <ResponseHandler>https://some.ddf/services/csw/subscription/event</ResponseHandler>
    <Query typeNames="Record">
        <ElementSetName>summary</ElementSetName>
        <Constraint version="1.1.0">
            <ogc:Filter>
                <ogc:PropertyIsLike wildCard="%" singleChar="_" escapeChar="\">
                    <ogc:PropertyName>xml</ogc:PropertyName>
                    <ogc:Literal>%</ogc:Literal>
                </ogc:PropertyIsLike>
            </ogc:Filter>
        </Constraint>
    </Query>
</GetRecords>
```

1.7.2.5.5. Subscription `GetRecords` HTTP PUT

The `HTTP PUT` request `GetRecords` is used to update an existing subscription. It is the same as the `POST`, except the `requestid` URN is appended to the url.

Subscription `GetRecords` *XML Request*

```
https://{FQDN}:{PORT}/services/csw/subscription/urn:uuid:4d5a5249-be03-4fe8-afea-
6115021dd62f
```

Subscription `GetRecords` *XML Response*

```
<?xml version="1.0" ?>
<Acknowledgement timeStamp="2008-09-28T18:49:45" xmlns=
"http://www.opengis.net/cat/csw/2.0.2"
xmlns:ogc="http://www.opengis.net/ogc"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/cat/csw/2.0.2 ../../../csw/2.0.2/CSW-
discovery.xsd">
  <EchoedRequest>
    <GetRecords
            requestId="urn:uuid:4d5a5249-be03-4fe8-afea-6115021dd62f"
            service="CSW"
            version="2.0.2"
            maxRecords="4"
            startPosition="1"
            resultType="results"
            outputFormat="application/xml"
            outputSchema="urn:catalog:metacard">
        <ResponseHandler>
https://some.ddf/services/csw/subscription/event</ResponseHandler>
        <Query typeNames="Record">
            <ElementSetName>summary</ElementSetName>
            <Constraint version="1.1.0">
                <ogc:Filter>
                    <ogc:PropertyIsLike wildCard="%" singleChar="_" escapeChar="\">
                        <ogc:PropertyName>xml</ogc:PropertyName>
                        <ogc:Literal>%</ogc:Literal>
                    </ogc:PropertyIsLike>
                </ogc:Filter>
            </Constraint>
        </Query>
    </GetRecords>
  </EchoedRequest>
  <RequestId>urn:uuid:4d5a5249-be03-4fe8-afea-6115021dd62f</ns:RequestId>
</Acknowledgement>
```

Subscription `GetRecords` *event Response*

The following is an example of an `application/xml` event sent to a subscribers `ResponseHandler` using an `HTTP POST` for a create, `HTTP PUT` for an update, and `HTTP DELETE` for a delete using the default `outputSchema` of `http://www.opengis.net/cat/csw/2.0.2` if you specified another supported schema format in the subscription it will be returned in that format.

Subscription `GetRecords` *event XML Response*

```
<csw:GetRecordsResponse version="2.0.2" xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns:dct="http://purl.org/dc/terms/" xmlns:ows="http://www.opengis.net/ows" xmlns:xs=
"http://www.w3.org/2001/XMLSchema"  xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <csw:SearchStatus timestamp="2014-02-19T15:33:44.602-05:00"/>
    <csw:SearchResults numberOfRecordsMatched="1" numberOfRecordsReturned="1" nextRecord
="5" recordSchema="http://www.opengis.net/cat/csw/2.0.2" elementSet="summary">
      <csw:SummaryRecord>
        <dc:identifier>182fb33103414e5cbb06f8693b526239</dc:identifier>
        <dc:title>Product10</dc:title>
        <dc:type>pdf</dc:type>
        <dct:modified>2014-02-19T15:22:51.563-05:00</dct:modified>
        <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
          <ows:LowerCorner>20.0 10.0</ows:LowerCorner>
          <ows:UpperCorner>20.0 10.0</ows:UpperCorner>
        </ows:BoundingBox>
      </csw:SummaryRecord>
    </csw:SearchResults>
  </csw:GetRecordsResponse>
```

1.7.2.5.6. Subscription HTTP GET or HTTP DELETE Request

The following is an example `HTTP GET` Request to retrieve an active subscription

Subscription `HTTP GET` *or* `HTTP DELETE`

```
https://{FQDN}:{PORT}/services/csw/subscription/urn:uuid:4d5a5249-be03-4fe8-afea-
6115021dd62f
```

1.7.2.5.7. Subscription `HTTP GET` or `HTTP DELETE` Response

The following is an example `HTTP GET` Response retrieving an active subscription

Subscription `HTTP GET` *or* `HTTP DELETE` *XML Response*

