OGC Testbed-14

Compliance Engineering Report

Table of Contents

Publication Date: 2019-02-07

Approval Date: 2018-12-12

Submission Date: 2018-11-12

Reference number of this document: OGC 18-034r3

Reference URL for this document: http://www.opengis.net/doc/PER/t14-D027

Category: Public Engineering Report

Editor: Andrea Aime, Emanuele Tajariol, Simone Giannecchini

Title: OGC Testbed-14: Compliance Engineering Report

OGC Engineering Report

COPYRIGHT

WARNING

This document is not an OGC Standard. This document is an OGC Public Engineering Report created as a deliverable in an OGC Interoperability Initiative and is not an official position of the OGC membership. It is distributed for review and comment. It is subject to change without notice and may not be referred to as an OGC Standard. Further, any OGC Engineering Report should not be referenced as required or mandatory technology in procurements. However, the discussions in this document could very well lead to the definition of an OGC Standard.

indicate compliance with any LICENSOR standards or specifications.

This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

None of the Intellectual Property or underlying information or technology may be downloaded or otherwise exported or reexported in violation of U.S. export laws and regulations. In addition, you are responsible for complying with any local laws in your jurisdiction which may impact your right to import, export or use the Intellectual Property, and you represent that you have complied with any regulations or registration procedures required by applicable law to make this license enforceable.

Chapter 1. Summary

The OGC Compliance Program provides a free online testing facility based on TeamEngine and a set of test suites dedicated to specific protocols and versions, as well as specification profiles and extension.

This document reviews the work that has been carried out as part of the Testbed-14 compliance activity, in particular covering the development of:

- A Web Feature Service (WFS) 3 core test suite, covering both the tests and the reference implementation servers
- A Defence Geospatial Information Working Group CATalog (DGIWG CAT) 2.0 extension for the Catalog Services for the Web 2.0.2 (CSW) test suite and server reference implementation

The WFS 3.0 protocol is the next iteration of the WFS specification, focusing on open specification, ease of implementation, and modern Representational State Transfer (REST) Application Program Interface (API) approaches.

The DGIWG CAT is an application profile of the CSW, which allows to query and get metadata following the DGIWG application profile of the ISO19139 standard, which augments the metadata elements to include information relevant to the defense organizations.

Both the test suites are meant to be run by the Test, Evaluation, And Measurement (TEAM) Engine and eventually land on the OGC beta compliance test engine [http://cite.opengeospatial.org/te2] (availability on the primary site is subject to the WFS 3.0 specification being finalized and the tests being adapted to it).

1.1. Requirements & Research Motivation

The WFS3 work aims at building a complete test suite for WFS3 core and output format extensions. WFS3 brings a set of novelties influencing the test development in a number of ways:

- Use of resource-oriented protocols developed by OGC, with server implementations being potentially distributed among different server endpoints
- Documents and responses are not strictly bound to schemas and are extensible with extra information out of the box, allowing for greater variability than in the past
- Requires both core and a set of extensions (output formats) to allow for automated testing

The security client work is important enough to merit its own Engineering Report, which can be found at D003 Secure Client Engineering Report [https://github.com/opengeospatial/D003-Secure_Client_Test_Engineering_Report].

The CSW compliance aims at having a metadata catalog compliant with the DGIWG Metadata Foundation (a profile of the ISO19115/ISO19139 standard); the catalog shall be able to:

- Handle the core elements defined in the DGIWG Metadata Foundation specification
- Be queried using an extended set of fields

- Return a defined set of elements

1.2. Prior-After Comparison

The previous Compliance Interoperability & Testing Evaluation (CITE) tests did not have the machinery to deal with a resource oriented service and its peculiarity, the new tests for WFS 3.0 thus set a valuable reference for future tests.

The CSW compliance activity did not share the same novelty, but was instrumental in reviewing the DGIWG CAT specification and providing feedback for it, as well as validating the ability to implement in an actual server and test it with a compliance suite.

The reference implementations for WFS 3.0 also enjoyed new test coverage which resulted in a number of improvements for increased compatibility and interoperability.

1.3. Recommendations for Future Work

The current setup of the specification and tests imply that a server can be certified compliant only if it implements the GeoJSON extension, however at the same time, the specification core mandates no output format implementation. This could be addressed by the specification mandating the GeoJSON format, making the test suite more useful and making it easier to write generic clients at the same time.

Failing that, the WFS 3.0 test suite should try to support as many output formats as possible, in particular, the GML profiles. The HTML output format would be harder to verify instead, as there is no indication of its contents, the test suite would be left verifying links between pages without meaningful access to the data.

Security plays an important part of data dissemination, however, the current test suite only addresses open, un-secured, servers. It would be useful to get future extensions understanding OpenAPI security declaration, being able to gather credentials at the start of the test, and thus being able to issue authenticated requests against a secured server.

Finally, it is to be noted that both tests and reference implementations for WFS 3.0 are based on a draft of the specification, and might need to be adjusted according to changes that might occur in the final WFS 3.0 release.

