DELIVERABLE

Project Acronym:	**OTN**
Grant Agreement number:	**620533**
Project Full Title:	OpenTransportNet – Spatially Referenced Data Hubs for Innovation in the Transport Section

D5.3 TECHNICAL TESTING REPORT

Version: 1.0

Authors:

Irene Matzakou (INTRA)
Leonidas Kallipolitis (ATC)
Premysl Vohnout (HSRS)
Jan Jezek (UWB)
Indulis Makens (EX)
Dieter de Paepe (iMinds)

Internal Reviewers:

Lieven Raes (CORVE)
Karel Charvat (HSRS)
Karel Jedlicka (UWB)
Pieter Colpaert (iMinds)
Richard Brown (ATC)

Dissemination Level		
P	Public	X
C	Confidential, only for members of the consortium and the Commission Services	

This project has received funding from the European Union's Competitiveness and Innovation Framework Programme under grant agreement no. 620533.

Table of Contents

List of Figures

List of Tables

Revision History

Version	Date	Author	Organization	Description
0.1	08/07/2015	Irene Matzakou	INTRA	ToC
0.2	10/07/2015	Irene Matzakou, Leonidas Kallipolitis	INTRA, ATC	Guidelines, Examples, Assignments
0.3	14/07/2015	Irene Matzakou	INTRA	Input and clarifications added
0.5	20/07/2015	Premysl Vohnout, Jan Jezek, Indulis Makens, Leonidas Kallipolitis, Dieter de Paepe	HSRS, UWB, EX, ATC, iMINDS	Input Consolidated, Draft for internal review
0.8	24/07/2015	Irene Matzakou	INTRA	Comments addressed, Draft for consortium review
0.9	28/07/2015	Irene Matzakou	INTRA	Consortium Review Comments addressed
1.0	29/07/2015	Irene Matzakou	INTRA	QA, Final Version for submission to the EC

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This deliverable contains original unpublished work except where clearly indicated otherwise.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.

Executive Summary

This document provides the methodology used for the technical testing of the components comprising the OTN platform. It defines the OTN Quality Model which includes a set of software quality characteristics to be tested. Furthermore, it presents the concrete test plan that was followed, its objectives, phases and team, as well as the results of the testing for all the components, both in terms of functional completeness, as well as in terms of functional correctness. Additionally, it includes testing about the performance efficiency of the OTN platform.

The results prove that the OTN platform adheres to the defined quality metrics and can perform well in a number of test cases and usage scenarios. The main characteristics that are tested are Functional Suitability, Performance Efficiency, Compatibility, Usability, Security, Maintainability and Portability.

1 Introduction

1.1 Purpose and scope

The objective of this document is to present **technical testing results of the OTN platform**. The current document outlines the methodology for the technical testing, the test plan and the results of the testing based on the defined OTN Quality Model.

The Quality Model describes a set of software characteristics that the components of the OTN platform should cover in order to ensure that the integrated result provides high quality services and functionalities to the end users.

1.2 Structure of the deliverable

The deliverable is structured in 6 sections:

Section 1 “Introduction”: Purpose and Scope of this Deliverable.

Section 2 “Methodology for the Technical Testing”: Description of the Software Assurance Process which is the basis for the definition of the OTN Quality Model.

Section 3 “Test Plan”: Description of the **objectives, phases and the team** that was responsible to run the tests of the OTN platform components.

Section 4 “Technical Testing Results”: Detailed presentation of the testing results. The results are presented per software characteristic according to the OTN Quality Model.

Section 5 “Conclusion”: Concludes the deliverable.

Section 6 “References”: Lists the references appearing in the document.

2 Methodology for the Technical Testing

2.1 Software Assurance Process

The OTN Platform will be tested in order to assess the maturity of the technical implementation and the alignment to the user requirements from a technical perspective. The technical assessment of the OTN platform is supported by monitoring the technical parameters of the platform performance and aims to determine how far the integrated prototype meets the technical requirements and the functional specifications.

OTN development tasks are tested according to established standards on software assurance process. This process aims to assess the efficiency of the platform functionalities and provide evidence that the integrated prototype is fully functional and available for release through a software assurance process. In principle, software assurance can be realised by evaluating both the software itself (the product) and how it has been developed (the process). Both aspects are important for a platform targeting to support scalable functionalities.

However, in prototypes such as the OTN platform, the software assurance of the process becomes quite difficult. This is mostly due to the constant change in both the design and the requirements of the software during the project cycle itself, as a result of the on-going work in the other work packages and the living lab approach used for the user requirements gathering and the user engagement process. Thus, the software assurance process means testing the outcome of the prototype, considering this as an integrated platform consisting of individual components which are integrated using the architecture principles that are analysed in details in deliverable D2.5.

Software validation is the "confirmation by examination and provision of objective evidence that software specifications conform to user needs and intended uses, and that the particular requirements implemented through software can be consistently fulfilled". Since software is usually part of a larger hardware system, the validation of software typically includes evidence that all software requirements have been implemented correctly and completely.

In general, software validation is the process of developing a "level of confidence" that the system meets all requirements, functionalities, and user expectations as set out during the design process. It is a critical tool used to assure the quality of its component and the overall system. It allows for improving/refining the end product.