```
<?xml version="1.0" ?>
<Acknowledgement timeStamp="2008-09-28T18:49:45" xmlns=
"http://www.opengis.net/cat/csw/2.0.2"
                                                 xmlns:ogc="http://www.opengis.net/ogc"
                                                 xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"
                                                 xsi:schemaLocation=
"http://www.opengis.net/cat/csw/2.0.2 ../../../../csw/2.0.2/CSW-discovery.xsd">
  <EchoedRequest>
    <GetRecords
            requestId="urn:uuid:4d5a5249-be03-4fe8-afea-6115021dd62f"
            service="CSW"
            version="2.0.2"
            maxRecords="4"
            startPosition="1"
            resultType="results"
            outputFormat="application/xml"
            outputSchema="urn:catalog:metacard">
        <ResponseHandler>
https://some.ddf/services/csw/subscription/event</ResponseHandler>
        <Query typeNames="Record">
            <ElementSetName>summary</ElementSetName>
            <Constraint version="1.1.0">
                <ogc:Filter>
                    <ogc:PropertyIsLike wildCard="%" singleChar="_" escapeChar="\">
                        <ogc:PropertyName>xml</ogc:PropertyName>
                        <ogc:Literal>%</ogc:Literal>
                    </ogc:PropertyIsLike>
                </ogc:Filter>
            </Constraint>
        </Query>
    </GetRecords>
  </EchoedRequest>
  <RequestId>urn:uuid:4d5a5249-be03-4fe8-afea-6115021dd62f</ns:RequestId>
</Acknowledgement>
```

1.7.2.5.8. Example Responses for CSW Endpoint Error Conditions

The following are example data and expected errors responses that will be returned for each error condition.

HTTP error codes are also returned. https://en.wikipedia.org/wiki/List_of_HTTP_status_codes#4xx_Client_errors

No Transaction Contents

This will not generate an error, but the response will tell you that nothing was processed as part of the transaction. For security purposes the `ows:ExceptionText` on invalid data is generic. The log file should be consulted for more information.

Example CSW Request with no payload

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction service="CSW" verboseResponse="true" version="2.0.2" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2">
</csw:Transaction>
```

No Payload CSW Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:TransactionResponse xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2" xmlns:gml="http://www.opengis.net/gml" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" version="2.0.2"
ns10:schemaLocation="http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <csw:TransactionSummary>
        <csw:totalInserted>0</csw:totalInserted>
        <csw:totalUpdated>0</csw:totalUpdated>
        <csw:totalDeleted>0</csw:totalDeleted>
    </csw:TransactionSummary>
</csw:TransactionResponse>
```

Malformed XML

The follow example sends malformed XML to the CSW Endpoint.

Example Malformed XML request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction
    service="CSW"
    version="2.0.2"
    verboseResponse="true"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2">
    <csw:Insert typeName="csw:Record">
        <csw:Record
            xmlns:ows="http://www.opengis.net/ows"
            xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:dct="http://purl.org/dc/terms/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <dc:identifier></dc:identifier>
            <dc:title>Aliquam fermentum purus quis arcu</dc:title>
            <dc:type>http://purl.org/dc/dcmitype/Text</dc:type>
            <dc:subject>Hydrography--Dictionaries</dc:subject>
            <dc:format>application/pdf</dc:format>
            <dc:date>2006-05-12</dc:date>
            <dct:abstract>Vestibulum quis ipsum sit amet metus imperdiet vehicula. Nulla
scelerisque cursus mi.</dct:abstract>
            <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
                <ows:LowerCorner>44.792 -6.171</ows:LowerCorner>
                <ows:UpperCorner>51.126 -2.228</ows:UpperCorner>
            </ows:BoundingBox>
        </csw:Record>
    </csw:Update>
</csw:Transaction>
```

An HTTP 400 Bad request response is returned. The error is logged in the log file and the following response body is returned.

Malformed XML CSW Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ows:ExceptionReport xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2" xmlns:gml="http://www.opengis.net/gml" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" version="1.2.0"
ns10:schemaLocation="http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <ows:Exception exceptionCode="MissingParameterValue">
        <ows:ExceptionText>Error parsing the request.  XML parameters may be missing or
invalid.</ows:ExceptionText>
    </ows:Exception>
</ows:ExceptionReport>
```

Non-CSW Request

The following example sends a non-CSW request to the CSW endpoint.

Example Non-CSW request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<document>
    <title>boucle dampish caulkers</title>
    <id>abc123</id>
</document>
```

An HTTP 400 Bad request response is returned, and the following response body is returned.

Non-CSW Data Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ows:ExceptionReport xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2" xmlns:gml="http://www.opengis.net/gml" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" version="1.2.0"
ns10:schemaLocation="http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <ows:Exception exceptionCode="InvalidParameterValue" locator="service">
        <ows:ExceptionText>Unknown Service</ows:ExceptionText>
    </ows:Exception>
</ows:ExceptionReport>
```

Request with Unknown Schema

This type of request will succeed and attribute names that match the expected names for the typeName (e.g. csw:Record) will get mapped into the metacard. In the example, the `title` attribute will get mapped to the metacard `title` attribute since it's the same attribute name as `<dc:title>` that `csw:Record` is configured to parse.

Example Unknown Schema

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction service="CSW" verboseResponse="true" version="2.0.2" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2">
    <csw:Insert typeName="csw:Record">
        <csw:Record
            xmlns:ows="http://www.opengis.net/ows"
            xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:dct="http://purl.org/dc/terms/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            xmlns:unk="http://example.com/unknown">
            <unk:id>123</unk:id>
            <unk:title>Aliquam fermentum purus quis arcu</unk:title>
        </csw:Record>
    </csw:Insert>
</csw:Transaction>
```

Metacard is created successfully.