1.4. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Contacts

Name	Organization
Andrea Aime	GeoSolutions
Emanuele Tajariol	GeoSolutions
Dirk Stenger	lat/lon

1.5. Foreword

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

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

Chapter 2. References

The following normative documents are referenced in this document.

NOTE: Only normative standards are referenced here, e.g. OGC, ISO or other SDO standards. All other references are listed in the bibliography.

- OGC: OGC 17-069, OGC Web Feature Service 3.0: Part 1 - Core - Draft 1 [https://cdn.rawgit.com/opengeospatial/WFS_FES/3.0.0-draft.1/docs/17-069.html]
- OGC: OGC 18-030, OGC Testbed-14 Secure Client Engineering Report [http://www.opengis.net/doc/PER/t14-D003]
- OGC: OGC 18-045, Secure Resource Oriented Geospatial Data Handling Engineering Report [http://www.opengis.net/doc/PER/t14-D021]
- DGIWG: Defence Profile of OGC Catalogue Service for the Web 2.0 [https://portal.opengeospatial.org/files/?artifact_id=78531]
- DGIWG: DGIWG Metadata Foundation v2.0.0 [https://portal.dgiwg.org/files/?artifact_id=67565&format=doc]

Chapter 3. Terms and definitions

For the purposes of this report, the following apply:

- dataset

  ```
  collection of data, published or curated by a single agent, and available for
  access or download in one or more formats
  ```

- feature

  ```
  abstraction of real world phenomena [ISO 19101-1:2014]
  ```

- feature collection

  ```
  a set of features from a dataset
  ```

- TestNG

  ```
  a java library used to implement unit, integration and functional tests
  ```

3.1. Abbreviated terms

- API Application Programming Interface
- ATS Abstract Test Suite
- CITE Conformance and Interoperability Testing and Evaluation
- CSW Catalog Services for the Web
- DGIWG Defence Geospatial Information Working Group
- GML Geography Markup Language
- JSON JavaScript Object Notation
- HTML HyperText Markup Language
- HTTP Hypertext Transfer Protocol
- ISO International Organization for Standardization
- OGC Open Geospatial Consortium
- SF Simple Feature
- TEAM Test, Evaluation, And Measurement (Engine)
- URL Uniform Resource Locator
- WFS Web Feature Service

Chapter 4. Overview

Section 5 describes the WFS 3.0 test suite development.

Section 6 discusses the WFS 3.0 reference implementations.

Section 7 discusses the DGIWG CAT 2.0.2 test suite development.

Section 8 discusses the DGIWG CAT 2.0.2 reference implementation.

Chapter 5. WFS 3.0 Test Suite

lat/lon, the company, provided an implementation of the WFS3 test suite that will run on the OGC Compliance Portal [http://cite.opengeospatial.org/teamengine/]. As all recent test suites, the test had the following characteristics:

- Executable via TEAM Engine
- Written in TestNG
- Based on the WFS 3.0 Abstract Test Suite (ATS), with test naming matching the test suite

The test covered the "core" conformance class, and part of what could become the OpenAPI Extension conformance class. As of November 2018, the WFS standard working group has not yet decided on the scope of the conformance class. The core is very general and not testable by itself and requires one implementable conformance class (e.g. OpenAPI).

Currently, in order to test feature results, some support for HyperText Markup Language (HTML), Geographic JavaScript Object Notation (GeoJSON) and Geography Markup Language (GML) outputs will be necessary. The tests explicitly cover the associated conformance classes, though part of them was needed to support testing the "core" class.

This test suite presents a set of novelties that will have been addressed during the development:

- Support for REST, based on link navigation as opposed to fixed Uniform Resource Locator (URL) structures
- The service can be potentially distributed over multiple servers
- The API is not strictly defined by the standard, the tests have to inspect an OpenAPI 3 document in order to find out all the missing details, and selectively ignore extensions that are not part of the test, or are not meant to be covered during this testbed (the servers under test will include both cases)

The tests are made available in a variety of ways:

- Source code in the associated Github repository [https://github.com/opengeospatial/ets-wfs30], recommended for those interested in running the tests from an Integrated Development Environment (IDE), or to contribute changes to them.
- Central Maven repository [https://search.maven.org/], for pre-build artifacts ready for use. Maven is a software project management and comprehension tool, commonly used for managing software project dependencies during compilation, test and packaging phases.
- Docker image [https://github.com/opengeospatial/teamengine-docker], for immediate local usage, start and run. Docker is an open source software platform to create, deploy and manage virtualized application containers on a common operating system.
- Eventually online in the beta test environment [http://cite.opengeospatial.org/te2/].

Any issue related to the tests development can be reported in the GitHub issue tracker [https://github.com/opengeospatial/ets-wfs30/issues].

5.1. Requirement classes tested

The WFS 3.0 CITE tests are currently testing a single requirement class, the "core" one.

In order for core to be testable, the OpenAPI and GeoJSON conformance classes also need to be implemented. They are not fully tested themselves, rather they are leveraged to test the core one (by being used in the tests, they get partially covered as well).

5.2. Test approach for a resource-oriented architecture

All Hypertext Transfer Protocol (HTTP) requests sent by the test suite follow the REST concept. Nevertheless, any HTTP client can be used to send the requests as REST, using the HTTP protocol.

