OGC Testbed-13

Workflows ER

Table of Contents

Publication Date: 2018-01-08

Approval Date: 2017-12-07

Posted Date: 2017-10-30

Reference number of this document: OGC 17-029r1

Reference URL for this document: http://www.opengis.net/doc/PER/t13-NG009

Category: Public Engineering Report

Editor: Benjamin Pross, Christoph Stasch

Title: OGC Testbed-13: Workflows ER

OGC Engineering Report

COPYRIGHT

Copyright © 2018 Open Geospatial Consortium. To obtain additional rights of use, visit http://www.opengeospatial.org/

WARNING

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

LICENSE AGREEMENT

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD. THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER’S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR’s sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize you or any third party to use certification marks, trademarks or other special designations to

indicate compliance with any LICENSOR standards or specifications.

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

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

Chapter 1. Summary

This Engineering Report (ER) addresses the development of a consistent, flexible, adaptable workflow that will run behind the scenes. A user should be able to discover existing workflows via a catalog and execute them using their own datasets. An expert should be able to create workflows and to publish them. Previous OGC Testbed initiatives investigated workflows in the geospatial domain:

- OWS 3 Imagery Workflow Experiments
- OWS 4 WPS IPR Workflow descriptions and lessons learned
- OWS 4 Topology Quality Assessment Interoperability Program Report
- OWS 5 Data View Architecture Engineering Report
- OWS 6 Geoprocessing Workflow Architecture Engineering Report

These initiatives mostly favored Business Processing Execution Language (BPEL) as the workflow execution language. More recent studies ([6], [7]) were performed using BPMN as a means for describing and executing workflows comprised of OGC Web services. This ER will give an overview about existing approaches to compose and execute geospatial workflows and will describe the approach taken in Testbed-13, taking into account security aspects.

Parts of this work have been funded by the COLABIS and Mudak-WRM projects funded by the German Federal Ministry of Education and Research under grant agreement numbers 03G0852C and 02WGR1431C.

1.1. Requirements

As stated above, the workflow is required to perform the following tasks:

- Gather data
- Check ellipsoid/projection
- Check data quality
- Run conflation
- Deliver data

A planned workflow (i.e. a workflow constructed ahead of time) shall be developed combining the conflation Web Processing Service (WPS) 2.0 and data quality WPS 2.0 implemented in Testbed-12. In order to secure the workflow, the following items need to be taken into account:

1. OGC Services and content encodings
2. Fine grained access control implemented through an Attribute Based Access Control (ABAC) infrastructure
3. Services and content are provided by more than one organization utilizing more than one security environment
4. Services are not accredited to the same level of trust

5. Some content may require a higher level of protection than some services are accredited to provide
6. All actions must be logged and associated with the user who initiated the service chain
7. Some of the content is sensitive (Personal Identifiable Information). Unauthorized release of this data must be avoided.

1.2. Key Findings and Prior-After Comparison

The workflow package in Testbed-13 has the goal of developing a consistent, flexible, adaptable workflow that will run behind the scenes, is described in a standardized format (BPMN) and could be shared to and executed by other users. In a nutshell, the workflow aims to automate conflation of road datasets and pre-processing steps (coordinate transformation, data quality checks). The processing steps are implemented as WPS processes and the data is passed by reference to data servers that include Web Feature Services (WFS) and Web Coverage Services (WCS) between the different processing steps.

The work and experiments that were done showed that it is possible to use OGC Web services in workflows that are protected by different security mechanisms, i.e. X.509 certificates and OAuth. Concepts of combining mainstream security with OGC Web services that have been developed during past testbeds and that are currently discussed in OGC have been proven to work in geospatial workflows. It has been shown that workflows can be created, wrapped in WPS processed and made discoverable in catalog services using current standards and technologies.

1.3. What does this ER mean for the Working Group and OGC in general

Workflows that transform geospatial datasets to derived geospatial information are usually composed of several processing steps. These steps may be provided by different WPS endpoints implementing different process profiles. Up to now, a common approach for defining, sharing and executing complex workflows (e.g. those composed of processes from different service providers) has been non-existent. This ER describes the approaches taken in OGC Testbed-13 to address these issues. Besides the general question of how to compose, share, and execute a workflow consisting of several WPS processes, different approaches for dealing with security issues such as identity mediation or delegation are also described in this ER.

Several approaches regarding workflow composition have been discussed in the Workflow DWG, [6] being the most recent document. However, no common approach has been formalized yet. This ER describes a further proof of concept of the approach described by [6], which could lead to the creation of a best practice document regarding geospatial workflow composition and execution.

1.4. Document contributor contact points

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

Table 1. Contacts

Name	Organization
Benjamin Proß (editor)	52°North GmbH
Christoph Stasch (editor)	52°North GmbH
Dimitar Misev	Rasdaman
Mark Lawrence	Compusult

1.5. Future Work

This ER shows that BPMN can be used to compose, discover and execute secured workflows in the geospatial domain. In the future, a general approach for BPMN engines to communicate with OGC Web services should be investigated, including the treatment of inputs and outputs. Also, the encoding of security aspects in BPMN should be defined. A transactional extension for WPS 2.0 and the extension of the WPS 2.0 StatusInfo document to add more information about the process status should be discussed in the WPS 2.0 SWG.

1.6. Foreword

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

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

Chapter 2. References

The following normative documents are referenced in this document.

- OGC 06-121r9, OGC® Web Services Common Standard [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2]
- OGC 14-065, OGC® WPS 2.0 Interface Standard Corrigendum 1 [http://docs.opengeospatial.org/is/14-065/14-065.html]
- OGC 09-025r2, OGC® Web Feature Service 2.0 Interface Standard - With Corrigendum [http://docs.opengeospatial.org/is/09-025r2/09-025r2.html]
- OGC 07-006r1, OpenGIS Catalogue Service Implementation Specification [http://portal.opengeospatial.org/files/?artifact_id=20555]
- OGC 09-110r4, OGC® WCS 2.0 Interface Standard- Core: Corrigendum [https://portal.opengeospatial.org/files/09-110r4]

Chapter 3. Terms and definitions

For the purposes of this report, the definitions specified in Clause 4 of the OWS Common Implementation Standard OGC 06-121r9 [https://portal.opengeospatial.org/files/?artifact_id=38867&version=2] shall apply.

Chapter 4. Abbreviated terms

- API Application Program Interface
- AWS Amazon Web Services
- BPEL Business Process Execution Language (also WS-BPEL)
- BPMN Business Process Model and Notation
- CSW Catalog Service for the Web
- CRS Coordinate Reference System
- ER Engineering Report
- OGC Open Geospatial Consortium
- OWS OGC Web Service
- WPS Web Processing Service
- WFS Web Feature Service
- WCS Web Coverage Service

Chapter 5. Overview

This Engineering Report describes the results of the OGC Testbed-13 Workflow thread.

In the following sections, the scenario is described, followed by an overview on workflow description formats and execution engines. The developed/used components are described in sections 8 - 11. Section 12 describes the security aspects and section 13 summarizes the recommendations.

Chapter 6. Scenario

The overall goal is to create a conflation workflow consisting of several (pre-)processing steps needed to conflate road datasets of different quality. Conflation, also known as map matching or merging is the process of combining two datasets based on a set of rules to produce a more complete dataset. One example is the combination of authoritative data, e.g. authoritative road data, with crowd-sourced data, e.g. OpenStreetMap (OSM). The authoritative data is collected using certain guidelines and must follow strict requirements to ensure data quality. This leads to a smaller frequency of updates. Crowd-sourced data on the other hand, can be collected by virtually everyone. In this case, the quality of the data is monitored by the community and usually has a high volume of updates and therefore much more prone to error. As a result, the data is usually of lower quality.

The general conflation workflow consists of three major steps as shown in the figure below.

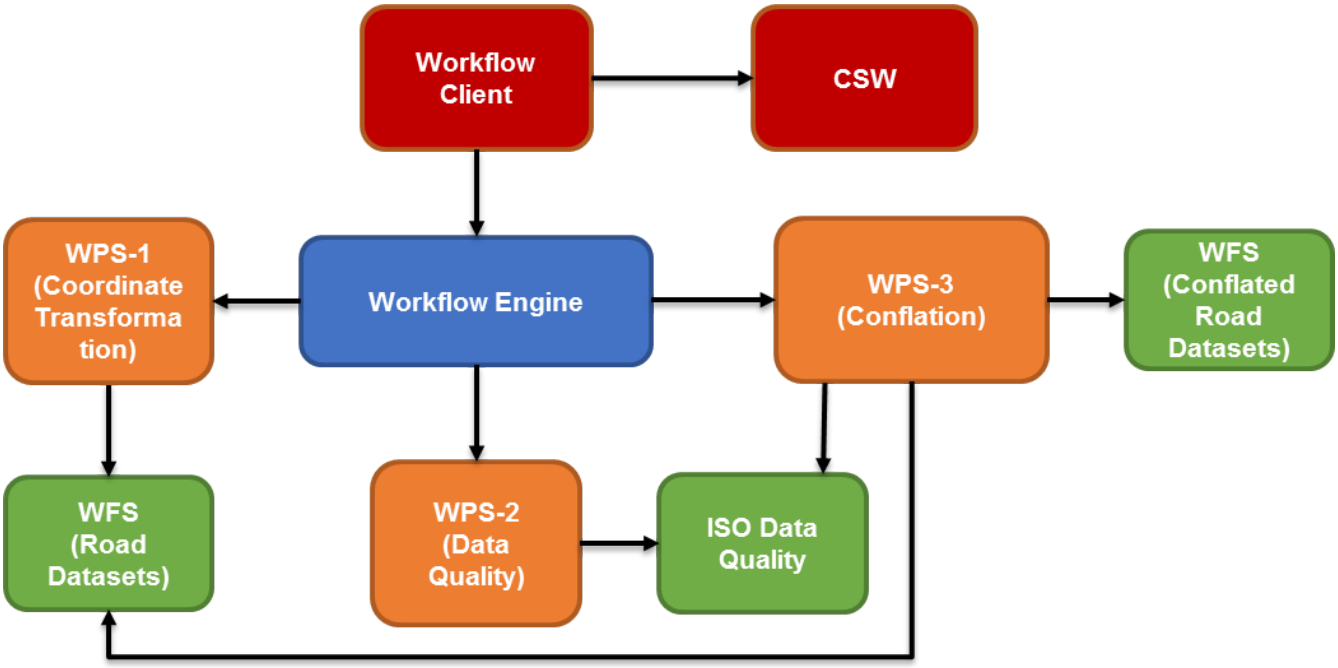

Figure 1. Workflow Overview

The first step is to check whether the road datasets are in the same coordinate reference system. If not, a coordinate transformation will be executed (WPS-1). The next step is to check the data quality of the road datasets (WPS-2). Thereby, the mean minimum distance between the road datasets and a set of control points is checked. The results are returned as ISO data quality. The last step in the workflow is the actual conflation (WPS-3) that is an extended version of the Conflation WPS developed in Testbed-12 that uses the Hootenanny conflation software [1: Hootenanny is an open source conflation tool developed to facilitate automated and semi-automated conflation of critical GEOINT features in the topographic domain. More information can be found at: https://github.com/ngageoint/hootenanny] at the backend.

The workflow client can be used for two purposes:

- **Workflow composition:** An *expert user* defines the workflow by discovering suitable WPS processes in a catalog service and composing them in a BPMN document.
- **Workflow execution:** A *workflow user* can use the client to execute a workflow process.

The following two sequence diagrams illustrate the general sequences for both use cases.

6.1. Workflow Creation

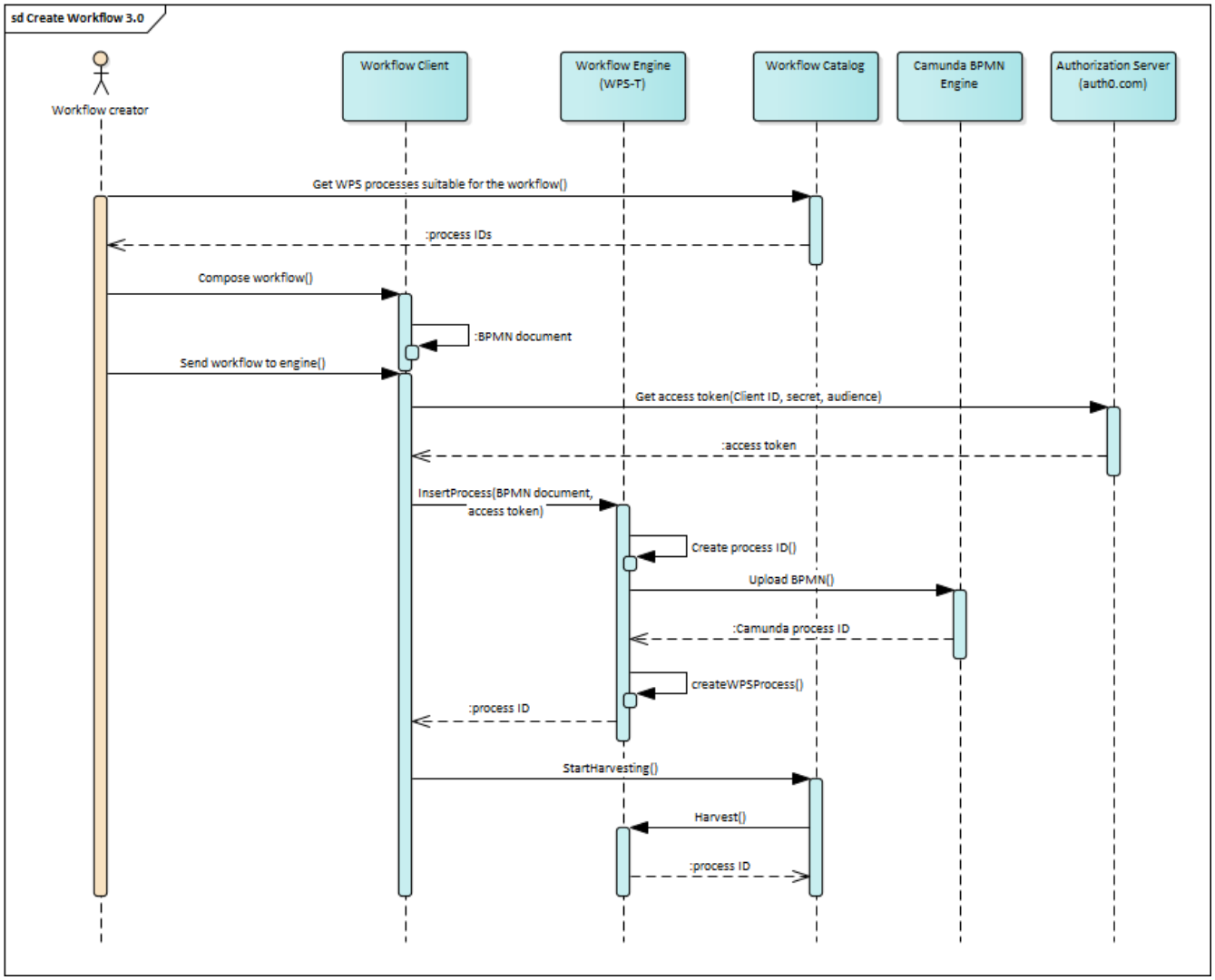

Figure 2. Workflow Creation

Step 1:

- The expert queries a catalog for processes suitable for the workflow.

Step 2:

- The expert composes the workflow using the workflow client (BPMN editor).

Step 3:

- The expert then selects the option in the client to send the workflow defined in the BPMN file to the workflow engine.

Step 4:

- In order to process such a request, the workflow engine requires a security access code. The workflow client (WPS client) sends a request to the OAuth Security Service to get an access code and then includes it as part of the request.

- The workflow engine creates a new process identifier and uploads the BPMN to the underlying BPMN engine. The BPMN engine returns an internal process identifier that can be used to start the process. A wrapper process is created in the workflow engine WPS. The WPS then returns the process identifier of the wrapper process.

Step 5:

- The workflow client then contacts the CSW and instructs it to harvest the new process in the workflow Engine WPS.

6.2. Workflow Execution

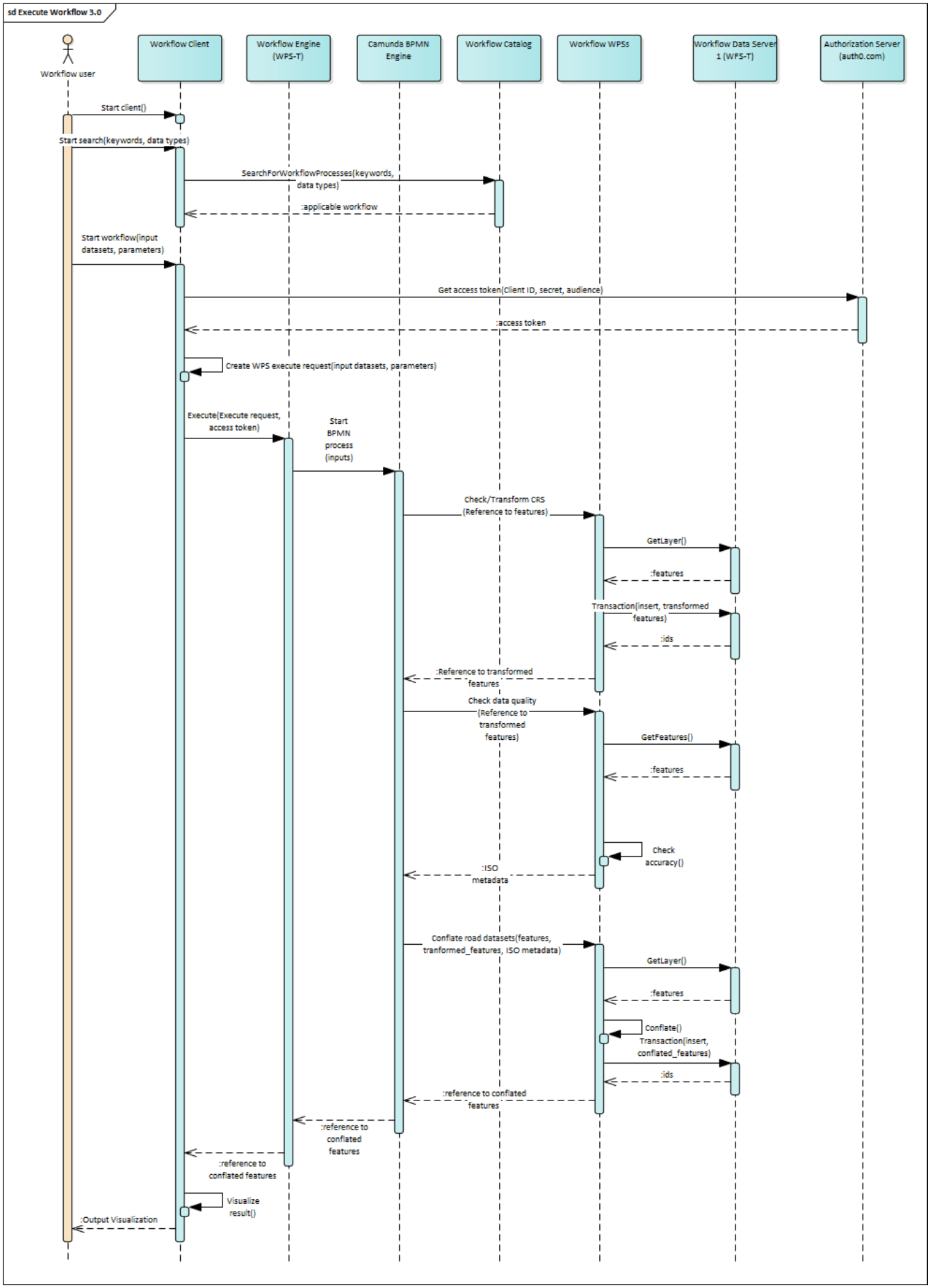

Figure 3. Workflow Execution Sequence

Note: Only the security aspects of the workflow client and -engine interaction are shown here. For details about the security aspects of the workflow in general, please refer to Workflow Security

Step 1:

- The user launches the workflow client and searches the catalog for workflows (i.e. WPS process descriptions) using keywords and data types that match their current needs.

Step 2:

- The user finds the applicable conflation WPS process and provides the list of datasets and other input parameters for processing.

Step 3:

- In order to process an WPS Execute request, the workflow engine WPS requires a security access code. The workflow client (WPS client) sends a request to the OAuth Security Service to get an access code.

Step 4:

- The WPS client then constructs the applicable WPS Execute request (including the security access code) and sends it to the workflow engine WPS.

Step 5:

- The workflow engine now invokes the first WPS by passing the references to the road datasets and a target CRS to the WPS. If the CRS is different from the target CRS, the coordinates are transformed.

Step 6:

- The data quality is checked using the Data Quality WPS. A coverage is requested from a WCS and quality checks are performed.

Step 7:

- If the quality is sufficient the actual conflation is executed by invoking another WPS process. The WPS then inserts the resulting dataset in a WFS-T (or WCS-T, depending on the result type) and returns a reference to it. The reference is then forwarded to the client and may be visualized in a map.

6.3. Datasets

The workflow is run for conflating road datasets in two regions: New York City in the United States of America and Za'atari in Jordan. The datasets available are described in the following two subsections.

New York City:

- The National Map
- OSM
- Digital Globe satellite images

Za'atari Datasets:

- UN
- OSM
- Digital Globe satellite images

6.4. Technology Integration Experiments

The following tables list the Technology Integration Experiments (TIEs) that were performed in the workflow scenario. See TIE results for additional information about the TIEs.

Involved components:

Abbreviation	Component name	Deliverable
WCL	Workflow Client	NG136
WCA	Workflow Catalog	NG135
WEN	Workflow Engine	NG131
CTP	Coordinate Transformation Process	NG130
DQP	Data Quality Check Process	NG130
COP	Conflation Process	NG130
WFS	Workflow WFS	NG132
ABU	Amazon Bucket	N/A
IBR	Internet Browser	N/A

TIEs:

TIE between	Security	Use Case	Successful
WEN - CTP	x.509, OAuth2: Client Credentials	Dominating Privileges	Y (04.10.2017)
WEN - DQP	x.509, OAuth2: Client Credentials	Tunneling Proxies	Y (04.10.2017)
WEN - COP	x.509, OAuth2: Client Credentials	N/A	Y (04.10.2017)
CTP - WFS	x.509, OAuth2: Client Credentials	Dominating Privileges	Y (04.10.2017)
DQP - WFS	x.509, OAuth2: Client Credentials	N/A	Y (04.10.2017)
COP - WFS	x.509, OAuth2: Client Credentials	N/A	Y (04.10.2017)
COP - ABU	x.509, IAM	Identity Mediation	Y (04.10.2017)
WCL - WEN InsertProcess	OAuth2: Client Credentials	N/A	Y (20.09.2017)

WCL - WEN Execute	OAuth2: Client Credentials	N/A	Y (04.10.2017)
WCL - CTP	N/A	N/A	Y (20.09.2017)
WCL - DQP	N/A	N/A	Y (20.09.2017)
WCL - COP	N/A	N/A	Y (20.09.2017)
WCL - WCA	N/A	N/A	Y (20.09.2017)
IBR - CTP	OAuth2: Authorization Code	N/A	Y (05.10.2017)

Chapter 7. Overview on Workflow Description Formats and Execution Engines

This chapter will give an overview of different workflow description formats and execution engines that were used to comprise geospatial workflows.

7.1. Workflow Description Formats

In this section, two formats are identified that have been used to describe workflows in the geospatial domain, the Web Services Business Process Execution Language (WS-BPEL, BPEL in the following) and the Business Process Model and Notation (BPMN).

BPEL was released by the Organization for the Advancement of Structured Information Standards (OASIS). The current version is 2.0. It is used to orchestrate Web services, i.e. to specify executable processes and how messages are exchanged between them. It was designed to be used together with the Web Services Description Language (WSDL) 1.1 [2: http://www.w3.org/TR/wsdl]. There is no standard graphical notation for BPEL, although there is a mapping from BPMN to BPEL 1.1. Former OGC Testbeds investigated the use of BPEL for the orchestration of OGC Web services. Also, a number of research papers were published dealing with this topic, see [2],[4],[5] and [10]. [8] has proposed a transactional profile for the Web Processing Service based on BPEL.

BPMN is a more recent standard for the graphical representation of business process models and also for describing the execution semantics. The use of BPMN for the orchestration of OGC Web services was investigated in recent studies (see [6], [7]). There is a growing number of engines that support BPMN, e.g. Orchestra [3: http://orchestra.ow2.org/], Activiti [4: http://www.activiti.org/], Camunda [5: http://camunda.org/] or jBPM [6: http://www.jbpm.org/]. BPMN was approved by the International Organization for Standardization (ISO) as ISO/IEC 19510:2013.

7.2. Workflow Execution Engines

In the following section, different workflow engines are described and approaches are shown that used them for orchestrating OGC Web services.

7.2.1. JBOSS jBPM

Open-source workflow engine released by the JBoss company. Its core is "a light-weight, extensible workflow engine written in pure Java" [7: https://docs.jboss.org/jbpm/release/7.3.0.Final/jbpm-docs/html_single/#_what_is_jbpm]. Several tools provide additional functionality like workflow modelling and monitoring. [6] used jBPM for their research.

Version	Programming language	BPMN 2.0 supported	License
7.3.0	Java	Yes	Apache License v2.0

7.2.2. Taverna

Open-source workflow management system started by the myGrid project [8: http://www.mygrid.org.uk/] and currently an Apache Incubator project. Taverna is used in different scientific domains like bioinformatics or astronomy. Workflows can be created and executed using a desktop client/command line interface or in the web browser. [3] described the composition of geospatial workflows using WSDL and Taverna. [5] examined Taverna orchestration of geospatial web services.

Version	Programming language	BPMN 2.0 supported	License
3.1.0 (Command Line Tool)	Java	No (SCUFL2)	Apache License v2.0

7.2.3. Amazon Simple Workflow Service

The Amazon Simple Workflow Service (Amazon SWF) is used to coordinate work tasks that can run asynchronously in distributed systems, e.g. Amazon Elastic Compute Cloud. Geospatial workflows in Cloud environments, such as Amazon Web services, have been investigated by previous OGC Testbeds and in research (e.g. [1], [9]). However, past approaches were focused on the composition of workflows using Web services deployed in Cloud environments, not the use of Amazon SWF itself for the composition.

Version	Programming language	BPMN 2.0 supported	License
1.11.208 (Java SDK)	Java	No (Flow Framework)	Apache License v2.0

7.2.4. Camunda

Open-source platform for business process management. I.a., Camunda focuses on a more complete coverage of BPMN 2.0. Camunda offers a Java and REST API to manage workflows and a modeler to create them. Previous experiences with Camunda showed an easy to setup and robust engine.

Version	Programming language	BPMN 2.0 supported	License
7.7.0	Java	Yes	Apache License v2.0

7.2.5. Windows Workflow Foundation

Windows Workflow Foundation enables the implementation of processes in Microsoft .NET. Provides an API, a workflow engine and a designer. It is possible to access Web services with Windows Workflow Foundation. However, literature shows no evidence of orchestration of geospatial workflows using Windows Workflow Foundation.

