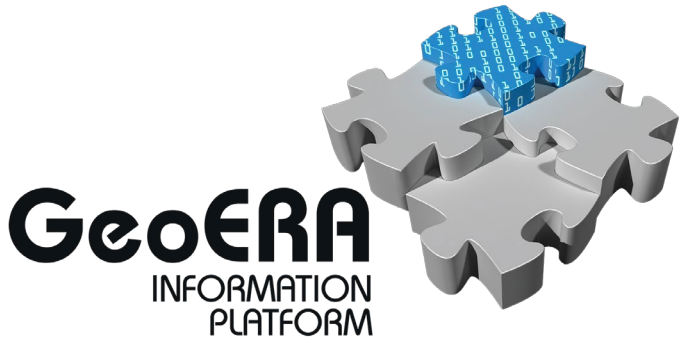

Establishing the European Geological Surveys Research

Area to deliver a Geological Service for Europe

Authors and affiliation:

BRGM	**M. Beaufils**
	S. Grellet
ISPRA	**C. Cipolloni**
	M. P. Congi
	E. Giulianelli
BGS	**M. Sen**
	E. Lewis
	J. Passmore

Deliverable 3.3

E-mail of lead author:
carlo.cipolloni@isprambiente.it

Version: 28-01-2020

Deliverable Data		
Deliverable number	D3.3	
Dissemination level	Public	
Deliverable name	Standards validation procedures	
Work package	WP3, Standard and Interoperability Issues	
Lead WP/Deliverable beneficiary	Carlo Cipolloni (ISPRA)	
Deliverable status		
Submitted (Author(s))	27/01/2020	Carlo Cipolloni (ISPRA)
Verified (WP leader)	30/01/2020	Carlo Cipolloni (ISPRA)
Approved (Coordinator)	31/01/2020	Jørgen Tulstrup (GEUS)

This report is part of a project that has received funding by the European Union's Horizon 2020 research and innovation programme under grant agreement number 731166.

GENERAL INTRODUCTION

The GeoERA Information Platform project (GIP-P) is established to support the 14 GeoERA geoscientific projects (GSPs) in organising and disseminating the geoinformation generated within their frameworks. The GIP-P is entitled to safeguard the results of the research performed by the various GeoERA projects in terms of geospatial data, reports and structured databases. This will be done by extending the current European Geological Data Infrastructure (EGDI), so that it can display and share the results from the various GeoERA projects with citizens, researchers and stakeholders.

The different GeoERA projects deal with multiple aspects of geosciences in the fields of groundwater, raw materials, and geo-energy. These projects will thus be generating a variety of products, which will require specific functionalities to be developed to store, show and share them properly. It is thus important that the GIP-P has a good understanding of the products that each project will generate, and the functionalities required to show them properly. This is assured by Work Package 2 (WP2), which coordinates the interactions between the various GeoERA projects and the GIP-P.

EXECUTIVE REPORT SUMMARY

Despite the diversity and multidisciplinary nature of the various GeoERA projects, all of them address geological topics. Hence, different projects might produce similar or complementary geoinformation. It is thus likely that some overlap occurs between the geoinformation produced by the different projects. That is especially true for projects delivering data from the same or neighbouring areas.

In order to support all GeoERA projects in data harmonization and service deployment this report provides some general elements to be implemented in the Central harvest system, and in the services provided by EGDI from the harvested data. The present report aims to explain existing procedures to validate network services and data models that can be implemented in the platform or used by data providers in different GeoERA projects.