Software validation is realised through quality models. In the past, different quality models have been proposed, each of which addresses different quality attributes that allow evaluating the developed software. Some of the most well-known are:

- McCall's model of software quality [1] (GE Model, 1977), which incorporates 11 criteria encompassing product operation, product revision, and product transition.
- Boehm's spiral model [2] (1978) based on a wider range of characteristics, which incorporates 19 criteria. The criteria in both, this and the GE model, are not independent as they interact with each other and often cause conflicts.
- ISO 9126-1 [3] incorporates six quality goals, each goal having a large number of attributes. These six goals are then further split into sub-characteristics, which represent measurable attributes (custom defined for each software product).

Recently the BS ISO/IEC 25010:2011 standard [4] about system and software quality models has replaced ISO 9126-1. Applying any of the above models is not a straightforward process. There are no automated means for testing software against each of the characteristics defined by each model. For each model, the final attributes must be matched against measurable metrics and thresholds for evaluating the results must be set. It is then possible to measure the results of the tests performed (either quantitative or qualitative/observed).

For the OTN case, we have adopted the ISO/IEC 25010:2011 standard, which is the most widespread reference model and it includes the common software quality characteristics that are supported by the other models. This standard defines two quality models providing a consistent terminology for specifying, measuring and evaluating system and software product quality:

- Quality in use model, which is composed of five characteristics that relate to the outcome of interaction with the system and characterises the impact that the product can have on the stakeholders.
- Product quality model, which is composed of eight characteristics that relate to static properties of software and dynamic properties of the computer system.

For our case, the product quality model is adopted. The eight characteristics, are further divided into sub-characteristics, as shown in the following figure:

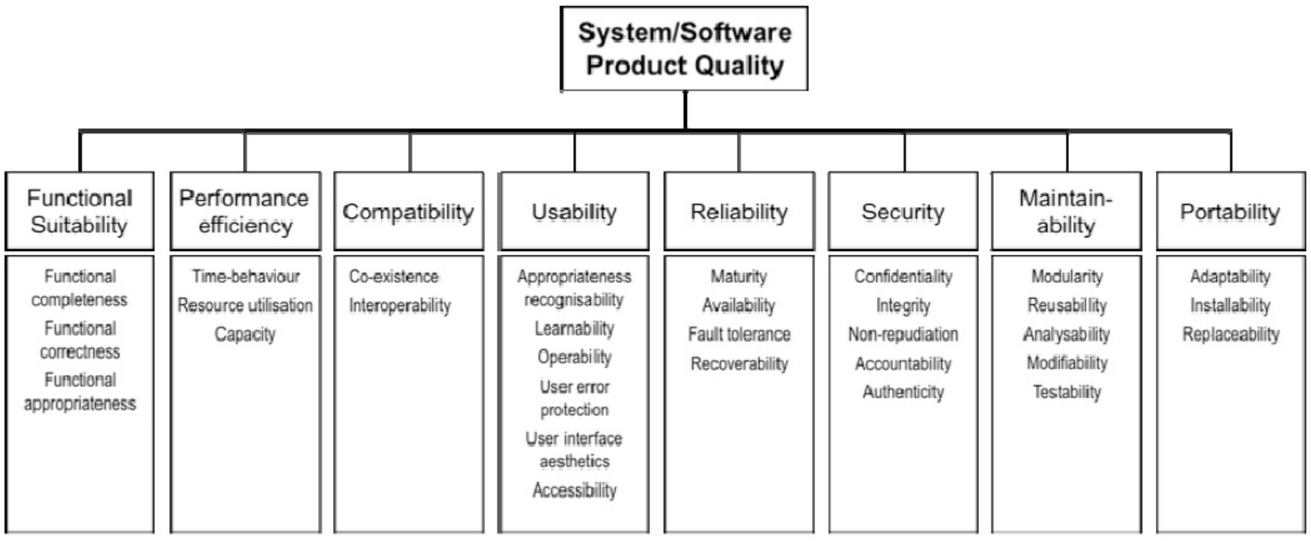

Figure 2-1: The ISO/IEC 25010:2011 system/software quality model characteristics

For each of the sub-characteristics, a metric/measurable attribute is defined, along with thresholds. These metrics and thresholds are customised for each software product, which in our case is the OTN platform (consisting of individual components). By evaluating the complete set of metrics, we will be able to assess the overall quality of our platform and the per cent to which we were able to meet the user requirements (reflected to system specifications and functionalities) defined during the design phase of the project.

2.2OTN Quality Model

The software quality model of the BS ISO/IEC 25010:2011 standard is adapted to the needs of the OTN platform, in order to define appropriate metrics and to be able to evaluate the platform capabilities. These metrics need to reflect the characteristics that they represent. They also need to allow appropriate measurements to be obtained, either through quantitative methods (e.g. by software tests/simulations, usability tests) or qualitative methods (e.g. through user observations). Three types of classes of metrics are defined in this standard:

- Internal metrics associated with static internal properties of a system such as number of function calls, number of rules.
- External metrics associated with dynamic external properties. These are metrics that are observable when the user interacts with the system (i.e. the user performs a task/function/operation and observes the response in the sense of time required, results obtained etc.).
- Quality-in-use metrics, which refer to metrics that evaluate the extent to which a system meets the needs of the user.