Version	Programming language	BPMN 2.0 supported	License
4.5	.NET (C#, VB.NET)	No (XAML)	MIT

Because of its support of BPMN and the ease in setup and execution we have chosen to use Camunda in this experiment.

Chapter 8. Testbed-13 Workflow Engine

The Testbed-13 Workflow Engine allows execution of workflows consisting of several WPS processes defined in a BPMN document.

8.1. Concept

The workflow engine provides two interfaces:

- **InsertProcess**: This endpoint can be used to create new WPS processes for workflow compositions by posting a BPMN document to this endpoint using the HTTP POST operation.
- **WPS 2.0 interface for executing workflow**: To execute the workflow, the engine will rely on the WPS 2.0 specification, i.e. for each workflow available, there will be a WPS process allowing to execute the workflow. Therefore, the execution engine supports the default operations of the WPS including:
  - GetCapabilities
  - DescribeProcess
  - Execute
  - GetStatus (for execution in asynchronous mode)
  - GetResult

8.2. Implementation

The implementation is based on the 52°North WPS framework (http://52north.org/wps) and Camunda. A transactional WPS interface (WPS-T) will be used to create new processes based upon BPMN documents.

The following image shows the components involved in the creation of a workflow WPS process:

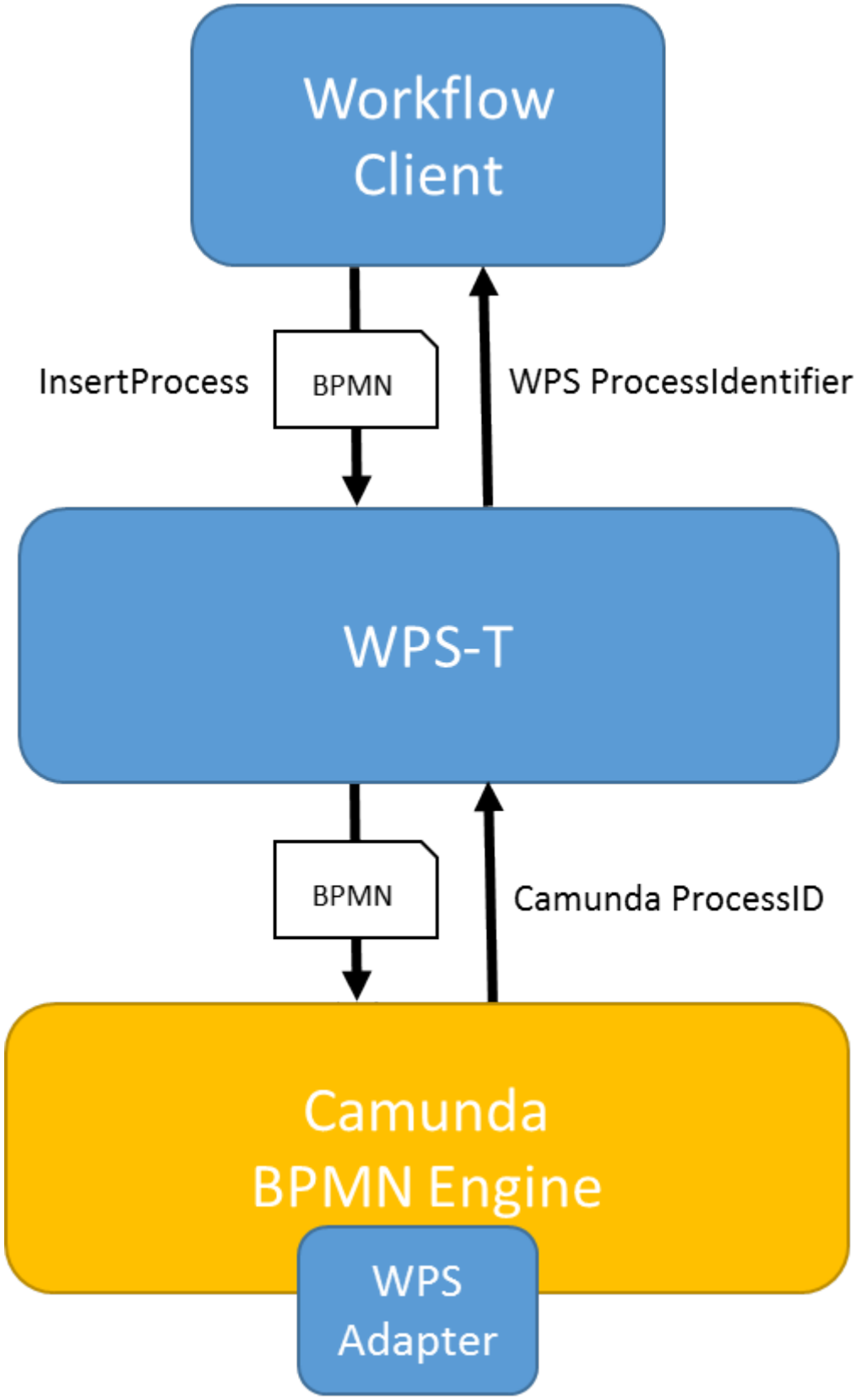

Figure 4. Components involved in workflow WPS process creation

The BPMN is sent to the WPS-T using the new InsertProcess operation. The current time in milliseconds is attached to the original process identifier. The BPMN is forwarded to the Camunda BPMN Engine that offers a REST interface. The engine checks the BPMN and returns a deployment identifier to the WPS-T. The WPS-T creates a process and returns the new process identifier to the client.

Below is an example of the InsertProcess request:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:InsertProcess xmlns:wps="http://www.opengis.net/wps/2.0"
    xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:xlink=
"http://www.w3.org/1999/xlink"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://tb12.dev.52north.org:80/workflow-wps/static/schemas/wpsInsertProcess.xsd"
    service="WPS" version="2.0.0">

    <wps:ProcessSpecification id="ConflationWorkflow">
        <wps:ProcessSpecificationAsValue
            mimeType="application/x-bpmn">
            <bpmn2:definitions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:camunda="http://camunda.org/schema/1.0/bpmn"  xmlns:bpmn2=
"http://www.omg.org/spec/BPMN/20100524/MODEL" xmlns:bpmndi=
"http://www.omg.org/spec/BPMN/20100524/DI" xmlns:dc=
"http://www.omg.org/spec/DD/20100524/DC" xmlns:di=
"http://www.omg.org/spec/DD/20100524/DI" xmlns:ext="http://org.eclipse.bpmn2/ext"
xmlns:xs="http://www.w3.org/2001/XMLSchema" id="Definitions_1" exporter=
"org.eclipse.bpmn2.modeler.core" exporterVersion="1.3.1.Final-v20161006-1425-B58"
targetNamespace="http://org.eclipse.bpmn2/default/process">
            ....
            </bpmn2:definitions>
        </wps:ProcessSpecificationAsValue>
    </wps:ProcessSpecification>

</wps:InsertProcess>
```

(See Process specification for the full BPMN document.) The BPMN is extracted and wrapped in a multipart message:

```
--3b30bbd9-ae0e-4e3f-a6cd-fb57db0fe400
Content-Disposition: form-data; name="deployment-name"

ConflationWorkflow_1507125390489
--3b30bbd9-ae0e-4e3f-a6cd-fb57db0fe400
Content-Disposition: form-data; name="enable-duplicate-filtering"

true
--3b30bbd9-ae0e-4e3f-a6cd-fb57db0fe400
Content-Disposition: form-data; filename="ConflationWorkflow_1507125390489.bpmn";
name="data"

<bpmn2:definitions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:camunda="http://camunda.org/schema/1.0/bpmn"
xmlns:bpmn2="http://www.omg.org/spec/BPMN/20100524/MODEL"
xmlns:bpmndi="http://www.omg.org/spec/BPMN/20100524/DI"
xmlns:dc="http://www.omg.org/spec/DD/20100524/DC"
xmlns:di="http://www.omg.org/spec/DD/20100524/DI"
xmlns:ext="http://org.eclipse.bpmn2/ext" xmlns:xs="http://www.w3.org/2001/XMLSchema"
id="Definitions_1" exporter="org.eclipse.bpmn2.modeler.core"
exporterVersion="1.3.1.Final-v20161006-1425-B58"
targetNamespace="http://org.eclipse.bpmn2/default/process">

....

</bpmn2:definitions>

--3b30bbd9-ae0e-4e3f-a6cd-fb57db0fe400--
```

This is sent to the REST-endpoint of the Camunda engine:

```
http://hostname.org/engine-rest/engine/default/deployment/create
```

The response contains the deployment-id and additional information:

```
{
    "links": [
        {
            "method": "GET",
            "href": "http://hostname.org/engine-
rest/engine/default/deployment/45d60c1f-a99b-11e7-ac41-000c2988332c",
            "rel": "self"
        }
    ],
    "id": "45d60c1f-a99b-11e7-ac41-000c2988332c",
    "name": "ConflationWorkflow_1507125390489",
    "source": null,
    "deploymentTime": "2017/10/04T14:56:32",
    "tenantId": null
}
```


The new process will appear on the Camunda dashboard. Also, if available, the BPMN diagram can be visualized as shown in the following figure (note that some elements, e.g. inputs and outputs, are not visualized by Camunda):

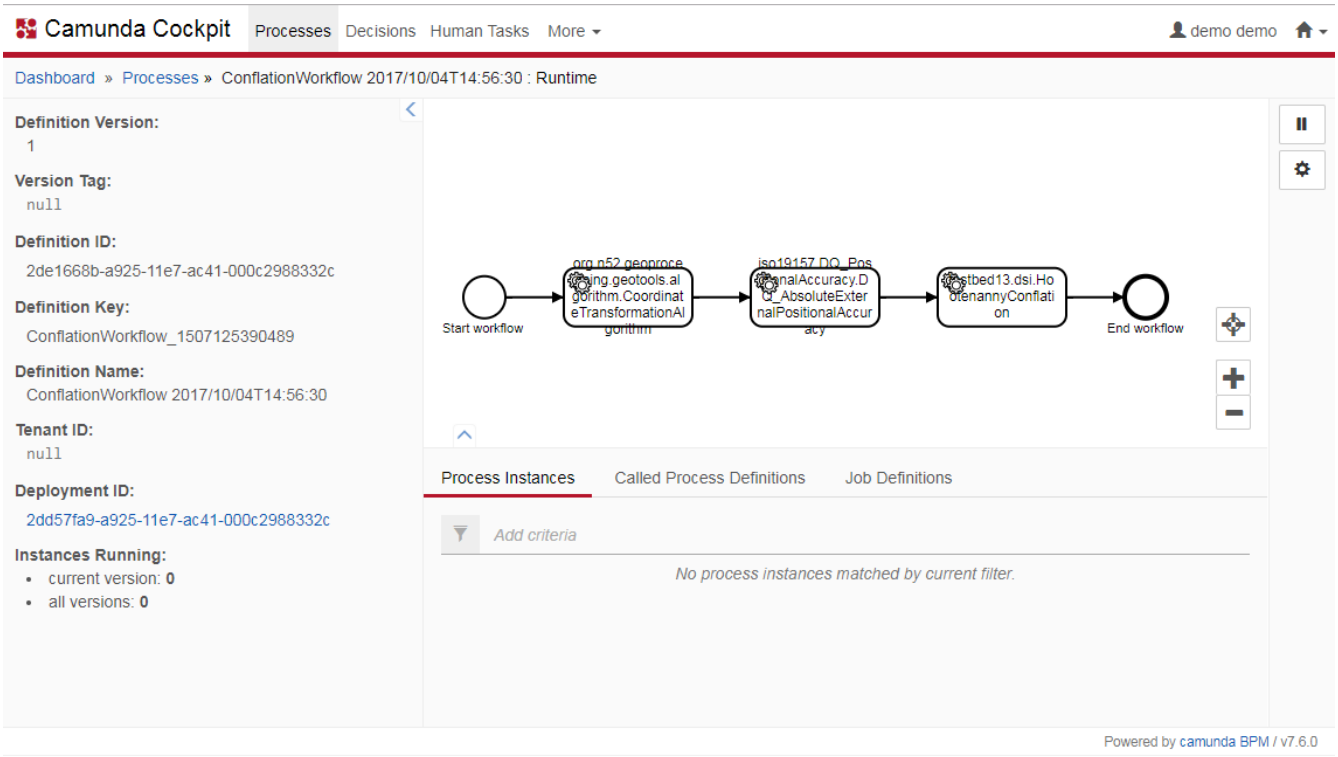

Figure 5. Workflow visualized in Camunda dashboard

After a successful upload to Camunda, the WPS sends a InsertProcessinfo response, as shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:InsertProcessInfo xmlns:wps="http://www.opengis.net/wps/2.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
"http://www.opengis.net/wps/2.0 http://tb12.dev.52north.org:80/workflow-
wps/static/schemas/wpsInsertProcess.xsd">
    <wps:processID>testbed13.dsi.ConflationWorkflow_1507125390489</wps:processID>
</wps:InsertProcessInfo>
```

The process description of the newly created process is below:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:ProcessOfferings xmlns:wps="http://www.opengis.net/wps/2.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xmlns:ows="http://www.opengis.net/ows/2.0"
xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd">
  <wps:ProcessOffering processVersion="1.0.0" jobControlOptions="sync-execute async-
execute" outputTransmission="value reference">
    <wps:Process>
      <ows:Title>Conflation workflow process 2017/10/04T14:56:30</ows:Title>
      <ows:Abstract>This process encapsulates a BPMN workflow that will perform data
quality checks, transforms coordinates if needed and finally conflates
datasets.</ows:Abstract>
      <ows:Identifier>testbed13.dsi.ConflationWorkflow_1507125390489</ows:Identifier>
      <wps:Input minOccurs="2" maxOccurs="4">
        <ows:Title>Datasets</ows:Title>
        <ows:Abstract>Vector datasets for conflation</ows:Abstract>
        <ows:Identifier>datasets</ows:Identifier>
        <wps:ComplexData xmlns:ns="http://www.opengis.net/wps/2.0">
          <ns:Format default="true" mimeType="application/x-zipped-shp"/>
          <ns:Format mimeType="application/x-openstreetmap+xml"/>
          <ns:Format mimeType="application/x-zipped-shp" encoding="base64"/>
          <ns:Format mimeType="application/x-zipped-gdb"/>
          <ns:Format mimeType="application/x-zipped-gdb" encoding="base64"/>
        </wps:ComplexData>
      </wps:Input>
      <wps:Output>
        <ows:Title>Conflated dataset</ows:Title>
        <ows:Identifier>conflated-dataset</ows:Identifier>
        <wps:ComplexData xmlns:ns="http://www.opengis.net/wps/2.0">
          <ns:Format default="true" mimeType="application/x-zipped-shp"/>
          <ns:Format mimeType="application/x-openstreetmap+xml"/>
          <ns:Format mimeType="application/x-zipped-shp" encoding="base64"/>
          <ns:Format mimeType="text/xml" schema=
"http://schemas.opengis.net/gml/3.1.1/base/feature.xsd"/>
          <ns:Format mimeType="application/vnd.geo+json"/>
          <ns:Format mimeType="application/x-zipped-gdb"/>
          <ns:Format mimeType="application/x-zipped-gdb" encoding="base64"/>
        </wps:ComplexData>
      </wps:Output>
    </wps:Process>
  </wps:ProcessOffering>
</wps:ProcessOfferings>
```

After the WPS process was created, it can be executed by the client. The following image shows the involved components in executing a workflow WPS process:

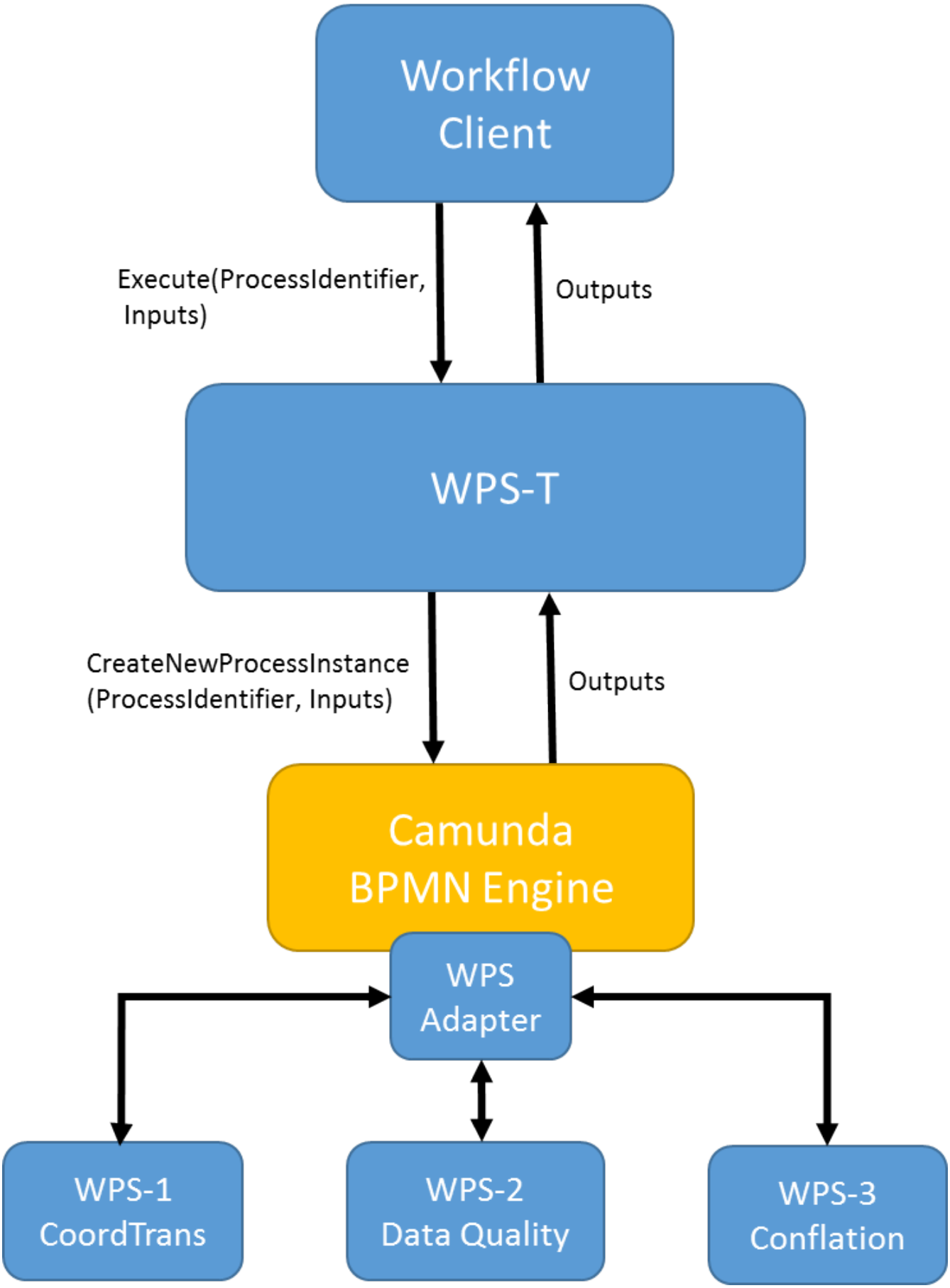

Figure 6. Components involved in workflow WPS process execution

The client sends a standard execute-request to the WPS-T based on the process description, e.g.:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute xmlns:wps="http://www.opengis.net/wps/2.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xmlns:ows="http://www.opengis.net/ows/2.0"
xmlns:xlin="http://www.w3.org/1999/xlink" xsi:schemaLocation=
"http://www.opengis.net/wps/2.0 http://schemas.opengis.net/wps/2.0/wps.xsd" service=
"WPS" version="2.0.0" response="document" mode="async">
  <ows:Identifier>testbed13.dsi.ConflationWorkflow_1507125390489</ows:Identifier>
  <wps:Input id="datasets">
    <wps:Reference mimeType="text/xml; subtype=gml/3.1.0" schema=
"http://schemas.opengis.net/gml/3.1.0/base/feature.xsd" xlin:href=
"https://tb12.dev.52north.org/security-proxy/service/wfs?service=WFS&amp;
version=1.0.0&amp;request=GetFeature&amp;typeName=tb13:tnm-manhattan-streets-
wgs84&amp;outputFormat=gml3"/>
  </wps:Input>
  <wps:Input id="datasets">
    <wps:Reference mimeType="text/xml; subtype=gml/3.1.0" schema=
"http://schemas.opengis.net/gml/3.1.0/base/feature.xsd" xlin:href=
"https://tb12.dev.52north.org/security-proxy/service/wfs?service=WFS&amp;
version=1.0.0&amp;request=GetFeature&amp;typeName=tb13:lion-manhattan-streets
&amp;outputFormat=gml3"/>
  </wps:Input>
  <wps:Input id="datasets">
    <wps:Reference xlin:href="https://tb12.dev.52north.org/aws-
proxy/service/aws?service=aws&amp;url=https://s3-eu-west-1.amazonaws.com/testbed13-
osm/manhattan/osm-manhattan-roads.osm" mimeType="application/x-openstreetmap+xml"/>
  </wps:Input>
  <wps:Output id="conflated-dataset" mimeType="text/xml" schema=
"http://schemas.opengis.net/gml/3.1.1/base/feature.xsd" transmission="reference"/>
</wps:Execute>
```

The process identifier and inputs are extracted and forwarded to the Camunda REST-endpoint to start a new instance of the BPMN process.

```
http://hostname.org/engine-rest/process-
definition/key/ConflationWorkflow_1507125390489/submit-form
```

```
{
  "variables": {
    "dataset1": {"value":"https://tb12.dev...","type":"String"},
    "dataset2":{"value":"https://tb12.dev...","type":"String"},
    "dataset3":{"value":"https://tb12.dev...","type":"String"}
  }
}
```

A WPS adapter deals with the (secured) communication with WPS internally in the Camunda BPMN Engine. After a successful execution of the BPMN process, the final output (link to resources) is sent to the WPS-T and from there forwarded to the client in a standard WPS ResultDocument.