Example Successful Unknown Schema Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:TransactionResponse xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml=
"http://www.opengis.net/gml" xmlns:csw="http://www.opengis.net/cat/csw/2.0.2" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" version="2.0.2"
ns10:schemaLocation="http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <csw:TransactionSummary>
        <csw:totalInserted>1</csw:totalInserted>
        <csw:totalUpdated>0</csw:totalUpdated>
        <csw:totalDeleted>0</csw:totalDeleted>
    </csw:TransactionSummary>
    <csw:InsertResult>
        <csw:BriefRecord>
            <dc:identifier>4ec3ec03f75344a7b4404773f97e5a03</dc:identifier>
            <dc:title>Aliquam fermentum purus quis arcu</dc:title>
            <dc:type/>
        </csw:BriefRecord>
    </csw:InsertResult>
</csw:TransactionResponse>
```

Well-formed, but Invalid TypeName

The `typeName` on the `csw:Insert` specifies the transformer to use when parsing the data. If the name specified is not configured, an error response is returned.

Example Invalid `typeName`

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:Transaction service="CSW" verboseResponse="true" version="2.0.2" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2">
    <csw:Insert typeName="invalid-data">
        <root>
            <id>abcd16df29413796b388b02ee017a315</id>
        </document>
    </csw:Insert>
</csw:Transaction>
```

An HTTP 400 Bad request response is returned. The error is logged in the log file and the following response body is returned.

Invalid typeName Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ows:ExceptionReport xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:csw=
"http://www.opengis.net/cat/csw/2.0.2" xmlns:gml="http://www.opengis.net/gml" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" version="1.2.0"
ns10:schemaLocation="http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <ows:Exception exceptionCode="MissingParameterValue">
        <ows:ExceptionText>Error parsing the request.  XML parameters may be missing or
invalid.</ows:ExceptionText>
    </ows:Exception>
</ows:ExceptionReport>
```

Request with Missing XML Prologue

The following example sends XML data to the CSW Endpoint without the XML prologue.

Example Missing XML Tag

```
<csw:Transaction
    service="CSW"
    version="2.0.2"
    verboseResponse="true"
    xmlns:csw="http://www.opengis.net/cat/csw/2.0.2">
    <csw:Insert typeName="csw:Record">
        <csw:Record
            xmlns:ows="http://www.opengis.net/ows"
            xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:dct="http://purl.org/dc/terms/"
            xmlns:xsd="http://www.w3.org/2001/XMLSchema">
            <dc:identifier></dc:identifier>
            <dc:title>Aliquam fermentum purus quis arcu</dc:title>
            <dc:type>http://purl.org/dc/dcmitype/Text</dc:type>
            <dc:subject>Hydrography--Dictionaries</dc:subject>
            <dc:format>application/pdf</dc:format>
            <dc:date>2006-05-12</dc:date>
            <dct:abstract>Vestibulum quis ipsum sit amet metus imperdiet vehicula. Nulla
scelerisque cursus mi.</dct:abstract>
            <ows:BoundingBox crs="urn:x-ogc:def:crs:EPSG:6.11:4326">
                <ows:LowerCorner>44.792 -6.171</ows:LowerCorner>
                <ows:UpperCorner>51.126 -2.228</ows:UpperCorner>
            </ows:BoundingBox>
        </csw:Record>
    </csw:Insert>
</csw:Transaction>
```

Metacard is created successfully.

Example Missing XML Tag Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<csw:TransactionResponse xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml=
"http://www.opengis.net/gml" xmlns:csw="http://www.opengis.net/cat/csw/2.0.2" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" version="2.0.2"
ns10:schemaLocation="http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <csw:TransactionSummary>
        <csw:totalInserted>1</csw:totalInserted>
        <csw:totalUpdated>0</csw:totalUpdated>
        <csw:totalDeleted>0</csw:totalDeleted>
    </csw:TransactionSummary>
    <csw:InsertResult>
        <csw:BriefRecord>
            <dc:identifier>c318d32e9c9a4bb5b1cd00bc1aafd704</dc:identifier>
            <dc:title>Aliquam fermentum purus quis arcu</dc:title>
            <dc:type>http://purl.org/dc/dcmitype/Text</dc:type>
            <ows:BoundingBox crs="EPSG:4326">
                <ows:LowerCorner>44.792 -6.171</ows:LowerCorner>
                <ows:UpperCorner>51.126 -2.228</ows:UpperCorner>
            </ows:BoundingBox>
        </csw:BriefRecord>
    </csw:InsertResult>
</csw:TransactionResponse>
```

Request with Non-XML Data

The following is a non-XML request sent to the CSW Endpoint.

Non-XML data Example

```
title: Non-XML title
id: abc123
```

An HTTP 400 Bad request response is returned. The error is logged in the log file and the following response body is returned.

