Technical Testing Methodology

MARCH 2015

DELIVERABLE

Project Acronym: **SDI4Apps**
Grant Agreement number: **621129**
Project Full Title: **Uptake of Open Geographic Information Through Innovative Services Based on Linked Data**

D4.4 TECHNICAL TESTING METHODOLOGY

Revision no. 01

Main Authors: Alessandro Pironi (HYPER)
Matteo Lorenzini (HYPER)

Project co-funded by the European Commission within the ICT Policy Support Programme		
Dissemination Level		
P	Public	X
C	Confidential, only for members of the consortium and the Commission Services	

REVISION HISTORY

Revision	Date	Author	Organisation	Description
01	17/03/2015	Matteo Lorenzini	HYPER	Initial draft and first implementation
02	20/03/2015	Alessandro Pironi	HYPER	Fixing of deliverable's structure and implementation of contents
03	27/03/2015	Martin Kuba	Masaryk University	Contribution with suggestion about functional and non-functional testing
04	27/03/2015	Michal Kepka	CCSS	Peer Review
05	27/03/2015	Tomas Sapak	Masaryk University	Peer Review
06	31/03/2015	Matteo Lorenzini	HYPER	Final Version and Formatting

Statement of originality:

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Disclaimer:

Views expressed in this document are those of the individuals, partners or the consortium and do not represent the opinion of the Community.

TABLE OF CONTENTS

LIST OF TABLES

EXECUTIVE SUMMARY

This deliverable, concerning the activities and main topics of WP4, presents the testing methodology adopted for checking both the integration of basic cloud components with the other advanced tools such as visualization and data management module and the interoperability of different type of data and source managed by SDI4APPs platform.

1 INTRODUCTION

The purpose of this document is to define the methodology by which we will test the components developed within the deliverable D4.2 Advanced Tools API Design and refers to a generic bibliography focused on the definition of qualitative and quantitative test. The deliverable describes the features of the library s4a.js made available by the platform SDI4Apps. Through this library we will provide web application's and mobile app's developers a set of API to integrate, within the pages of their application, the appropriate components that will show data coming from the SDI4Apps platform. The purpose of the API is to provide developers with a predefined set of visualization and interaction tools concerning primarily - but not only - map-based representations of data.

The API library is structured in 5 modules:

1. Map module
2. Information retrieval module
3. Advanced visualization module
4. Mobile module
5. Analytical and Modelling module

For each of these modules we will apply the methodology provided in this document, to understand how to design the tests that will have to be performed in order to assess the fulfilment of the module requirements.

2 USER AND USE CASE

In the definition of a test methodology for a software component the first task is the identification of the users involved and their relative use cases[1].

The software component we are going to test is not an end user application, but rather a tool (actually an API library) to be used by developers to implement third-party applications. In the context of SDI4Apps project the API library is used for the development of pilot applications.

The main high level use case can therefore be described as:

> *a web application developer uses the SDI4Apps library to integrate within his application the tools made available by the library itself*.

This implies that the scope of the test will be the API library itself and not the pilot applications that make use of the library. Having this scope, we point out that some of the characteristics typically tested in a software module will have a slightly different interpretation in this context.

For example, if we consider the non-functional requirement of *usability*, in this context we mean the ease with which *the developer* uses the library to develop his application, and not the usability experienced by an end user that interacts with the application itself.

[1] Armour, F., Granville M., "Advanced Use Case Modelling: software system". Addison-Wesley, 2000.

3 TEST METHODOLOGY

The methodology we outline in this document focuses on the testing of the modules of the library described above. For each module the definition of the test will follow the same procedure:

1. identification of functional and non-functional requirements to test
2. definition of the test matrix to be used for the test, that details:
    1. definition of the success condition for each requirement
    2. definition of the indicators for each success condition
    3. definition of the metric of each indicator
    4. definition of the thresholds of success to be used for each metric

The following sections will address the details of the procedure detailed above.

3.1 Identification of functional and non-functional requirements

Each module of the library will be tested in both its functional and non-functional aspects, which have to be identified in each module.

The testing of **functional aspects** covers the testing of the specific goals of each module of the library. For this kind of testing we will identify the goals of each module and for each goal we will define at least one test that will cover the goal. More than one test will be appropriate only where the goal has more than one application in the context of the library.

For the identification of the **non-functional aspects** to be tested for each module we will consider two possible sources. First we will identify the explicit non-functional requirements for the module, if any are stated in its definition. Then we will identify other non-functional aspects that are anyway important to address, due to the nature of the library as a set of modules inter-connected with one another. To do this we will select for each module the appropriate non-functional requirement from a list defined in this same document. The selection of the non-functional requirements to be tested depends on the nature of the module, since some are defined as pure system services, while others are to be used directly in an interface with a real user.