TABLE OF CONTENTS

GENERAL INTRODUCTION 3
EXECUTIVE REPORT SUMMARY 3
DEFINITIONS 5
ABBREVIATIONS 5
INTRODUCTION 6
Objectives and contents of D3.3: 6
Sources of needs expression from the projects and versions considered for D3.3: 6
Relevant data models identified and considered for D3.3: 6
VALIDATION PROCEDURE 7
Network Service Validation 9
Data Validation Process 11
DATA VALIDATION TOOLS 12
Validate the transformed data inside Hale studio 13
Schematron validation 14
Abstract Test Suite validation 16
DATASET VALIDATION RECOMMENDATIONS 18

DEFINITIONS

GeoERA: Establishing the European Geological Surveys Research Area to deliver a Geological Service for Europe

ABBREVIATIONS

API: Application Programming Interface

EGDI: European Geological Data Infrastructure

ETS: Executable Test Suite

GIP-P: GeoERA Information Platform Project

GSPs: Geoscientific projects within GeoERA

SD-TGs: Data Specification technical guidelines

WP: work package

.

INTRODUCTION

Objectives and contents of D3.3:

The main objective of this report is to help GSPs validate data harmonization and, in the case that some web services are set-up to deploy data, validate these services in a conformant way. In fact D3.1 identified existing standards and data models that address the topics of the thematic projects, based on the UML models defined and, on the vocabularies, and ontologies identified and archived in WP4, the user can perform some validation tests in order to guarantee a minimum quality level.

Some inputs of this report need to be considered in the Architecture and in the implementation activity, in order to made available in the central system some of those tools, as well as are presented some tools and reccomadations that we need to explore in the WP8 to support other projects in validation procedure.

D3.2.2 document provides technical requirements and guidance to expose the data identified in D2.2.1 with technologies identified by D3.1, applying the expected conceptual mapping described in D3.2.1 and it has also provided a set of recommendations for the other GSPs that need to validate.

In this report is presented the validation procedure for web services and datasets, also suggesting some tools. For the dataset in the data validation two main concepts could be used: the schematron rule and the use of Abstract Test Suites (ATS) that are a set of technical rules to implement in a real case the data model, where it must take the semantic content into consideration.

D3.2.1 identified possible matches and conceptual mapping between the projects' data and those existing standards.

Sources of needs expression from the projects and versions considered for D3.3:

Name of the document	Date / version
D2.1.1, Potential synergies and overlaps between the projects.	Version: 30/06/2019
D3.1, Data models, Standard Guidelines and Toolkits.	Version: 02/05/2019
D3.2.1, Gaps analysis and path extension	Version: 30/07/2019

Relevant data models identified and considered for D3.3:

Name of the document	Date / version
OGC GeoSciML	4.1 Rev 16-008
OGC GWML2	2.2 Rev 16-032r2
EarthResourceML	2.0 October 2013
INSPIRE AC (Atmospheric Conditions)	Revision 4618 This version corresponds to the content of the Implementing Rules
INSPIRE AF (Agricultural and aquaculture facilities)	
INSPIRE AM (Area Management)	

INSPIRE EF (Environmental Monitoring Facility)	(EU) No 1089/2010, No 102/2011, No 1253/2013 and the latest publicly available version of the data specifications of Annex I, II+III.
INSPIRE EL (Elevation)	
INSPIRE ER (Earth Resources)	
INSPIRE GE (Geology)	
INSPIRE LU (Land Use)	
INSPIRE MR (Mineral Resources)	
INSPIRE OF (Ocean Features)	
INSPIRE SO (Soil)	
INSPIRE NZ (Natural Risk Zones)	
EPOS BoreholeView	1.0.0
EPOS ModelView	1.0.0
ISO 19156 : Observations & Measurements	2.0 Rev 10-025r1 (OGC)
WaterML 2 - Part 1 / Timeseries	2.0.1 Rev 10-126r4
ISO 19115 / ISO 19139	
OGC Coverage Implementation Schema with Corrigendum (09-146r8)	Version 1.1.1 Published 2019-10-28

VALIDATION PROCEDURE

A validation procedure is a process for checking the accuracy and quality of source data or a service before using, importing or otherwise processing it. There are many different types of validation that can be performed, the main two are: formal schema validation rather compliancy with standard UML model defined for a service or a dataset and Data Specification conformance classes validation that evaluate also the interoperability between data, metadata and service.