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Result xmlns:wps="http://www.opengis.net/wps/2.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xmlns:xlin="http://www.w3.org/1999/xlink"
xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd">
    <wps:JobID>2206815e-4824-4b19-a790-68be5986b4c4</wps:JobID>
    <wps:Output id="conflated-dataset">
        <wps:Reference schema="http://schemas.opengis.net/gml/3.1.1/base/feature.xsd"
mimeType="text/xml" xlin:href=
"http://tb12.dev.52north.org/SecurityProxy/service/wps?request=GetOutput&amp;amp;version=2
.0.0&amp;amp;service=WPS&amp;amp;id=2206815e-4824-4b19-a790-68be5986b4c4conflated-
dataset.cf30b672-7df2-405f-a5ee-8f6a5c558c88"/>
    </wps:Output>
</wps:Result>
```

The communication of the workflow engine with the different services is then logged. See Workflow engine log for an example.

8.3. Results

In the following, the final results of the workflow are shown and compared to the input datasets.

Figure 7. The conflated result

First, this is set as a contrast to the different input datasets.

- OpenStreetMap

Figure 8. The conflated result compared to the OSM input dataset

- The National Map

Figure 9. The conflated result compared to the TNM input dataset

- LION (New York State data)

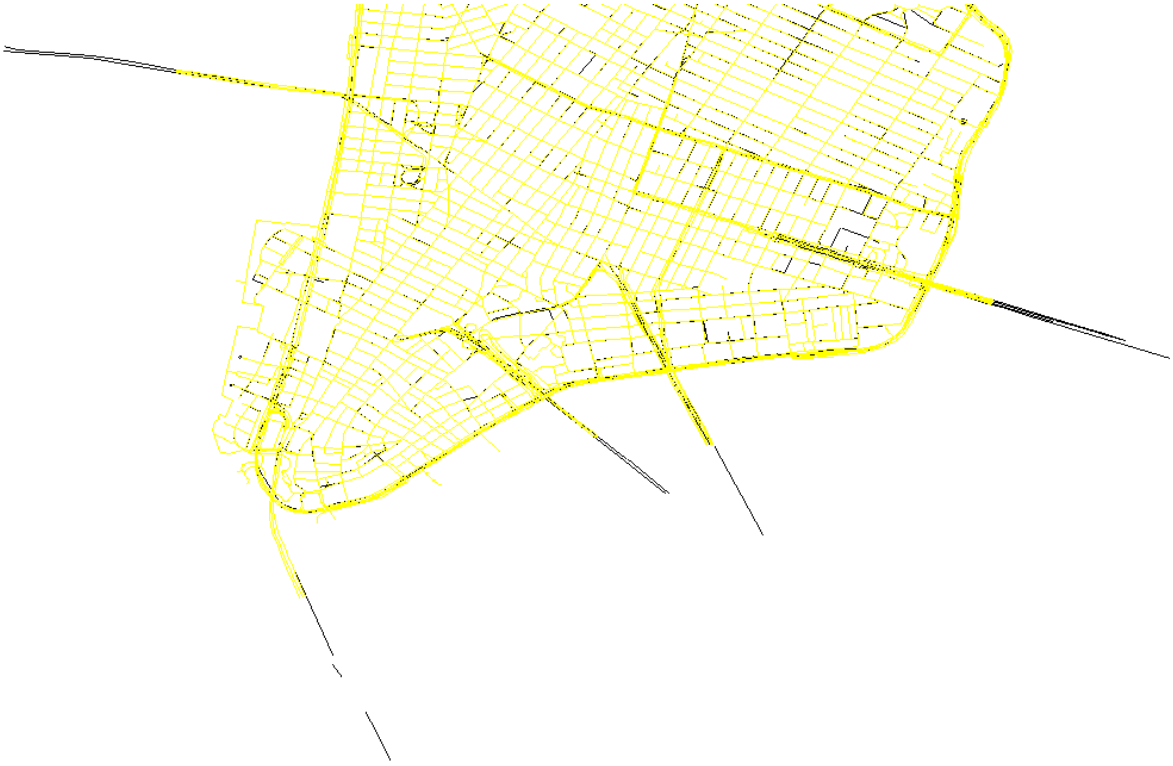

Figure 10. The conflated result compared to the LION input dataset

From these complete views, one can see that there are slight differences between each of the input datasets and the conflated result. This means that the respective geometries were derived from the input datasets and not just blindly inserted. Next, it is worth looking at some details of the result.

- Missing data in the result

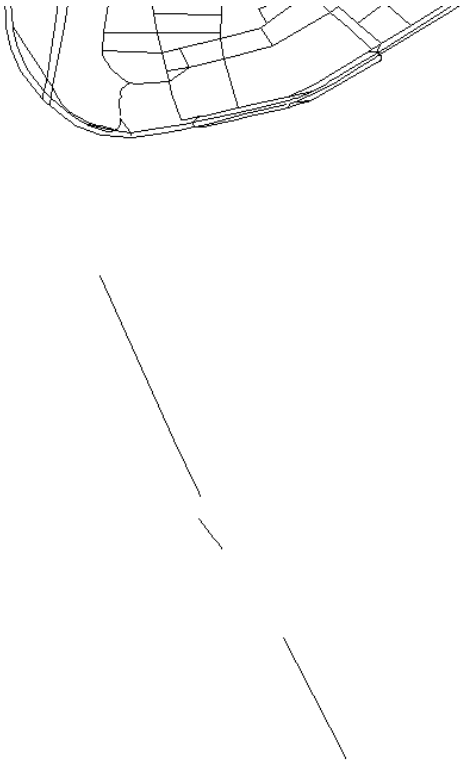

Figure 11. Missing data in the result

Parts of this segment are missing in the result dataset, although the segment is complete in the

input datasets. The reason is presumably a conflict in the naming. *Brooklyn Battery Tunnel* vs *Hugh L. Carey Tunnel*. The different names are stored in either the name or alt_name attribute and only one of them is selected for the conflation.

- Second lane added to two-lane street

The following figure shows the conflated result for *Edgar Street*, a street with separate one-way-lanes.

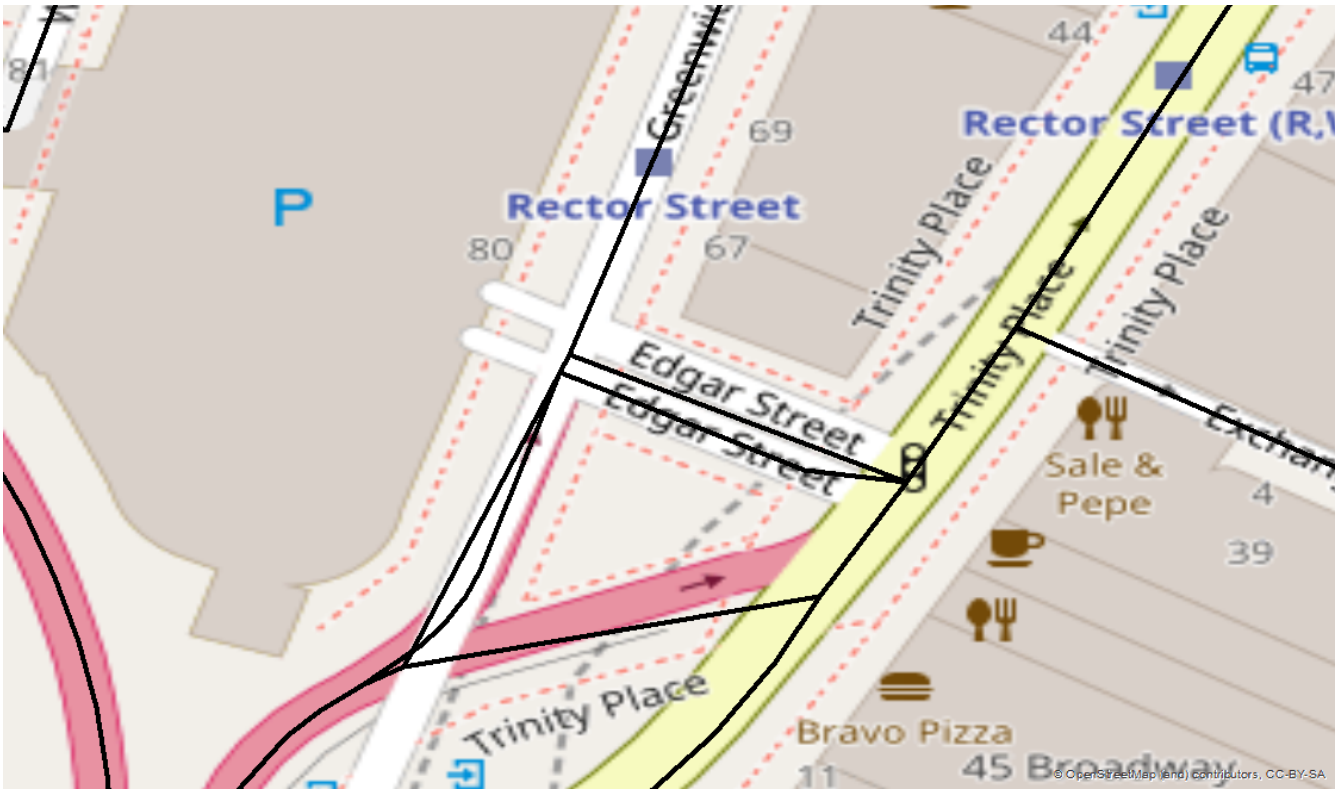

Figure 12. Conflated Edgar Street

The TNM input dataset shows only one lane for this street.

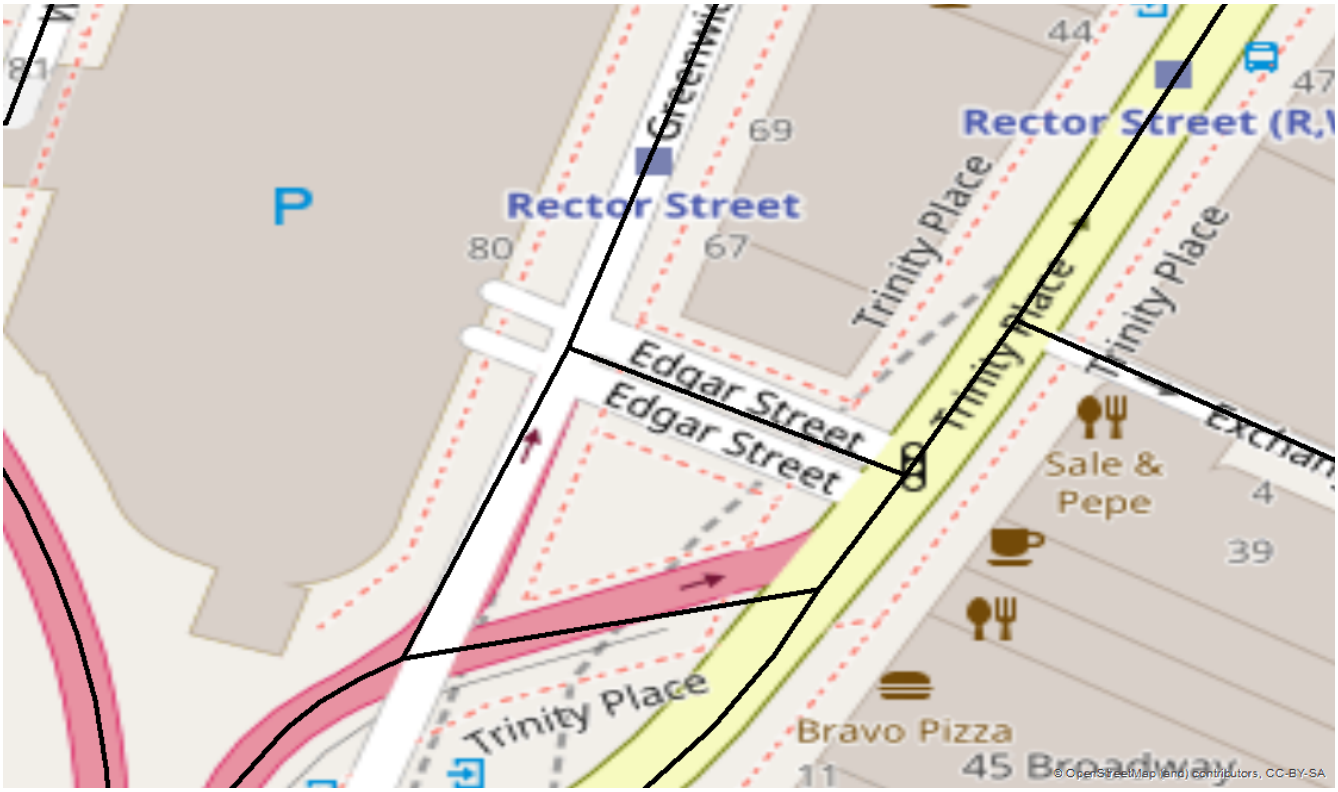

Figure 13. Edgar Street in TNM dataset

In the OSM dataset, two lanes exist.

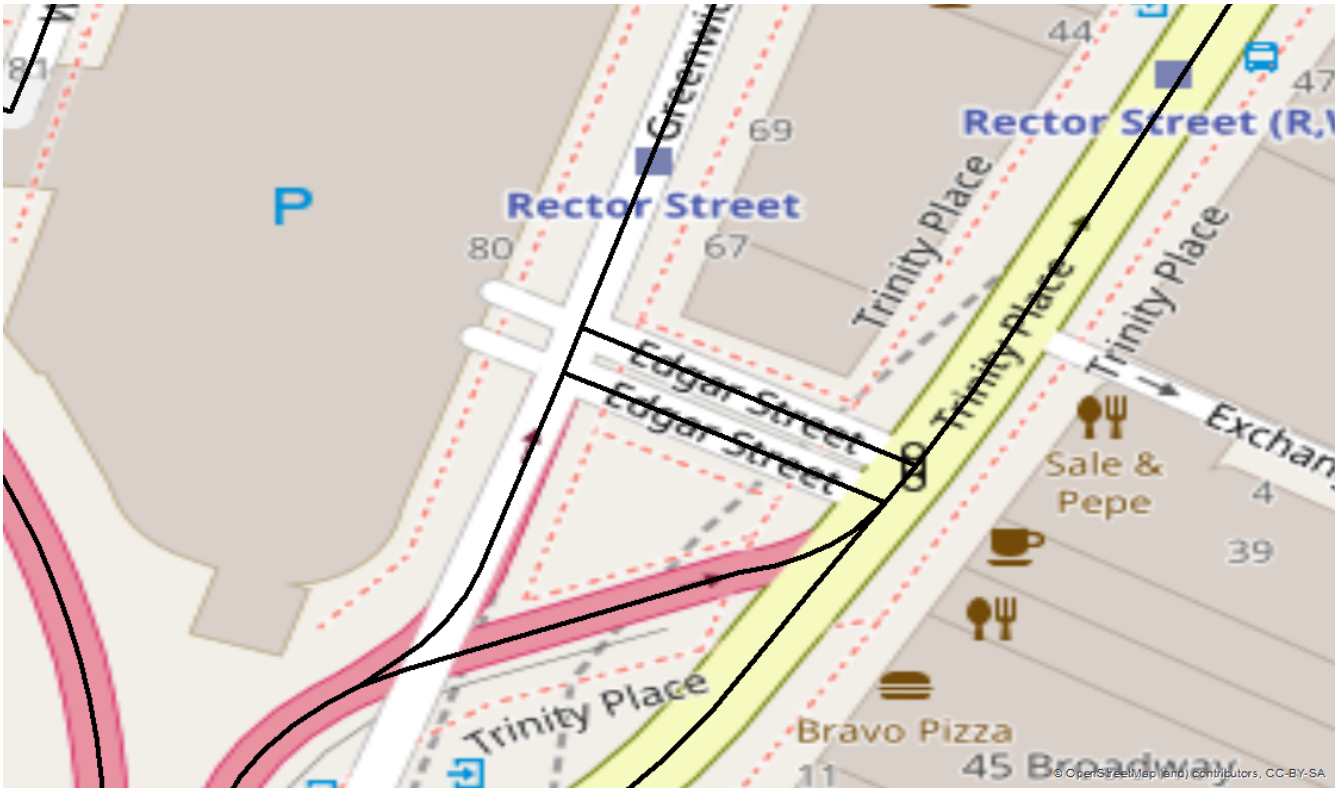

Figure 14. Edgar Street in OSM Dataset

- Added a street from OSM data

The following figure shows a segment of *Greenwich Street* in the conflated result dataset.

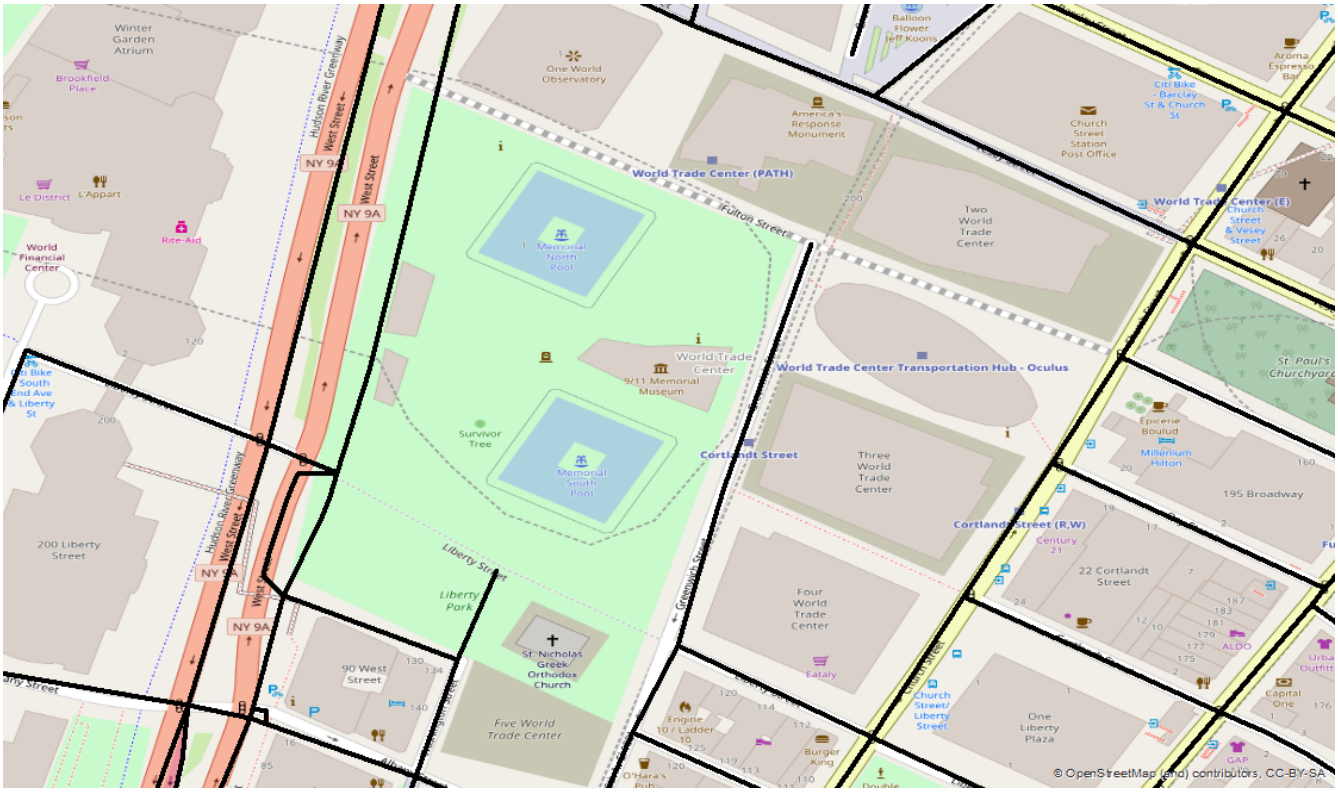

Figure 15. Conflated GreenwichStreet

This street doesn't exist in the TNM dataset.

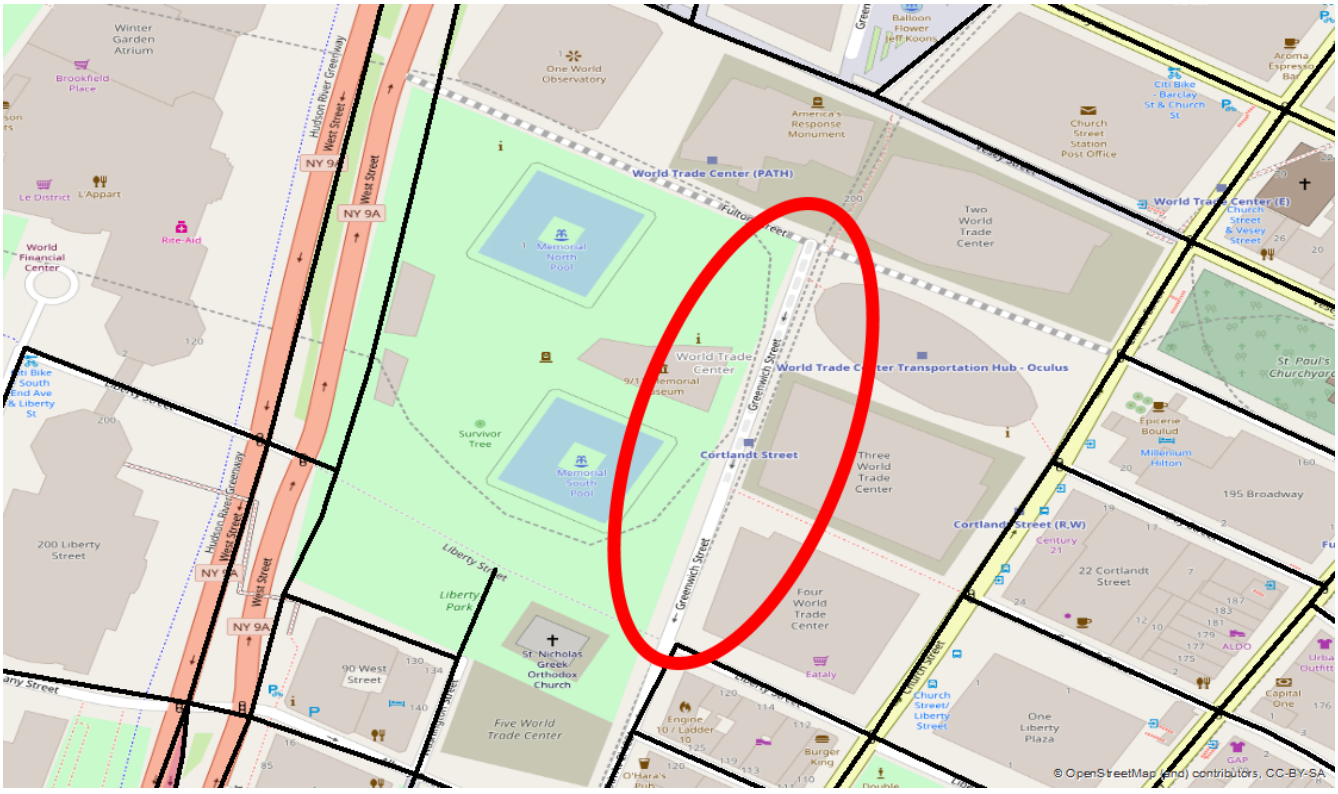

Figure 16. Greenwich Street missing in TNM dataset

The OSM dataset contains the street so it was taken over in the conflated result. However, it was shifted due to a different geometry of a street segment in the TNM dataset.

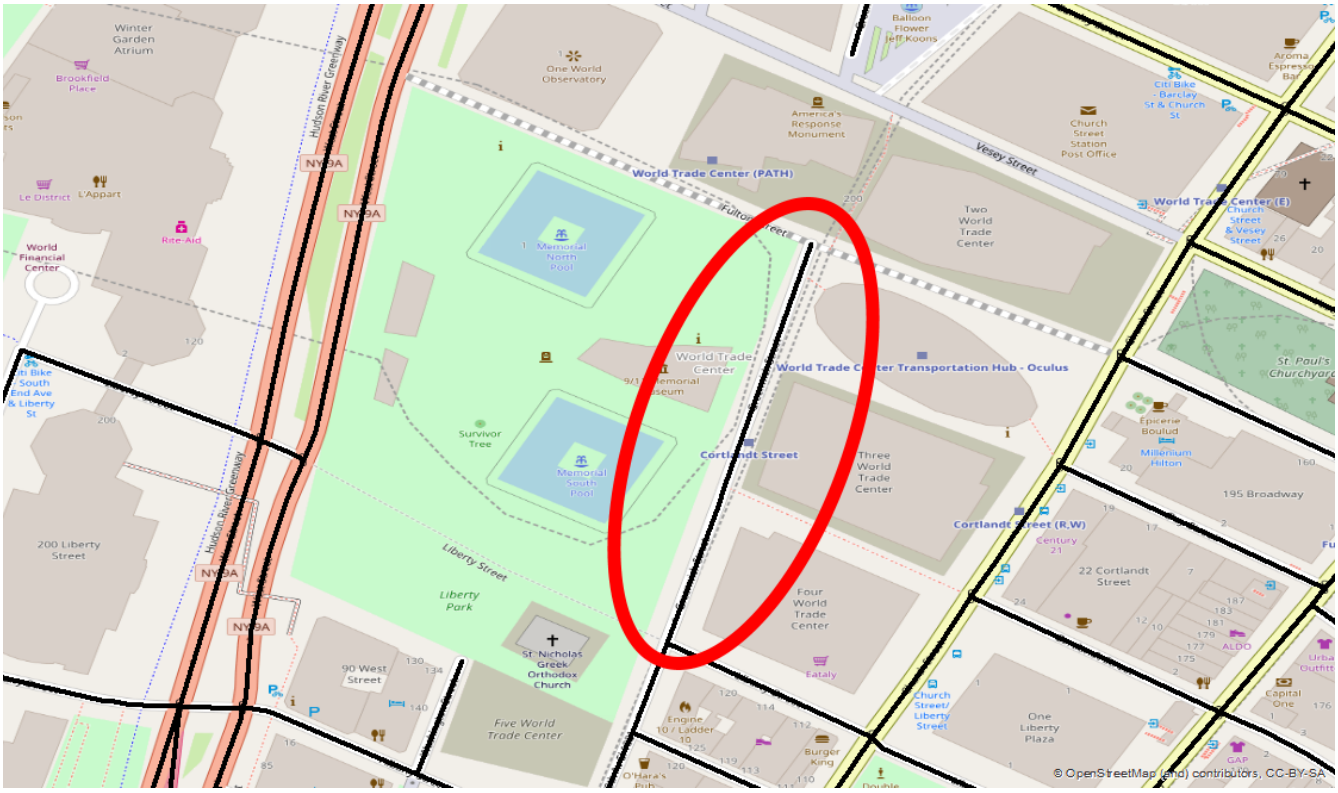

Figure 17. Greenwich Street existing in OSM dataset

Chapter 9. Processing Services

In this chapter, the processes involved in the workflow and their corresponding implementations in Web Processing Services are described.

9.1. Coordinate Transformation Service

The different data that is used in the workflow can have different coordinate reference systems (CRS). Therefore a process using the GeoTools library [9: http://www.geotools.org/] was implemented that can be used for coordinate transformation of vector datasets.

The process is able to transform vector data in the GML 3.1.1 format. Additional inputs are the source and target CRS in form of the EPSG code. The transformed data is returned also in the GML 3.1.1 format. The process was used to transform authoritative data from projected coordinate reference systems to WGS 84 that is used for OpenStreetMap data.

Running instance: http://ows.dev.52north.org:8080/SecurityProxy/service/wps?request=GetCapabilities&service=WPS [http://ows.dev.52north.org:8080/javaps/service?request=GetCapabilities&service=WPS]

Example execute-request:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute
 xmlns:wps="http://www.opengis.net/wps/2.0"
 xmlns:ows="http://www.opengis.net/ows/2.0"
 xmlns:xlink="http://www.w3.org/1999/xlink"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wpsExecute.xsd"
 service="WPS" version="2.0.0" response="raw" mode="sync">

<ows:Identifier>org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgor
ithm</ows:Identifier>
    <wps:Input id="data">
        <wps:Reference xlink:href="http://tb12.dev.52north.org/security-
proxy/service/wfs?service=WFS&amp;amp;version=2.0.0&amp;amp;request=GetFeature&amp;amp;typeName=tb
13:tnm-manhattan-streets&amp;amp;OUTPUTFORMAT=GML3" mimeType="text/xml" schema=
"http://schemas.opengis.net/gml/3.1.1/base/feature.xsd" />
    </wps:Input>
    <wps:Input id="source_epsg">
        <wps:Data mimeType="text/plain">EPSG:32118</wps:Data>
    </wps:Input>
    <wps:Input id="target_epsg">
        <wps:Data mimeType="text/plain">EPSG:4326</wps:Data>
    </wps:Input>
    <wps:Output id="result" mimeType="text/xml" schema=
"http://schemas.opengis.net/gml/3.1.1/base/feature.xsd" transmission="reference"/>
</wps:Execute>
```

9.2. Data Quality Service

Another step in the workflow is checking datasets for quality aspects like positional accuracy. The Data Quality WPS that was implemented in Testbed-12 by Helyx will be re-used in the workflow (see OGC 16-041r1 [10: http://docs.opengeospatial.org/per/16-041r1.html] for more information).