Non-XML Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ows:ExceptionReport xmlns:ows="http://www.opengis.net/ows" xmlns:ns2=
"http://www.w3.org/1999/xlink" xmlns:ogc="http://www.opengis.net/ogc" xmlns:gml=
"http://www.opengis.net/gml" xmlns:csw="http://www.opengis.net/cat/csw/2.0.2" xmlns:ns6=
"http://www.w3.org/2001/SMIL20/" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dct=
"http://purl.org/dc/terms/" xmlns:ns9="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns10="http://www.w3.org/2001/XMLSchema-instance" version="1.2.0"
ns10:schemaLocation="http://www.opengis.net/csw /ogc/csw/2.0.2/CSW-publication.xsd">
    <ows:Exception exceptionCode="MissingParameterValue">
        <ows:ExceptionText>Error parsing the request.  XML parameters may be missing or
invalid.</ows:ExceptionText>
    </ows:Exception>
</ows:ExceptionReport>
```

1.7.3. FTP Endpoint

The FTP Endpoint provides a method for ingesting files directly into the DDF catalog using the FTP protocol.

The FTP endpoint can be accessed from any FTP client of choice. Some common clients are FileZilla, PuTTY, or the FTP client provided in the terminal. The default port number is **8021**. If FTPS is enabled with 2-way TLS, a client that supports client authentication is required.

Custom Ftplets can be implemented by extending the `DefaultFtplet` class provided by Apache FTP Server. Doing this will allow custom handling of various FTP commands by overriding the methods of the `DefaultFtplet`. Refer to https://mina.apache.org/ftpserver-project/ftplet.html for available methods that can be overridden. After creating a custom Ftplet, it needs to be added to the FTP server's Ftplets before the server is started. Any Ftplets that are registered to the FTP server will execute the FTP command in the order that they were registered.

Table 5. Operations

Operation	FTP Request Type	Details	Example URL
ingest	`PUT`		`ftp://<FQDN>:8021/`

The FTP endpoint supports the `PUT`, `MPUT DELE`, `RETR`, `RMD`, `APPE`, `RNTO`, `STOU`, and `SITE` operations.

The FTP endpoint supports files being uploaded as a dot-file (e.g., `.foo`) and then being renamed to the final filename (e.g., `some-file.pdf`). The endpoint will complete the ingest process when the rename command is sent.

1.7.4. KML Endpoint

Keyhole Markup Language (*KML*) is an XML notation for describing geographic annotation and visualization for 2- and 3- dimensional maps.

The KML Network Link endpoint allows a user to generate a view-based KML Query Results Network Link. This network link can be opened with Google Earth, establishing a dynamic connection between Google Earth and DDF. The root network link will create a network link for each configured source, including the local catalog. The individual source network links will perform a query against the OpenSearch Endpoint periodically based on the current view in the KML client. The query parameters for this query are obtained by a bounding box generated by Google Earth. The root network link will refresh every 12 hours or can be forced to refresh. As a user changes their current view, the query will be re-executed with the bounding box of the new view. (This query gets re-executed two seconds after the user stops moving the view.)

After the above request is sent, a KML Network Link document is returned as a response to download or open. This KML Network Link can then be opened in Google Earth.

Table 6. KML Endpoint Operations

Operation	HTTP Request Type	Details	Example URL

Example Output from KML Endpoint

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<kml xmlns="http://www.opengis.net/kml/2.2" xmlns:ns2="http://www.google.com/kml/ext/2.2"
  xmlns:ns3="http://www.w3.org/2005/Atom" xmlns:ns4=
"urn:oasis:names:tc:ciq:xsdschema:xAL:2.0">
  <NetworkLink>
    <name>DDF</name>
    <open xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xs=
"http://www.w3.org/2001/XMLSchema" xsi:type="xs:boolean">true</open>
    <Snippet maxLines="0"/>
    <Link>
      <href>http://0.0.0.0:8181/services/catalog/kml/sources</href>
      <refreshMode>onInterval</refreshMode>
      <refreshInterval>43200.0</refreshInterval>
      <viewRefreshMode>never</viewRefreshMode>
      <viewRefreshTime>0.0</viewRefreshTime>
      <viewBoundScale>0.0</viewBoundScale>
    </Link>
  </NetworkLink>
</kml>
```

The KML endpoint can also serve up Icons to be used in conjunction with the KML style document. The request below shows the format to return an icon.

NOTE	`<icon-name>` must be the name of an icon contained in the directory being served.

Return KML Icon

```
https://{FQDN}:{PORT}/services/catalog/kml/icons?<icon-name>
```

1.7.5. OpenSearch Endpoint

The OpenSearch Endpoint enables a client to send query parameters and receive search results. This endpoint uses the input query parameters to create an OpenSearch query. The client does not need to specify all of the query parameters, only the query parameters of interest.

The OpenSearch specification defines a file format to describe an OpenSearch endpoint. This file is XML-based and is used to programatically retrieve a site's endpoint, as well as the different parameter options a site holds. The parameters are defined via the OpenSearch and CDR IPT Specifications.

Many modern web browsers currently act as OpenSearch clients. The request call is an HTTP GET with the query options being parameters that are passed.

Table 7. OpenSearch Endpoint Operations

Operation	FTP Request Type	Details	Example URL
query	`GET`	This request performs a full-text search for the phrase 'Predator' on the DDF providers and provides the results as Atom-formatted XML for the web browser to render.	`https://{FQDN}:{PORT}/services/catalog/query?q=Predator`

1.7.5.1. Parameter List

Table 8. Main OpenSearch Standard

OS Element	HTTP Parameter	Possible Values	Comments
`searchTerms`	`q`	URL-encoded, space-delimited list of search terms	Complex contextual search string.