Data validation can help data cleansing and help data interoperability and integration in a platform. Taking into consideration the validation processes, already in place in relation to the standards analyzed in D.3.2.1 and D.3.2.2 at international level, there are two main validation board for services and datasets: Open Geospatial Consortium (OGC) and INSPIRE. The first is more related to set-up a list of available standards (table 1 below) that can be tested by software online tool (OGC Team Engine). The second is developed to support the SDI implementation process in Europe, following the INSPIRE Technical guidelines.

Table 1 - Standard services and package that can be tested by OGC TEAM Engine

Specification	Version
Catalogue Service - Web (CSW)	2.0.2
Catalogue Service - Web (CSW)	3.0.0
GeoPackage	1.0

GeoPackage	1.0
Geography Markup Language (GML)	3.2.1
Sensor Observation Service (SOS)	1.0.0
Sensor Observation Service (SOS)	2.0
Sensor Planning Service (SPS)	1.0
Sensor Planning Service (SPS)	2.0
SensorThings API	1.0
Simple Feature Access - SQL (SFS)	1.1
Simple Feature Access - SQL (SFS)	1.2.1
Web Coverage Service (WCS)	1.0.0
Web Coverage Service (WCS)	2.0.1
Web Feature Service (WFS)	1.0.0
Web Feature Service (WFS)	1.1.0
Web Feature Service (WFS)	2.0
Web Map Service (WMS)	1.3.0
Web Map Service (WMS) - Client	1.3.0
Web Map Tile Service (WMTS)	1.0.0

The INSPIRE validation procedure has been developed by the European Commission in order to support the Public Administrations that should provide data and services in the countries.

The network services validation exploits the basic engine offered by the OGC Team Engine, but at the same time extends it with respect to the INSPIRE technical guidelines by introducing the concept of extension of capabilities and interlinking between dataset and service.

The validation of the datasets versus the reference data models deserves a separate discussion, in fact this procedure can take place mechanically with respect to general rules summarized in Abstract Test Suite (ATS) or in the schematron, but sometimes it, also requires, a valued and reasoned evaluation on the semantic concepts used towards vocabularies fixed.

In the following sections are described the different validation procedures and systems that can be used or implemented in the GeoERA.

Network Service Validation

The validation process of web services based on OGC and/or INSPIRE standards can be carried out: by installing on the centralized system of the GIP, on remote tools residing on cloud systems managed by standard bodies (OGC and INSPIRE), or using the systems web API in the cloud or by installing the validation software on server in-premise.

The OGC TEAM Engine (Test, Evaluation, And Measurement Engine) is a Java-based application for testing web services and other information resources, the other way around the INSPIRE Validation service is based on the Testing framework for spatial data and services (ETF) and it is an open source testing framework for validating spatial data, metadata and web services following the main technical guidelines of INSPIRE Directive.
In the case of TEAM Engine runs test suites developed using the popular TestNG framework, OGC Compliance Test Language (CTL) scripts and possibly other JVM compatible languages. It is light and easy to run from the command line or as a web application. As for ETFs, this is based on the ISO 19105 framework and on the OGC specification model that underlies the standards used in the Spatial Data Infrastructures (SDIs). The tests are organized in Executable Test Suites, which can be developed and run using different tools to properly support all types of resources in an SDI.
In the following part are presented the resources, where to find online cloud validation systems of the OGC and INSPIRE services, these may require the creation of a specific account, and the resources where to find the source code repository if you want to install the system locally.

The source code GitHub repositories are available:

Table 2 – source code repository list.