Since the aim of the OTN platform is that of a proof of concept and not a commercial product by the end of the project, there will be less focus on the internal metrics, which are essentially used in the development phase. As such, the different components and the platform itself will be mainly optimised for their functional suitability and usability. We will therefore concentrate on external metrics and quality in use metrics, which are respectively used in the testing phase and the piloting phase. Various target users should actually make use of these metrics, as follows:

- Developers and IT experts will make use of external metrics prior to the release of the OTN platform prototype to the evaluation phase
- Platform end users (i.e. developers, Data Providers and Service Providers) will make use of quality-in-use metrics, during evaluation to assess the suitability of the prototype to address the needs in helping cities and citizens to find insights in transport data and create meaningful map compositions.

Due to the nature of the OTN platform, we concentrate on the following BS ISO/IEC 25010:2011 characteristics, as shown in the following table:

Table 2-1: OTN Quality Model

ISO/IEC 25010:2011 characteristics	Description	Method for measuring quality metric
Functional Suitability		
Functional Completeness	Assess the implemented functionalities with respect to requirements and project objectives	Observation Tests
Functional Correctness	Assess whether the OTN platform provides the correct results with the needed degree of precision	Log Analysis, Observation Tests
Functional Appropriateness	Assess whether the implemented functionalities facilitate the accomplishment of specified tasks and objectives	Log Analysis, Observation Tests
Performance Efficiency		
Time Behaviour	Assess the response and processing times and throughput rate per user	Log Analysis, Observation Tests, Simulation Tests
Resource Utilisation	Assess the resource usage	Log Analysis, Simulation Tests
Capacity	Assess the number of concurrent users	Simulation Tests
Compatibility		
Interoperability	Assess the interoperability capabilities of the OTN platform	Observation Tests
Usability: It is left to WP6 activities		
Security		
Confidentiality	Assess whether the OTN platform ensures that data are accessible only to those authorised to have access	Observation Tests
Maintainability		
Modularity	Assess whether the OTN platform operation is affected by changes to one or more components	Observation Tests
Portability		
Adaptability	Assess whether the OTN platform can be adapted effectively and efficiently for different hardware, software or other operational or usage environments	Observation Tests

3 Test Plan

Following the OTN Quality Model, this section describes the plan for testing the functional suitability of the OTN platform. The tests are based on the defined quality metrics and are designed taking into account the functional and technical requirements of the platform. All the tests are performed using real data of the production instance of the platform. Analytical testing of all the functionalities has been conducted in order to verify the correct operation and completeness of the platform.

3.1 Objectives

The objectives of the tests are to check the functionalities offered by the OTN platform following the user journeys defined during the user requirements gathering and the identified system workflows presented in D2.5. These workflows are briefly listed below to ensure the current deliverable's completeness and help the reader better understand the test results presented in the next paragraphs.

- Test geospatial data uploading to a Hub
    - Upload geospatial datasets
    - Harmonise geospatial datasets
    - Publish geospatial datasets (make them available for use in map compositions)
- Test Link/Upload existing datasets (open data) to a Hub
    - Upload dataset
    - Harvest/link datasets from remote catalogues
- Test the creation of map compositions
    - Use geospatial and open data in a map composition
    - Save map compositions
    - Share map compositions
- Test the use of mobile devices to upload VGI data to a Hub
    - Upload sensor(mobile) data
    - Upload crowdsourcing data (e.g. reporting of issues)
    - View visualisations in a mobile device
- Test the Marketplace and Business Forum functionalities
    - Create Challenges and Services
    - Buy Services and respond to challenges
    - Create posts in the forum
- Test the discoverability of data and map compositions
    - Browse the data catalogue
    - Browse the map compositions

3.2 Phases

The creation process of the test plan can be summarised in the following phases:

1. Test plan: The test plan includes the strategic design of the validation and verification process, the definition of the test requirements, the definition of resources and the development of the testing procedure.
2. Definition of tests: the functionalities and modules that will be tested are defined. The means of testing are identified and the complete coverage of the functional requirements with the defined tests is assessed.
3. Test preparation: The software environment is prepared and the modules and functionalities to be tested are identified within the software. The Test Team gets acquainted with the testing environment and performs some preliminary runs of the test cases.
4. Test execution: the tests are performed during this phase, their execution course is evaluated and the results are recorded.
5. Test Evaluation: this is the final phase of the testing plan where the complete results are evaluated and the possible errors are analysed.

3.3 Team

In order to execute the set of tests for the platform, a team of IT experts has been assembled. Each member of the team has a specific role during the execution of the test plan. The following table (**Error! Reference source not found.**-1) analyses the roles of the test plan team:

Table 3-1: Test team

Role	Duties
Test Manager	• Coordinates the correct implementation of the OTN testing methodology • Provides technical guidance • Ensures the availability of the necessary resources • Compiles the administrative reports
Test Designer	• Identifies, creates and prioritises the test cases • Develops the test plan and evaluates the test results
Tester	• Executes the test plan • Records the test results • Records possible problems during the execution of the tests
OTN Platform Administrator	• Supervises the good operation of the OTN platform • Ensures proper access to the platform functionalities during testing • Manages data exchanged between the platform and the OTN apps

4 Technical Testing Results

The results of the tests are presented in this section. The presentation follows the parameters of the ISO/IEC 25010:2011 standard and the metrics selected for the OTN Quality Model as described in Table 2-1 above.