OS Element	HTTP Parameter	Possible Values	Comments
`count`	`count`	Integer >= 0	Maximum # of results to retrieve. default: `10`
`startIndex`	`start`	integer > 0	Index of first result to return. This value uses a one-based index for the results. default: `1`
`format`	`format`	Requires a transformer shortname as a string, possible values include, when available, `atom`, `html`, and `kml`. See Query Response transformers for more possible values.	Defines the format that the return type should be in. default: `atom`

Table 9. Temporal Extension

OS Element	HTTP Parameter	Possible Values	Comments
`start`	`dtstart`	RFC-3399-defined value (e.g. `YYYY-MM-DDTHH:mm:ssZ`, `yyyy-MM-dd’T’HH:mm:ss.SSSZZ`)	Specifies the beginning of the time slice of the search. Default value of "1970-01-01T00:00:00Z" is used when `dtend` is specified but `dtstart` is not specified.
`end`	`dtend`	RFC-3399-defined value (e.g. `YYYY-MM-DDTHH:mm:ssZ`, `yyyy-MM-dd’T’HH:mm:ss.SSSZZ`)	Specifies the ending of the time slice of the search Current GMT date/time is used when `dtstart` is specified but `dtend` is not specified.

NOTE	The start and end temporal criteria must be of the format specified above. Other formats are currently not supported. Example: `2011-01-01T12:00:00.111-04:00`. **The start and end temporal elements are based on modified timestamps for a metacard.**

Geospatial Extension

These geospatial query parameters are used to create a geospatial `INTERSECTS` query, where `INTERSECTS` means geometries that are not `DISJOINT` of the given geospatial parameters.

OS Element	HTTP Parameter	Possible Values	Comments
`lat`	`lat`	`EPSG:4326 (WGS84)` decimal degrees	Used in conjunction with the `lon` and `radius` parameters.
`lon`	`lon`	`EPSG:4326 (WGS84)` decimal degrees	Used in conjunction with the `lat` and `radius` parameters.
`radius`	`radius`	`EPSG:4326 (WGS84)` meters along the Earth's surface > 0	Specifies the search distance in meters from the `lon`,`lat` point. Used in conjunction with the `lat` and `lon` parameters. default: `5000`
`polygon`	`polygon`	Comma-delimited list of lat/lon (`EPSG:4326 (WGS84)` decimal degrees) pairs, in clockwise order around the polygon, where the last point is the same as the first in order to close the polygon. (e.g. `-80, -170,0,-170,80, -170,80,170,0,170, -80,170,-80,-170`)	According to the OpenSearch Geo Specification this is **deprecated**. Use the `geometry` parameter instead.
`box`	`bbox`	4 comma-delimited `EPSG:4326 (WGS84)` decimal degrees coordinates in the format West,South,East,North	

OS Element	HTTP Parameter	Possible Values	Comments

`geometry`	`geometry`	WKT Geometries
		
		Examples:
		
		`POINT(10 20)` where 10 is the longitude and 20 is the latitude.
		
		`POLYGON ( ( 30 10, 10 20, 20 40, 40 40, 30 10 ) )`. 30 is longitude and 10 is latitude for the first point.
		
		`MULTIPOLYGON (  40 40, 20 45, 45 30, 40 40,   20 35, 10 30, 10 10, 30 5, 45 20, 20 35), (30 20, 20 15, 20 25, 30 20)`
		
		`GEOMETRYCOLLECTION(POINT(4 6),LINESTRING(4 6,7 10))`
			Make sure to repeat the starting point as the last point to close the polygon.

Table 10. Extensions

OS Element	HTTP Parameter	Possible Values	Comments

`sort`	`sort`	`<sbfield>:<sborder>` where
		
		`<sbfield>` is `date` or `relevance`
		
		`<sborder>` is `asc` or `desc`
			`<sborder>` is optional but has a value of `asc` or `desc` (default is `desc`). However, when `<sbfield>` is `relevance`, `<sborder>` must be `desc`.
			
			Sorting by `date` will sort the results by the `effective` date.
			
			default: `relevance:desc`

`maxResults`	`mr`	Integer >= 0	Maximum # of results to return. If `count` is also specified, the `count` value will take precedence over the `maxResults` value. default: `1000`

OS Element	HTTP Parameter	Possible Values	Comments
`maxTimeout`	`mt`	Integer > 0	Maximum timeout (milliseconds) for query to respond. default: `300000` (5 minutes)

Table 11. Federated Search

OS Element	HTTP Parameter	Possible Values	Comments
`routeTo`	`src`	Comma-delimited list of site names to query. Varies depending on the names of the sites in the federation. `local` specifies to query the local site.	If `src` is not provided, the default behavior is to execute an enterprise search to the entire federation.

Table 12. DDF Extensions

OS Element	HTTP Parameter	Possible Values	Comments
`dateOffset`	`dtoffset`	Integer > 0	Specifies an offset (milliseconds), backwards from the current time, to search on the modified time field for entries.
`type`	`type`	Any valid datatype (e.g. `Text`)	Specifies the type of data to search for.
`version`	`version`	Comma-delimited list of strings (e.g. 20,30)	Version values for which to search.
`selector`	`selector`	Comma-delimited list of XPath string selectors (e.g. `//namespace:example`, //example`)	Selectors to narrow the query.

1.7.5.1.1. Supported Complex Contextual Query Format

The OpenSearch Endpoint supports the following operators: `AND`, `OR`, and `NOT`. These operators are case sensitive. Implicit `AND`s are also supported.

Using parentheses to change the order of operations is supported. Using quotes to group keywords into literal expressions is supported.

See the OpenSearch specification for more syntax specifics.

1.7.6. WPS Endpoint

NOTE	EXPERIMENTAL

The WPS endpoint enables a client to execute and monitor long running processes.