Web Service type	GitHub
TEAM Engine and the tests are available at OGC GitHub	https://github.com/opengeospatial/teamengine
INSPIRE ETF Validator is available at INSPIRE GitHub	https://github.com/etf-validator/etf-webapp
INSPIRE ETF Web Application software download	https://github.com/etf-validator/etf-webapp/releases/latest

To execute directly the test using cloud resources you can use for TEAM Engine the following table for the different Testbed:

Table 3 – OGC Test suite availability

Web Service type	Test Suite Revision
Catalogue Service - Web (CSW)	http://cite.opengeospatial.org/teamengine/about/cat30/3.0.0/site

GeoPackage	http://cite.opengeospatial.org/teamengine/about/gpkg10/1.0/site
GeoPackage (1.2)	http://cite.opengeospatial.org/teamengine/about/gpkg12/1.2/site/
Geography Markup Language (GML)	http://cite.opengeospatial.org/teamengine/about/gml32/3.2.1/site
Sensor Observation Service (SOS)	http://cite.opengeospatial.org/teamengine/about/sos/1.0.0/site
SensorThings API	http://cite.opengeospatial.org/teamengine/about/sta10/1.0/site
Web Coverage Service (WCS)	http://cite.opengeospatial.org/teamengine/about/wcs/2.0.1/site/
Web Feature Service (WFS)	http://cite.opengeospatial.org/teamengine/about/wfs/1.1.0/site
Web Map Service (WMS)	http://cite.opengeospatial.org/teamengine/about/wms/1.3.0/site
Web Map Service (WMS) - Client	http://cite.opengeospatial.org/teamengine/about/wms-client/1.3.0/site
Web Map Tile Service (WMTS)	http://cite.opengeospatial.org/teamengine/about/wmts/1.0.0/site

At the same time to execute directly the test using cloud resources for public INSPIRE web services you can use the following online resource, where is possible to select one or more web service to test:

Table 4 – INSPIRE Reference Validator

Web Service type	INSPIRE Validation
ETF Validator	http://inspire.ec.europa.eu/validator/

Another option to be able to perform the validation test remotely is to use the web API, exploiting the interactive Web interface for the use of the Web API version 2 of the test framework ETF.
The ETF Web API is published and documented in detail as a Swagger definition on SwaggerHub (https://app.swaggerhub.com/api/etf/etf/). The ETF web application instance also provides the Swagger web interface which is available at http://your-etf-server/path/swagger-ui.html.
In order to support the API implementation in the central system must be considered that the response model is based on a XML schema and the interactive XML schema documentation is available at http://resources.etf-validator.net/schema/v2/doc/service.html.

More information about how to implement the API is available at: http://docs.etf-validator.net/v2.0/Developer_manuals/WEB-API.html, moreover the documentation on how to develop
Executable Test Suites for ETFs using SoapUI, BaseX and TEAM Engine test engines is available at the following address:
http://docs.etf-validator.net/v2.0/Developer_manuals/Developing_Executable_Test_Suites.html
The key concepts introduced by these two standards are Compliance Classes and Executable Test Suites, where a compliance class is a set of requirements defined in a specification. To pass the compliance class, a resource that requires compliance with that class must meet all requirements. An executable test suite is a collection of tests that validates a resource against all the requirements of the associated compliance class.

Data Validation Process

The validation procedures of the datasets with respect to the services require the correct identification of the reference data model, therefore before undertaking a datasets validation activity, it is essential to understand what you are validating in your dataset.

Understanding what the validation process will evaluate and what you can declare on your dataset once the validation report is obtained, it is determined to obtain quality data. Most validation tools perform only a so-called formal "schema" validation. This means that the dataset is tested against those requirements that can be expressed through the XML schema grammar (i.e. the xsd requirements) by evaluating whether all the elements of the dataset adhere to the structure defined in the schema of the addressed application.

The first level of dataset validation is represented by formal Schema validation to deepen the level of validation, it is necessary to perform tests on formal exchange language such as GML 3.2.1 validation. In this validation process the data set is validated against the requirements of ISO 19136: 2007, which defines the GML standard. This means that not only the requirement of the gml.xsd schema is taken into account, but, for example, the constraints relating to the geometry of the data set and the coordinate reference system are also considered.
To have a complete validation process of a dataset, it is necessary to define validation rules of the conformity classes of the same. An accurate dataset validation process is independent of the fact that a schematron file is addressed directly in the data set or is selected from a list of Abstract Test Suites, in fact the content of the data set is validated against the constraints specified therein.