4.1 Functional Suitability

In order to test the functional suitability of the implemented functionalities, the OTN platform has been used by developers and IT experts to identify the prototype behaviour with respect to the planned one and to check whether the user requirements, as they are presented in D2.4 [5], and the functional requirements, as they are presented in D2.5 [6], have been addressed. The testing process follows a number of test use cases, which are iteratively repeated. These tests should be successfully accomplished from the OTN platform and in all iterations they should provide the same results. The test cases relate to the OTN requirements and for each of them the result of the testing process is reported. For all the test cases, an http client (web browser) has been used as the method for observation testing, while direct access to the logs of the OTN Server ensure that the intended use of the functionalities is performed (with no programming level exceptions to be occurred).

4.1.1 Functional Completeness

The functional completeness of the OTN platform refers to satisfying the user requirements and functional requirements as introduced in the first year of the project. Currently, the development of the OTN platform meets the planned functionalities in an extended degree, but updates have still to be performed during the setup of the personalised instances of the Hub. The degree of fulfilment is justifiable from the fact that the project is currently planning the pilots in all the partner cities and the Hub core functionalities have to be upgraded and implemented according to the comments received from real stakeholders. Moreover, functionalities with lower priority have an understandable smaller degree of fulfilment and will be revisited after receiving the feedback from the first round of user evaluation.

Table 4-1 lists the functional requirements identified in D2.5. For better clarity only the reference number and the name of the requirement have been included in the table. The two new columns are the ‘Current Status’ which provides a brief description of the work done for the specific requirement and the ‘Degree of fulfilment’ as a percentage defined by the relative technical partners.

Table 4-1: Functional Requirements Fulfilment Status

No	Name	Current Status	Degree of fulfilment
FR01	White-Label Front-end	The Hub front-end is totally customisable through an online administrative environment (CMS functionalities of Liferay).	100%
FR02	Hub Access	Preliminary work to register via social media account has been done.	50%
FR03	User Profile	A user profile page is available in the Hub.	100%

FR04	Landing Page	Most of the Hub key features are accessible directly from the landing page.	85%
FR05	Visible stakeholder paths	The Hub structure guides the users through activity flows based on what they want to do.	90%
FR06	Data Search	Datasets are searchable but an easier way (data Catalogue) will be implemented for the personalised instances.	60%
FR07	Data Download	Datasets are downloadable but an easier way (data Catalogue) will be implemented for the personalised instances.	60%
FR08	Data Upload	Datasets can be uploaded but an easier way (Data Upload form) will be implemented for the personalised instances.	60%
FR09	Volunteer Data	Data and data formats (Senslog) have been identified. Functional analysis of mobile apps is in place and a plan for the development of the apps has been agreed.	50%
FR10	External APIs	No external APIS have been used so far. FA of the pilots has indicated a number of APIs to consider.	20%
FR11	Data Notification	No data notification mechanism has been developed yet.	-
FR12	Data Licensing	Data licenses are not yet displayed to user. Planned support for both public and user-owned (non-public) data.	5%
FR13	Map Visualizations	Map visualisations are available in the platform, but adaptations are required according to preliminary user testing in order to increase usability of the respective tools.	80%
FR14	Map Notifications	No map notification mechanism has been developed yet.	-
FR15	Map Use	Map compositions can be embedded and saved in the user's profile	100%
FR16	Optimal Routing Service	Routing service is available in the environment of SQL. The Rest API has not been developed yet.	30%
FR17	Other Visualizations	Prototype of advanced visualization is available however the API focused on preparation of visualization for third party users is missing.	40%
FR18	Social Game Feature	No social game features have been added so far. Initial concept has to be validated during the first validation period from end users.	-
FR19	Visualisations Catalogue	The created map visualisations which are public can be found in the map composition tools. A distinct separate section for these visualisation will be developed for the personalised instances	65%

FR20	Skill Matching Engine	The skill matching engine has not been implemented yet. Initial concept has to be validated during the first validation period from end users.	-
FR21	Predictive Analysis Engine	Predictive analysis focused on traffic volumes has been developed as a prototype service. No other predictive analysis has been requested so far.	20%
FR22	Map mash-up application	The Hub supports combining of different sources on a single map. Updates will include data coming from CKAN as described in the revised D2.5	70%
FR23	Community	The Hub contains a forum page	100%
FR24	Business Mentoring Interface	The Hub contains a 'business mentors' section which enables the communication with experts in various domains	100%
FR25	Payment Module	No payment module has been developed/integrated yet.	-
FR26	VGI Interface	A number of mobile applications will be developed as a result of the functional analysis of the pilots. The VGI Interface will be part of these application which are currently in the design phase	25%
FR27	Users Notifications	No user notification mechanism has been developed yet.	-
FR28	Find all amenities in close distance	Relying on the available datasets. Maps can support this functionality	100%
FR29	Multi-language	The Hub can support multilingualism using the Liferay content localisation features	100%
FR30	Open APIs	Open API is available for visualization libraries (HSLayers, WebGLayer) as well as Web Services (Laymen, SensLog).	50%
FR31	Pluggable Front-end	An instance of the OTN Hub will be able to be installed in various infrastructures. The Open APIs will be available to provide functionalities and data to existing web portals.	50%

4.1.2 Functional Correctness and Appropriateness

The functional correctness and appropriateness has been tested through iterative test cases performed in 20 cycles and for a period of time of one week. These test cases include both correctness and appropriateness testing, even though the appropriateness of the system will be further tested and analysed in the context of WP6 as it mostly refers to the degree where the platform functionalities facilitate the accomplishment of specified tasks and objectives.