For a typical sequence of WPS requests, a client would first issue a GetCapabilities request to the server to obtain an up-to-date listing of available processes. Then, it may issue a DescribeProcess request to find out more details about the particular processes offered, including the supported data formats. To run a process with the desired input data, a client will issue an Execute request. The operations GetStatus and GetResult are used in conjunction with asynchronous execution.

For brevity the examples below use GET Key-value pair requests but POST is also supported. See the OGC WPS 2.0 Interface Standard for more details.

Table 13. WPS Endpoint Operations

Operation	HTTP Request Type	Details	Example URL
			`https://{FQDN}:{PORT}/services/WPS`

`GetCapabilities` *Operation*

This operation allows a client to request information about the server's capabilities and processes offered.

`GetCapabilities` *KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/wps?service=WPS&version=2.0.0&request=GetCapabilities&acce
ptVersions=2.0.0&sections=Contents,OperationsMetadata,ServiceIdentification,ServiceProvid
er
```

Capabilities (Capabilities)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns4:Capabilities xmlns:ns2="http://www.opengis.net/ows/2.0" xmlns:ns3=
"http://www.w3.org/1999/xlink" xmlns:ns4="http://www.opengis.net/wps/2.0" service="WPS"
version="2.0.0">
    <ns2:ServiceIdentification>
        <ns2:Title>Web Processing Service</ns2:Title>
        <ns2:Abstract>DDF WPS Endpoint</ns2:Abstract>
        <ns2:ServiceType>WPS</ns2:ServiceType>
        <ns2:Fees>NONE</ns2:Fees>
        <ns2:AccessConstraints>NONE</ns2:AccessConstraints>
    </ns2:ServiceIdentification>
    <ns2:ServiceProvider>
        <ns2:ProviderName>DDF</ns2:ProviderName>
        <ns2:ProviderSite/>
        <ns2:ServiceContact/>
    </ns2:ServiceProvider>
    <ns2:OperationsMetadata>
        <ns2:Operation name="GetCapabilities">
            <ns2:DCP>
                <ns2:HTTP>
                    <ns2:Get ns3:href="https://host:8993/services/wps"/>
                    <ns2:Post ns3:href="https://host:8993/services/wps"/>
                </ns2:HTTP>
            </ns2:DCP>
        </ns2:Operation>
        <ns2:Operation name="DescribeProcess">
            <ns2:DCP>
                <ns2:HTTP>
                    <ns2:Get ns3:href="https://host:8993/services/wps"/>
                    <ns2:Post ns3:href="https://host:8993/services/wps"/>
                </ns2:HTTP>
            </ns2:DCP>
        </ns2:Operation>
        <ns2:Operation name="Execute">
            <ns2:DCP>
                <ns2:HTTP>
                    <ns2:Post ns3:href="https://host:8993/services/wps"/>
                </ns2:HTTP>
            </ns2:DCP>
        </ns2:Operation>
        <ns2:Operation name="GetStatus">
            <ns2:DCP>
                <ns2:HTTP>
                    <ns2:Get ns3:href="https://host:8993/services/wps"/>
                    <ns2:Post ns3:href="https://host:8993/services/wps"/>
                </ns2:HTTP>
            </ns2:DCP>
        </ns2:Operation>
        <ns2:Operation name="GetResult">
            <ns2:DCP>
                <ns2:HTTP>
                    <ns2:Get ns3:href="https://host:8993/services/wps"/>
                    <ns2:Post ns3:href="https://host:8993/services/wps"/>
                </ns2:HTTP>
            </ns2:DCP>
        </ns2:Operation>
        <ns2:Operation name="Dismiss">
            <ns2:DCP>
                <ns2:HTTP>
                    <ns2:Get ns3:href="https://host:8993/services/wps"/>
                    <ns2:Post ns3:href="https://host:8993/services/wps"/>
                </ns2:HTTP>
            </ns2:DCP>
        </ns2:Operation>
    </ns2:OperationsMetadata>
    <ns4:Contents>
        <ns4:ProcessSummary jobControlOptions="async-execute" outputTransmission=
"reference" processVersion="1.0">
            <ns2:Title>Test Primitives</ns2:Title>
            <ns2:Abstract>Test for modeled, primitive data types.</ns2:Abstract>
            <ns2:Identifier>testPrimitives</ns2:Identifier>
        </ns4:ProcessSummary>
    </ns4:Contents>
</ns4:Capabilities>
```

`DescribeProcess` *Operation*

This operation allows a client to request detailed metadata on selected processes offered by a server.