A schematic correct process of validation that takes in account all the levels can be represented in figure 1.

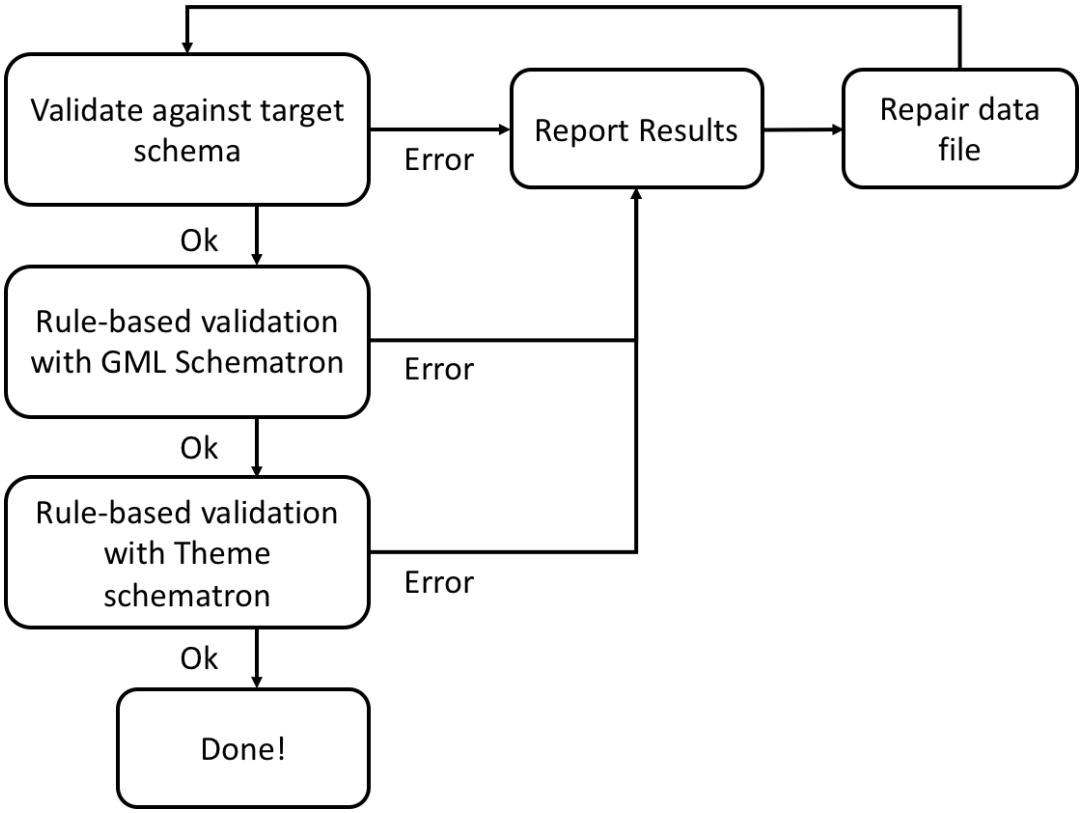

Figure 1 – Dataset validation procedure workflow.

DATA VALIDATION TOOLS

Following the general rules defined in the previous section, here will be presented a set of tools useful for validating datasets at different levels.
The first step can be tested validating syntax against formal application schema, to do that can be used different XML/GML editor that allow to set-up reference schema, like oXygen XML Editor, Altova XMLSpy or Hale Studio.

In this software, once the project has been set up and the reference xsd file has been indicated or, if possible, the URL where the reference scheme application resides, we can proceed with the validation of the Schema, when XML/GML has been processed the software responds with a positive check value.