The test suites requires the `/api` resource as an OpenAPI document. Thus, a parser that is capable of processing those responses is needed.

As the test suite is written in TestNG, any Java HTTP client (for HTTP requests) and any Java parser (for HTTP responses) can be used for the executable tests.

The following libraries/frameworks can be used to make creation of tests easier and readability better:

- Jersey (HTTP client)
- Spring RestTemplate (HTTP client)
- REST-assured (HTTP client)
- KaiZen-OpenAPI-Parser (OpenAPI parser)

All options regarding the Java HTTP client were evaluated closely. Finally, the decision fell on REST-assured, as this framework also offers other useful functionalities (besides a Java HTTP client) which are not provided by Jersey and Spring RestTemplate.

The main functionality added by REST-assured is the capability of validating REST services with Java. As this is the main focus of a test suite, it was clear that REST-assured was the framework best fitting the needs at hand.

Regarding the HTTP responses a parser is needed that provides OpenAPI documents parsing support. As KaiZen-OpenAPI-Parser is currently the only OpenAPI parser, it was decided to use this library.

5.3. Strategy to implement test suite

The implementation of the test suite is kept as close as possible to the Abstract Test Suite defined in the specification.

First, the Core conformance class is implemented by creating one or more tests (Java methods) per abstract test defined in specification. The description of each implemented test references the corresponding test of the Abstract Test Suite and the requirement which is verified by that executable test.

The optional conformance classes, HTML, GeoJSON, GML Simple Feature Level 0 (SF-0) and GML SF-2 are not part of the test suite. The reason of this decision is that these conformance classes are not covered by the Abstract Test Suite of the specification.

Instead, the implementation work concentrated on creating a stable and robust executable test suite of the Core conformance class.

Also, the test suites have two important preconditions: the JSON encoding and OpenAPI interface must be supported.

5.4. Challenges during implementation of test suite

This section covers noteworthy issues that needed to be handled during the test development.

5.4.1. New API paradigm

One of the major challenges encountered was that there are currently no TestNG test suites validating a resource-oriented architecture, yet. Thus, new tools and techniques had to be identified to handle this approach. This challenge is covered by the chapter "Test approach for a resource oriented-architecture".

5.4.2. Capabilities document replacement

Preview test implementations had a single obvious entry point, the capabilities document. In WFS3 the contents of a typical capabilities document have been split into four separate resources, landing page, conformance declaration, OpenAPI description, and collections resource. While the landing page and OpenAPI description seem equally interesting as an entry point to describe the service as a whole, the tests currently require the specification of the landing page.

5.4.3. Multiple valid ways to expose the API

Another challenge was identified during actual verification of the test suite with service implementations. There are two approaches for how to implement WFS 3.0 API. One is the explicit approach which offers two API resources per collection. The second approach is a compact implementation of the API meaning that two API resources cover all collections via parametrization. The first release of the test suite just covers the explicit approach. Handling of the compact approach was added for the second release. This is a good example of how the test suite was made more robust and flexible as a direct result of actual verification with service implementations.

It is also to be noted that filtering support by attribute equality can only be exposed via the explicit collection API approach, meaning servers using the compact approach will have feature filtering tested only based on bounding box (bbox) and time, but not on attribute equality.

5.4.4. Building a suite with no mandated output format

The tests cannot be fully performed without the server also supporting the optional GeoJSON conformance class. This means that a compliant server implementing, for example, only HTML and GML would not be supported by the test suite.

This can also be seen as a challenge for clients trying to connect to a server, there is no guarantee that a compliant client and a compliant server will actually be able to exchange data, the likeliness of successful communication will grow the more optional output conformance classes are implement on both sides.

5.4.5. Building a test suite against random data sets and schemaless output

The WFS3 service is 'schemaless', meaning there is no resource describing the schema of a collection. A schema for JSON outputs might eventually be included in the OpenAPI document, but it is not mandatory and can only be supported while using the explicit collection API style.

At the same time, the test suite does not mandate specific datasets to be published, the server can have few or many collections, small or large ones, structured or unstructured in contents. This limits the ability to run specific tests on the data.

5.4.6. Running tests within a reasonable amount of time, and test repeatability

The lack of a fixed test dataset implies the test suite has to work against whatever the server proposes. An exhaustive test of all collections and all data contained is not always practical, as it might require loading very large amounts of data from the server, or performing millions of requests paging though the datasets or covering all requests.

Picking random collections to run the tests on, or random sizes for collection paging, while making the test more amenable to find issues in the server, would make the test not repeatable, which would be an obvious problem certification wise.

The tests have been thus modified to work against a fixed (configurable) set of collections, and have no elements of randomness in the run.

5.4.7. Authentication

The test servers in this round of development have been exposed both un-authenticated, and protected via OpenID connect authentication. The tests, at the time of writing, support the open and non-authenticated version. Future developments might add support for a variety of authentication mechanisms.