Running instance (client certificate needed): https://tb12.dev.52north.org/data-quality-wps-proxy/service/wps?service=wps&request=getcapabilities [https://tb12.dev.52north.org/data-quality-wps-proxy//WebProcessingService?service=wps&request=getcapabilities]

Example execute-request:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute
  xmlns:wps="http://www.opengis.net/wps/2.0"
  xmlns:ows="http://www.opengis.net/ows/2.0"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd"
  service="WPS" version="2.0.0" response="document" mode="sync">
  <ows:Identifier>
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy</ows:Identifier>
  <wps:Input id="inputTargetDataset">
  <wps:Reference schema="http://schemas.opengis.net/gml/3.1.0/base/feature.xsd"
mimeType="text/xml; subtype=gml/3.1.0" xlink:href=
"http://geoprocessing.demo.52north.org:8080/geoserver/testbed13/ows?service=WFS&versio
n=1.0.0&request=GetFeature&typeName=testbed13:new_york_lines_excerpt_streets_projected
&maxFeatures=50&outputFormat=gml3"/>
</wps:Input>
   <wps:Input id="inputReferenceDataset">
  <wps:Reference schema="http://schemas.opengis.net/gml/3.1.0/base/feature.xsd"
mimeType="text/xml; subtype=gml/3.1.0" xlink:href=
"http://geoprocessing.demo.52north.org:8080/geoserver/testbed13/ows?service=WFS&versio
n=1.0.0&request=GetFeature&typeName=testbed13:new_york_streets_projected_massive_shift
&maxFeatures=50&outputFormat=gml3"/>
</wps:Input>
  <wps:Input id="threshold">
    <wps:Data>
    <wps:LiteralValue>10</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Input id="inputTargetField">
    <wps:Data>
    <wps:LiteralValue>osm_id</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Input id="inputReferenceField">
    <wps:Data>
    <wps:LiteralValue>osm_id</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Output id="outputMetadataChunk" transmission="value"/>
</wps:Execute>
```

Corresponding response:

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

```
  <wps:JobID>a1dbf17c-35f5-4643-97b1-8357042a845f</wps:JobID>
  <wps:Output id="outputMetadataChunk">
    <wps:Data mimeType="text/xml">
      <DQ_AbsoluteExternalPositionalAccuracy>
        <nameOfMeasure>
          <CharacterString>Test of accuracy of the target data against an
authoritative reference</CharacterString>
        </nameOfMeasure>
        <evaluationMethodType>
          <DQ_EvaluationMethodTypeCode codeList=
"http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#EvaluationMethodTypeC
ode" codeListValue="directExternal">Direct external</DQ_EvaluationMethodTypeCode>
        </evaluationMethodType>
        <dateTime>
          <DateTime>2006-11-10T00:00:00</DateTime>
        </dateTime>
        <result>
          <DQ_ConformanceResult>
            <specification>
              <CI_Citation>
                <title>
                  <CharacterString>Accuracy of position test</CharacterString>
                </title>
                <date>
                  <CI_Date>
                    <date>
                      <Date>2017-06-14</Date>
                    </date>
                    <dateType>
                      <CI_DateTypeCode codeList=
"http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#CI_DateTypeCode"
codeListValue="creation">creation</CI_DateTypeCode>
                    </dateType>
                  </CI_Date>
                </date>
              </CI_Citation>
            </specification>
            <explanation>
              <CharacterString>The mean displacement from the authoritative data is
28.552085167979794</CharacterString>
            </explanation>
            <pass>
              <Boolean>0</Boolean>
            </pass>
          </DQ_ConformanceResult>
        </result>
      </DQ_AbsoluteExternalPositionalAccuracy>
    </wps:Data>
  </wps:Output>
</wps:Result>
```

Data Quality process execute request with result from coordinate transformation

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute
  xmlns:wps="http://www.opengis.net/wps/2.0"
  xmlns:ows="http://www.opengis.net/ows/2.0"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd"
  service="WPS" version="2.0.0" response="document" mode="sync">
  <ows:Identifier>
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy</ows:Identifier>
  <wps:Input id="inputTargetDataset">
    <wps:Reference xlink:href=
"http://ows.dev.52north.org:8080/SecurityProxy/service/wps?request=GetOutput&version=2
.0.0&service=WPS&id=95c0dfd3-2b3a-4c66-a2ed-09c192a31d61" mimeType="text/xml;
subtype=gml/3.1.0" schema="http://schemas.opengis.net/gml/3.1.0/base/feature.xsd"/>
</wps:Input>
   <wps:Input id="inputReferenceDataset">
     <wps:Reference schema="http://schemas.opengis.net/gml/3.1.0/base/feature.xsd"
mimeType="text/xml; subtype=gml/3.1.0" xlink:href=
"http://geoprocessing.demo.52north.org:8080/geoserver/testbed13/ows?service=WFS&versio
n=1.0.0&request=GetFeature&typeName=testbed13:manhatten-streets-reference-projected
&outputFormat=gml3"/>
</wps:Input>
  <wps:Input id="threshold">
    <wps:Data>
    <wps:LiteralValue>10</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Input id="inputTargetField">
    <wps:Data>
    <wps:LiteralValue>osm_id</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Input id="inputReferenceField">
    <wps:Data>
    <wps:LiteralValue>osm_id</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Output id="outputMetadataChunk" transmission="value"/>
</wps:Execute>
```

9.3. Conflation Service

The Conflation process developed in Testbed-12 will be re-used in the workflow to generate the final result (see OGC 16-022 [11: http://docs.opengeospatial.org/per/16-022.html] for more details). Two inputs were added to capture the data quality of the two datasets that should be used for

conflation. As a result, the Reference_Layer input was removed. This input was used to specify, which dataset should be used as reference layer (i.e. the one that should be completed by the conflation). This decision now is based upon the data quality of the inputs. The data quality is passed from the data quality process to the conflation process by the workflow engine. Further information about the Hootenanny software can be found in Scenario.

Running instance (client certificate needed): https://tb12.dev.52north.org/conflation-wps-proxy/service/wps?Request=GetCapabilities&Service=WPS [https://tb12.dev.52north.org/conflation-wps-proxy/service/wps?Request=GetCapabilities&Service=WPS]

The process description can be found here: Conflation service process description

An example execute request looks like the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:Execute xmlns:wps="http://www.opengis.net/wps/2.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ows=
"http://www.opengis.net/ows/2.0"
    xmlns:xlin="http://www.w3.org/1999/xlink"
    xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd"
    service="WPS" version="2.0.0" response="document" mode="sync">
    <ows:Identifier>testbed13.dsi.HootenannyConflation</ows:Identifier>
    <wps:Input id="INPUT1">
        <wps:Reference mimeType="application/x-zipped-shp" xlin:href=
"http://tb12.dev.52north.org/security-proxy/service/wfs?service=WFS&amp;
version=1.0.0&amp;request=GetFeature&amp;typeName=tb13:tnm-manhattan-streets
&amp;outputFormat=SHAPE-ZIP"/>
    </wps:Input>
    <wps:Input id="INPUT1_TRANSLATION">
        <wps:Reference
            xlin:href="http://geoprocessing.demo.52north.org:8080/data/TNM_Roads.py"
            mimeType="text/x-script.phyton" />
    </wps:Input>
    <wps:Input id="INPUT1_DATA_QUALITY">
        <wps:Data mimeType="text/xml">
            <DQ_AbsoluteExternalPositionalAccuracy>
                <nameOfMeasure>
                    <CharacterString>Test of accuracy of the target data against an
                        authoritative reference</CharacterString>
                </nameOfMeasure>
                <evaluationMethodType>
                    <DQ_EvaluationMethodTypeCode
                        codeList=
"http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#EvaluationMethodTypeC
ode"
                        codeListValue="directExternal">Direct
external</DQ_EvaluationMethodTypeCode>
                </evaluationMethodType>
                <dateTime>
```

```
                    <DateTime>2006-11-10T00:00:00</DateTime>
                </dateTime>
                <result>
                    <DQ_ConformanceResult>
                        <specification>
                            <CI_Citation>
                                <title>
                                    <CharacterString>Accuracy of position
test</CharacterString>
                                </title>
                                <date>
                                    <CI_Date>
                                        <date>
                                            <Date>2017-06-14</Date>
                                        </date>
                                        <dateType>
                                            <CI_DateTypeCode
                                                codeList=
"http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#CI_DateTypeCode"
                                                codeListValue="creation">
creation</CI_DateTypeCode>
                                        </dateType>
                                    </CI_Date>
                                </date>
                            </CI_Citation>
                        </specification>
                        <explanation>
                            <CharacterString>The mean displacement from the
authoritative
                                data is 28.552085167979794</CharacterString>
                        </explanation>
                        <pass>
                            <Boolean>0</Boolean>
                        </pass>
                    </DQ_ConformanceResult>
                </result>
            </DQ_AbsoluteExternalPositionalAccuracy>
        </wps:Data>
    </wps:Input>
    <wps:Input id="INPUT2">
        <wps:Reference
            xlin:href="https://tb12.dev.52north.org/aws-
proxy/service/aws?service=aws&amp;url=https://s3-eu-west-1.amazonaws.com/testbed13-
osm/manhattan/osm-manhattan-roads.osm"
            mimeType="application/x-openstreetmap+xml" />
    </wps:Input>
    <wps:Input id="INPUT2_DATA_QUALITY">
        <wps:Data mimeType="text/xml">
            <DQ_AbsoluteExternalPositionalAccuracy>
                <nameOfMeasure>
                    <CharacterString>Test of accuracy of the target data against an
```

```
                        authoritative reference</CharacterString>
                </nameOfMeasure>
                <evaluationMethodType>
                    <DQ_EvaluationMethodTypeCode
                        codeList=
"http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#EvaluationMethodTypeC
ode"
                        codeListValue="directExternal">Direct
external</DQ_EvaluationMethodTypeCode>
                </evaluationMethodType>
                <dateTime>
                    <DateTime>2006-11-10T00:00:00</DateTime>
                </dateTime>
                <result>
                    <DQ_ConformanceResult>
                        <specification>
                            <CI_Citation>
                                <title>
                                    <CharacterString>Accuracy of position
test</CharacterString>
                                </title>
                                <date>
                                    <CI_Date>
                                        <date>
                                            <Date>2017-06-14</Date>
                                        </date>
                                        <dateType>
                                            <CI_DateTypeCode
                                                codeList=
"http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#CI_DateTypeCode"
                                                codeListValue="creation">
creation</CI_DateTypeCode>
                                        </dateType>
                                    </CI_Date>
                                </date>
                            </CI_Citation>
                        </specification>
                        <explanation>
                            <CharacterString>The mean displacement from the
authoritative
                                data is 0.5542467979794</CharacterString>
                        </explanation>
                        <pass>
                            <Boolean>0</Boolean>
                        </pass>
                    </DQ_ConformanceResult>
                </result>
            </DQ_AbsoluteExternalPositionalAccuracy>
        </wps:Data>
    </wps:Input>
    <wps:Output id="CONFLATION_OUTPUT" mimeType="application/x-zipped-shp"
```

```
        transmission="reference" />
    <wps:Output id="CONFLATION_REPORT" mimeType="text/plain"
        transmission="reference" />
</wps:Execute>
```

Chapter 10. Data Services

In the following chapter, the data services involved in the workflow scenario are described. Note that the Web Coverage Service was used in the Fit-for-Purpose workflow (see OGC 17-038).

10.1. Web Coverage Service

The rasdaman Array DBMS is the principal provider of raster data in the Workflows package. Rasdaman has been implemented from scratch to support management and retrieval on massive multi-dimensional arrays in a domain agnostic way. It is the only tool which implements all WCS specifications and passes the corresponding OGC conformance tests; as such, it is officially acknowledged as the WCS Core Reference Implementation. The supported OGC standards include:

- Web Coverage Service (WCS) 2.0.1
- Web Coverage Processing Service (WCPS) 1.0.0
- WCS Transaction Extension (WCS-T) 2.0.0
- Web Map Service (WMS) 1.3.0
- Coverage Implementation Schema (CIS) 1.1.0

On the server side, query evaluation involves a variety of optimizations, including adaptive data partitioning and compression, heuristic query rewriting, multi-core parallelization and distributed processing, semantic location-aware caching, and hybrid CPU / GPU processing.

10.1.1. Service information

The Workflows scenario is supported with several raster datasets offered by a rasdaman installation:

Table 2. Service endpoint and access credentials

Service endpoint	http://138.201.18.85/tb13/rasdaman/ows
WCS Capabilities	http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=GetCapabilities
WMS Capabilities	http://138.201.18.85/tb12/rasdaman/ows?service=WMS&version=1.3.0&request=GetCapabilities
Username	testbed13
Password	K9M3yXrM33uD

10.1.2. Datasets

All datasets have been provided by DigitalGlobe and ingested into rasdaman via WCS-T. Following are more details of each dataset.

Zaatari

Three coverages of the Zaatari refugee camp (opened mid-2012) from years 2013, 2016, and 2017 are available; they illustrate the fast changes of the camp with the influx of refugees. In 2015 the camp population was 83,000 refugees.

Year	DescribeCoverage request	GetCoverage request, 4x downscaled
2013	DescribeCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=DescribeCoverage&coverageId=Zaatari_2013]	GetCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=GetCoverage&coverageId=Zaatari_2013&format=image/jpeg&subset=Lat(32.280,32.305)&subset=Long(36.310,36.346)&scaleFactor=4]
2016	DescribeCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=DescribeCoverage&coverageId=Zaatari_2016]	GetCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=GetCoverage&coverageId=Zaatari_2016&format=image/jpeg&subset=Lat(32.280,32.305)&subset=Long(36.310,36.346)&scaleFactor=4]
2017	DescribeCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=DescribeCoverage&coverageId=Zaatari_2017]	GetCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=GetCoverage&coverageId=Zaatari_2017&format=image/jpeg&subset=Lat(32.280,32.305)&subset=Long(36.310,36.346)&scaleFactor=4]

Screenshots of a subset of the coverages:

- 2013

Figure 18. Zaatari 2013

- 2017

Figure 19. Zaatari 2017

Manhattan

Three coverages of lower Manhattan are available, two normal from 2016 and 2017, and an "oblique" coverage from 2016.

Year	DescribeCoverage request	GetCoverage request, 4x downscaled
2016	DescribeCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=DescribeCoverage&coverageId=Manhattan_20161013]	GetCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=GetCoverage&coverageId=Manhattan_20161013&format=image/jpeg&subset=Lat(40.693,40.73)&subset=Long(-74.025,-73.95)&scaleFactor=4]
2017	DescribeCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=DescribeCoverage&coverageId=Manhattan_20170529]	GetCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=GetCoverage&coverageId=Manhattan_20170529&format=image/jpeg&subset=Lat(40.693,40.73)&subset=Long(-74.025,-73.95)&scaleFactor=4]
2016 "oblique"	DescribeCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=DescribeCoverage&coverageId=ManhattanOblique_20161108]	GetCoverage [http://138.201.18.85/tb13/rasdaman/ows?service=WCS&version=2.0.1&request=GetCoverage&coverageId=ManhattanOblique_20161108&format=image/jpeg&subset=Lat(40.70,40.73)&subset=Long(-74.025,-73.95)&scaleFactor=2]

Screenshots of smaller subsets of each coverage:

- 2016

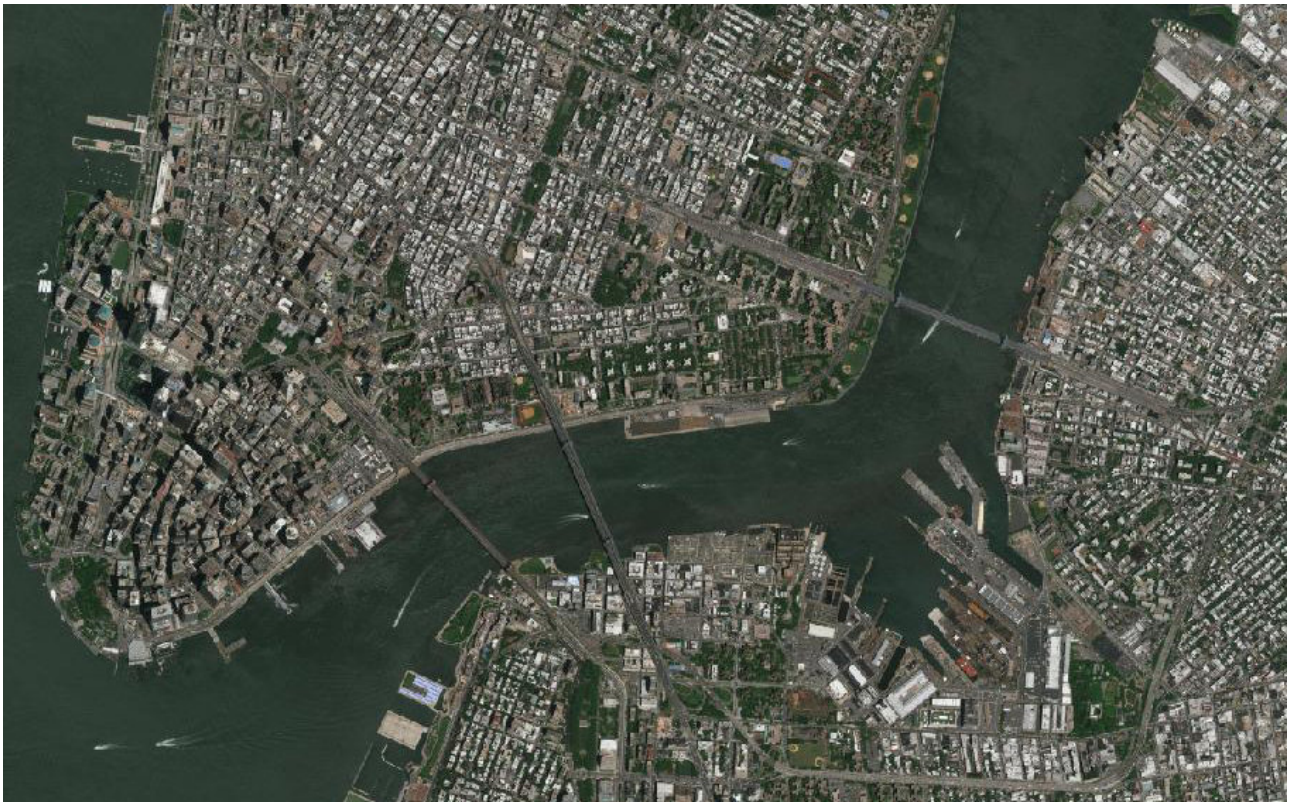

Figure 20. Manhattan 2016

- 2017

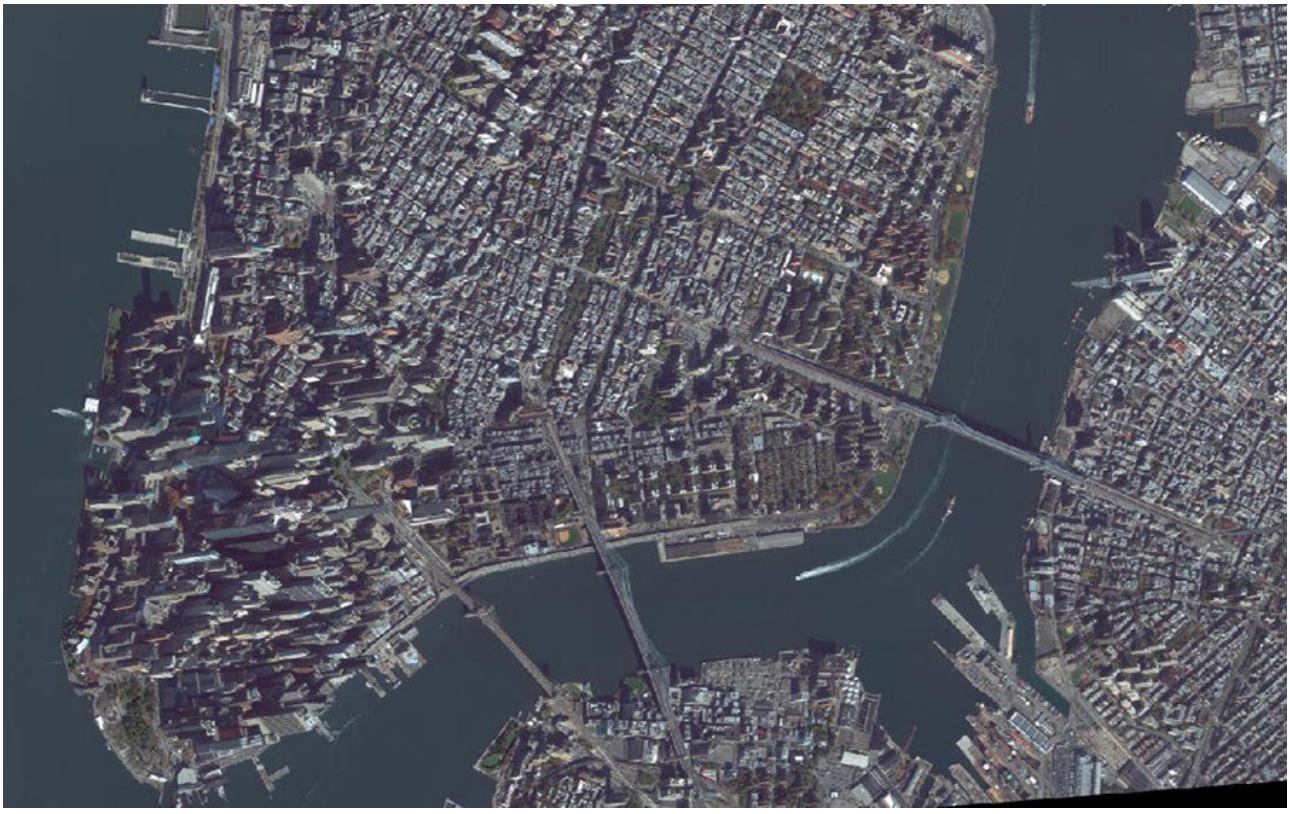

Figure 21. Manhattan 2016 Oblique

Daraa

The Daraa dataset contains time-series of imagery over a 100 sq km area in the Daraa province in Syria. The imagery can be used to see how agriculture production has changed, before and after

Syrian refugee crisis.

This is WorldView data in 2m resolution, with 8 float bands (ingested as band1, ..., band8); it is a 3D time-series cube with 4 time slices (each from around early May): 2011, 2012, 2013, 2014. Full details can be found in the DescribeCoverage document [http://138.201.18.85/tb12/rasdaman/ows?service=WCS&version=2.0.1&request=DescribeCoverage&coverageId=Daraa].

Example queries are shown below:

- Get an RGB view of the slice at year 2013 with a WCPS request:

```
for c in (Daraa)
return encode( {
    red: (unsigned char) (750.0 * c[unix("2013-04-30"), Lat(33.05:33.15)].band5);
    green: (unsigned char) (1000.0 * c[unix("2013-04-30"),
Lat(33.05:33.15)].band3);
    blue: (unsigned char) (1000.0 * c[unix("2013-04-30"), Lat(33.05:33.15)].band2)
}, "jpeg")
```

  The query can be executed with a corresponding

WCS ProcessCoverages request [http://138.201.18.85/tb13/rasdaman/ows?service%3DWCS%26version%3D2.0.1%26request%3DProcessCoverages%26query%3Dfor%20c%20in%20%28Daraa%29%20return%20encode%28%7Bred%3A%20%28unsigned%20char%29%20%28750.0%20%2A%20c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%3B%20green%3A%20%28unsigned%20char%29%20%281000.0%20%2A%20c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band3%29%3B%20blue%3A%20%28unsigned%20char%29%20%281000.0%20%2A%20c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band2%29%20%7D%2C%20%22jpeg%22%29] to get result as shown below:

Figure 22. Daraa 2013 RGB

- "Reverse" NDVI view of the slice at year 2013, that highlights roads and built up areas:

```
for c in (Daraa)
return encode((
    (c[unix("2013-04-30"), Lat(33.05:33.15)].band7 -
     c[unix("2013-04-30"), Lat(33.05:33.15)].band5) /
    (c[unix("2013-04-30"), Lat(33.05:33.15)].band7 +
     c[unix("2013-04-30"), Lat(33.05:33.15)].band5)
) < 0.091, "jpeg")
```

The query can be executed with a corresponding WCS ProcessCoverages request [http://138.201.18.85/tb13/rasdaman/ows?service%3DWCS%26version%3D2.0.1%26request%3DProcessCoverages%26query%3Dfor%20c%20in%20%28Daraa%29%20return%20encode%28%28%28c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band7%20-%20c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%20%2F%20%28c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band7%20%2B%20c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%29%20%3C%200.091%2C%20%22jpeg%22%29] to get result as shown below:

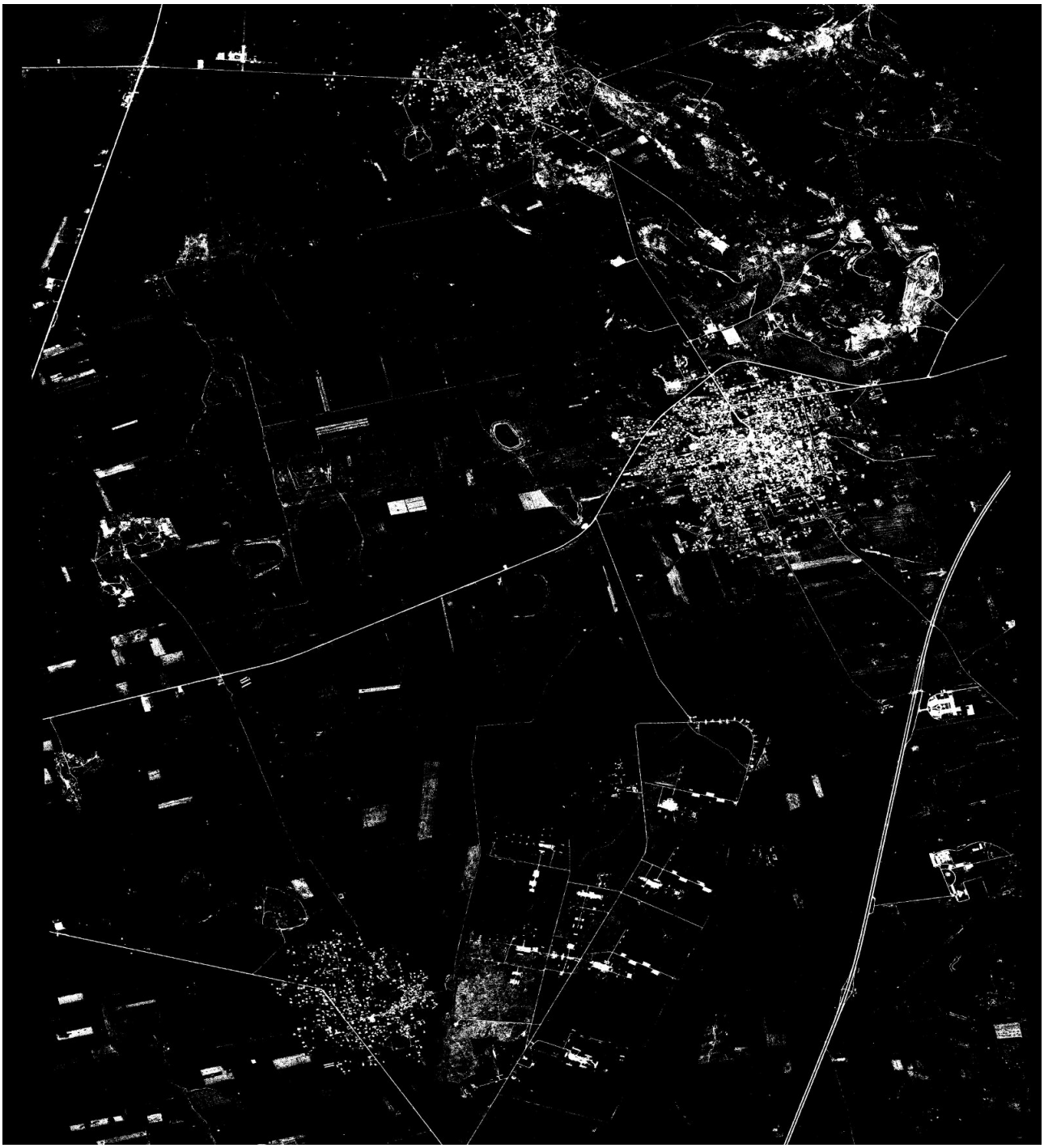

Figure 23. Daraa 2013 NDVI

- Change detection: blue areas denote vegetation lost since May 2011, cyan and green mark vegetation lost since May 2012, and red/orange areas mark vegetation gained in May 2013:

```
for c in (Daraa)
return encode(
    { r:500; b:0; g:0 } * (
        (c[unix("2013-04-30"), Lat(33.05:33.15)].band7 -
         c[unix("2013-04-30"), Lat(33.05:33.15)].band5) /
        (c[unix("2013-04-30"), Lat(33.05:33.15)].band7 +
         c[unix("2013-04-30"), Lat(33.05:33.15)].band5)) +
    {r:0; b:500; g:0} * (
        (c[unix("2012-05-12"), Lat(33.05:33.15)].band7 -
         c[unix("2012-05-12"), Lat(33.05:33.15)].band5) /
        (c[unix("2012-05-12"), Lat(33.05:33.15)].band7 +
         c[unix("2012-05-12"), Lat(33.05:33.15)].band5)) +
    {r:0; b:0; g:500} * (
        (c[unix("2011-05-11"), Lat(33.05:33.15)].band7 -
         c[unix("2011-05-11"), Lat(33.05:33.15)].band5) /
        (c[unix("2011-05-11"), Lat(33.05:33.15)].band7 +
         c[unix("2011-05-11"), Lat(33.05:33.15)].band5))
, "jpeg")
```

The query can be executed with a corresponding WCS ProcessCoverages request [http://138.201.18.85/tb13/rasdaman/ows?service%3DWCS&version%3D2.0.1&request%3DProcessCoverages&query%3Dfor%20c%20in%20%28Daraa%29%20return%20encode%28%7B%20r%3A500%3B%20b%3A0%3B%20g%3A0%20%7D%20%2A%20%28%28c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band7%20-%20c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%20%2F%20%28c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band7%20%2B%20c%5Bunix%28%222013-04-30%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%29%20%2B%20%7Br%3A0%3B%20b%3A500%3B%20g%3A0%7D%20%2A%20%28%28c%5Bunix%28%222012-05-12%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band7%20-%20c%5Bunix%28%222012-05-12%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%20%2F%20%28c%5Bunix%28%222012-05-12%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band7%20%2B%20c%5Bunix%28%222012-05-12%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%29%20%2B%20%7Br%3A0%3B%20b%3A0%3B%20g%3A500%7D%20%2A%20%28%28c%5Bunix%28%222011-05-11%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band7%20-%20c%5Bunix%28%222011-05-11%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%20%2F%20%28c%5Bunix%28%222011-05-11%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band7%20%2B%20c%5Bunix%28%222011-05-11%22%29%2C%20Lat%2833.05%3A33.15%29%5D.band5%29%29%2C%20%22jpeg%22%29] to get result as shown below:

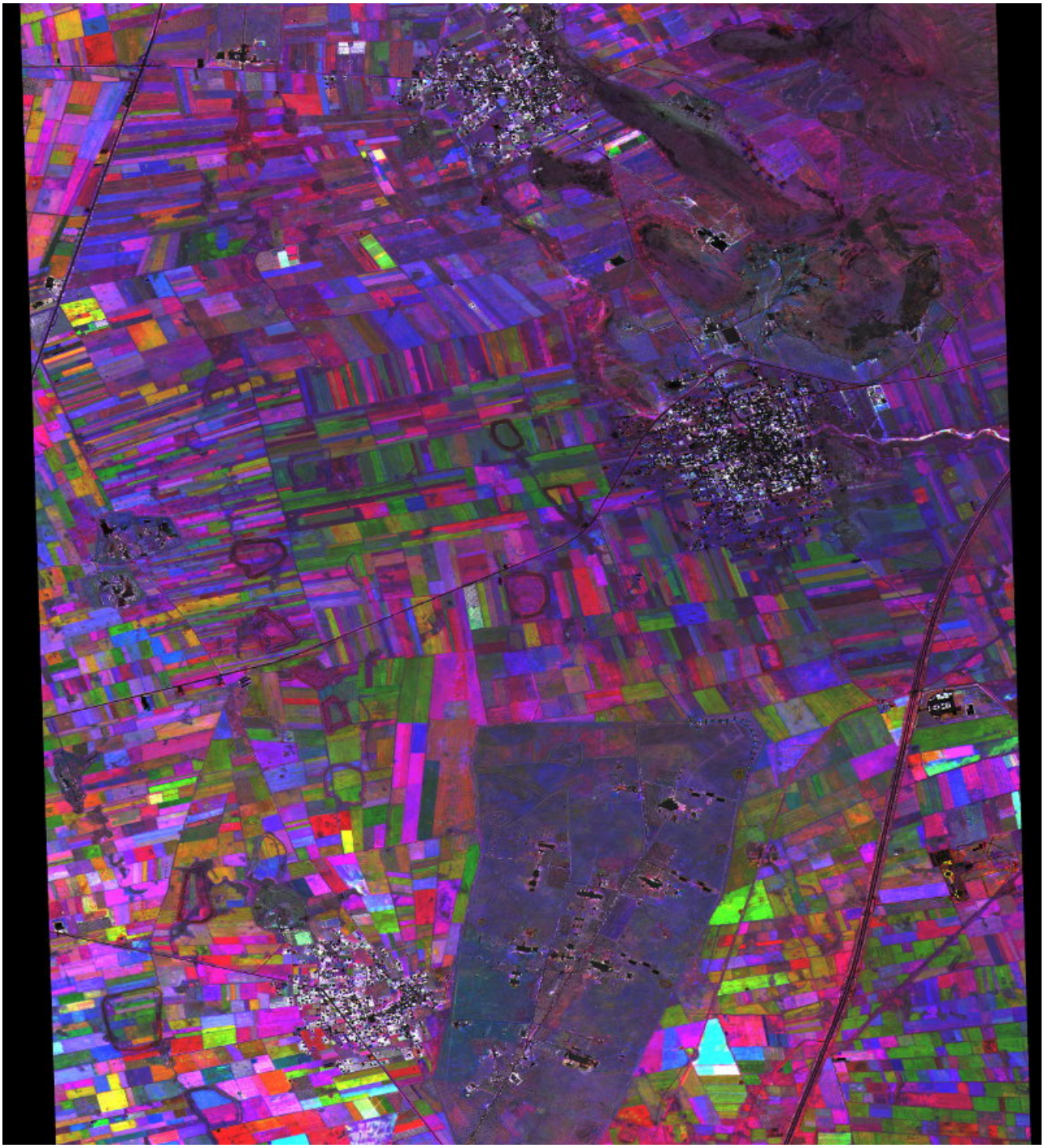

Figure 24. Daraa 2013 change detection

10.2. Web Feature Service

52°North has provided a GeoServer instance that serves road data in the area of Manhattan, New York City.

Service endpoint:

- http://tb12.dev.52north.org/geoserver/tb13/ows [http://tb12.dev.52north.org/geoserver/tb13/ows]
- http://tb12.dev.52north.org/geoserver/tb13/ows?service=wfs&version=1.0.0&request=GetCapabilities [http://tb12.dev.52north.org/geoserver/tb13/ows?service=wfs&version=1.0.0&request=GetCapabilities]

Three different datasets are served:

- OpenStreetMap (OSM) data
- Data from The National Map (TNM)
- Data from the state of New York (LION [12: LION is a single line representation of New York City streets containing address ranges and other information])

The following image shows the OSM data:

Figure 25. OSM data in Manhattan

The coordinate reference system for this data is WGS84 (EPSG:4326).

The next image shows the TNM data in NAD83 (EPSG:32118):

Figure 26. TNM data in Manhattan

Finally, the following image shows the LION data, also in NAD83 (EPSG:32118):

Figure 27. LION data in Manhattan

The three datasets were checked for their positional accuracy using a WPS process. Based upon this quality check, it was decided, which datasets were used for the conflation. The transactional interface of the WFS was used to store the results of the conflation.

Chapter 11. Workflow Client and Catalog

In this chapter, the workflow client and catalog are described. The client was used to create, upload and execute workflows. The catalog was used to discover existing workflows.

11.1. Client

The WPS Client served two purposes in Testbed-13. These included:

- The ability to compose a workflow.
- The ability to execute a workflow.

11.1.1. Workflow Composition

Workflow composition essentially is the ability to a use a WPS Client to create a BPMN (Business Process Model and Notation) workflow document from a WPS process chain.

To create the WPS process chain, the WPS client relies on the WPS processes published in the CSW catalog. Therefore, the ability to generate a BPMN document from a WPS process chain and publish it in the WES (Web Enterprise Suite) CSW catalog is a two step process.

In the first step, applicable WPS services are published in the WES CSW catalog. In the second step, the WPS client will be used to query the CSW for applicable WPS processes and communicate with each applicable WPS service to present the appropriate process information to start the WPS process chain. Once the process chain has been defined, a BPMN document will be generated and published in the WES CSW for discovery.

Step 1: The workflow user publishes all applicable WPS services in the WES CSW registry

- Publishing WPS Services

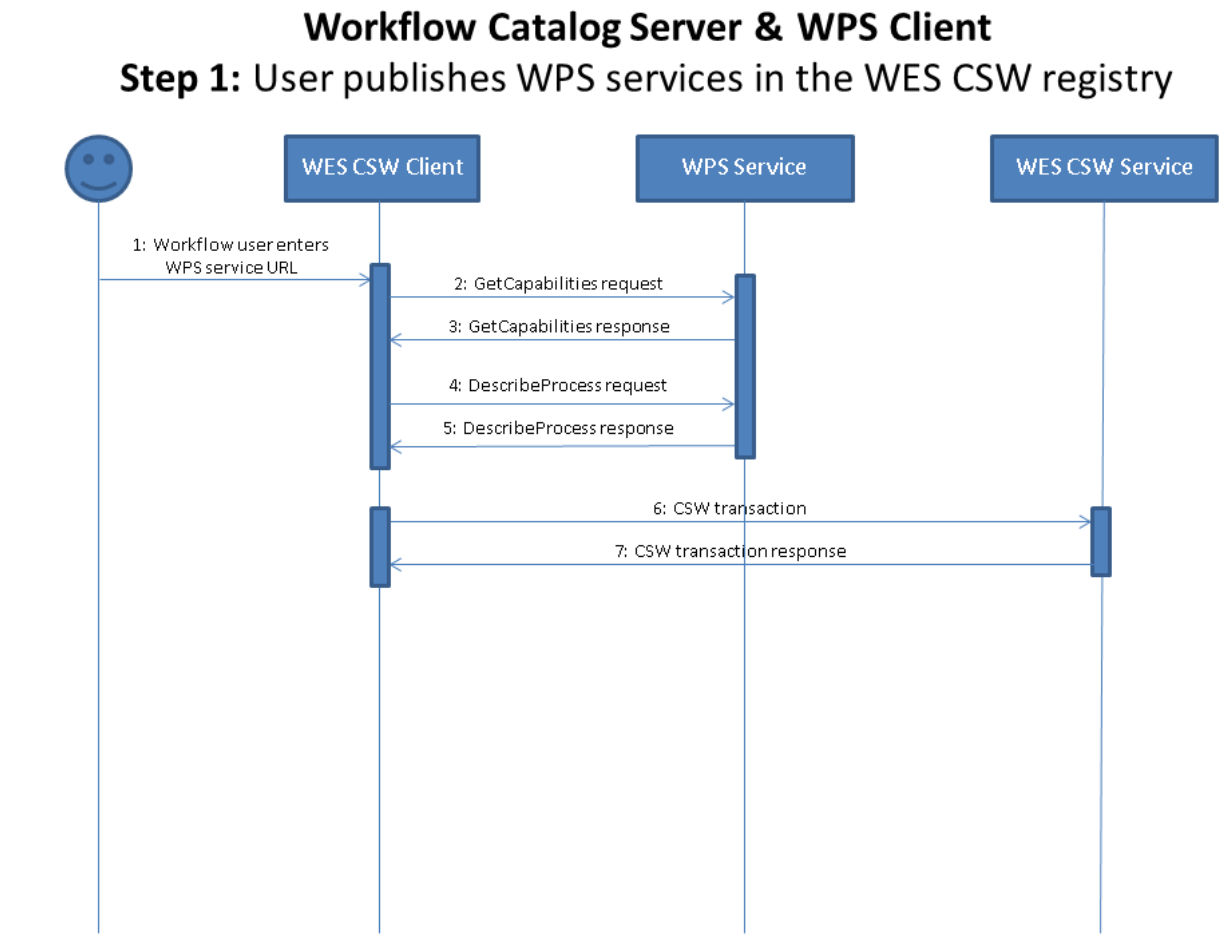

Figure 28. Publishing WPS Services

1. The workflow user launches the WES CSW client and enters the GetCapabilities URL of a WPS service.
2. The CSW client does a GetCapabilities request to the WPS service.
3. The WPS service responds with a GetCapabilities document.
4. To gather more information about each WPS process advertised in the capabilities document, the CSW client does a WPS DescribeProcess request.
5. The WPS service responds with a DescribeProcess response.
6. The CSW client packages all the WPS service information it needs and sends a CSW Transaction to the WES CSW service to publish the service.
7. The CSW service responds with a CSW Transaction response.

*Step 1 will be repeated for each WPS service that is required in the BPMN document.

Step 2: The workflow user launches the WES WPS Client to generate a BPMN document from a WPS process chain and publishes it in the WES CSW registry.

- Process Selection

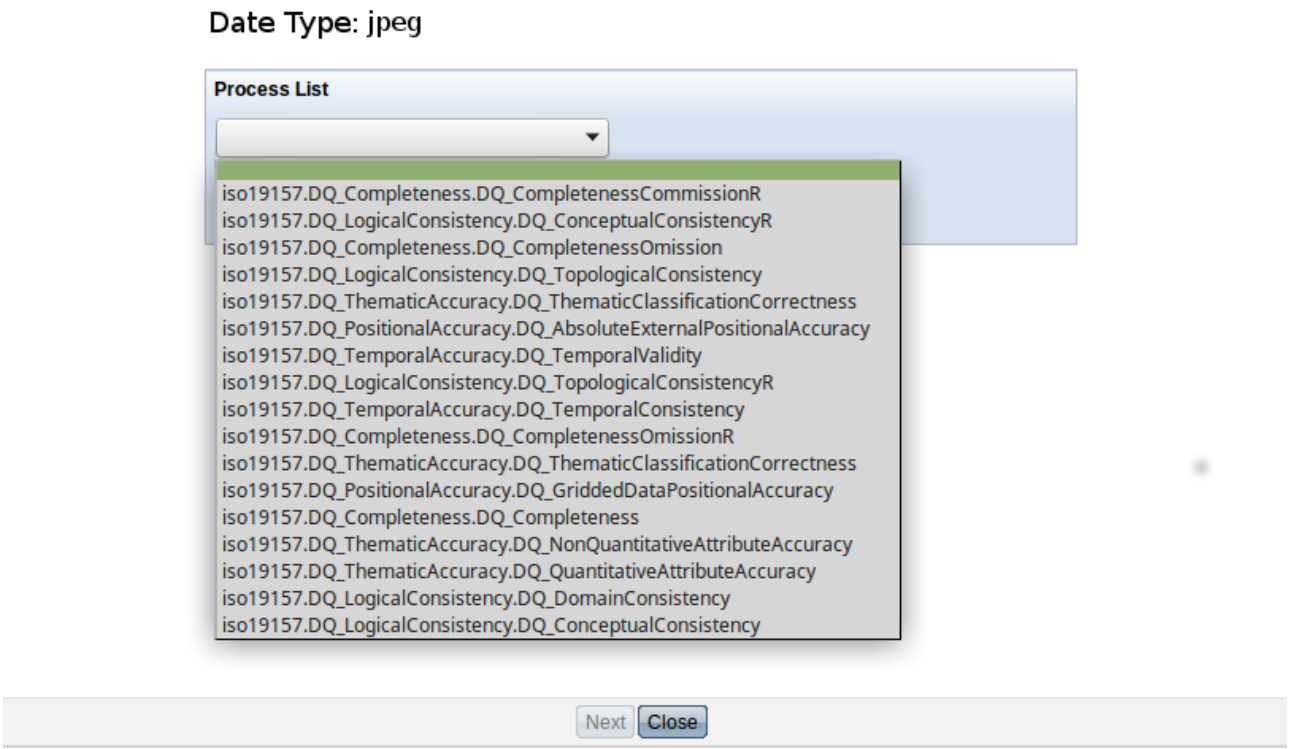

Figure 29. WPS Process Selection

1. The workflow user launches the WES WPS Client and specifies a keyword and/or a data type.
2. The WPS Client queries the WES CSW for applicable WPS processes matching the specified criteria.
3. The CSW service responds with the WPS process information.
4. Using the WPS process information provided by the CSW, the WPS client generates a WPS DescribeProcess request for each process returned above so that the appropriate input and output information can be presented to the user.
5. The WPS service responds with a DescribeProcess response.

- Process Configuration

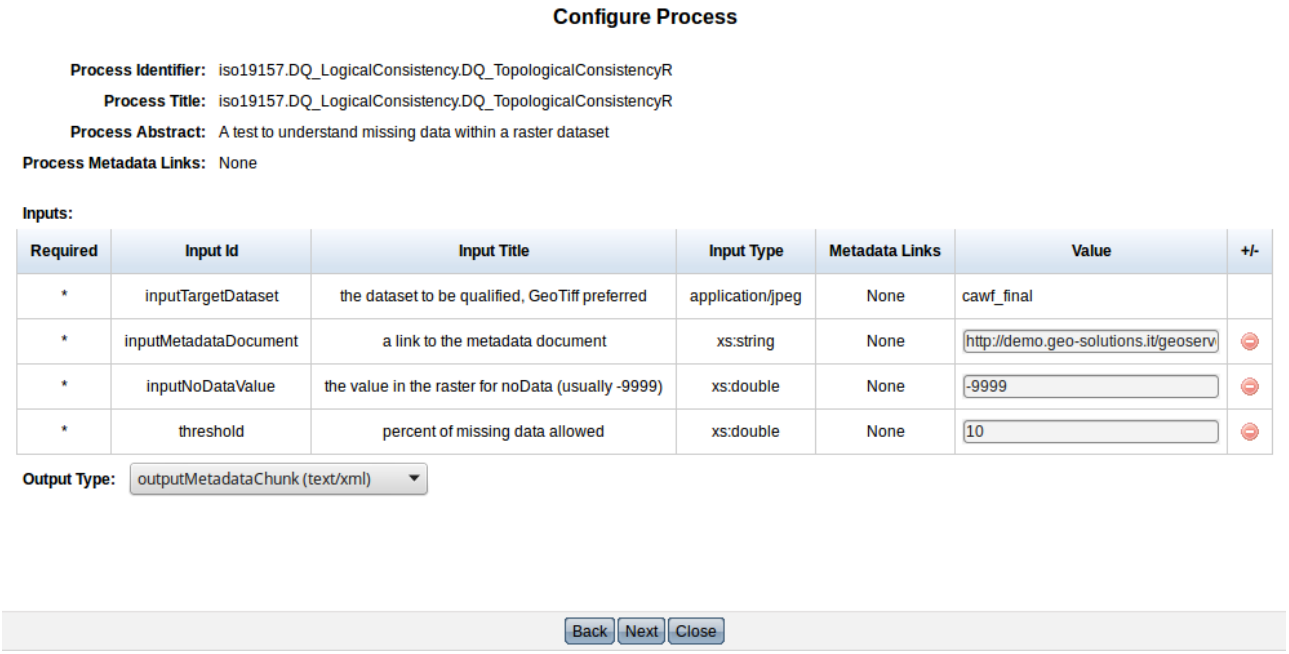

Figure 30. WPS Process Configuration

1. The workflow user selects a process and can provide the values for each input and output type.

- Process Chaining

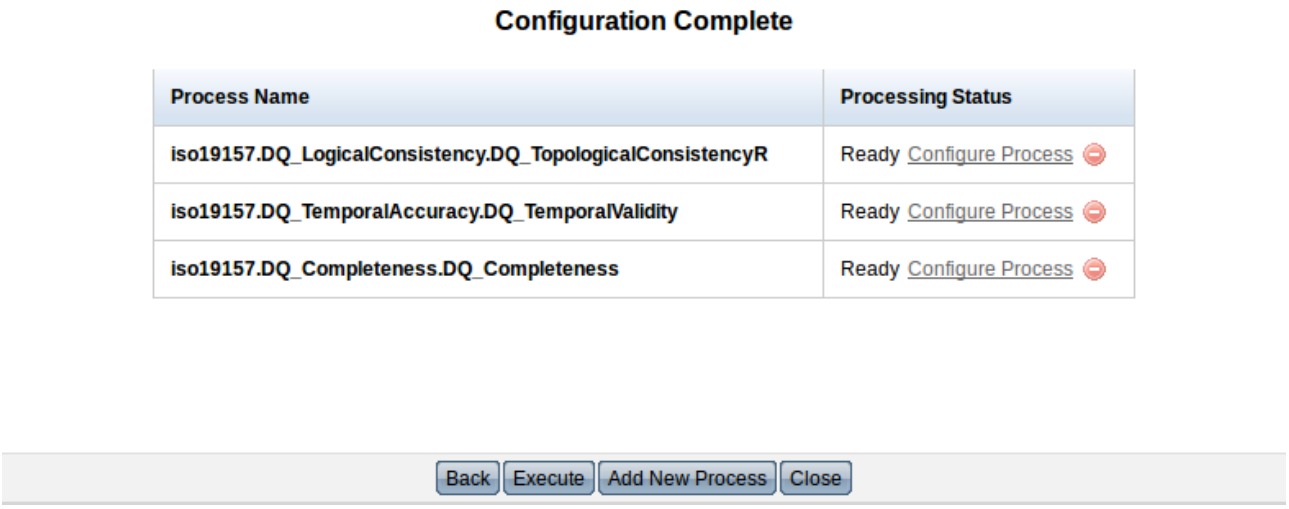

Figure 31. WPS Process Chaining

1. During the Workflow creation, the user can decide to add another process in the chain. This process is repeated until the desired WPS process chain is complete.
2. Once the user has completed the WPS process chain, they select the option to generate a BPMN document and publish it to the WES CSW catalog for discovery. All items in the WES CSW catalog are available via CSW v2.0.2 and v3.0.0 specifications.
3. As highlighted in Workflow Composition Overview, the WPS Client then pushes the BPMN document to the Workflow Engine which dynamically generates a new WPS process based on the workflow defined in the BPMN. This new WPS process is then published in the WES CSW catalog available to be used in the WES WPS Client.

- Workflow Composition Overview

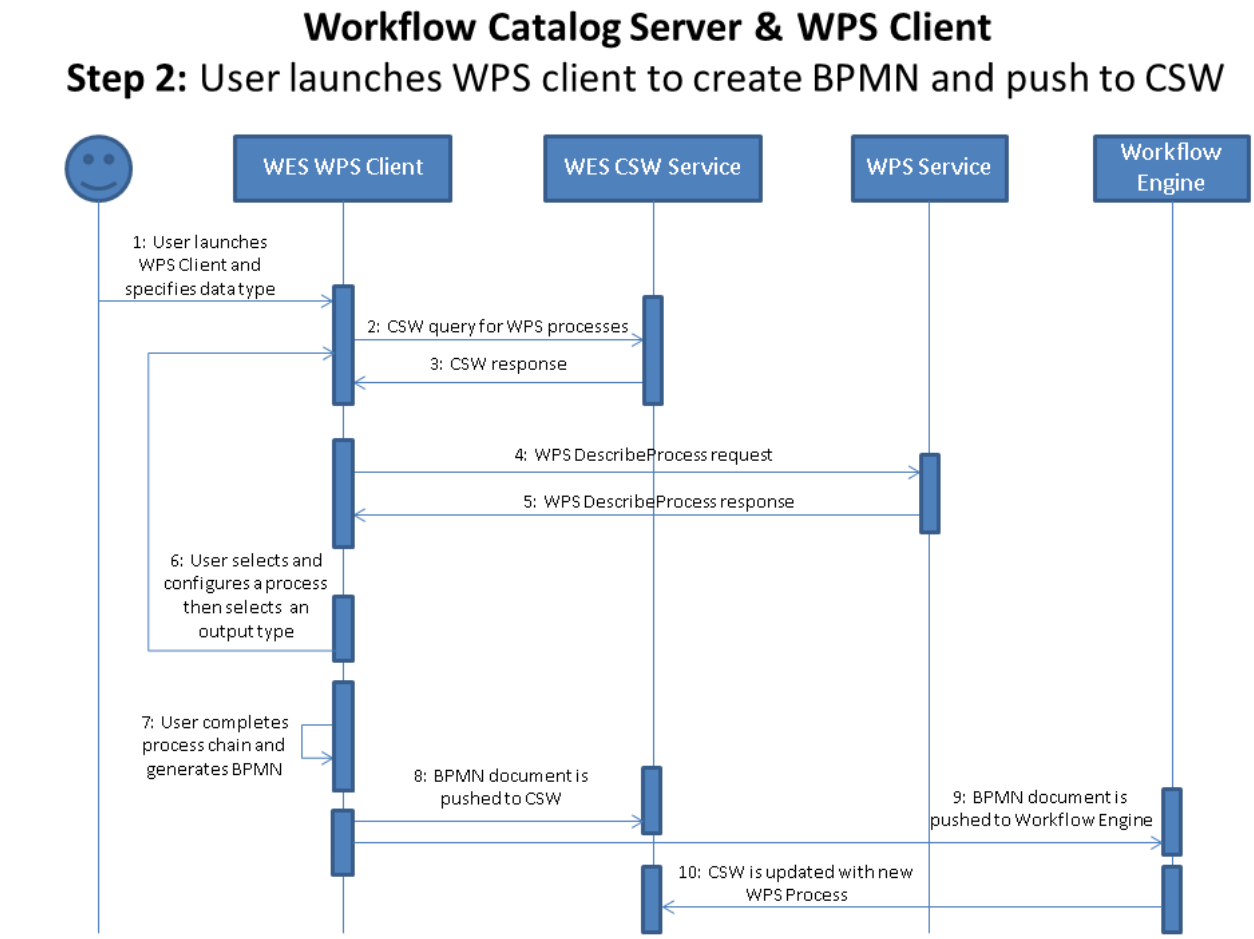

Figure 32. Workflow Composition Overview

1. The workflow user launches the WES WPS Client and specifies a data type (not specifying a data type will return all WPS processes).
2. The WPS Client queries the WES CSW for applicable WPS processes matching the specified data type.
3. The CSW service responds with the WPS process information.
4. Using the WPS process information provided by the CSW, the WPS client generates a WPS DescribeProcess request for each process returned above so that the appropriate input and output information can be presented to the user.
5. The WPS service responds with a DescribeProcess response.
6. The workflow user selects a process, provides the values for each input and selects an output type. At this point the user can then decide to add another process in the chain. If so, the selected output type of the current process is fed back into step 2. This process is repeated until the desired WPS process chain is complete.
7. The user has completed the WPS process chain and selects the option to generate a BPMN document.
8. The BPMN document is generated and the user selects the option to publish it into the WES CSW catalog for discovery. All items in the WES CSW catalog will be available via CSW v2.0.2 and v3.0.0.
9. The BPMN is pushed to the workflow engine.

10. The workflow engine responds with a new WPS process which is then re-harvested into the CSW catalog.

11.1.2. Workflow Execution

Workflow execution essentially is the ability to use the WPS Client to select a WPS process, provide the values for each of the inputs and select the desired output format to accomplish a desired task. The WPS Client creates the appropriate execute request and sends it to the WPS service to be executed. The generated product is then made available for download.

As discussed earlier, once the workflow engine dynamically generates the new WPS process, this process is re-harvested into the WES CSW catalog. As part of this process keywords are pulled out of the capabilities document and the WPS describe process response and associated with the new entry.

The WPS Client will enable users to search for WPS processes via keywords and data type. Once this information is provided the WPS Client queries the WES CSW for applicable WPS processes matching the specified criteria (see Step 2 above). The user selects the appropriate WPS process, supplies the input values and output format and selects the option to execute the process. The WPS Client will wait for the WPS service to respond and then make the new generated product available for download to the user.

11.2. Catalog

For Testbed-13, the WES (Web Enterprise Suite) CSW Catalog not only worked as a data warehouse for WPS Services and their associated processes but also as a search and discover service for generated workflows.

As mentioned above, the WPS Client queries the CSW for WPS processes that match specified criteria. The WPS client then presents the list of matching processes found in the CSW to the user for selection.

The CSW is also used to store the auto-generated BPMN workflow documents. Once published in the CSW, these documents are available via the CSW v2.0.2 and v3.0.0 specification.

11.2.1. The WES Catalog

WES Catalog is made up of three main components:

- WES Catalog Services for the Web - The WES CSW is an OGC-compliant CSW service that supports v3.0.0 (implemented as part of Testbed-12) with backwards capability for v2.0.2. As its data store, it uses the electronic business XML (ebXML) registry information data model (v3.0) as defined by OASIS. This service supports the ability to return and ingest data in a wide variety of XML formats, including electronic business registry information model (ebRIM), Dublin Core (DC), ISO19115, ISO19119, and Federal Geographic Data Committee (FGDC).