Validate the transformed data inside Hale studio

Within *hale studio* validation of instance file can be done on the exported transformation result or on the transformed instances currently available in the application, but if you target a specific schema it is vital for your transformation result to not only follow the schema's structure, but to also meet the other constraints defined by the schema, like mandatory properties or restrictions on property values. In addition to schema-based checks, *hale studio* supports validation of the exported transformation result with Schematron. To perform Schematron validation, you can add the Schematron validator to the list of validators and configure it when exporting the transformed instances. Alternatively, you can load Schematron schemas permanently into the project. Validation will then be performed by the Project validator which is enabled in the export wizard by default.

If a mapping exists and source data is loaded, each mapping change will trigger the live transformation (if activated). When the transformed data changes, schema-based validation is started automatically to validate the instances. Live validation can be enabled and disabled in the main menu or via the tool bar (figure 2).

Figure 2 – toolbar icon.

A status item in the lower right corner of the hale studio window shows the status of the validation. If there is a small warning sign in the icon, it informs us about validation errors (figure 3).

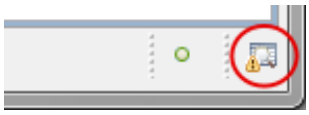

Figure 3 – Warning message icon.

Clicking on the status item opens the Report List and selects the last validation report (figure 4).

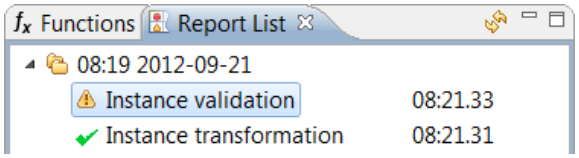

Figure 4 – Validation report window.

If the Properties view is not yet shown, you can open it by double-clicking on the report. If there were problems during the validation you will see the Warnings section in the report's properties. There the

validation warnings are listed, grouped by the property where they occur. If there are many warnings for a property, only a few are displayed, because usually they originate from the same problem.
You can inspect the instances for which these warnings were generated by double-clicking on a warning message or a property. The transformed data view is opened, and the instance associated to the message or the instances with problems for this property are displayed. Please note that this action pins the Properties view, so it shows the report information even if another view is selected.

Schematron validation

Once the XML file has been successfully validated against the corresponding schema, we can proceed with the Schematron validation.

Schematron is a structural based validation language, defined by Rick Jelliffe, as an alternative to existing grammatical approaches. Tree models, defined as XPath expressions, are used to formulate assertions and provide user-centred reports on XML documents. Expressing validation rules using models is often easier than defining the same rule using a content model. The tree motifs are collected together to form a Schematron pattern.
Schematron is a useful and accessible supplement to other schema languages. The open source XSLT implementation is based on an open basic framework for extension and customization.
This document introduces the Schematron language and available implementations. An overview of the architecture is also provided in order to produce customized versions.
The implementation of Schematron derives from the observation that tree pattern-based validators can be trivially constructed using XSLT stylesheets. For example, a simple stylesheet that validates that `house`s must have `wall`s can be defined as follows:

```
         <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">

            <!-- select all houses -->
            <xsl:template match="//house">
               <!-- test whether it has any walls -->
               <xsl:if test="not(wall)">
               This house has no walls!
               </xsl:if>
            </xsl:template>

         </xsl:stylesheet>
```

It should be obvious from the above that if a house does not have any walls, a simple error message will be displayed to the user.