Chapter 6. WFS 3.0 Reference Implementations

The WFS 3.0 tests have been verified against three separate implementations (in alphabetical order):

- CubeServer, by CubeWerx
- GeoServer WFS 3, by GeoSolutions
- ld-proxy, by Interactive Instrument

The following text provides an overview of each components, while a detailed description can be found in chapter 7 of OGC 18-045, "Secure Resource Oriented Geospatial Data Handling Engineering Report [https://github.com/opengeospatial/D021-Secure_Resource_Oriented_Geospatial_Data_Handling_Engineering_Report].

6.1. Cubeserver by CubeWerx

The server provides data from direct data sources and implements the following requirement classes:

- Core [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_core]
- HTML [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_html]
- GeoJSON [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_geojson]
- GML SF0 - SF2 [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_gmlsf0]

The server also offers a number of experimental extensions that include:

- CRS extension
- Geometry simplification extension
- OpenSearch query extension
- Property selection extension
- Theme extension
- Map extension
- Tile extension
- Asynchronous request extension
- Advanced query extension
- Transaction extension

The CubeWerx D113 deliverables consist of servers deployed at the following endpoints:

- Unsecured, development, servers:
  - USGS Framework Data [http://www.pvretano.com/cubewerx/cubeserv/default/wfs/3.0.0/framework/api]

  - VMAP Level 0 Foundation Data [http://www.pvretano.com/cubewerx/cubeserv/default/wfs/3.0.0/foundation/api]
  - Daraa Refugee Camp [http://www.pvretano.com/cubewerx/cubeserv/default/wfs/3.0.0/daraa/api]
  - Zaztari Refugee Camp [http://www.pvretano.com/cubewerx/cubeserv/default/wfs/3.0.0/zaatari/api]
  - US Building Footprints [http://www.pvretano.com/cubewerx/cubeserv/default/wfs/3.0.0/buildings/api]
- Secured, stable, servers:
  - VMAP Level 0 Foundation Data [https://tb14.cubewerx.com/cubewerx/cubeserv/default/wfs/3.0.0/Foundation/api]
  - Daraa Refugee Camp [https://tb14.cubewerx.com/cubewerx/cubeserv/default/wfs/3.0.0/Daraa/api]
  - Zaatari Refugee Camp [https://tb14.cubewerx.com/cubewerx/cubeserv/default/wfs/3.0.0/Zaatari/api]

6.2. GeoServer WFS 3 by GeoSolutions

This deliverable consists of a single component, a WFS 3.0 server implementation developed by GeoSolutions [https://www.geo-solutions.it/] as a community plugin to the GeoServer [http://geoserver.org/] open source software.

The source code of the plugin has been made available to the GeoServer community, from the start, through the public GitHub repository [https://github.com/geoserver/geoserver/tree/master/src/community/wfs3], while its binary form can be found among the nightly builds of the development series [https://build.geoserver.org/geoserver/master/community-latest/geoserver-2.15-SNAPSHOT-wfs3-plugin.zip].

The plugin can be installed by simply un-packing the contents of the zip file in the GeoServer `WEB-INF/lib` directory and re-starting GeoServer.

This component is published as four separate endpoints, exposing the OSM data for Daraa and Zaatari areas, both as unsecured and secured services:

- Daraa Refugee Camp, unsecured [http://cloudsdi.geo-solutions.it/geoserver/daraa/wfs3]
- Daraa Refugee Camp, secured [https://cloudsdi.geo-solutions.it/geoserver/daraas/wfs3]
- Zaatari Refugee Camp, unsecured [http://cloudsdi.geo-solutions.it/geoserver/zaatari/wfs3]
- ZaatariRefugee Camp, secured [https://cloudsdi.geo-solutions.it/geoserver/zaataris/wfs3]

The server provides data from direct data sources and implements the following requirement classes:

- Core [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_core]
- HTML [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_html]
- GeoJSON [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_geojson]
- GML SF0 [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_gmlsf0]

6.3. ld-proxy, by Interactive Instrument

The deliverable consists of two components, a WFS 2.0 component and a WFS 3.0 component that is a facade providing access to the WFS 2.0 services.

The ldproxy tool has been developed as a proxy for WFS endpoints to explore how WFS could be better aligned with today's Web and the expectations of developers and users. See, for example, the report "Spatial Data on the Web using the current SDI" from the Geonovum tested in 2016. The results have been input to the development of the W3C/OGC Spatial Data on the Web Best Practice and the OGC Change Request for WFS 3.0.

The WFS 3.0 landing pages are:

- Daraa Refugee Camp [https://services.interactive-instruments.de/t14/wfs3/daraa]
- Zaatari Refugee Camp [https://services.interactive-instruments.de/t14/wfs3/zaatari]

The server serves data from direct data sources and implements the following requirement classes:

- Core [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_core]
- HTML [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_html]
- GeoJSON [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_geojson]
- GML SF0 / SF2 [https://rawgit.com/opengeospatial/WFS_FES/master/docs/17-069.html#rc_gmlsf0]

The server also offers the following experimental extensions:

- CRS extension
- Geometry simplification extension
- Random paging based on offset and limit parameters
- ResultType parameter with "hits" support

Chapter 7. WFS 3.0 Implementation challenges

In terms of implementation challenges for the server side, the protocol appeared to be straightforward to implement and is widely advertised as being implementable in a few days.