- WES Publisher - WES Publisher is an integrated CSW client that provides the ability for data custodians to publish and register geospatial applications, services, styles, data, and data products into the catalog. Tightly integrated with GeoNetwork, the publishing wizard provides

an easy and intuitive way for users to add various types of geospatial data to the catalog and is fully configurable to include other data types as required.

- WES Explorer - WES Explorer is an integrated CSW client interface that provides discovery access into the items published in the service registry. This component allows users to define search criteria through a fully configurable interface and enables users to execute various actions on the search results.

As displayed in the figure below each ebRIM object shares a number of common attributes (i.e. name, description, slots, Classification, User, External Links and Extents).

- The WES Catalog ebRIM Core View

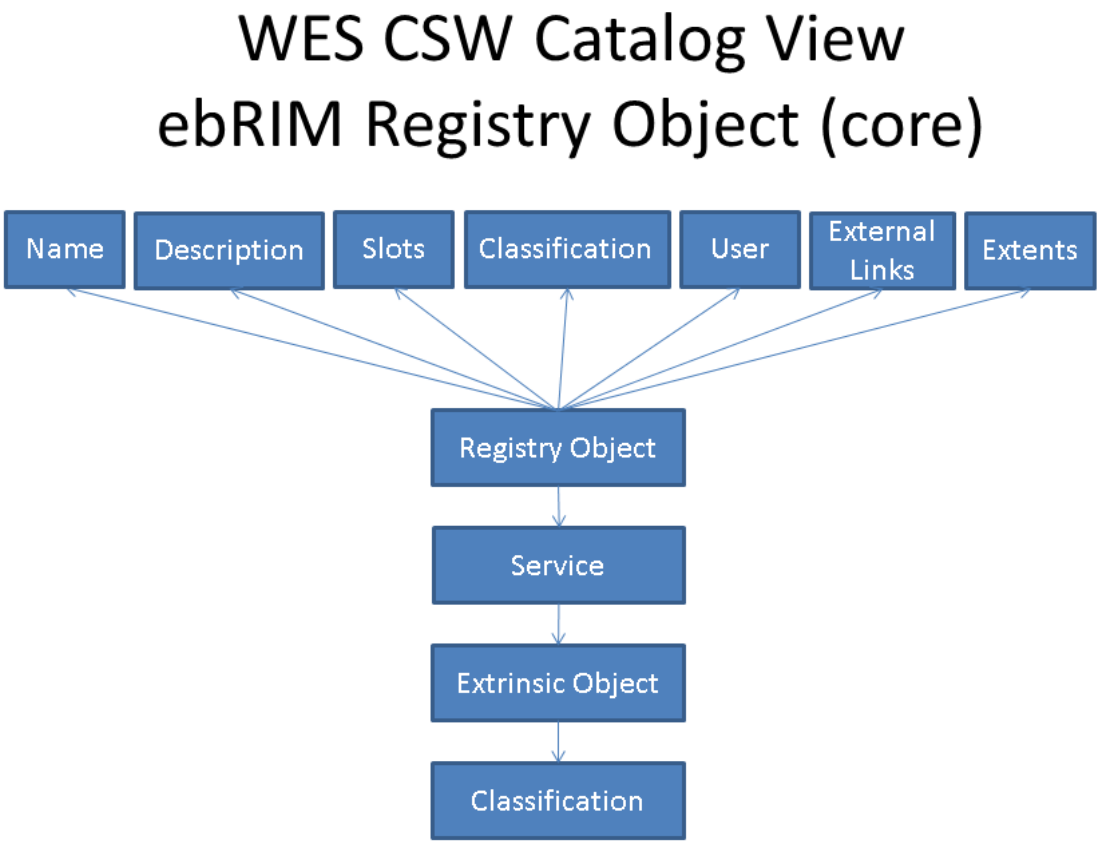

Figure 33. The WES Catalog ebRIM Core View

11.2.2. Representing the WPS Service in the WES CSW Catalog

In the following section, the representation of WPS services in the workflow catalog is described. The figure below presents a closer look of how an OGC WPS service is represented in the WES catalog.

As indicated, each Service object is associated with a Process object. Each Process in turn is then associated with an Input object. It should be noted that each Input can also be associated with another Input object.

To store additional metadata, an ISO document is associated with each Service and Process object.

This document is primarily used for keyword searching and metadata presentation.

- The WPS Service WES Catalog View

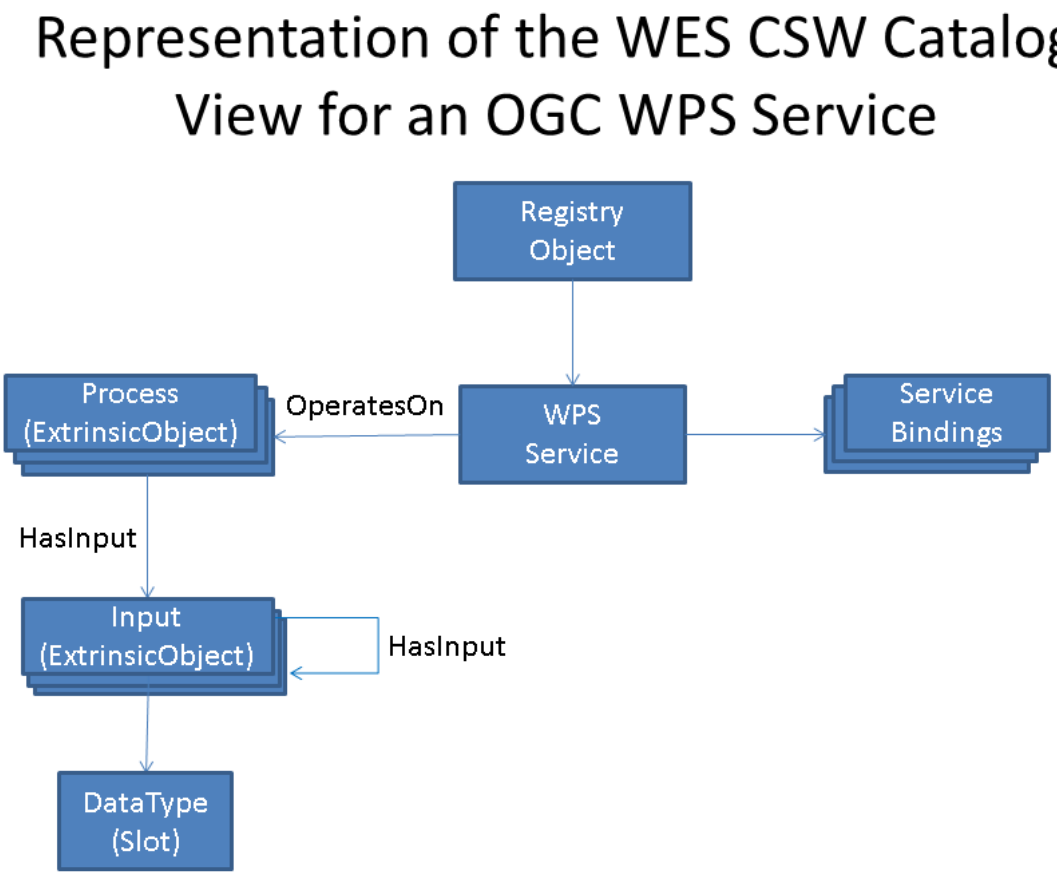

Figure 34. The WPS Service WES Catalog View

11.2.3. The BPMN Workflow Document

The WES Catalog also has the ability to store BPMN workflow documents. These documents not only are used to create new WPS processes in the Workflow Engine WPS but are also available for discovery and can be used by other applications such as Alfresco & Canunda that can process BPMN.

The WES Catalog represents BPMN documents using an ebRIM ExtrinsicObject type with a classification of 'urn:ogc:def:ebRIMObjectType:OGC:WorkFlow'. They can be discovered using the CSW v2.0.2 or the v3.0.0 specification.

Chapter 12. Workflow Security

The workflow security use cases and concepts developed in Testbed-13 are also described in the OGC Testbed-13 Security ER (OGC 17-021).

12.1. Overview

When executing geoprocessing workflows consisting of WPS processes and intermediate transactional W*S data services, it must be ensured that differing access rights and security levels can be dealt with. More specific, three major use cases are considered that need to be addressed in the Testbed-13 workflow scenario.

Dominating Privileges: This use case describes the situation when a user has more privileges than the computer system he is using. Access control based solely on the computer's system privileges will result in a security violation.

Tunneling Proxies: If a service runs behind a proxy/firewall, certain security attributes that were sent by the client might not reach the service. As an example, a mutual TLS Authentication between client and service will not work, since the proxy is creating a new connection between the service and the proxy and the client certificate will hence not reach the service.

Identity mediation: Many different forms of Identification and Authorization (I&A) are available. This is of importance for service-based workflows where the I&As used may differ between the workflow services. In this case, a mediation between identity and the different models used for access control needs to be implemented.

The solutions developed to address these use cases are described in the following subsection.

12.2. Dominating Privileges

The workflow scenario addresses the problem of dominating privileges, if the WPS needs to access a protected resource provided by a data server, e.g. a road dataset provided in a WFS. Though the WPS may be authenticated by the data server and may have general access, operations for retrieving resources of a user may still be protected and may require authorization by the owning user. OAuth addresses this issue, since it is made for delegating authorization from the user to applications to access the user's resources.

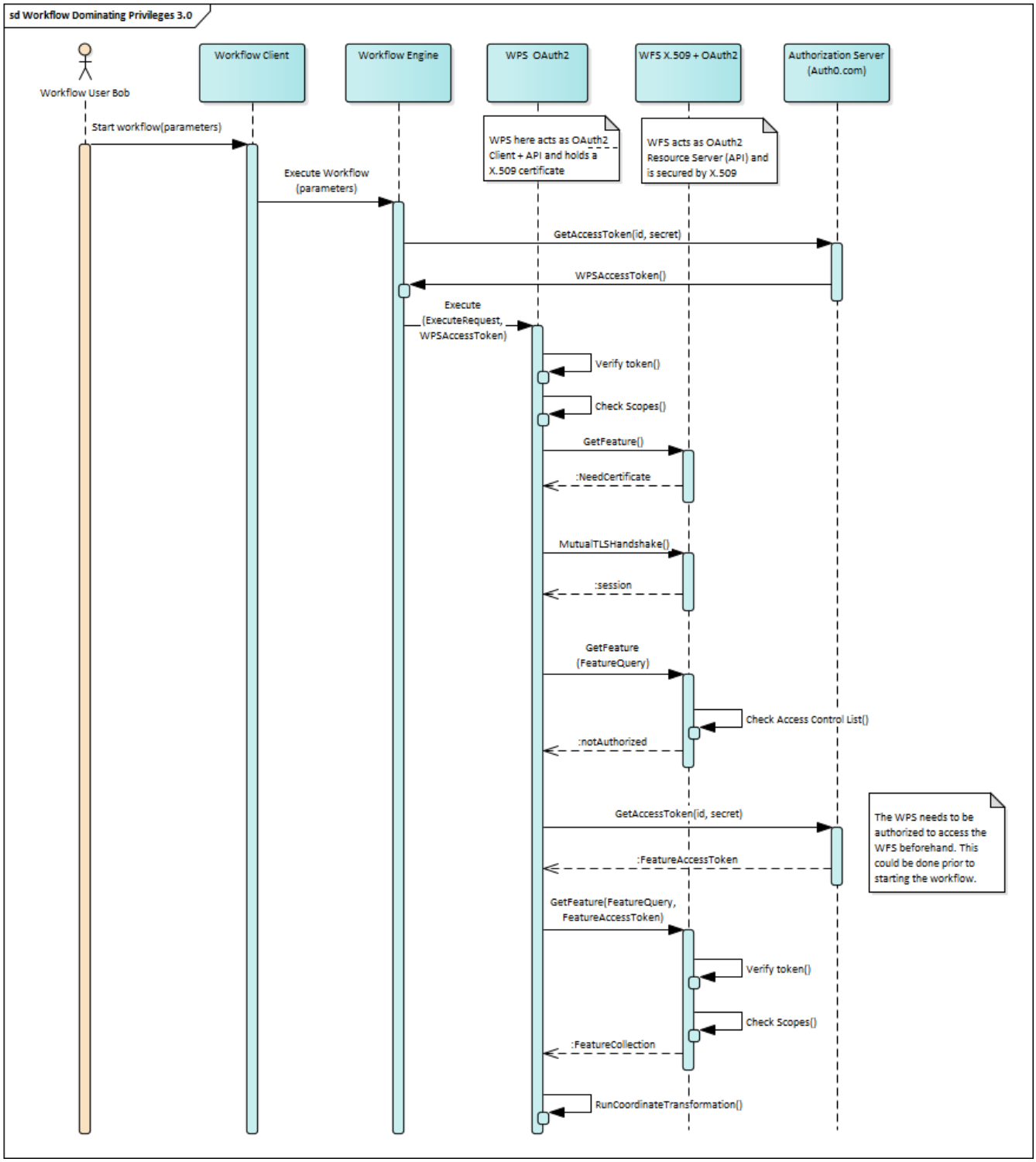

Figure 35. Sequence diagram illustrating the concept for dominating privileges use case in Testbed-13 workflow scenario.

In the Testbed-13 workflow, a WPS needs to retrieve a road dataset from a WFS. The road dataset is passed by reference to the WPS. The WPS at first queries the WFS providing the dataset. The WPS is authenticated by a X.509 certificate that allows the WPS to retrieve the Capabilities document (managed by an Access Control List), but not to retrieve the actual feature data, which is owned by the user running the whole workflow. In this case, authorization needs to be granted from the user owning resources in the WFS to the WPS. This can either be done beforehand using the Client Credentials flow of OAuth (as shown in the diagram above). In this case, the user has authorized the WPS beforehand to access the roads in the WFS and the WPS can directly use its client credentials to retrieve access tokens. Another option is to utilize the Authorization Code Flow to request the

authorization using the authorization server and the user agent. This is described in detail in the section on the OAuth-enabled Web Processing Service above.

12.3. Tunneling Proxies

To address the issue that a service behind a proxy will not retrieve all client security information, consider the case when the client is authenticated by an X.509 certificate that would not be transferred through the proxy on the protocol level. Instead, the certificate needs to be provided on the application level as illustrated in the diagram below.

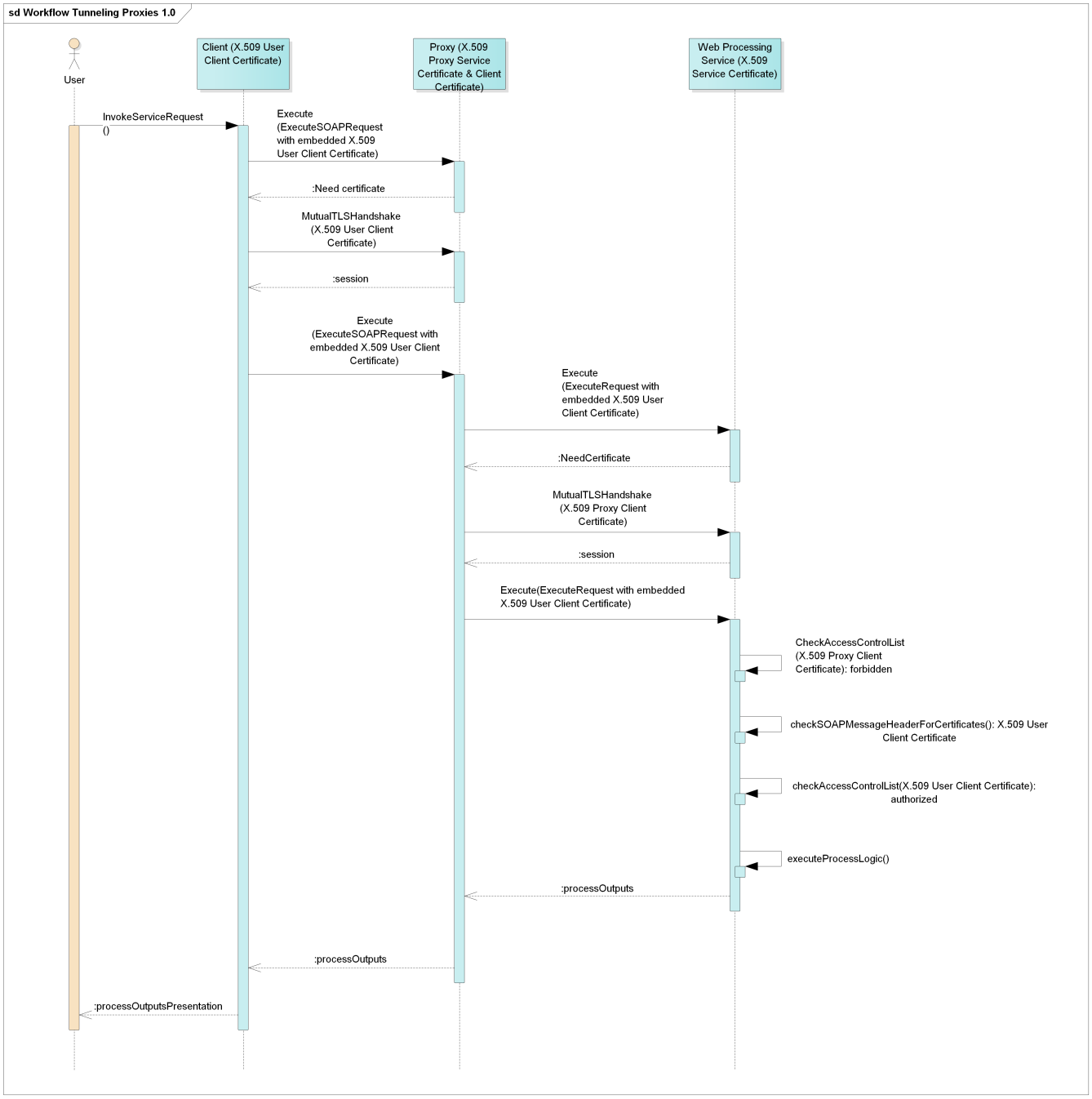

Figure 36. Sequence diagram illustrating the concept for the tunneling proxies use case in Testbed-13 workflow scenario.

The client holds a X.509 user client certificate and sends an Execute Request to WPS. The Execute requests reaches the proxy at first. The Proxy requires client to provide its certificate. The Proxy holds two certificates, a X.509 server certificate and a X.509 client certificate that is used to

communicate with services behind proxy. After the secured session between client and proxy is established, the client can send an Execute request again. The execute request is now forwarded to the WPS. Again, the WPS asks for a client certificate. On the protocol level, the secured session between proxy and WPS is established with the proxy client certificate and WPS service certificate. But when authenticating the actual client and providing access checking the ACL, the WPS needs the user client certificate.

The solution to this problem is to transfer the client certificate on the application level. A standard that can be utilized is the Web Services Security X.509 Certificate Token Profile that describes how to embed X.509 certificates in SOAPMessageHeaders. Hence, if the proxy client certificate is not allowed to run the actual process execution, the WPS can check, whether a certificate is contained in the SOAPMessageHeader and can then check, whether this client has permission to run the execution. In this case, the execution can be done, and the process outputs are then returned.

12.4. Identity Mediation

In the workflow, the requirement is that there needs to be a mediation between X.509 certificate authentication with ACL and a road dataset that is available in a WFS running in the Amazon cloud. The solution to the issue is to deploy a Policy Enforcement Point (PEP) that translates the identities and maps the attributes to roles for access control as shown in the diagram below.

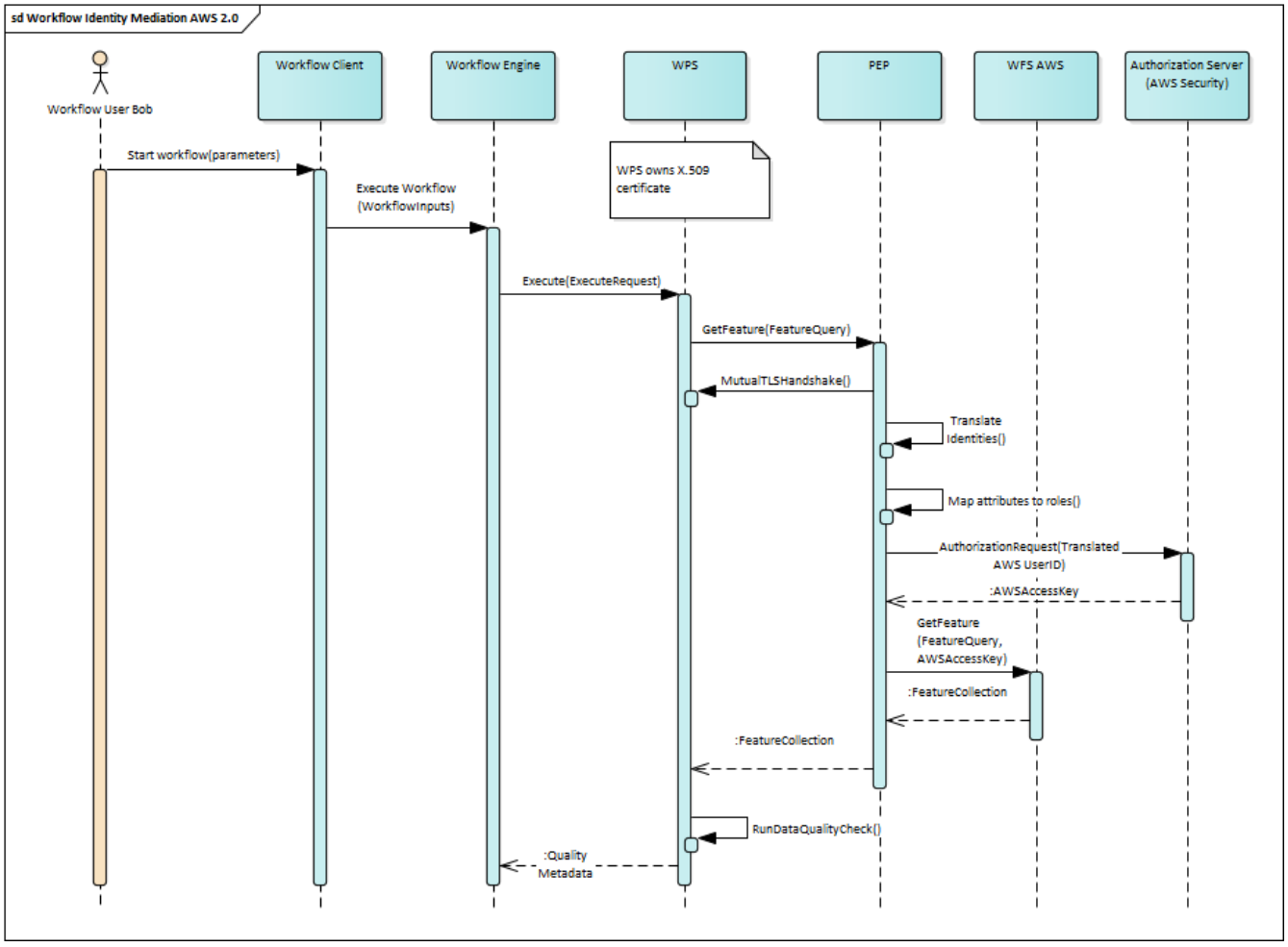

Figure 37. Sequence diagram illustrating the concept for the identity mediation use case in Testbed-13 workflow scenario.

The user connects to the workflow client and starts the workflow. The client triggers the workflow

engine. At some point during the workflow execution, the data quality of road features is checked. A service offering the road data (WFS) is protected by AWS security. The data quality WPS holds a X.509 certificate. A PEP is used as proxy in front of the WFS. A GetFeature request is sent directly to the PEP, along with the certificate. The PEP mediates between the certificate and AWS security. Attribute based access control is mapped to role based access control. The PEP then requests AWS Authorization server for a token. Once it has received the access token, the PEP forwards GetFeature request together with the access token to the WFS and the road dataset is returned. The PEP forwards the features to the WPS. The WPS can then run the data quality check.

12.5. Implementation

For implementing the OAuth Resource Servers and W*S that use X.509 certificates for authentication and access control lists for authorization, 52°North implements a security proxy. The security proxy is implemented as a Java Web Service that can be deployed with OGC Web Services (52°North is providing WPS and WFS for the Testbed-13 Workflow Subthread).

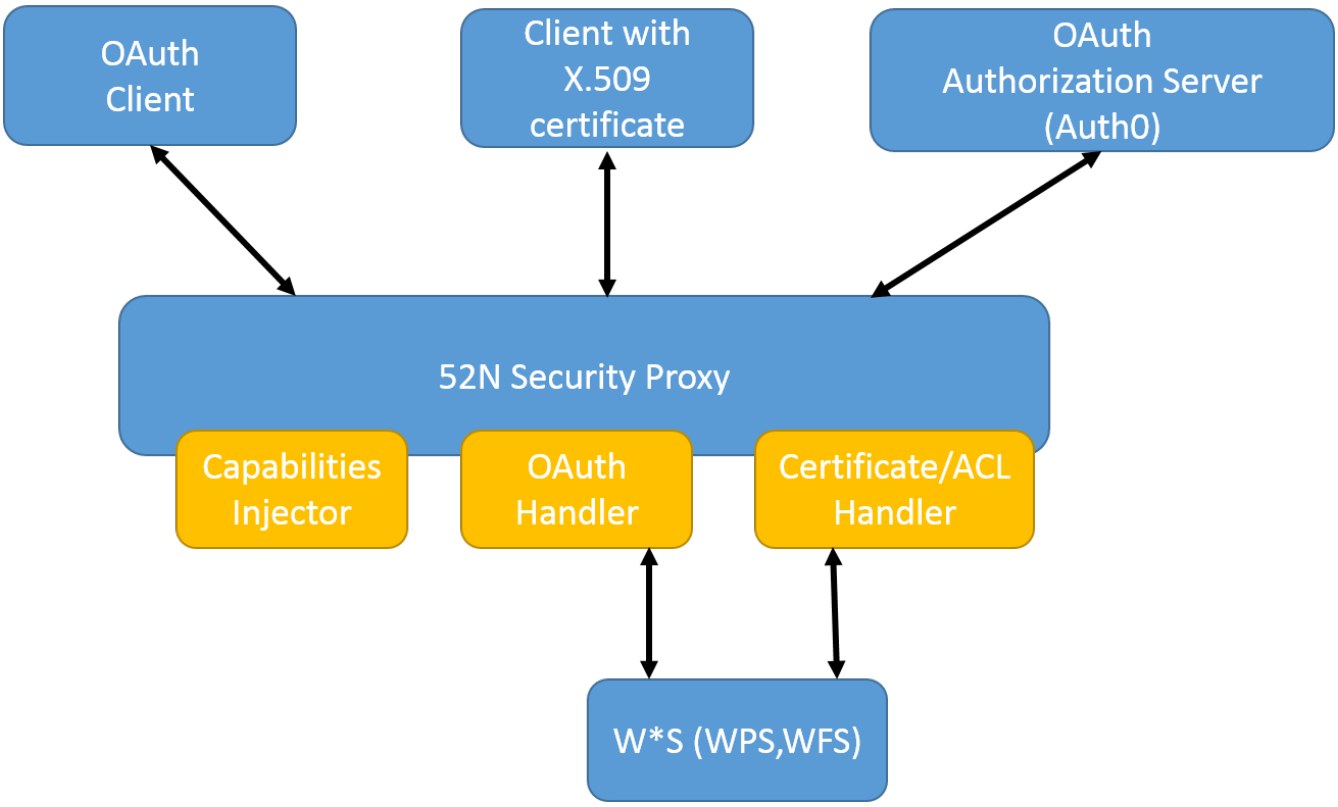

Figure 38. Overview on the 52N security proxy implementation.

An overview of the security proxy implementation is given in the figure above. The 52N security proxy acts as a proxy that receives requests for OGC web services, executes security checks, and, if permission is granted, forwards the requests to the OGC web services and forwards the responses back to the clients. As illustrated in the subsections above, OAuth resource servers as well as servers supporting X.509 certificate authentication and access control lists are needed to address the issues in the workflow security use cases. Thus, the security proxy implements two handlers, the OAuth handler and the Certificate/ACL handler.

The **OAuth handler** has two major tasks:

- Token Verification: It verifies the access token that is sent by the client with the service request.

- Scope Verification: The proxy also needs to verify the list of scopes associated with the access token. If the scope to perform the requested service operation is not included, the proxy denies the request and returns an HTTP 401 "Unauthorized" response with an WWW-Authenticate header indicating that a bearer token is required. If the scope is sufficient, the service request is forwarded to the actual service.

Since auth0 [13: https://auth0.com] is used as an authorization server and JSON Web Tokens (JWT) are provided as access tokens, the verification is implemented in the proxy using a token verification Java library provided by auth0. This library is also used to extract the scopes and implement the scope checks. The scopes itself can be configured in a configuration file.

The **Certificate/ACL handler** handles requests that include a X.509 certificate for authentication. For authorization, it implements an access control list that defines the rights of the user to access operations and/or resources in the secured service. For the implementation of the X.509 certificate authentication the Spring Security framework is used. The access control list can be defined in a configuration file that is read when starting the proxy.

Below is an example of a SimplePermissions policy:

```
<SimplePermissions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://www.52north.org/security/simple-permission/1.0">
    <PermissionSet name="AWS1 Permission">
        <ResourceDomain value="https://s3-eu-west-1.amazonaws.com/testbed13-osm/" />
        <ActionDomain value="https://s3-eu-west-1.amazonaws.com/testbed13-osm/" />
        <SubjectDomain value="urn:n52:security:subject:role" />
        <Permission name="e.h.juerrens_GetObject">
            <Resource value="manhattan/osm-manhattan-roads.osm" />
            <Action value="GetObject" />
            <Subject value="e.h.juerrens" />
        </Permission>
    </PermissionSet>
</SimplePermissions>
```

In both cases, the **Capabilities injector** is used to add the security constraints for the service in the constraints element of the OperationsMetadata section in the Capabilities (see OGC OGC 16-048r1).

Below is an excerpt from a WFS capabilities document describing the access constraints:

```
<ows:Operation name="GetFeature">
    <ows:DCP>
        <ows:HTTP>
            <ows:Get xlink:href="https://tb12.dev.52north.org/security-
proxy/service/wfs" />
            <ows:Post xlink:href="https://tb12.dev.52north.org/security-
proxy/service/wfs" />
        </ows:HTTP>
    </ows:DCP>
    <ows:Parameter name="resultType">
        <ows:AllowedValues>
```

```
            <ows:Value>results</ows:Value>
            <ows:Value>hits</ows:Value>
        </ows:AllowedValues>
    </ows:Parameter>
    <ows:Parameter name="outputFormat">
        <ows:AllowedValues>
            <ows:Value>text/xml; subtype=gml/3.2</ows:Value>
            <ows:Value>GML2</ows:Value>
            ...
            <ows:Value>KML</ows:Value>
        </ows:AllowedValues>
    </ows:Parameter>
    <ows:Constraint name="PagingIsTransactionSafe">
        <ows:NoValues />
        <ows:DefaultValue>FALSE</ows:DefaultValue>
    </ows:Constraint>
    <ows:Constraint name="CountDefault">
        <ows:NoValues />
        <ows:DefaultValue>1000000</ows:DefaultValue>
    </ows:Constraint>
    <ows:Constraint>
        <ows:ValuesReference
            ows:reference="https://www.tb13.secure-
dimensions.de/authnCodeList#OAUTH2_BEARER_TOKEN">
            urn:ogc:def:security:authentication:ietf:6750:Bearer
        </ows:ValuesReference>
        <ows:Metadata xlink:role="AuthorizationServer"
            xlink:href="https://bpross-52n.eu.auth0.com/authorize" />
    </ows:Constraint>
    <ows:Constraint name="urn:ogc:def:security:oauth2:scopes">
        <ows:AllowedValues>
            <ows:Value>GetFeature</ows:Value>
            <ows:Value>GetFeature/TypeName=tb13:manhattan-streets-
reference</ows:Value>
            <ows:Value>GetFeature/TypeName=tb13:tnm-manhattan-streets</ows:Value>
            <ows:Value>GetFeature/TypeName=tb13:osm-manhattan-streets</ows:Value>
            <ows:Value>GetFeature/TypeName=tb13:lion-manhattan-streets</ows:Value>
            <ows:Value>GetFeature/TypeName=tb13:un-zataari-roads</ows:Value>
            <ows:Value>GetFeature/TypeName=tb13:tnm-manhattan-streets-
wgs84</ows:Value>
        </ows:AllowedValues>
        <ows:Meaning ows:reference="https://github.com/52North/testbed13-security-
proxy/blob/master/scopes.adoc"/>
    </ows:Constraint>
</ows:Operation>
```

12.6. OAuth scopes

Scopes were defined on two levels: (1) operation and (2) feature type/process level. On the operation level, scopes like Execute, DescribeProcess or GetFeature were defined. If a client is

authorized for the respective scope, it can call the operation with no limitation, e.g. call Execute on all processes or get all feature types. To allow more fine-grained access control, the scopes were defined on the feature type/process level. The scheme for these scopes is: Operation/AttributeName=AttributeValue, e.g. Execute/ProcessID=my.process or DescribeFeatureType/TypeName=my.feature.

The first level of scopes works well, as the available operations of an OGC Web services instance rarely change. The second level of scopes works well for a fixed set of feature types/processes. However, adding new feature types or processes (resulting in new scopes) is quite common and this can cause a problem. The clients initially request authorization for a given set of scopes. If new scopes are added within the authorization server, the client will not be authorized to access resources connected to these scopes. It would need to make a new authorization request for each additional scope.

Chapter 13. Recommendations

Using a BPMN engine as backend for a transactional WPS to execute workflows worked well with respect to creating new workflows/processes and starting them. The InsertProcess and Execute requests could be mapped straightforward to the encodings used by the REST endpoint of the workflow engine. The service task element of the BPMN 2.0 specification can be used to specify OCG Web services, although a client class was created in Java to do the actual communication with the services during the execution of the workflow. As the service task is a generic element, other BPMN engines will need a similar helper class. The class that was developed is specific to the Camunda BPMN Engine and cannot be used directly with other engines. Also, different BPMN engines handle the treatment of inputs and outputs to/from service tasks differently. Because of this restriction, the approach taken with Camunda will not work with other engines as well.

In this implementation, the identifier of the newly created processes was composed of the name-attribute of the process-element in the BPMN, together with the current time in milliseconds. This allowed for unique process identifiers but made discovery of the processes difficult, due to the similarity if the identifiers.

The security mechanism tested, X.509 certificates and Oauth, worked well in the workflow scenario. The engine was checking each service capabilities for constraints and did act accordingly. This was possible since the necessary grants and certificates were available directly in the workflow engine. The usage of new protected resources was possible in principle in the current implementation, e.g. by using the OAuth Authorization Code Grant Flow. However, this was not either required nor supported by the client.

Based on these findings, the following are recommendations for future work:

- Investigate how client functionality to communicate with OGC Web services (e.g. XML en- and decoding) can be made usable by different workflow engines
- Investigate a common approach to handle inputs/outputs defined in the BPMN
- Investigate how newly created workflows/WPS processes can be discovered, e.g. considering the identifier, title, abstract and/or keywords
- Investigate how new resources can be dynamically used in the workflow, e.g. by using the OAuth Authorization Code Grant Flow
- Investigate how to encode security aspects in BPMN
- Investigate how the StatusInfo document of WPS 2.0 can be extended, also for displaying intermediate results
- Create a transactional extension for WPS based on [8] and BPMN.

Appendix A: Listings

Larger sections code is included in the Appendix. This helps improve readability in the body of the ER.

1.1. Process specification

A BPMN 2 document that can be sent in a InsertProcess request and will be passed to the Camunda workflow engine.

```
<?xml version="1.0" encoding="UTF-8"?>
<bpmn2:definitions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:camunda="http://camunda.org/schema/1.0/bpmn"  xmlns:bpmn2=
"http://www.omg.org/spec/BPMN/20100524/MODEL" xmlns:bpmndi=
"http://www.omg.org/spec/BPMN/20100524/DI" xmlns:dc=
"http://www.omg.org/spec/DD/20100524/DC" xmlns:di=
"http://www.omg.org/spec/DD/20100524/DI" xmlns:ext="http://org.eclipse.bpmn2/ext"
xmlns:xs="http://www.w3.org/2001/XMLSchema" id="Definitions_1" exporter=
"org.eclipse.bpmn2.modeler.core" exporterVersion="1.3.1.Final-v20161006-1425-B58"
targetNamespace="http://org.eclipse.bpmn2/default/process">
  <bpmn2:itemDefinition id="ITEM_DEF_STRING" isCollection="false" structureRef=
"xs:string"/>
  <bpmn2:process id="$process_id$" name="ConflationWorkflow $timestamp$" isExecutable
="true">
    <bpmn2:ioSpecification id="InputOutputSpecification_1">
      <bpmn2:dataInput id="source_epsg" itemSubjectRef="ITEM_DEF_STRING" name=
"source_epsg"/>
      <bpmn2:dataInput id="data" itemSubjectRef="ITEM_DEF_STRING" name="data"/>
      <bpmn2:dataInput id="target_epsg" itemSubjectRef="ITEM_DEF_STRING" name=
"target_epsg"/>
      <bpmn2:dataInput id="inputTargetDataset" itemSubjectRef="ITEM_DEF_STRING" name=
"Vector dataset to be qualified"/>
      <bpmn2:dataInput id="inputReferenceDataset" itemSubjectRef="ITEM_DEF_STRING"
name="the reference dataset with positions according that represent the universe of
discourse"/>
      <bpmn2:dataInput id="inputReferenceField" itemSubjectRef="ITEM_DEF_STRING" name
="The name of the field to match to the reference dataset"/>
      <bpmn2:dataInput id="inputMetadataDocument" itemSubjectRef="ITEM_DEF_STRING"
name="Link to the metadata document, optional"/>
      <bpmn2:dataInput id="threshold" itemSubjectRef="ITEM_DEF_STRING" name="a
threshold value for the displacement allowed (same units as the input data)"/>
      <bpmn2:dataInput id="inputTargetField" itemSubjectRef="ITEM_DEF_STRING" name=
"The name of the field to match to the reference dataset"/>
      <bpmn2:dataInput id="INPUT1" itemSubjectRef="ITEM_DEF_STRING" name="INPUT1"/>
      <bpmn2:dataInput id="INPUT1_TRANSLATION" itemSubjectRef="ITEM_DEF_STRING" name=
"INPUT1_TRANSLATION"/>
      <bpmn2:dataInput id="INPUT1_DATA_QUALITY" itemSubjectRef="ITEM_DEF_STRING" name
="INPUT1_DATA_QUALITY"/>
```

```
      <bpmn2:dataInput id="INPUT2" itemSubjectRef="ITEM_DEF_STRING" name="INPUT2"/>
      <bpmn2:dataInput id="INPUT2_DATA_QUALITY" itemSubjectRef="ITEM_DEF_STRING" name
="INPUT2_DATA_QUALITY"/>
      <bpmn2:dataInput id="CONFLATION_TYPE" itemSubjectRef="ITEM_DEF_STRING" name=
"CONFLATION_TYPE"/>
      <bpmn2:dataInput id="MATCH_THRESHOLD" itemSubjectRef="ITEM_DEF_STRING" name=
"MATCH_THRESHOLD"/>
      <bpmn2:dataInput id="MISS_THRESHOLD" itemSubjectRef="ITEM_DEF_STRING" name=
"MISS_THRESHOLD"/>
      <bpmn2:dataOutput id="result" itemSubjectRef="ITEM_DEF_STRING" name="result"/>
      <bpmn2:dataOutput id="conformanceStatement" itemSubjectRef="ITEM_DEF_STRING"
name="The result of the test"/>
      <bpmn2:dataOutput id="outputMetadataDocument" itemSubjectRef="ITEM_DEF_STRING"
name="The output metadata document in full (only available if input metadata document
supplied)"/>
      <bpmn2:dataOutput id="outputMetadataChunk" itemSubjectRef="ITEM_DEF_STRING"
name="The updated quality element supplied as an XML chunk"/>
      <bpmn2:dataOutput id="CONFLATION_OUTPUT" itemSubjectRef="ITEM_DEF_STRING" name=
"CONFLATION_OUTPUT"/>
      <bpmn2:dataOutput id="CONFLATION_REPORT" itemSubjectRef="ITEM_DEF_STRING" name=
"CONFLATION_REPORT"/>
      <bpmn2:inputSet id="InputSet_69ce3a0d-98a3-900c-6652-1eb72ca94fcf" name="Input
Set"/>
      <bpmn2:outputSet id="OutputSet_39ce27ca-b185-9d66-357f-9ba6a6ffe311" name=
"Output Set"/>
    </bpmn2:ioSpecification>
    <bpmn2:startEvent id="StartEvent_1" name="Start workflow">
      <bpmn2:outgoing>SequenceFlow_Start</bpmn2:outgoing>
    </bpmn2:startEvent>
    <bpmn2:endEvent id="EndEvent_1" name="End workflow">
      <bpmn2:incoming>SequenceFlow_End</bpmn2:incoming>
    </bpmn2:endEvent>
    <bpmn2:serviceTask id="serviceTask-
org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm"
camunda:class="org.n52.testbed13.workflow.WPSClient" name=
"org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm"
implementation="http://ows.dev.52north.org:8080/SecurityProxy/service/wps">
      <bpmn2:incoming>SequenceFlow_Start</bpmn2:incoming>

<bpmn2:outgoing>SequenceFlow_org.n52.geoprocessing.geotools.algorithm.CoordinateTransf
ormationAlgorithm</bpmn2:outgoing>
      <bpmn2:ioSpecification id=
"InputOutputSpecification_org.n52.geoprocessing.geotools.algorithm.CoordinateTransform
ationAlgorithm">
        <bpmn2:dataInput id="DataInput_source_epsg" itemSubjectRef="ITEM_DEF_STRING"
name="source_epsg"/>
        <bpmn2:dataInput id="DataInput_data" itemSubjectRef="ITEM_DEF_STRING" name=
"data"/>
        <bpmn2:dataInput id="DataInput_target_epsg" itemSubjectRef="ITEM_DEF_STRING"
name="target_epsg"/>
        <bpmn2:dataOutput id="DataOutput_result" itemSubjectRef="ITEM_DEF_STRING"
```

```
name="result"/>
        <bpmn2:inputSet id="InputSet_3b355c27-1fb0-4323-ea4d-a65e1b8fc0f6" name="Input
Set">
          <bpmn2:dataInputRefs>DataInput_target_epsg</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_source_epsg</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_data</bpmn2:dataInputRefs>
        </bpmn2:inputSet>
        <bpmn2:outputSet id="OutputSet_5ca5e45e-2a23-8479-7504-0a029b41cdf1" name=
"Output Set">
          <bpmn2:dataOutputRefs>DataOutput_result</bpmn2:dataOutputRefs>
        </bpmn2:outputSet>
      </bpmn2:ioSpecification>
      <bpmn2:dataInputAssociation id="DataInputAssociation_target_epsg">
        <bpmn2:sourceRef>target_epsg</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_target_epsg</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_source_epsg">
        <bpmn2:sourceRef>source_epsg</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_source_epsg</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_data">
        <bpmn2:sourceRef>data</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_data</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataOutputAssociation id="DataOutputAssociation_result">
        <bpmn2:sourceRef>DataOutput_result</bpmn2:sourceRef>
        <bpmn2:targetRef>result</bpmn2:targetRef>
      </bpmn2:dataOutputAssociation>
    </bpmn2:serviceTask>
    <bpmn2:sequenceFlow id="SequenceFlow_Start" sourceRef="StartEvent_1" targetRef=
"serviceTask-
org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm"/>
    <bpmn2:serviceTask id="serviceTask-
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy" camunda:class=
"org.n52.testbed13.workflow.WPSClient" name=
"iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy" implementation
="https://tb12.dev.52north.org/data-quality-wps-proxy/service/wps">

<bpmn2:incoming>SequenceFlow_org.n52.geoprocessing.geotools.algorithm.CoordinateTransf
ormationAlgorithm</bpmn2:incoming>

<bpmn2:outgoing>iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy</
bpmn2:outgoing>
      <bpmn2:ioSpecification id=
"InputOutputSpecification_iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositional
Accuracy">
        <bpmn2:dataInput id="DataInput_inputTargetDataset" itemSubjectRef=
"ITEM_DEF_STRING" name="Vector dataset to be qualified"/>
        <bpmn2:dataInput id="DataInput_inputReferenceDataset" itemSubjectRef=
"ITEM_DEF_STRING" name="Lookup field for the target dataset"/>
        <bpmn2:dataInput id="DataInput_inputReferenceField" itemSubjectRef=
```

```
"ITEM_DEF_STRING" name="Lookup field for the target dataset"/>
        <bpmn2:dataInput id="DataInput_inputMetadataDocument" itemSubjectRef=
"ITEM_DEF_STRING" name="Link to the metadata document, optional"/>
        <bpmn2:dataInput id="DataInput_threshold" itemSubjectRef="ITEM_DEF_STRING"
name="Threshold for the failure rate (percentage)"/>
        <bpmn2:dataInput id="DataInput_inputTargetField" itemSubjectRef=
"ITEM_DEF_STRING" name="The format described as a string (for example MMDDYYYY or DD-
MM-YYYY) must be the same for both start and end date fields"/>
        <bpmn2:dataOutput id="DataOutput_conformanceStatement" itemSubjectRef=
"ITEM_DEF_STRING" name="The result of the test"/>
        <bpmn2:dataOutput id="DataOutput_outputMetadataDocument" itemSubjectRef=
"ITEM_DEF_STRING" name="The output metadata document in full (only available if input
metadata document supplied)"/>
        <bpmn2:dataOutput id="DataOutput_outputMetadataChunk" itemSubjectRef=
"ITEM_DEF_STRING" name="The updated quality element supplied as an XML chunk"/>
        <bpmn2:inputSet id="InputSet_bd8bb0db-6de4-f50d-a1ff-3ad7208cee29" name="Input
Set">
          <bpmn2:dataInputRefs>DataInput_threshold</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_inputReferenceField</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_inputMetadataDocument</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_inputReferenceDataset</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_inputTargetField</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_inputTargetDataset</bpmn2:dataInputRefs>
        </bpmn2:inputSet>
        <bpmn2:outputSet id="OutputSet_6142b575-e869-2615-cae9-40c39b8d9ad4" name=
"Output Set">
          <bpmn2:dataOutputRefs>DataOutput_conformanceStatement</bpmn2:dataOutputRefs>
          <bpmn2:dataOutputRefs>
DataOutput_outputMetadataDocument</bpmn2:dataOutputRefs>
          <bpmn2:dataOutputRefs>DataOutput_outputMetadataChunk</bpmn2:dataOutputRefs>
        </bpmn2:outputSet>
      </bpmn2:ioSpecification>
      <bpmn2:dataInputAssociation id="DataInputAssociation_threshold">
        <bpmn2:sourceRef>threshold</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_threshold</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_inputReferenceField">
        <bpmn2:sourceRef>inputReferenceField</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_inputReferenceField</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_inputMetadataDocument">
        <bpmn2:sourceRef>inputMetadataDocument</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_inputMetadataDocument</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_inputReferenceDataset">
        <bpmn2:sourceRef>inputReferenceDataset</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_inputReferenceDataset</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_inputTargetField">
        <bpmn2:sourceRef>inputTargetField</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_inputTargetField</bpmn2:targetRef>
```

```
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_inputTargetDataset">
        <bpmn2:sourceRef>inputTargetDataset</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_inputTargetDataset</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataOutputAssociation id="DataOutputAssociation_conformanceStatement">
        <bpmn2:sourceRef>DataOutput_conformanceStatement</bpmn2:sourceRef>
        <bpmn2:targetRef>conformanceStatement</bpmn2:targetRef>
      </bpmn2:dataOutputAssociation>
      <bpmn2:dataOutputAssociation id="DataOutputAssociation_outputMetadataDocument">
        <bpmn2:sourceRef>DataOutput_outputMetadataDocument</bpmn2:sourceRef>
        <bpmn2:targetRef>outputMetadataDocument</bpmn2:targetRef>
      </bpmn2:dataOutputAssociation>
      <bpmn2:dataOutputAssociation id="DataOutputAssociation_outputMetadataChunk">
        <bpmn2:sourceRef>DataOutput_outputMetadataChunk</bpmn2:sourceRef>
        <bpmn2:targetRef>outputMetadataChunk</bpmn2:targetRef>
      </bpmn2:dataOutputAssociation>
    </bpmn2:serviceTask>
    <bpmn2:sequenceFlow id=
"SequenceFlow_org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorit
hm" sourceRef="serviceTask-
org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm" targetRef
="serviceTask-iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy"/>
    <bpmn2:serviceTask id="serviceTask-testbed13.dsi.HootenannyConflation"
camunda:class="org.n52.testbed13.workflow.WPSClient" name=
"testbed13.dsi.HootenannyConflation" implementation=
"https://tb12.dev.52north.org/conflation-wps-proxy/service/wps">

<bpmn2:incoming>iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy</
bpmn2:incoming>
      <bpmn2:outgoing>SequenceFlow_End</bpmn2:outgoing>
      <bpmn2:ioSpecification id=
"InputOutputSpecification_testbed13.dsi.HootenannyConflation">
        <bpmn2:dataInput id="DataInput_INPUT1" itemSubjectRef="ITEM_DEF_STRING" name=
"INPUT1"/>
        <bpmn2:dataInput id="DataInput_INPUT1_TRANSLATION" itemSubjectRef=
"ITEM_DEF_STRING" name="INPUT1_TRANSLATION"/>
        <bpmn2:dataInput id="DataInput_INPUT1_DATA_QUALITY" itemSubjectRef=
"ITEM_DEF_STRING" name="INPUT1_DATA_QUALITY"/>
        <bpmn2:dataInput id="DataInput_INPUT2" itemSubjectRef="ITEM_DEF_STRING" name=
"INPUT2"/>
        <bpmn2:dataInput id="DataInput_INPUT2_DATA_QUALITY" itemSubjectRef=
"ITEM_DEF_STRING" name="INPUT2_DATA_QUALITY"/>
        <bpmn2:dataInput id="DataInput_CONFLATION_TYPE" itemSubjectRef=
"ITEM_DEF_STRING" name="CONFLATION_TYPE"/>
        <bpmn2:dataInput id="DataInput_MATCH_THRESHOLD" itemSubjectRef=
"ITEM_DEF_STRING" name="MATCH_THRESHOLD"/>
        <bpmn2:dataInput id="DataInput_MISS_THRESHOLD" itemSubjectRef="
ITEM_DEF_STRING" name="MISS_THRESHOLD"/>
        <bpmn2:dataOutput id="DataOutput_CONFLATION_OUTPUT" itemSubjectRef=
"ITEM_DEF_STRING" name="CONFLATION_OUTPUT"/>
```

```
        <bpmn2:dataOutput id="DataOutput_CONFLATION_REPORT" itemSubjectRef=
"ITEM_DEF_STRING" name="CONFLATION_REPORT"/>
        <bpmn2:inputSet id="InputSet_bfddea9b-2743-6d7e-26a7-75fda90fcd2f" name="Input
Set">
          <bpmn2:dataInputRefs>DataInput_INPUT2</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_MATCH_THRESHOLD</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_INPUT1</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_MISS_THRESHOLD</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_INPUT1_DATA_QUALITY</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_INPUT1_TRANSLATION</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_INPUT2_DATA_QUALITY</bpmn2:dataInputRefs>
          <bpmn2:dataInputRefs>DataInput_CONFLATION_TYPE</bpmn2:dataInputRefs>
        </bpmn2:inputSet>
        <bpmn2:outputSet id="OutputSet_9cf6e47e-72c6-d22d-8b07-93076c00a9da" name=
"Output Set">
          <bpmn2:dataOutputRefs>DataOutput_CONFLATION_OUTPUT</bpmn2:dataOutputRefs>
          <bpmn2:dataOutputRefs>DataOutput_CONFLATION_REPORT</bpmn2:dataOutputRefs>
        </bpmn2:outputSet>
      </bpmn2:ioSpecification>
      <bpmn2:dataInputAssociation id="DataInputAssociation_INPUT2">
        <bpmn2:sourceRef>INPUT2</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_INPUT2</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_MATCH_THRESHOLD">
        <bpmn2:sourceRef>MATCH_THRESHOLD</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_MATCH_THRESHOLD</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_INPUT1">
        <bpmn2:sourceRef>INPUT1</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_INPUT1</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_MISS_THRESHOLD">
        <bpmn2:sourceRef>MISS_THRESHOLD</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_MISS_THRESHOLD</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_INPUT1_DATA_QUALITY">
        <bpmn2:sourceRef>INPUT1_DATA_QUALITY</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_INPUT1_DATA_QUALITY</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_INPUT1_TRANSLATION">
        <bpmn2:sourceRef>INPUT1_TRANSLATION</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_INPUT1_TRANSLATION</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_INPUT2_DATA_QUALITY">
        <bpmn2:sourceRef>INPUT2_DATA_QUALITY</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_INPUT2_DATA_QUALITY</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
      <bpmn2:dataInputAssociation id="DataInputAssociation_CONFLATION_TYPE">
        <bpmn2:sourceRef>CONFLATION_TYPE</bpmn2:sourceRef>
        <bpmn2:targetRef>DataInput_CONFLATION_TYPE</bpmn2:targetRef>
      </bpmn2:dataInputAssociation>
```