In order to present the test cases, a test case template based on the IEEE Std. 829-1998 [7] standard is used as presented in the following table:

Table 4-2: Test Case Template

Test case Title	*Title*	*A/A*
Goal	*Which is the goal of this test case*	
Prerequisites	*What are the prerequisites to run the test case*	
Steps	▪ *Which are the steps to successfully run the test case*	
Expected successful result	*What is the expected result in case of successful execution of the test case*	
Expected failed result	*What is the expected result in case of failure of the execution of the test case*	
Result	*What is the result after running the test case*	

The following tables include the Test Cases acknowledged:

Table 4-3: Test Case 1

Test case Title	*Use the Marketplace to sell a service*	*TC1*
Goal	Test if a user can add a service to the marketplace.	
Prerequisites	• User is a registered user of the platform • User is logged in	
Steps	1. User navigates to the marketplace->sell a service menu item 2. User fills in the form fields and clicks save	
Expected successful result	1. The service page is created and presented to the user 2. The service is available in the marketplace under the selected category 3. The service is available in the 'My services' section under the 'Explore Data' section	
Expected failed result	The page notifies the user about missing required fields.	
Result	Successful. A new service was created and became available in the marketplace and in the user's personal page.	

Table 4-4: Test Case 2

Test case Title	Use the Marketplace to buy a service	TC2
Goal	Test if a user can buy a service from the marketplace	
Prerequisites	• User is a registered user of the platform • User is logged in	
Steps	1. User navigates to the marketplace->buy a service menu item 2. User searches for a service using the search and filtering functionality	

Expected successful result	1. The results page is presented to the user 2. The user selects a service to see its details 3. The user clicks the buy button 4. The service is available in the 'My services' section under the 'Explore Data' section
Expected failed result	The page notifies the user about errors caused during searching or buying a service.
Result	Successful. The bought service is available in the user's personal page.

Table 4-5: Test Case 3

Test case Title	*Use the Marketplace to create a challenge*	*TC3*
Goal	Test if a user can add a challenge to the marketplace	
Prerequisites	• User is a registered user of the platform • User is logged in	
Steps	1. User navigates to the marketplace->post a challenge menu item 2. User fills in the form fields and clicks save	
Expected successful result	1. The challenge page is created and presented to the user 2. The challenge is available in the marketplace under the challenges section 3. The service is available in the 'My challenges' section under the 'Explore Data' section	
Expected failed result	The page notifies the user about missing required fields.	
Result	Successful. A new challenge was created and became available in the marketplace and in the user's personal page.	

Table 4-6: Test Case 4

Test case Title	*Use the Marketplace to search and respond to a challenge*	*TC4*
Goal	Test if a user can respond to a challenge.	
Prerequisites	• User is a registered user of the platform • User is logged in	
Steps	1. User navigates to the marketplace and goes to "respond to a challenge" menu item 2. User filters the list of challenges and selects one of them 3. The challenge page is presented to the user 4. The user clicks the respond button	
Expected successful result	1. An email is sent to the challenge creator, notifying him about the responding user	
Expected failed result	The page notifies the user about errors caused during searching or responding to a challenge.	
Result	Successful. A new challenge was created and became available in the marketplace and in the user's personal page.	

Table 4-7: Test Case 5

Test case Title	*Use the OTN Forum*	*TC5*
Goal	Test if a user can post a message to the OTN Forum	
Prerequisites	• User is a registered user of the platform • User is logged in	
Steps	1. User navigates to the Discussion Forum 2. User browses the topic threads and selects one 3. User creates a new post	
Expected successful result	1. A new post is created in the selected thread. 2. The post is available in the user's posts at his personal page 3. Subscribed users to the particular thread receive a notification of the new post	
Expected failed result	The page notifies the user about errors caused during creating a new post.	
Result	Successful. A new post was created.	

Table 4-8: Test Case 6

Test case Title	*Use the Business Mentoring Page*	*TC6*
Goal	Test if a user can contact a business mentor	
Prerequisites	• User is a registered user of the platform • User is logged in	
Steps	1. User navigates to the Business Mentors page 2. User clicks the contact link of a Mentor 3. Users fills in the pop up contact form and clicks send	
Expected successful result	1. The mentor receives an email notification of the user contacting him and initiates the discussion	
Expected failed result	The page notifies the user about errors caused during sending the contact details to the mentor.	
Result	Successful. A notification was send to the mentor with the details of the user.	