However, in order to pass the compliance tests there are a number of little details to look out for, specifically they are: HTTP headers, proper links with the right ref support, and the eventuality of supporting multiple API styles (compact and explicit) which can grow the overall implementation effort considerably.

While the XML and JSON outputs are well documented in the specification itself, the HTML output is open ended, which allows a lot of flexibility, and might be implemented either in a very basic fashion, but also in a very rich way, with graphical previews inside the page, dynamic filtering and sorting, and the like. This in particular could pose challenges on services that are secured, as the dynamic mapping clients in the response pages need to perform secured requests back to the server as well.

Finally, while the JSON based outputs are covered by the test suite, nothing can currently be said about the compliance of the HTML and XML conformance classes, which are currently not being verified.

Chapter 8. DGIWG Catalog Tests

lat/lon provided an implementation of the DGIWG CAT 2.0.2 test suite that runs on the OGC compliance portal [http://cite.opengeospatial.org/teamengine/]. As all recent test suites, the test has the following characteristics:

- Executable via TEAM Engine
- Written in TestNG
- Based on the WFS 3.0 Abstract Test Suite (ATS), with test naming matching the test suite

The test suite is a classic extension test, assuming a compliant CSW 2.0.2 behavior and testing all the extra behavior required in the DGIWG profile. There are 5 tests covering all of the requirements, with the exception of requirement 20, not mentioned in the ATS. The initial delivery tested the DGIWG_Basic_CSW conformance class, while the final delivery covered the DGIWG_CSWT conformance class.

The tests are made available in a variety of ways:

- Source code in the associated Github repository [https://github.com/opengeospatial/ets-cat20-dgiwg10], recommended for those interested in running the tests from an IDE, or contribute changes to them.
- Central maven repository [https://search.maven.org/], for pre-build artifacts ready for use.
- Docker image [https://github.com/opengeospatial/teamengine-docker], for immediate local usage, start and run.
- Eventually online in the beta test environment [http://cite.opengeospatial.org/te2/]

Any issue related to the development of the tests can be reported in the GitHub issue tracker [https://github.com/opengeospatial/ets-cat20-dgiwg10/issues].

8.1. Conformance classes tested

The tests verify compliance for the following conformance classes:

- DGIWG_Basic_CSW
- DGIWG_CSWT

8.2. Strategy to implement test suite

The implementation of the test suite is kept as close as possible to the ATS defined in specification.

First, the DGIWG_Basic_CSW conformance class is implemented by creating several tests (Java methods) per abstract test defined in the specification (this is necessary as there are just three abstract tests being too complex to become only three executable test). The description of each implemented test references the corresponding test of the ATS and the requirement which is verified by that executable test.

After having completed the implementation of the DGIWG_Basic_CSW conformance class, the DGIWG_CSWT conformance class is implemented. Again, there are just two abstract tests which are too complex for only two executable tests. Thus, there are more than two executable tests. The description is populated as described above.

The final release covers all conformance classes defined by specification, DGIWG_Basic_CSW and DGIWG_CSWT.

8.3. Challenges during implementation of test suite

The major challenge found during implementation is related to inaccuracies found in the ATS itself. For example, references are missing and there are contradictory requirements. All inaccuracies have been documented in GitHub issues.

Chapter 9. DGIWG Catalog Implementation

9.1. GeoNetwork Opensource

9.1.1. Introduction

GeoNetwork Opensource [https://geonetwork-opensource.org/] is a catalog application to manage spatially referenced resources. It provides powerful metadata editing and search functions as well as an interactive web map viewer. It is currently used in numerous Spatial Data Infrastructure (SDI) initiatives across the world.

GeoNetwork has been developed to connect spatial information communities and their data using a modern architecture, which is at the same time powerful and low cost, based on the principles of Free and Open Source Software (FOSS) and International and Open Standards for services and protocols (a.o. from ISO/TC211 and OGC).

GeoNetwork implements, among others, the following protocols:

- OGC CSW
- OAI-PMH
- OpenSearch
- Z39.50

and also provides its own API to interact with other systems and a DCAT/RDF search service.

9.1.2. GeoNetwork & CSW

GeoNetwork OpenSource has been one of the first reference implementations [https://geonetwork-opensource.org/manuals/2.10.4/eng/users/quickstartguide/geographic_management/index.html#background-and-evolution] for the ISO19139:2007 profile of CSW 2.0.2.

GeoNetwork implements a full CSW-T interface; this means that it also implements the optional `harvest` and `transaction` operations.

9.1.3. GeoNetwork and DGIWG Metadata

GeoNetwork can handle more metadata profiles than the ones "natively" provided (i.e. ISO19139/2007, ISO19119, FGDC, Dublin core) by implementing a "schema plugin".

A schema plugin allows GeoNetwork to:

- recognize a new profile,
- provide specific editor form(s) to facilitate the editing (offering, for instance, specific forms for mandatory fields in the given profile)
- index and search for specific parts of the metadata documents
- present the metadata documents of the given profile in an appropriate way

- validate the documents according to the profile specific constraints

A guide for implementing new schema plugins can be found at this link [https://geonetwork-opensource.org/manuals/3.4.x/is/customizing-application/implementing-a-schema-plugin.html].