A more relevant example is like below that validates ISO 19139 XML metadata confirms to a rule for Spatial Representation Type according to the the latest INSPIRE metadata guidance.

```
<sch:pattern fpi="metadata/2.0/req/isdss/spatial-representation-typeNN">
    <sch:title>Spatial Representation Type is not nillable for
dataset/series</sch:title>
    <sch:p>Dataset and dataset series must have a
MD_SpatialRepresentationTypeCode</sch:p>
    <sch:rule

context="//gmd:MD_Metadata[1]/gmd:identificationInfo[1]/gmd:MD_DataIdentification[1]/gmd
:spatialRepresentationType">
      <sch:assert
        test="($hierarchyLevelCLValue = 'dataset' or $hierarchyLevelCLValue = 'series')
and count(gmd:MD_SpatialRepresentationTypeCode) &gt; 0"
        > MI-50c (Spatial representation type): Dataset and dataset series metadata must
have at least one
        gmd:spatialRepresentationType with gmd:MD_SpatialRepresentationTypeCode. The
codeListValue
        must be one of 'vector', 'grid', 'tin', or 'textTable' </sch:assert>
    </sch:rule>
  </sch:pattern>
```

Schematron is therefore a simple layer above XPath and XSLT allowing it to leverage existing tools and benefit from a framework which is already familiar to XSLT developers. Yet from a user perspective, the details of XSLT are hidden; the end-user need only grapple with the XPath expressions used to define constraints.

In the INSPIRE domain some dataset general ETS conformance classes are already predefined in the Annex I context, but if some specific application schema should be tested other ETS following the general INSPIRE community rules will be implemented and developed.
In the international standard contest, some schematron are already predefined and available i.e. GeoSciML 4.1 and EarthResourceML (Table 2), but in the case of OGC GeoSciML schema a cloud service to support the schematron validation against GeoSciML model is available; in the below table all these resources are presented.

Table 5 – OGC and CGI Schema location

Data Model	INSPIRE Validation
GeoSciML 4.1	http://schemas.opengis.net/gsml/4.1/
EarthResourceML 2.0.1	http://www.earthresourceml.org/schemas/

GeoSciML 4.1 validator	https://validation-service.inspire-helpdesk.eu/#GeoSciML

Abstract Test Suite validation

In all the INSPIRE Data Specification technical guidelines (SD-TGs) to validate the conformity to data model concept are presented a set of rules to test classes of model, Abstract Test Suite (ATS) included in the Annex A of the INSPIRE Data Specifications is the starting point for the conformance testing process of datasets.

The requirements to be tested are grouped in several conformance classes. Each of these classes covers a specific aspect: one conformance class contains tests reflecting the requirements on the application schema, another on the reference systems and if a dataset is not yet conformant with all requirements of the data specification, conformity to individual conformance classes can be claimed. In order to be conformant to a specific conformance class, a data set has to pass all tests defined for that conformance class. The following table provides an example of conformant classes based on ATS.

Table 6 – Abstract Test Suite example list

Conformance Class	Tests
A.1 Application Schema Conformance Class	A.1.1 Schema element denomination test
	A.1.2 Value type test
	A.1.3 Value test
	A.1.4 Attributes/associations completeness test
	A.1.5 Abstract spatial object test
	A.1.6 Constraints test
	A.1.7 Geometry representation test
A.2 Reference Systems Conformance Class	A.2.1 Datum test
	A.2.2 Coordinate reference system test
	A.2.3 Grid test
	A.2.4 View service coordinate reference system test
	A.2.5 Temporal reference system test
	A.2.6 Units of measurements test
	A.3.1 Unique identifier persistency test

A.3 Data Consistency Conformance Class	A.3.2 Version consistency test
	A.3.3 Life cycle time sequence test
	A.3.4 Validity time sequence test
	A.3.5 Update frequency test
A.4 Data Quality Conformance Class	A.4.1 Data quality target results test
A.5 Metadata IR Conformance Class	A.5.1 Metadata for interoperability test
A.6 Information Accessibility Conformance Class	A.6.1 Code list publication test
	A.6.2 CRS publication test
	A.6.3 CRS identification test
	A.6.4 Grid identification test
A.7 Data Delivery Conformance Class	A.7.1 Encoding compliance test
A.8 Portrayal Conformance Class	A.8.1 Layer designation test

If we want consider a single conformance class, namely: *A.1 Application Schema Conformance Class,* and we want describe how to implement the abstract tests associated to this conformance class with reference to a GML dataset, in order to evaluate that it fulfils the requirements included in the relevant data specification. In the specific example:

A.1.4 Attributes/associations completeness test

Test Method: Examine whether all attributes and association roles defined for a spatial object type or data type are present for each instance in the dataset.