Table 4-9: Test Case 7

Test case Title	*Data Publishing tool - Upload*	*TC7*
Goal	Upload a file containing geodata	
Prerequisites	• User is logged-in • Data publishing tool is open	
Steps	Upload zipped shapefile	
Expected successful result	The file is unzipped and shown in the Files panel	
Expected failed result	Error message is shown	
Result	Data is successfully uploaded	

Table 4-10: Test Case 8

Test case Title	*Data Publishing tool - Publish*	*TC8*
Goal	Publish uploaded geodata as a map layer	
Prerequisites	• User is logged-in • Data publishing tool is open • Some geodata is successfully uploaded	
Steps	1. Select the uploaded file and choose 'Publish' 2. Fill in the publication form 3. The 'Publish as' field must have the default "As new" value. 4. Click 'Publish'	
Expected successful result	1. The system processes the request and finally new data appears in the Data panel and new layer appears in the Layers panel. 2. The data in Data panel and the layer in the Layers panel is shown in the group it has been published to.	
Expected failed result	Error message is shown	
Result	Data is successfully published	

Table 4-11: Test Case 9

Test case Title	*Data publishing tool - Styler*	*TC9*
Goal	Open the published layer in Styler	
Prerequisites	• User is logged-in • Data publishing tool is open • Some geodata is successfully uploaded and published.	
Steps	1. Select the published layer 2. Click 'Styler' in the layer menu.	
Expected successful result	1. The Styler opens. 2. The layer can be selected in the Layers panel. 3. Select the layer and the layer is shown.	
Expected failed result	Error message is shown	
Result	Layer is open in the Styler and can be styled.	

Table 4-12: Test Case 10

Test case Title	*Search for a dataset in the Catalogue*	*TC10*
Goal	Find required dataset	
Prerequisites	Catalogue client application is opened	
Steps	1. Fill form with search terms 2. Click on search 3. Wait for result 4. Examine results	
Expected successful result	Search result contains searched dataset	
Expected failed result	Search doesn't work	
Result	Dataset can be found using search	

Table 4-13: Test Case 11

Test case Title	*Search for a composition in the Map Compositions Directory*	*TC11*
Goal	Open desired composition	
Prerequisites	Thematic Map Viewer app opened	
Steps	1. Zoom in to desired location 2. Filter compositions by keywords 3. Select desired composition	
Expected successful result	Data from composition is displayed	
Expected failed result	Composition can’t be found	
Result	Desired composition displayed	

Table 4-14: Test Case 12

Test case Title	*Create a map composition - Add layer to map*	*TC12*
Goal	Add published layer to map	
Prerequisites	• User is logged-in • Data publishing tool is open • Some geodata is successfully uploaded and published. It may be styled.	
Steps	1. Select the published layer. Click 'Add to map'. 2. Click on save composition button 3. Fill information about composition 4. Save composition on server	
Expected successful result	Composition is saved on server	
Expected failed result	Error message is shown.	
Result	Composition is stored	

Table 4-15: Test Case 13

Test case Title	*Use an interactive map visualization*	*TC13*
Goal	Visualize and interactively filter the multidimensional spatial data by using different views on the data (map and histogram).	
Prerequisites	Modern browser (e.g. Chrome) with WebGL support that complies with this test: https://www.khronos.org/registry/webgl/sdk/tests/conformance/rendering/point-size.html?webglVersion=1	
Steps	1. Open a http://jezekjan.github.io/webglayer/examples/ukaccidents/ 2. Drag and zoom in and out the map 3. Drag a rectangle over histogram to select the data, click outside the selection and then cancel the selection.	
Expected successful result	The map shows the point data and the heat map displaying the data density. Map view and histograms are ´coordinated´. Dragging the mouse over histogram will select the particular data that are instantly highlighted in the map view. Dragging the map will instantly update the histograms according to the actual bounding box of the map.	
Expected failed result	The map or histogram is blank, the heat map or the point data are not displayed. The histogram does not show any results.	
Result	Successful. Tested in Chrome 43.0.2357.132 and Firefox 39.0.	

Table 4-16: Test Case 14

Test case Title	*Create an interactive map visualization*	*TC14*
Goal	Create a visualization of multidimensional data by using a map view and histogram	
Prerequisites	• Data in CSV format that contains point data expressed by coordinates in UTM and some other numeric attributes. • Basic knowledge of JavaScript and HTML	
Steps	Follow the developer guide available under OTN developer guide.	
Expected successful result	Web page that contains a visualization of a Map view that displays the points and histograms of chosen attribute. Map view and histograms are ´coordinated´. Dragging the mouse over histogram will select the particular data that are instantly highlighted in the map view. Dragging the map will instantly update the histograms according to the actual bounding box.	
Expected failed result	The documentation and API is not sufficient to do that.	
Result	The API and documentation is not yet finished. A basic level of Javascript and WebGL is necessary for achieving this test case.	

Table 4-17: Test Case 15

Test case Title	*Use a mobile device to upload VGI to the Hub*	*TC15*
Goal	Test if a user can add upload VGI to the Hub	
Prerequisites	• User is logged in	
Steps	1. User start track logging 2. User save track route 3. User navigates to menu item->upload track to the Hub	
Expected successful result	Track file is uploaded	
Expected failed result	Track file is not uploaded	
Result	Successful. User receives message “Track file is uploaded!”	

4.2 Performance Efficiency

In order to test the performance of the services offered by the OTN platform to the users, the popular open source tool JMeter[1] was used. JMeter is designed to load test functional behaviour and measure performance of web applications and web services. Web Services Test Plans were created to test the performance of the web services that are used inside the OTN platform to. These services support the portlets of the platform, which offer the desired functionalities to the Hub administrators, City administrators and registered users.

The conducted tests include the creation of indicative scenarios with 5/10/15 concurrent users that send http requests (service calls) to the server and run this test 4 times.