9.1.4. Code availability

GeoNetwork opensource is available at this github repository [https://github.com/geonetwork/core-geonetwork].

The "official" schema plugins are available at this github repository [https://github.com/metadata101]. This is a hub organization for all the schema plugins available for GeoNetwork, so the DGIWG plugin has been stored here [https://github.com/metadata101/iso19139.dgiwg].

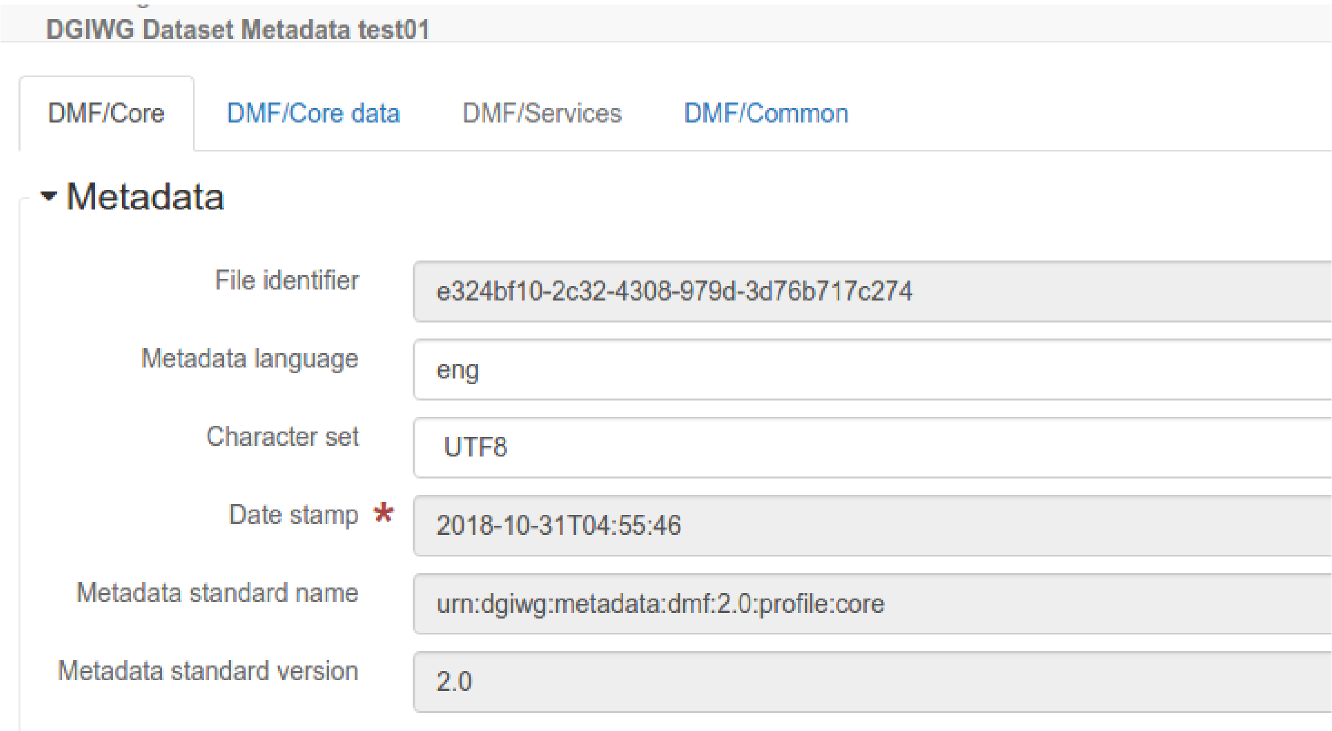
9.2. Implementation

The schema plugin has been implemented following the step described in the implementation guide referenced above; since the guide is a bit old, also some adjustments have been done in order to comply with the requirements needed in the latest versions.

The implementation followed the requirements described in this document. [https://portal.dgiwg.org/files/?artifact_id=67565&format=doc] The DGIWG 2.0.0 schema files needed for GeoNetwork are available at this link [https://portal.dgiwg.org/files/?artifact_id=68100]. They have been slightly edited in order to simplify the directory hierarchy.

While the Testbed only required schema compliance, a GUI for editing DGIWG metadata has been implemented in order to ease creation of DGIWG compliant metadata. Some examples follow.