A.1.5 Abstract spatial object test

Test Method: Examine that there are NO instances of abstract spatial object / data types in the dataset provided.

A.1.6 Constraints test

Test Method: Examine all instances of data for the constraints specified for the corresponding spatial object / data type. Each instance shall adhere to all constraints specified in the target application schema(s).

A.1.7 Geometry representation test

Test Method: Check whether all spatial properties only use 0, 1 and 2-dimensional geometric objects that exist in the right 2-, 3- or 4-dimensional coordinate space, and where all curve interpolations respect the rules specified in the reference documents.
Some of the tests required in the conformance classes can be automated by means of validation tools or designed specific API. To transform the ATS in executable test we need to transform ATS to Executable Test Suite (ETS) write specific code following schema presented in figure 5 below. Some documentation about ETS can be found in the INSPIRE GitHub repository at: https://github.com/inspire-eu-validation/ets-repository.

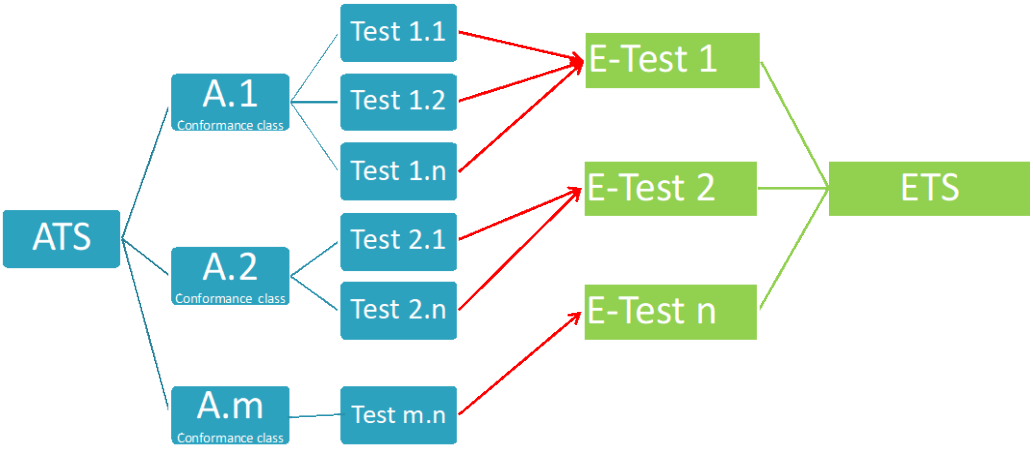

Figure 5 – ATS versu ETS transformation.

DATASET VALIDATION RECOMMENDATIONS

Within the GIP project such as WP3 it was decided to support the use of the GeoPackage standard as an exchange file format, this also in light of an openness of the INSPIRE Community towards this format. At the moment, however, the dataset validation systems are all oriented towards the validation of the GML format, therefore below are some recommendations for the validation and use of GeoPackages.

We have to consider that the GeoPackage coding standard describes a series of conventions for storing the following in a SQLite database:

- vector characteristics
- set of image tile arrays and raster maps on various scales
- attributes (non-spatial data)
- extensions

It must be considered that, since the SQLite container is GeoPackage encoding standard that defines the rules and requirements of the content stored in a GeoPackage container, it will be necessary to set the reference data model of the INSPIRE or OGC standard to be harmonized as target schema.

Having set a specific standard model as target scheme, the data that will be imported into this container will be largely validated in the transfer procedure. For more validation checks instead it will be necessary to proceed with a test performed by an extraction or transformation from GeoPackage to the GML in order to be able to use all the tools previously described.