In order to derive these numbers of concurrent users, the following assumptions were made:

- 90.000 visitors/month or 3000 visitors/day to the OTN Hub were assumed, a number significant large considering the purposes of the OTN tools
- The average time the user spends on each OTN page is 5 minutes or 300 seconds
- The number of concurrent users consuming the same service at the same second is derived by the formula: **Number of Concurrent users = Rate of incoming visitors * time of visit** (where Rate of incoming visitors is the number of visitors per second)

By using the numbers mentioned above the Rate of incoming visitors is 0.0347 visitors per second and the concurrent users are 10.41, so the 5/10/15 concurrent users cover the basic assumptions and even surpass the presumed number.

To make it even more difficult, we consider that a user is able to make more actions, thus call many web services, at the same time, so the services were not tested individually but as sets, as it will be demonstrated below.

The following tables present the results of the Web Tests for the different number of users. The first column shows the services that were tested and the rest of the columns represent the various metrics measured by JMeter. More specifically, through JMeter the following results are retrieved:

- Service - The name of the service (endpoint) that has been tested.
- Samples - The number of samples for the service;
- Average - The average time of a set of results;
- Min - The shortest time for the samples of the service;
- Max - The longest time for the samples of the service;
- Error % - Percentage of requests with errors;
- Throughput – the throughput is measured in requests per time unit (second/minute/hour) and in Kbytes/sec

4.2.1 Web Test 1: Marketplace

The conducted Web Test covers the GET services of the Marketplace component.

The base url of the Service endpoints (first column) presented in all the tables below is *<ONT HUB base Url>*/otnServices/platform/marketplace/. The tested services provide the required functionality of the Marketplace component, namely: listing of all services, listing of all challenges, listing of Service categories, listing of Challenge categories, listing of cities, retrieval of a Service’s details (a random service is chosen), retrieval of a Challenge’s details (a random challenge is chosen).

[1] http://jmeter.apache.org

Table 4-18: Web Test for 5 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
/getServices	20	40	27	242	0	21.3	167.08
/getChallenges	20	39	26	248	0	21.6	58.27
/getServiceCategories	20	35	31	44	0	21.2	12.77
/getChallengeCategories	20	35	33	42	0	21	12.69
/getCities	20	35	31	42	0	20.9	13.86
/getChallengesById?id=12	20	40	32	62	0	20.4	45.44
/getServiceDetails? serviceID=31	20	38	30	60	0	20.8	19.61

Table 4-19: Web Test for 10 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
/getServices	40	72	42	272	0	36	281.71
/getChallenges	40	65	40	258	0	36.9	99.63
/getServiceCategories	40	49	20	1035	0	26	15.7
/getChallengeCategories	40	22	20	28	0	39.9	24.07
/getCities	40	22	20	34	0	40	26.55
/getChallengesById?id=12	40	33	23	40	0	38	84.47
/getServiceDetails? serviceID=31	40	33	22	45	0	37.6	.5.5

Table 4-20: Web Test for 15 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
/getServices	60	113	27	678	0	41.1	322.1
/getChallenges	60	110	26	674	0	41.3	111.48
/getServiceCategories	60	33	20	78	0	52.7	31.79

/getChallengeCategories	60	33	21	73	0	52.8	31.85
/getCities	60	33	20	74	0	53	35.11
/getChallengesById?id=12	60	35	22	77	0	55.6	52.51
/getServiceDetails? serviceID=31	60	37	24	76	0	55.5	123.44

The results above show that the services can handle the number of tested concurrent users without significant delays in response times or errors. The average times get higher as the concurrent users and number of requests increases but in an acceptable rate which doesn't result in loss of responsiveness. These results prove that the internal services of the Hub can handle an increasing number of users accessing the various components without any change of the current implementation.

4.2.2 Web Test 2: Layman

The conducted Web Test covers the GET services of the Layman component.

The base url of the Service endpoints (first column) presented in all the tables below is <*ONT HUB base Url*>/ /cgi-bin/layman/layman/. The tested services provide the required functionality of the Layman component, namely: getting list of data (/fileman), list of tables (/data), list of published layers (/layed)

Table 4-21: Web Test for 5 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
/fileman	20	595	388	723	0	6.6	3,23
/data	20	672	490	937	0	5.9	9.36
/layed	20	1312	824	1462	0	3.2	61.61

Table 4-22: Web Test for 10 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
/fileman	40	530	370	799	0	13.9	6.79
/data	40	641	510	1130	0	12.5	19.96
/layed	40	1273	770	1801	0	6.5	126.15

Table 4-23: Web Test for 15 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
/fileman	60	546	379	1040	0	19.1	9.32
/data	60	770	560	1220	0	15.9	25.39
/layed	60	1509	901	2129	0	7.6	146.22

4.2.3 Web Test 3: MIcKA

The conducted Web Test covers the GET services of the MIcKA component.

The base url of the Service endpoints (first column) presented in all the tables below is <*ONT HUB base Url*>/php/catalogue/libs/cswclient/cswClientRun.php. The tested services provide namely: getting list of map compositions, list of datasets.