- The metadata editor is split in different tabs, each one containing data belonging to different DMF metadata classes:

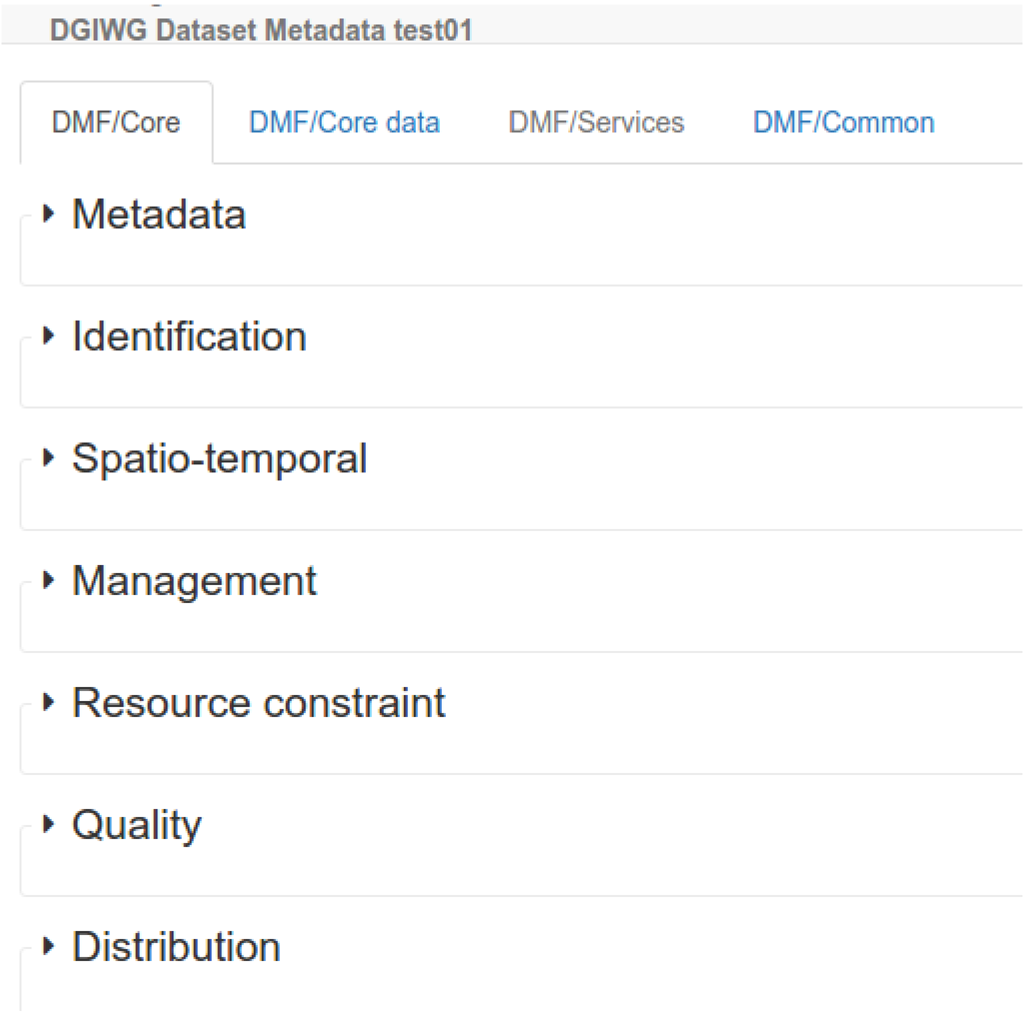


DGIWG Dataset Metadata test01

Metadata

File identifier	e324bf10-2c32-4308-979d-3d76b717c274
Metadata language	eng
Character set	UTF8
Date stamp *	2018-10-31T04:55:46
Metadata standard name	urn:dgiwg:metadata:dmf:2.0:profile:core
Metadata standard version	2.0

- Metadata fields and sections are ordered in the same sequence they appear in the DMF specifications:



DGIWG Dataset Metadata test01

DMF/Core | DMF/Core data | DMF/Services | DMF/Common

▸ Metadata

▸ Identification

▸ Spatio-temporal

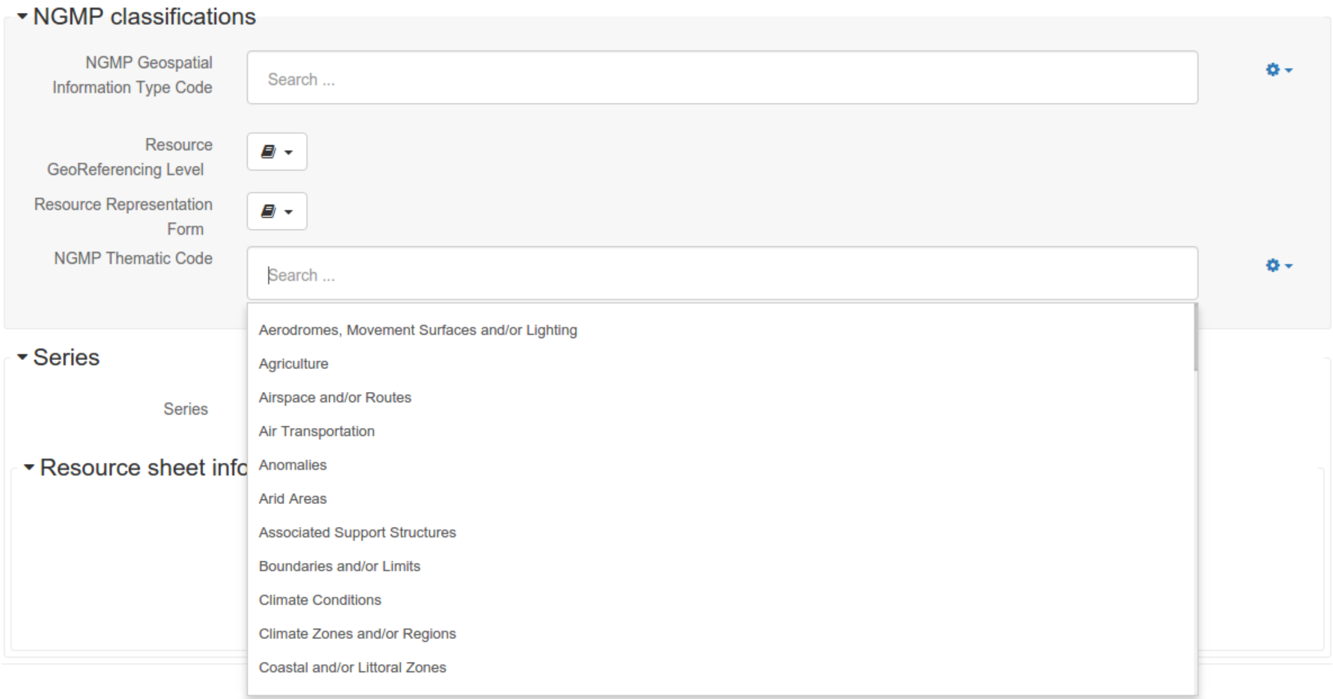
▸ Management

▸ Resource constraint

▸ Quality

▸ Distribution

- • NATO Geospatial Metadata Profile (NGMP) classifications, that should be expressed as simple keywords, have their own dropdown selection menus



▾ NGMP classifications

NGMP Geospatial Information Type Code: Search ...

Resource GeoReferencing Level

Resource Representation Form

NGMP Thematic Code: Search ...

Aerodromes, Movement Surfaces and/or Lighting
Agriculture
Airspace and/or Routes
Air Transportation
Anomalies
Arid Areas
Associated Support Structures
Boundaries and/or Limits
Climate Conditions
Climate Zones and/or Regions
Coastal and/or Littoral Zones

▾ Series

Series

▾ Resource sheet info

9.3. Configuration

In order to satisfy all of the requirements of the "Defence Profile of OGC Catalogue Service for the Web 2.0" [https://portal.dgiwg.org/files/?artifact_id=68270&format=pdf] GeoNetwork needed some configuration to be done outside the schema plugin. All of the related information is reported in the documentation files of the DGIWG metadata plugin [https://github.com/metadata101/iso19139.dgiwg].

9.4. Implementation challenges

9.4.1. Documentation

During the implementation some issues were found in the documentation.

The errors found in "Defence Profile of OGC Catalogue Service for the Web 2.0" [https://portal.dgiwg.org/files/?artifact_id=68270&format=pdf] have been filed as issues in the ets-cat20-dgiwg10 [https://github.com/opengeospatial/ets-cat20-dgiwg10] repository:

- "Errors in DGIWG125 v1.0.1" [https://github.com/opengeospatial/ets-cat20-dgiwg10/issues/2]: various errors in the specifications.
- "Req19 overrides req4" [https://github.com/opengeospatial/ets-cat20-dgiwg10/issues/24]: in the comments you can find a reference to a possible typo in a text required to be returned by the service.

In DGIWG Metadata Foundation 2.0 [https://portal.dgiwg.org/files/?artifact_id=67565&format=doc] the only error found is about trying to redefine the **topic category enumeration** (ISO19115 mandates that enumerations cannot be extended).

9.4.2. Framework

A bug in GeoNetwork [https://github.com/geonetwork/core-geonetwork/issues/3104] prevented a correct response to the `Insert operation. It has been fixed both on the stable branch [https://github.com/geonetwork/core-geonetwork/pull/3108] used as reference for this testbed, and in the master branch [https://github.com/geonetwork/core-geonetwork/pull/3107]. The fix has already been merged into both branches.

9.4.3. Authentication

The CSW specification does not require authentication for the Transaction operation. Since such an operation will modify the content of the catalog, GeoNetwork requires valid credentials to allow the operation. The Test Suite has been enhanced [https://github.com/opengeospatial/ets-cat20-dgiwg10/pull/42] in order to use HTTP Basic Authentication; this means that in order to complete all of the tests, credentials for a valid user need to be supplied.

Appendix A: Revision History

Table 1. Revision History

Date	Editor	Release	Primary clauses modified	Descriptions
May 14, 2018	Andrea Aime		all	initial version
July 2, 2018	Andrea Aime and Emanuele Tajariol		all	Added contents for WFS and CAT server implementations
September 17, 2018	Andrea Aime and Dirk Stenger		all	Added contents for ETS test suites
September 17, 2018	Andrea Aime and Dirk Stenger		all	Added contents for ETS test suites
November 12, 2018	Andrea Aime	r1	all	Added more contents, first official release in pending documents
December 3, 2018	Andrea Aime	r2	all	Fixes based on feedback
December 11, 2018	Andrea Aime	r3	all	Minor fixes based on feedback