3.2 Definition of test matrix per requirement

The purpose of the test matrix is to provide the base for the testing of functional and non-functional requirements as identified above. The matrix describes what the tests will have to cover both in the functional and non-functional domain. In this light the matrix provides guidance for the definition of tests that need to be carried out in order to assess the fulfillment of the requirements of the module under test. With this in mind, we will not list all the specific

tests that will be included in the test plan, but rather we describe the methodology to come up with the tests to be performed.

For each requirement identified in a module, the test matrix lists the **success conditions**. Success conditions describe input given and the expected outcomes of a successfully tested requirement. For example, if a requirement of a module states that the module “provides a service API to add a document to a remote folder”, a success condition is that providing to the module service API a document and a reference to a folder, the document is actually added to the specified remote folder. The test matrix also identifies the indicators to be measured, to get evidence of the level of success achieved. Indicators can be anything that provides such evidence, including for example module statistics, data mining in the module logs, queries in the module domain, or even user surveys in the usage of the module. On the trail of the example above, an indicator could be a query on the module folder contents, looking for the document that has been added through the service.Each indicator has an associated metric that is used to express a quantity to the evidence that results from a given test. The metric can be *quantitative* or *qualitative*. A quantitative metric can be used where the indicator can be expressed as a number in a given unit. A qualitative metric is either a free text description or anyway a response that cannot be given with a number. Qualitative information, such as information that ensues from free text surveys, needs to be evaluated as a discrete value in order to be useful in assessing the outcome of the test (see below). Following the example above, the metric for the service could be whether the document is found in the folder contents or not; which is a qualitative, non numeric, metric but with discrete values. Finally, each test provides a scale that reads the measurement as a certain outcome. Basic outcomes are either *success* of the test, or its *failure*. Success implies that the requirement is fully met. A failure, on the other hand, implies that the requirement is not fulfilled. Other possible outcomes can be given on specific metrics. These include *error*, meaning that an error occurred in the testing procedure and an outcome cannot be established. Other tests might provide other values, to be explained in the specific section describing the test. With reference to a responsiveness requirement, we can take into account the response time of a page as indicator. We can measure the time between the user input and when the page is fully loaded as a metric, and assume that a time under 2 seconds has to be considered as a successful result. In this example, the page loading time is the metric, and 2 second is the scale adopted.

3.3 Test execution

The execution of the tests can either be automated or manual.

Automated execution is taken care of by a program specifically designed to carry out the test. The program is set up for the success condition, evaluates the indicators using the appropriate

metric and then provides response in terms of success or failure of the test, including the possibility of error in the testing itself. Automated tests can be programmed as suites of tests, where each suite can be thought as a collection of related tests. The execution of the suite actually runs all the tests in the suite and provides a report of the results given by each test, putting emphasis on the feedback reported by failed tests (where any is provided by the test itself).

Manual testing is taken by a user of the module and cannot be automated as a program to be run. Asking the user to describe the usability of a service, or asking a tester to try out a given service on a scripted procedure, are both good examples of a manual test, the first for non-functional testing and the second for functional testing. Being manual, this type of testing is much more costly than automated execution tests. Manual testing can also make use of test suites to collect related tests. For example a user survey can include many questions (tests) for the user to answer.

3.4 Available tools

Given the scope of the modules in the library, we can already identify a set of tools that can be used to carry out the tests. With this set of tools in mind, the design of the test can be steered so that the test can be easily implemented with one of the provided tools.

Tools can cover both type of requirements of a module (functional and non-functional), and types of metric of an indicator (quantitative or qualitative). For each tool listed below, we also provide information about its suitability for manual and automated testing. The list of tools below can be extended in the course of the project.

Tool	Type of requirement	Type of metric	Type of testing
User surveys	Functional and non-functional	Quantitative and qualitative	Manual
Stress tests (with Selenium and/or SOAP-UI)	Non-functional	Quantitative	Automated
SOAP-UI WS tests	Functional	Quantitative	Automated
Selenium and Java Web Driver tests	Functional	Quantitative	Automated

Table 1: testing tools

4 MAP MODULE

4.1 Module definition

The Map module of s4a.js provides a set of functionalities related to the possibility to embed a map component in a web page.

On top of the map the component will show geometry of objects derived from one or more data layers.
This goal is very common in GIS-enabled application, where we want to provide the user a visual representation of objects linked with their geographical position (and shape).
The main tasks of the Map module are:

1. provide a way to embed a map in a pre-existing HTML DOM.
2. provide a method to add data layers (identified by unique URIs) to be shown on top of the map
3. allow the developer to add interactive controls (tools) to the map to let the user perform some kind of operation on the map itself.