`DescribeProcess` *KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/wps?service=WPS&version=2.0.0&request=DescribeProcess&iden
tifier=testPrimitives
```

Describe Process Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns4:ProcessOfferings xmlns:ns2="http://www.opengis.net/ows/2.0" xmlns:ns3=
"http://www.w3.org/1999/xlink" xmlns:ns4="http://www.opengis.net/wps/2.0">
    <ns4:ProcessOffering jobControlOptions="async-execute" outputTransmission="reference"
processVersion="1.0">
        <ns4:Process>
            <ns2:Title>Test Primitives</ns2:Title>
            <ns2:Abstract>Test for modeled, primitive data types.</ns2:Abstract>
            <ns2:Identifier>testPrimitives</ns2:Identifier>
            <ns4:Input minOccurs="1" maxOccurs="1">
                <ns2:Title>intParam</ns2:Title>
                <ns2:Abstract>An integer value [-2^31, 2^31-1]</ns2:Abstract>
                <ns2:Identifier>intParam</ns2:Identifier>
                <ns4:LiteralData>
                    <ns4:Format encoding="UTF-8" default="true"/>
                    <LiteralDataDomain default="true">
                        <ns2:AnyValue/>
                        <ns2:DataType ns2:reference="http://www.w3.org/TR/xmlschema-
2/#integer">Integer</ns2:DataType>
                        <ns2:DefaultValue>3</ns2:DefaultValue>
                    </LiteralDataDomain>
                </ns4:LiteralData>
            </ns4:Input>
            <ns4:Input minOccurs="1" maxOccurs="1">
                <ns2:Title>doubleParam</ns2:Title>
                <ns2:Abstract>A double-precision floating point value</ns2:Abstract>
                <ns2:Identifier>doubleParam</ns2:Identifier>
                <ns4:LiteralData>
                    <ns4:Format encoding="UTF-8" default="true"/>
                    <LiteralDataDomain default="true">
                        <ns2:AllowedValues>
                            <ns2:Range ns2:rangeClosure="open">
                                <ns2:MinimumValue>15.0</ns2:MinimumValue>
                                <ns2:MaximumValue>50.0</ns2:MaximumValue>
                            </ns2:Range>
                        </ns2:AllowedValues>
                        <ns2:DataType ns2:reference="http://www.w3.org/TR/xmlschema-
2/#double">Double</ns2:DataType>
                        <ns2:DefaultValue>50.0</ns2:DefaultValue>
                    </LiteralDataDomain>
                </ns4:LiteralData>
            </ns4:Input>
            <ns4:Input minOccurs="1" maxOccurs="1">
                <ns2:Title>byteParam</ns2:Title>
                <ns2:Abstract>A byte value [-128, 127]</ns2:Abstract>
                <ns2:Identifier>byteParam</ns2:Identifier>
                <ns4:LiteralData>
                    <ns4:Format encoding="UTF-8" default="true"/>
                    <LiteralDataDomain default="true">
                        <ns2:AnyValue/>
                        <ns2:DataType ns2:reference="http://www.w3.org/TR/xmlschema-
2/#byte">Byte</ns2:DataType>
                        <ns2:DefaultValue>1</ns2:DefaultValue>
                    </LiteralDataDomain>
                </ns4:LiteralData>
            </ns4:Input>
            <ns4:Input minOccurs="1" maxOccurs="1">
                <ns2:Title>shortParam</ns2:Title>
                <ns2:Abstract>A short value [-32768, 32767]</ns2:Abstract>
                <ns2:Identifier>shortParam</ns2:Identifier>
                <ns4:LiteralData>
                    <ns4:Format encoding="UTF-8" default="true"/>
                    <LiteralDataDomain default="true">
                        <ns2:AnyValue/>
                        <ns2:DataType ns2:reference="http://www.w3.org/TR/xmlschema-
2/#short">Short</ns2:DataType>
                        <ns2:DefaultValue>2</ns2:DefaultValue>
                    </LiteralDataDomain>
                </ns4:LiteralData>
            </ns4:Input>
            <ns4:Input minOccurs="1" maxOccurs="1">
                <ns2:Title>longParam</ns2:Title>
                <ns2:Abstract>A long value [-2^63, 2^63-1]</ns2:Abstract>
                <ns2:Identifier>longParam</ns2:Identifier>
                <ns4:LiteralData>
                    <ns4:Format encoding="UTF-8" default="true"/>
                    <LiteralDataDomain default="true">
                        <ns2:AnyValue/>
                        <ns2:DataType ns2:reference="http://www.w3.org/TR/xmlschema-
2/#long">Long</ns2:DataType>
                        <ns2:DefaultValue>4</ns2:DefaultValue>
                    </LiteralDataDomain>
                </ns4:LiteralData>
            </ns4:Input>
            <ns4:Input minOccurs="1" maxOccurs="1">
                <ns2:Title>booleanParam</ns2:Title>
                <ns2:Abstract>A boolean value [false, true]</ns2:Abstract>
                <ns2:Identifier>booleanParam</ns2:Identifier>
                <ns4:LiteralData>
                    <ns4:Format encoding="UTF-8" default="true"/>
                    <LiteralDataDomain default="true">
                        <ns2:AnyValue/>
                        <ns2:DataType ns2:reference="http://www.w3.org/TR/xmlschema-
2/#boolean">Boolean</ns2:DataType>
                        <ns2:DefaultValue>false</ns2:DefaultValue>
                    </LiteralDataDomain>
                </ns4:LiteralData>
            </ns4:Input>
            <ns4:Input minOccurs="1" maxOccurs="1">
                <ns2:Title>floatParam</ns2:Title>
                <ns2:Abstract>A long value [-2^63, 2^63-1]</ns2:Abstract>
                <ns2:Identifier>floatParam</ns2:Identifier>
                <ns4:LiteralData>
                    <ns4:Format encoding="UTF-8" default="true"/>
                    <LiteralDataDomain default="true">
                        <ns2:AnyValue/>
                        <ns2:DataType ns2:reference="http://www.w3.org/TR/xmlschema-
2/#float">Float</ns2:DataType>
                        <ns2:DefaultValue>5.0</ns2:DefaultValue>
                    </LiteralDataDomain>
                </ns4:LiteralData>
            </ns4:Input>
            <ns4:Input minOccurs="1" maxOccurs="1">
                <ns2:Title>Product Id</ns2:Title>
                <ns2:Abstract>Product Identifier</ns2:Abstract>
                <ns2:Identifier>productId</ns2:Identifier>
                <ns4:LiteralData>
                    <ns4:Format encoding="UTF-8" default="true"/>
```

```
                        <LiteralDataDomain default="true">
                            <ns2:AnyValue/>
                            <ns2:DataType ns2:reference="http://www.w3.org/TR/xmlschema-
2/#string">String</ns2:DataType>
                        </LiteralDataDomain>
                    </ns4:LiteralData>
                </ns4:Input>
                <ns4:Output>
                    <ns2:Title>Product</ns2:Title>
                    <ns2:Abstract>Raw output</ns2:Abstract>
                    <ns2:Identifier>product</ns2:Identifier>
                    <ns4:ComplexData>
                        <ns4:Format encoding="raw" default="true"/>
                    </ns4:ComplexData>
                </ns4:Output>
            </ns4:Process>
        </ns4:ProcessOffering>
</ns4:ProcessOfferings>
```