Table 4-24: Web Test for 5 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
mapCompositions	20	82	70	136	0	18.1	277.93
datasets	20	341	298	426	0	9.1	277.95

Table 4-25: Web Test for 10 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
mapCompositions	40	105	70	210	0	32.2	494.55
datasets	40	406	292	1510	0	16	486.15

Table 4-26: Web Test for 15 users * 4 requests per user

Service	Samples	Average (msec)	Min (msec)	Max (msec)	Error (%)	Throughput	
						Requests/sec	KB/sec
mapCompositions	60	152	72	344	0	37.6	576.92
datasets	60	469	296	1193	0	19.3	587.47

4.3 Compatibility

4.3.1 Interoperability

The OTN platform is designed and developed under the principle of a loosely coupled architecture, taking into consideration international standards on the data transformation and information exchange, as well as the potential interoperability with third party systems. The individual components communicate with each other through well-defined APIs over REST Web Services, while JSON data exchange schema has been adopted.

For example, the Map Visualisation Tools can consume WFS services residing in external systems and the frontend elements of the Marketplace make use of RESTful services to communicate with the database.

4.4 Usability

Usability refers to the interaction between the target users and the platform interface. Testing the user interface of the platform will verify the correct communication of the users with the underlying software. Testing the user environment will furthermore ensure that the interface elements of the platform operate as they should and that they are in agreement with the technical and functional requirements.

Testing usability involves a step-to-step navigation among all the pages, fields and functionalities of the platform. These exhaustive tests reveal the perception of the platform's functionalities by the users and their understanding of the current status of the platform. Therefore, the evaluation of the platform's usability will not be analysed in this deliverable since there is a separate work package that deals with this process. WP6 includes a series of four pilot evaluation cycles which will provide an extensive feedback on the OTN platform usability performance.

4.5 Security

4.5.1 Confidentiality

The OTN Hub defines role-based access to specific tasks and operations of the platform. The different users of the OTN platform should have different access privileges. Specific test cases were created and executed in order to verify information confidentiality as presented in paragraph 4.1.2**Error! Reference source not found.**. In these tests, the platform services are invoked deliberately in an improper way. The platform receives the malformed requests, e.g., wrong user credentials, request to access private data, etc., records these attempts and denies access. This behaviour ensures that the platform can handle malformed requests, which can be either malicious or accidentally wrong efforts to gain access to unauthorized resources or services.

4.6 Maintainability

4.6.1 Modularity

The OTN Hub has been built, based on the SOA-oriented modular architecture, thus the different modules follow a loosely coupled approach to connect to each other. The core modules of the platform have been implemented as java portlets using the standard JSR-286 portlet configuration. They have been already used in other platforms (e.g. Plan4business[2]) with success proving their re-usability which comes as a result of a number of adaptations according to the needs of OTN.

[2] http://www.plan4business.eu/ A service platform for aggregation, processing and analysis of urban and regional planning data

Moreover, implementing these portlets following an object-oriented approach, having a proper documentation and software packaging, as well as complete installation instructions increases the maintainability of the code and allows for the re-use of the existing modules and their re-use in future enhancements of the platform's functionality.

4.7 Portability

4.7.1 Adaptability

The implementation technologies of the OTN platform ensure the performance of the components and tools of the OTN Hub and provide high capability of adaptation to various installation and production environments. The platform is developed in a development server which has been set up in the integrator's infrastructure and is accessible to all the technical partners. Upon finishing the first cycle of development, the Hub was ported to the production environment which is a different server machine without any issues.

The personalised instances of the Hub will be rolled out in the near future, as will be described in the coming deliverable D6.2 'Personalised Instances of the Hub". This process will also test the portability of the solution to different hosting environments as the ones of the pilot cities.

Finally, the Hub functionalities have been tested with all the modern browsers in order to ensure adaptability in terms of client side technology used to access the various Hub components and tools.

5 Conclusion

The current deliverable includes the methodology for the technical testing of the OTN Platform components, the test plan and the results of the tests. The methodology is based on the Software Assurance Process defined in the BS ISO/IEC 25010:2011 standard, from which the OTN Quality Model was defined.

After describing the objectives, phases and team that will run the tests, the Test Plan was executed and the produced results were recorded. The Functional Suitability and the performance Efficiency were tested through test cases and extensive web tests respectively. Observation tests and log analysis provided the results for the rest of the software characteristics of the OTN Quality Model.

The tests showed that the components of the OTN platform fulfil the user requirements and the quality metrics in a high degree. A number of adjustments and updates will take place during the next period while the personalised instances of the OTN Hub are defined and deployed. These adaptations will yield the need to test the quality of the produced results from a technical as well as a functional perspective. The methodology and test plan defined in the current document will be the means to ensure that the end product will meet the desired requirements and quality without compromising the OTN Quality Model characteristics.

6 References

[1] J.A. McCall, P.K. Richards, and G.F. Walters, “Factors in Software Quality,” vols. 1, 2, and 3, AD/A-049-014/015/055, Nat'l Tech. Information Service, Springfield, Va., 1977.

[2] Barry W. Boehm, “ A spiral model of software development and enhancement”, TRW, Defence Systems Group

[3] ISO/IEC 9126-1:2001, Software engineering -- Product quality -- Part 1: Quality model

[4] BS ISO/IEC 25010:2011, Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models

[5] OTN Consortium, “Deliverable D2.4: Co-Design Workshop reports”, June, 2014

[6] OTN Consortium, “Deliverable D2.5: Architecture Blueprint And Hub Technical Specification”, June, 2015 (Resubmission)

[7] IEEE Standard 829-1998, IEEE Standard for Software Test Documentation, 1998.