Once the data layers will be added to the map, it will be possible to show or to hide every single layer.
It will be possible to add other tools to add functionalities as pan, zoom, filter object, draw new geometries on top of the map and so on.
While defining testing methodology of this (and other) components we will use test cases whose expected result is strictly related to the specific functionality: testing the zoom tools the expected result will probably be the ability to zoom in and zoom out the map.

4.1.1 Functional test

Success Condition	Indicators	Metric and Scale
Invoking the API to insert the map within the HTML page results in the HTML page filled with the map	Whether the map is present in the HTML DOM	If the map is present the test is successful
Geographical objects visualized in the map are correctly geo-located	Correspondence between CRS and geographical item	If the marker matches with Point of Interest the test is successful
The geometry of the geographical objects is properly defined	Correspondence between different kind of geometry (line, point and polygon) and geographical representation	If the geometry definition correspond to geographical representation test is successful

Table 2: Map module functional test

4.1.2 Non-functional test

Success Condition	Indicators	Metric and Scale
The map is loaded in a time acceptable for user experience (responsiveness)	The time T needed to load the map and insert it into HTML page by an automated test procedure	if T < 1 sec test is successful
It's easy for the developer to use the module and integrate it into his application (usability)	A user survey set of questions addresses the issue	The average score in the user survey is at least 4 (on a 1-5 scale)

Table 3: Map module non-functional test

5 INFORMATION RETRIEVAL MODULE

5.1 Module definition

The goal of this module is to provide methods to interact with data layers: perform queries on objects, filter object depending on certain attributes, provide facets classifications.

While the Map module is focused on data geometries and data visualization, this module exposes methods to interact with alphanumerical attributes of the datasets and to perform faceted browsing.

5.1.1 Functional test

Success Condition	Indicators	Metric and Scale
Information retrieval module finds the terms specified in the query in one or more textual or digital objects	terms are identified in textual objects	if the terms specified in the request are underlined in the text, the test is successful
Information retrieval module identifies geographical objects corresponding to query parameters	geographical objects are represented on the map	if the parameters specified in the query correspond to the represented geographical objects, the test is successful
Information retrieval module provides to a well defined facet definition following project's data-model	data and metadata are categorized following classes and concepts of project's data-model	if the data and metadata follow a well defined semantic structure and also they follow the data-model's concept, the test is successful

Table 4: Information retrieval module functional test

5.1.2 Non-functional test

Success Condition	Indicators	Metric and Scale
The information resulting from the query are loaded in a time acceptable for user experience (responsiveness)	The time T needed to show the query results and insert them in the HTML/DOM page by an automated test procedure	if T < 1 set, test is successful
It's easy for the developer to	A user survey set of questions	The average score in the user

use the module and integrate it into his application (usability)	addresses the issue	survey is at least 4 (on a 1-5 scale)

Table 5: Information retrieval module non-functional test

6 ADVANCED VISUALIZATION MODULE

6.1 Module definition

Using the Advanced visualization module the developer will be able to enrich his applications with graphics (both map based or not).

This module will provide the common set of graphics used to represent statistical information in a visual way: pie chart, bar chart, and so on.
It will be possible to show statistical analysis on geographically referenced data using map-based graphics like diagram map, heat map, prism map and so on.
Moreover, this module will provide the capability to link different data views based on the same dataset with functionality inspired to Crossfilter and D3.js[2] library.

6.1.1 Functional test

Success Condition	Indicators	Metric and Scale
Management of large scale dataset and repository	Dataset and repository can be visualized using different type of visualization i.e. charts and maps.	If dataset's value are properly visualized using different type of scale and representation, the test is successful.
Dynamic visualization and representation of data and metadata	Data and metadata can be visualized in a dynamic system using different kind level of detail	If the visualization tool allow a dynamic representation of different kind of data, the test is successful.
Different scale of representation	Data and metadata are represented using different kind of level of details from a macro to a micro level	If the visualization tool guarantee a useful data representation on a different level of detail using different type of visualization i.e. map or chart, the test is successful.

Table 6: advanced visualization module functional test

6.1.2 Non-functional test

Success Condition	Indicators	Metric and Scale
It's easy for the developer to	A user survey set of questions	The average score in the user

[2] http://square.github.io/crossfilter/, http://d3js.org/

use the module and integrate it into his application (usability)	addresses the issue	survey is at least 4 (on a 1-5 scale)
Data derived from the advanced visualizations can be exported in common_open data formats (interoperability)	The module offers data export for each advanced visualization model	If true, test is successful
The graphs are loaded in a time acceptable for user experience (responsiveness)	The time T needed to load the graphs and insert them in the HTML/DOM page by an automated test procedure	if T < 1 sec per graph, test is successful

Table 7: Advanced visualization module non-functional test

7 MOBILE MODULE

7.1 Module definition

Map module and Advanced visualization module provide tools to represent data both on graphics and maps (and on map based graphics too) and to embed the results on a web page.