`GetStatus` *Operation*

This operation allows a client to query status information of a processing job.

`GetStatus` *KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/wps?service=WPS&version=2.0.0&request=GetStatus&jobId=FB6D
D4B0-A2BB-11E3-A5E2-0800200C9A66
```

Status Info

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns4:StatusInfo xmlns:ns2="http://www.opengis.net/ows/2.0" xmlns:ns3=
"http://www.w3.org/1999/xlink" xmlns:ns4="http://www.opengis.net/wps/2.0">
    <ns4:JobID>FB6DD4B0-A2BB-11E3-A5E2-0800200C9A66</ns4:JobID>
    <ns4:Status>Running</ns4:Status>
    <ns4:PercentCompleted>50</ns4:PercentCompleted>
</ns4:StatusInfo>
```

`GetResult` *Operation*

This operation allows a client to query the results of a processing job. The response can be in several formats depending on the request: * If the response attribute in the request is `document` the response will be in the Result format if the response attribute is `raw` then response will be in the format defined in the output definition. * If the job failed an ExceptionReport will be returned. * If the response format is 'raw' and no data is returned than an empty response with an HTTP status of 204 will be returned.

`GetResult` *KVP (Key-Value Pairs) Encoding*

```
https://{FQDN}:{PORT}/services/wps?service=WPS&version=2.0.0&request=GetResult&jobId=FB6D
D4B0-A2BB-11E3-A5E2-0800200C9A66
```

Result

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns4:Result xmlns:ns2="http://www.opengis.net/ows/2.0" xmlns:ns3=
"http://www.w3.org/1999/xlink" xmlns:ns4="http://www.opengis.net/wps/2.0">
  <ns4:JobID>FB6DD4B0-A2BB-11E3-A5E2-0800200C9A66</wps:JobID>
  <ns4:ExpirationDate>2014-12-24T24:00:00Z</wps:ExpirationDate>
  <ns4:Output id="BUFFERED_GEOMETRY">
  <ns4:Reference xlink:href="http://result.data.server/FB6DD4B0-A2BB-11E3-A5E2-
0800200C9A66/BUFFERED_GEOMETRY.xml"/>
  </ns4:Output>
</ns4:Result>
```

`Execute` *Operation*

This operation allows a client to execute a process comprised of a process identifier, the desired data inputs, and the desired output formats. The response can be in several formats depending on the request: * If the mode is `async` the response will be in the StatusInfo format. * If the mode is `sync` and the response attribute in the request is `document` the response will be in the Result format if the response attribute is `raw` then response will be in the format defined in the output definition`. * If the mode is 'auto' then the response can be either of the aforementioned response formats. * If the job failed an ExceptionReport will be returned. * If the response format is 'raw' and no data is returned than an empty response with an HTTP status of 204 will be returned.

`PostAsyncExecutionRequest` *HTTP POST*

```
https://{FQDN}:{PORT}/services/wps?service=WPS&version=2.0.0&request=Execute
```

Async Execution Request

```xml
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute
    xmlns:wps="http://www.opengis.net/wps/2.0"
    xmlns:ows="http://www.opengis.net/ows/2.0"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/wps/2.0 ../wps.xsd"

    service="WPS"
    version="2.0.0"
    response="document"
    mode="async">

<ows:Identifier>reprocess</ows:Identifier>
    <wps:Input id="imagery_id">
        <wps:Input id="mission_id">
            <wps:Data>A123</wps:Data>
        </wps:Input>
        <wps:Input id="scene_id">
            <wps:Data>10</wps:Data>
        </wps:Input>
    </wps:Input>
    <wps:Output id="product" transmission="reference"/>

</wps:Execute>
```

Execution Request Response

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ns4:StatusInfo xmlns:ns2="http://www.opengis.net/ows/2.0" xmlns:ns3=
"http://www.w3.org/1999/xlink" xmlns:ns4="http://www.opengis.net/wps/2.0">
    <ns4:JobID>615f5ed6-adac-4630-8b3e-4ec97b154cf6</ns4:JobID>
    <ns4:Status>Accepted</ns4:Status>
    <ns4:PercentCompleted>0</ns4:PercentCompleted>
</ns4:StatusInfo>
```