The same functionalities can be embedded also in mobile apps using the same modules.
The goals of a specific Mobile module in the s4a.js library are:

1. provide different navigation and user interactions mechanisms for mobile devices
2. allow the mobile application to interact with cached version of data to provide data access also when connectivity is not available any more (obviously, data are to be downloaded for caching when connection is available).

The details of the interaction between the app and the backend server to manage the synchronization of data (especially related to data modification performed by the app) are described in D4.2 Advanced Tools API Design deliverable.

7.1.1 Functional test

Success Condition	Indicators	Metric and Scale
The module provides navigation mechanisms dedicated to mobile devices	A user driven procedure on a test case shows the navigation on a mobile device	The navigation shown is appropriate for a mobile device, evaluated by the user with a score of at least 4 (on a 1-5 scale)
The module provides interaction mechanisms dedicated to mobile devices	A user driven procedure on a test case shows interaction elements on a mobile device	The interactivity shown is appropriate for a mobile device, evaluated by the user with a score of at least 4 (on a 1-5 scale)
The module allows to work offline	An automatic scripted procedure tests interaction with the module on an off-line mobile device	The device responds to the scripted procedure with cached data

Table 8: mobile module functional test

7.1.2 Non-functional test

Success Condition	Indicators	Metric and Scale
It's easy for the developer to	A user survey set of questions	The average score in the user

use the module and integrate it into his application (usability)	addresses the issue	survey is at least 4 (on a 1-5 scale)
The GUI elements specific to a mobile device are loaded in a time acceptable for user experience (responsiveness)	The time T needed to load the mobile specific GUI elements and insert them in the HTML/DOM page by an automated test procedure	if T < 1 sec for a page containing a restricted (less than 5) specific GUI elements, test is successful

Table 9: mobile module non-functional test

8 ANALYTICAL AND MODELLING MODULE

8.1 Module definition

Through the Analytical and Modelling module's functions user can perform analysis and simulations, following designed and predefined patterns, on specific sets of data.

These analysis let the user specify some kind of parameter (e.g.: geographic boundaries) to be used for calculations.

This kind of operations are very expensive in terms of CPU and memory usage, and are not to be performed in a typical request-response synchronous handshake.

The common scenario for user interaction is:

1. user choose the simulation or analysis to be performed (from a predefined set), and inputs needed parameters
2. user submits the analysis request; an asynchronous process starts in background
3. the applications polls the server at a preset time interval to know if results are available
4. When results are ready, they are returned to application. The application formats results and show it to the user.

8.1.1 Functional test

Success Condition	Indicators	Metric and Scale
The analytics and modelling module must elaborate data request managed by SDI4APPS data repository and model	Point of interest are represented from a geographical point of view within a boundary defined by the request	If the results of the analysis correspond to a well defined geographical scale and data are analysed properly corresponding to the model structure, the test is successful

Table 10: Analytical and modelling module functional test

8.1.2 Non-functional test

Success Condition	Indicators	Metric and Scale
It's easy for the developer to use the module and integrate it into his application (usability)	A user survey set of questions addresses the issue	The average score in the user survey is at least 4 (on a 1-5 scale)
The module offers a wide	An automated procedure set	If the polling results in an

range of status reporting during the polling of the response (reliability)	up to result in an error (forced crash or out of bounds analysis) is run on the module	appropriate error are reported to the pooler, the test is successful

Table 11: Analytical and modelling module non-functional test

9 CONCLUSION

The adopted methodology, based on both SDI4APPS DoW and D4.2 Advanced Tools API Design, defines a very practical and useful solution for testing both the integration and interoperability between different components within SDI4APPS framework, and the scalability of SDI4APPS advanced tools towards SDI4APPS framework architecture taking in to consideration both functional and non-functional aspects.

REFERENCES

Lisa Crispin, Janet Gregory - Agile Testing: A Practical Guide for Testers and Agile Teams (Addison Wesley)

SQA - Software quality assurance IEEE 730

STD Software test documentation IEEE 829

V&V - Software verification and validation IEEE 1012

1012-2012 IEEE Standard for System and Software Verification and Validation. 2012

ISO/IEC/IEEE 29119-1:2013 - Software and Systems Engineering - Software Testing

http://en.wikipedia.org/wiki/Software_testing

http://en.wikipedia.org/wiki/Non-functional_requirement

http://en.wikipedia.org/wiki/Functional_requirement

http://users.ece.cmu.edu/~koopman/des_s99/sw_testing/