```
      <bpmn2:dataOutputAssociation id="DataOutputAssociation_CONFLATION_OUTPUT">
        <bpmn2:sourceRef>DataOutput_CONFLATION_OUTPUT</bpmn2:sourceRef>
        <bpmn2:targetRef>CONFLATION_OUTPUT</bpmn2:targetRef>
      </bpmn2:dataOutputAssociation>
      <bpmn2:dataOutputAssociation id="DataOutputAssociation_CONFLATION_REPORT">
        <bpmn2:sourceRef>DataOutput_CONFLATION_REPORT</bpmn2:sourceRef>
        <bpmn2:targetRef>CONFLATION_REPORT</bpmn2:targetRef>
      </bpmn2:dataOutputAssociation>
    </bpmn2:serviceTask>
    <bpmn2:sequenceFlow id=
"iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy" sourceRef=
"serviceTask-iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy"
targetRef="serviceTask-testbed13.dsi.HootenannyConflation"/>
    <bpmn2:sequenceFlow id="SequenceFlow_End" sourceRef="serviceTask-
testbed13.dsi.HootenannyConflation" targetRef="EndEvent_1"/>
  </bpmn2:process>
  <bpmndi:BPMNDiagram id="BPMNDiagram_1">
    <bpmndi:BPMNPlane id="BPMNPlane_Process_1" bpmnElement="test-2">
      <bpmndi:BPMNShape id="BPMNShape_StartEvent_1" bpmnElement="StartEvent_1">
        <dc:Bounds height="36.0" width="36.0" x="50.0" y="57.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_1" labelStyle="BPMNLabelStyle_1">
          <dc:Bounds height="15.0" width="76.0" x="30.0" y="93.0"/>
        </bpmndi:BPMNLabel>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape id="BPMNShape_EndEvent_1" bpmnElement="EndEvent_1">
        <dc:Bounds height="36.0" width="36.0" x="616.0" y="57.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_2" labelStyle="BPMNLabelStyle_1">
          <dc:Bounds height="15.0" width="73.0" x="598.0" y="93.0"/>
        </bpmndi:BPMNLabel>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape id="BPMNShape_ServiceTask_1" bpmnElement="serviceTask-
org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm">
        <dc:Bounds height="50.0" width="110.0" x="136.0" y="50.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_3" labelStyle="BPMNLabelStyle_1">
          <dc:Bounds height="90.0" width="108.0" x="137.0" y="30.0"/>
        </bpmndi:BPMNLabel>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape id="BPMNShape_ServiceTask_2" bpmnElement="serviceTask-
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy">
        <dc:Bounds height="50.0" width="110.0" x="296.0" y="50.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_4" labelStyle="BPMNLabelStyle_1">
          <dc:Bounds height="60.0" width="107.0" x="297.0" y="45.0"/>
        </bpmndi:BPMNLabel>
      </bpmndi:BPMNShape>
      <bpmndi:BPMNShape id="BPMNShape_ServiceTask_3" bpmnElement="serviceTask-
testbed13.dsi.HootenannyConflation">
        <dc:Bounds height="50.0" width="110.0" x="456.0" y="50.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_5" labelStyle="BPMNLabelStyle_1">
          <dc:Bounds height="30.0" width="108.0" x="457.0" y="60.0"/>
        </bpmndi:BPMNLabel>
      </bpmndi:BPMNShape>
```

```
      <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_1" bpmnElement="SequenceFlow_Start"
sourceElement="BPMNShape_StartEvent_1" targetElement="BPMNShape_ServiceTask_1">
        <di:waypoint xsi:type="dc:Point" x="86.0" y="75.0"/>
        <di:waypoint xsi:type="dc:Point" x="111.0" y="75.0"/>
        <di:waypoint xsi:type="dc:Point" x="136.0" y="75.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_6"/>
      </bpmndi:BPMNEdge>
      <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_2" bpmnElement="SequenceFlow_End"
sourceElement="BPMNShape_ServiceTask_3" targetElement="BPMNShape_EndEvent_1">
        <di:waypoint xsi:type="dc:Point" x="566.0" y="75.0"/>
        <di:waypoint xsi:type="dc:Point" x="591.0" y="75.0"/>
        <di:waypoint xsi:type="dc:Point" x="616.0" y="75.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_7"/>
      </bpmndi:BPMNEdge>
      <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_3" bpmnElement=
"SequenceFlow_org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorit
hm" sourceElement="BPMNShape_ServiceTask_1" targetElement="BPMNShape_ServiceTask_2">
        <di:waypoint xsi:type="dc:Point" x="246.0" y="75.0"/>
        <di:waypoint xsi:type="dc:Point" x="271.0" y="75.0"/>
        <di:waypoint xsi:type="dc:Point" x="296.0" y="75.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_8"/>
      </bpmndi:BPMNEdge>
      <bpmndi:BPMNEdge id="BPMNEdge_SequenceFlow_4" bpmnElement=
"iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy" sourceElement=
"BPMNShape_ServiceTask_2" targetElement="BPMNShape_ServiceTask_3">
        <di:waypoint xsi:type="dc:Point" x="406.0" y="75.0"/>
        <di:waypoint xsi:type="dc:Point" x="431.0" y="75.0"/>
        <di:waypoint xsi:type="dc:Point" x="456.0" y="75.0"/>
        <bpmndi:BPMNLabel id="BPMNLabel_9"/>
      </bpmndi:BPMNEdge>
    </bpmndi:BPMNPlane>
    <bpmndi:BPMNLabelStyle id="BPMNLabelStyle_1">
      <dc:Font name="arial" size="9.0"/>
    </bpmndi:BPMNLabelStyle>
  </bpmndi:BPMNDiagram>
</bpmn2:definitions>
```

1.2. Conflation service process description

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:ProcessDescriptions xmlns:wps="http://www.opengis.net/wps/1.0.0"
    xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ows="http://www.opengis.net/ows/1.1"
    xsi:schemaLocation="http://www.opengis.net/wps/1.0.0
http://schemas.opengis.net/wps/1.0.0/wpsDescribeProcess_response.xsd"
    xml:lang="en-US" service="WPS" version="1.0.0">
    <ProcessDescription statusSupported="true"
        storeSupported="true" wps:processVersion="1.0.0">
```

```
        <ows:Identifier>testbed13.dsi.HootenannyConflation</ows:Identifier>
        <ows:Title>Hootenanny Conflation Process</ows:Title>
        <ows:Metadata
            xlink:role="http://www.opengis.net/spec/wps/2.0/def/process-
profile/concept"
            xlink:href="http://52north.github.io/wps-
profileregistry/concept/conflation" />
        <ows:Metadata
            xlink:role="http://www.opengis.net/spec/wps/2.0/def/process-
profile/generic"
            xlink:href="http://52north.github.io/wps-
profileregistry/generic/conflation" />
        <ows:Metadata
            xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
            xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation" />
        <DataInputs>
            <Input maxOccurs="1" minOccurs="1">
                <ows:Identifier>INPUT1</ows:Identifier>
                <ows:Title>INPUT1</ows:Title>
                <ows:Abstract>Conflation input 1</ows:Abstract>
                <ows:Metadata
                    xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
                    xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation#input1" />
                <ComplexData>
                    <Default>
                        <Format>
                            <MimeType>application/x-zipped-shp</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>application/x-zipped-shp</MimeType>
                        </Format>
                        <Format>
                            <MimeType>application/x-openstreetmap+xml</MimeType>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-shp</MimeType>
                            <Encoding>base64</Encoding>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-gdb</MimeType>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-gdb</MimeType>
                            <Encoding>base64</Encoding>
                        </Format>
```

```
                    </Supported>
                </ComplexData>
            </Input>
            <Input maxOccurs="1" minOccurs="0">
                <ows:Identifier>INPUT1_TRANSLATION</ows:Identifier>
                <ows:Title>INPUT1_TRANSLATION</ows:Title>
                <ows:Abstract>Translation file for conflation input 1</ows:Abstract>
                <ows:Metadata
                    xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
                    xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation#input1_translation" />
                <ComplexData>
                    <Default>
                        <Format>
                            <MimeType>text/x-script.phyton</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>text/x-script.phyton</MimeType>
                        </Format>
                        <Format>
                            <MimeType>text/plain</MimeType>
                        </Format>
                    </Supported>
                </ComplexData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>INPUT1_DATA_QUALITY</ows:Identifier>
                <ows:Title>INPUT1_DATA_QUALITY</ows:Title>
                <ows:Abstract>DQ_AbsoluteExternalPositionalAccuracy object for input
                    1 as XML</ows:Abstract>
                <ComplexData>
                    <Default>
                        <Format>
                            <MimeType>text/xml</MimeType>
                            <Encoding>UTF-8</Encoding>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>text/plain</MimeType>
                            <Encoding>UTF-8</Encoding>
                        </Format>
                    </Supported>
                </ComplexData>
            </Input>
            <Input maxOccurs="1" minOccurs="1">
                <ows:Identifier>INPUT2</ows:Identifier>
                <ows:Title>INPUT2</ows:Title>
```

```
                <ows:Abstract>Conflation input 2</ows:Abstract>
                <ows:Metadata
                    xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
                    xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation#input2" />
                <ComplexData>
                    <Default>
                        <Format>
                            <MimeType>application/x-openstreetmap+xml</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>text/xml</MimeType>
                        </Format>
                        <Format>
                            <MimeType>application/x-openstreetmap+xml</MimeType>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-shp</MimeType>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-shp</MimeType>
                            <Encoding>base64</Encoding>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-gdb</MimeType>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-gdb</MimeType>
                            <Encoding>base64</Encoding>
                        </Format>
                    </Supported>
                </ComplexData>
            </Input>
            <Input minOccurs="1" maxOccurs="1">
                <ows:Identifier>INPUT2_DATA_QUALITY</ows:Identifier>
                <ows:Title>INPUT2_DATA_QUALITY</ows:Title>
                <ows:Abstract>DQ_AbsoluteExternalPositionalAccuracy object for input
                    2 as XML</ows:Abstract>
                <ComplexData>
                    <Default>
                        <Format>
                            <MimeType>text/xml</MimeType>
                            <Encoding>UTF-8</Encoding>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>text/plain</MimeType>
```

```
                        <Encoding>UTF-8</Encoding>
                    </Format>
                </Supported>
            </ComplexData>
        </Input>
        <Input maxOccurs="1" minOccurs="0">
            <ows:Identifier>CONFLATION_TYPE</ows:Identifier>
            <ows:Title>CONFLATION_TYPE</ows:Title>
            <ows:Metadata
                xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
                xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation#conflation_type" />
            <LiteralData>
                <ows:DataType ows:reference="xs:string" />
                <ows:AllowedValues>
                    <ows:Value>average</ows:Value>
                    <ows:Value>reference</ows:Value>
                </ows:AllowedValues>
            </LiteralData>
        </Input>
        <Input maxOccurs="1" minOccurs="0">
            <ows:Identifier>MATCH_THRESHOLD</ows:Identifier>
            <ows:Title>MATCH_THRESHOLD</ows:Title>
            <ows:Abstract>The threshold for calling a relationship a match.
                Defaults to 0.6. The higher the value the lower the TPR, but
likely
                also the lower the FPR.</ows:Abstract>
            <ows:Metadata
                xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
                xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation#miss_threshold" />
            <LiteralData>
                <ows:DataType ows:reference="xs:double" />
                <ows:AnyValue />
                <DefaultValue>0.6</DefaultValue>
            </LiteralData>
        </Input>
        <Input maxOccurs="1" minOccurs="0">
            <ows:Identifier>MISS_THRESHOLD</ows:Identifier>
            <ows:Title>MISS_THRESHOLD</ows:Title>
            <ows:Abstract>The threshold for calling a relationship a miss.
                Defaults to 0.6. The higher the value the lower the TNR, but
likely
                also the lower the FNR.</ows:Abstract>
            <ows:Metadata
                xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
                xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation#match_threshold" />
```

```
                <LiteralData>
                    <ows:DataType ows:reference="xs:double" />
                    <ows:AnyValue />
                    <DefaultValue>0.6</DefaultValue>
                </LiteralData>
            </Input>
        </DataInputs>
        <ProcessOutputs>
            <Output>
                <ows:Identifier>CONFLATION_OUTPUT</ows:Identifier>
                <ows:Title>CONFLATION_OUTPUT</ows:Title>
                <ows:Metadata
                    xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
                    xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation#conflation_output" />
                <ComplexOutput>
                    <Default>
                        <Format>
                            <MimeType>application/x-zipped-shp</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>application/x-zipped-shp</MimeType>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-shp</MimeType>
                            <Encoding>base64</Encoding>
                        </Format>
                        <Format>
                            <MimeType>application/x-zipped-gdb</MimeType>
                        </Format>
                        <Format>
                            <MimeType>text/xml</MimeType>
<Schema>http://schemas.opengis.net/gml/3.1.1/base/feature.xsd</Schema>
                        </Format>
                        <Format>
                            <MimeType>application/vnd.geo+json</MimeType>
                        </Format>
                    </Supported>
                </ComplexOutput>
            </Output>
            <Output>
                <ows:Identifier>CONFLATION_REPORT</ows:Identifier>
                <ows:Title>CONFLATION_REPORT</ows:Title>
                <ows:Metadata
                    xlink:role=
"http://www.opengis.net/spec/wps/2.0/def/process/description/documentation"
                    xlink:href="http://52north.github.io/wps-
profileregistry/implementing/hootenanny-conflation#conflation_report" />
```

```
                <ComplexOutput>
                    <Default>
                        <Format>
                            <MimeType>text/plain</MimeType>
                        </Format>
                    </Default>
                    <Supported>
                        <Format>
                            <MimeType>text/plain</MimeType>
                        </Format>
                    </Supported>
                </ComplexOutput>
            </Output>
        </ProcessOutputs>
    </ProcessDescription>
</wps:ProcessDescriptions>
```

1.3. Workflow engine log

```
2017-10-04 13:44:38,723 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Starting workflow execution.
2017-10-04 13:44:38,723 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Checking access to wps capabilities:
http://ows.dev.52north.org:8080/SecurityProxy/service/wps?request=GetCapabilities&serv
ice=WPS&acceptVersions=2.0.0
2017-10-04 13:44:39,020 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got status code 200
2017-10-04 13:44:39,020 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting capabilities from:
http://ows.dev.52north.org:8080/SecurityProxy/service/wps
2017-10-04 13:44:39,597 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting process description for process:
org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm from:
http://ows.dev.52north.org:8080/SecurityProxy/service/wps
2017-10-04 13:44:40,424 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got process: Id:
org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm
Title: org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm
            Abstract:

2017-10-04 13:44:40,424 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Creating execute request for process
org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgorithm
2017-10-04 13:44:40,424 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: <Execute version="2.0.0" service="WPS"
response="document" mode="sync" xmlns="http://www.opengis.net/wps/2.0"
xmlns:ns="http://www.opengis.net/ows/2.0" xmlns:xlin="http://www.w3.org/1999/xlink">

<ns:Identifier>org.n52.geoprocessing.geotools.algorithm.CoordinateTransformationAlgori
```

```
thm</ns:Identifier>
  <Input id="source_epsg">
    <Data mimeType="text/plain">EPSG:4326</Data>
  </Input>
  <Input id="target_epsg">
    <Data mimeType="text/plain">EPSG:32118</Data>
  </Input>
  <Input id="data">
    <Reference mimeType="text/xml"
schema="http://schemas.opengis.net/gml/3.1.1/base/feature.xsd"
xlin:href="https://tb12.dev.52north.org/security-
proxy/service/wfs?service=WFS&amp;version=1.0.0&amp;request=GetFeature&amp;typeName=tb
13:tnm-manhattan-streets-
wgs84&amp;maxFeatures=50&amp;outputFormat=gml3&amp;maxFeatures=50"/>
  </Input>
  <Output id="result" mimeType="text/xml"
schema="http://schemas.opengis.net/gml/3.1.1/base/feature.xsd"
transmission="reference"/>
</Execute>
2017-10-04 13:44:40,424 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Checking execute operation.
2017-10-04 13:44:40,970 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got status code 401
2017-10-04 13:44:40,970 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got HTTP error code: 401
2017-10-04 13:44:40,970 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting access token.
2017-10-04 13:44:40,970 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.OAuth2Client: Sending request for access token:
{"client_id":"AhsXezU....quMzt39mO2ID","client_secret":"LScXw6WH....8roo6gXLG8gnE4","a
udience":"http://ows.dev.52north.org:8080/javaps/service","grant_type":"client_credent
ials"}
2017-10-04 13:44:41,017 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.OAuth2Client: Received:
{"access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJ....b3jzJy1p8aMrCdqk7GfvAav1jcCmQxJbA","sc
ope":"Execute/ProcessID=org.n52.geoprocessing.geotools.algorithm.CoordinateTransformat
ionAlgorithm Execute","expires_in":86400,"token_type":"Bearer"}
2017-10-04 13:44:41,017 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.util.JSONUtil: Trying to parse JSON String:
{"access_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJ....b3jzJy1p8aMrCdqk7GfvAav1jcCmQxJbA","sc
ope":"Execute/ProcessID=org.n52.geoprocessing.geotools.algorithm.CoordinateTransformat
ionAlgorithm Execute","expires_in":86400,"token_type":"Bearer"}
2017-10-04 13:44:41,017 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Successfully requested access token.
2017-10-04 13:44:45,229 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Response: <wps:Result
xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd" xmlns:wps="http://www.opengis.net/wps/2.0"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <wps:JobID>998b0f75-07ec-4a85-93cd-d480ff97da0a</wps:JobID>
```

```
  <wps:ExpirationDate>2017-10-04T15:44:42Z</wps:ExpirationDate>
  <wps:Output id="result">
    <wps:Reference
xlink:href="http://ows.dev.52north.org:8080/SecurityProxy/service/wps?request=GetOutpu
t&amp;amp;version=2.0.0&amp;amp;service=WPS&amp;amp;id=998b0f75-07ec-4a85-93cd-d480ff97da0a"
mimeType="text/xml" schema="http://schemas.opengis.net/gml/3.1.1/base/feature.xsd"/>
  </wps:Output>
</wps:Result>
2017-10-04 13:44:45,229 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting WPS: https://tb12.dev.52north.org/data-
quality-wps-proxy/service/wps with process:
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy
2017-10-04 13:44:45,229 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Checking access to wps capabilities:
https://tb12.dev.52north.org/data-quality-wps-
proxy/service/wps?request=GetCapabilities&service=WPS&acceptVersions=2.0.0
2017-10-04 13:44:45,244 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got status code 400
2017-10-04 13:44:45,244 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Cannot access wps, trying to set client
certificate: ogc-tb-13-x509-test-client.p12
2017-10-04 13:44:45,244 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting capabilities from:
https://tb12.dev.52north.org/data-quality-wps-proxy/service/wps
2017-10-04 13:44:45,291 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting process description for process:
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy from:
https://tb12.dev.52north.org/data-quality-wps-proxy/service/wps
2017-10-04 13:44:45,307 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got process: Id:
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy
Title: iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy
            Abstract:

2017-10-04 13:44:45,307 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Creating execute request for process
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy
2017-10-04 13:44:45,307 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got result from previous process:
http://ows.dev.52north.org:8080/SecurityProxy/service/wps?request=GetOutput&version=2.
0.0&service=WPS&id=998b0f75-07ec-4a85-93cd-d480ff97da0a
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: <Execute version="2.0.0" service="WPS"
response="document" mode="sync" xmlns="http://www.opengis.net/wps/2.0"
xmlns:ns="http://www.opengis.net/ows/2.0" xmlns:xlin="http://www.w3.org/1999/xlink">

<ns:Identifier>iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy</n
s:Identifier>
  <Input id="inputTargetDataset">
    <Reference mimeType="text/xml; subtype=gml/3.1.0"
schema="http://schemas.opengis.net/gml/3.1.0/base/feature.xsd"
```

```
xlin:href="http://ows.dev.52north.org:8080/SecurityProxy/service/wps?request=GetOutput
&amp;version=2.0.0&amp;service=WPS&amp;id=998b0f75-07ec-4a85-93cd-d480ff97da0a"/>
  </Input>
  <Input id="inputReferenceDataset">
    <Reference mimeType="text/xml; subtype=gml/3.1.0"
schema="http://schemas.opengis.net/gml/3.1.0/base/feature.xsd"
xlin:href="https://tb12.dev.52north.org/security-
proxy/service/wfs?service=WFS&amp;version=1.0.0&amp;request=GetFeature&amp;typeName=tb
13:manhattan-streets-reference&amp;outputFormat=gml3&amp;maxFeatures=50"/>
  </Input>
  <Input id="threshold">
    <Data mimeType="text/xml">
      <LiteralValue>10</LiteralValue>
    </Data>
  </Input>
  <Input id="inputTargetField">
    <Data mimeType="text/xml">
      <LiteralValue>TEMP</LiteralValue>
    </Data>
  </Input>
  <Input id="inputReferenceField">
    <Data mimeType="text/xml">
      <LiteralValue>TEMP</LiteralValue>
    </Data>
  </Input>
  <Output id="outputMetadataChunk" mimeType="text/xml" transmission="value"/>
</Execute>
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Checking execute operation.
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got status code 401
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got HTTP error code: 401
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting access token.
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.OAuth2Client: Sending request for access token:
{"client_id":"AhsXe....YquMzt39mO2ID","client_secret":"LScXw6WHdP....8roo6gXLG8gnE4","
audience":"http://tb12.dev.52north.org/data-quality-
wps/WebProcessingService","grant_type":"client_credentials"}
2017-10-04 13:44:45,353 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.OAuth2Client: Received:
{"access_token":"eyJ0eXAiOiJKV....3o1DF-
tI7syTmw","scope":"Execute","expires_in":86400,"token_type":"Bearer"}
2017-10-04 13:44:45,353 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.util.JSONUtil: Trying to parse JSON String:
{"access_token":"eyJ0eXAiOiJKV....3o1DF-
tI7syTmw","scope":"Execute","expires_in":86400,"token_type":"Bearer"}
2017-10-04 13:44:45,353 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Successfully requested access token.
2017-10-04 13:44:47,459 [http-nio-8081-exec-4] INFO
```

```
org.n52.testbed13.workflow.LogUtil: Response: <wps:Result
xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd" xmlns:wps="http://www.opengis.net/wps/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <wps:JobID>5f3fa6f8-f2aa-42ce-b397-f572033949f4</wps:JobID>
  <wps:Output id="outputMetadataChunk">
    <wps:Data mimeType="text/xml">
      <DQ_AbsoluteExternalPositionalAccuracy>
        <nameOfMeasure>
          <CharacterString>Test of accuracy of the target data against an
authoritative reference</CharacterString>
        </nameOfMeasure>
        <evaluationMethodType>
          <DQ_EvaluationMethodTypeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#EvaluationMe
thodTypeCode" codeListValue="directExternal">Direct
external</DQ_EvaluationMethodTypeCode>
        </evaluationMethodType>
        <dateTime>
          <DateTime>2006-11-10T00:00:00</DateTime>
        </dateTime>
        <result>
          <DQ_ConformanceResult>
            <specification>
              <CI_Citation>
                <title>
                  <CharacterString>Accuracy of position test</CharacterString>
                </title>
                <date>
                  <CI_Date>
                    <date>
                      <Date>2017-10-04</Date>
                    </date>
                    <dateType>
                      <CI_DateTypeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#CI_DateTypeC
ode" codeListValue="creation">creation</CI_DateTypeCode>
                    </dateType>
                  </CI_Date>
                </date>
              </CI_Citation>
            </specification>
            <explanation>
              <CharacterString>The mean displacement from the authoritative data is
0.5778691972299712</CharacterString>
            </explanation>
            <pass>
              <Boolean>1</Boolean>
            </pass>
          </DQ_ConformanceResult>
        </result>
```

```
      </DQ_AbsoluteExternalPositionalAccuracy>
    </wps:Data>
  </wps:Output>
</wps:Result>
2017-10-04 13:44:47,459 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting WPS:
https://tb12.dev.52north.org/conflation-wps-proxy/service/wps with process:
testbed13.dsi.HootenannyConflation
2017-10-04 13:44:47,475 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Checking access to wps capabilities:
https://tb12.dev.52north.org/conflation-wps-
proxy/service/wps?request=GetCapabilities&service=WPS&acceptVersions=2.0.0
2017-10-04 13:44:47,475 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got status code 400
2017-10-04 13:44:47,475 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Cannot access wps, trying to set client
certificate: ogc-tb-13-x509-test-client.p12
2017-10-04 13:44:47,491 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting capabilities from:
https://tb12.dev.52north.org/conflation-wps-proxy/service/wps
2017-10-04 13:44:47,522 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting process description for process:
testbed13.dsi.HootenannyConflation from: https://tb12.dev.52north.org/conflation-wps-
proxy/service/wps
2017-10-04 13:44:47,553 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got process: Id:
testbed13.dsi.HootenannyConflation
Title: Hootenanny Conflation Process
            Abstract:

2017-10-04 13:44:47,553 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Creating execute request for process
testbed13.dsi.HootenannyConflation
2017-10-04 13:44:47,553 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got result from previous process:
<DQ_AbsoluteExternalPositionalAccuracy><nameOfMeasure><CharacterString>Test of
accuracy of the target data against an authoritative
reference</CharacterString></nameOfMeasure><evaluationMethodType><DQ_EvaluationMethodT
ypeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#EvaluationMe
thodTypeCode" codeListValue="directExternal">Direct
external</DQ_EvaluationMethodTypeCode></evaluationMethodType><dateTime><DateTime>2006-
11-
10T00:00:00</DateTime></dateTime><result><DQ_ConformanceResult><specification><CI_Cita
tion><title><CharacterString>Accuracy of position
test</CharacterString></title><date><CI_Date><date><Date>2017-10-
04</Date></date><dateType><CI_DateTypeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#CI_DateTypeC
ode"
codeListValue="creation">creation</CI_DateTypeCode></dateType></CI_Date></date></CI_Ci
tation></specification><explanation><CharacterString>The mean displacement from the
```

```
authoritative data is
0.5778691972299712</CharacterString></explanation><pass><Boolean>1</Boolean></pass></D
Q_ConformanceResult></result></DQ_AbsoluteExternalPositionalAccuracy>
2017-10-04 13:44:47,553 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: <Execute version="2.0.0" service="WPS"
response="document" mode="sync" xmlns="http://www.opengis.net/wps/2.0"
xmlns:ns="http://www.opengis.net/ows/2.0" xmlns:xlin="http://www.w3.org/1999/xlink">
  <ns:Identifier>testbed13.dsi.HootenannyConflation</ns:Identifier>
  <Input id="INPUT1">
    <Reference mimeType="application/x-zipped-shp"
xlin:href="https://tb12.dev.52north.org/security-
proxy/service/wfs?service=WFS&amp;version=1.0.0&amp;request=GetFeature&amp;typeName=tb
13:tnm-manhattan-streets-wgs84&amp;maxFeatures=50&amp;outputFormat=SHAPE-ZIP"/>
  </Input>
  <Input id="INPUT2">
    <Reference mimeType="application/x-openstreetmap+xml"
xlin:href="https://tb12.dev.52north.org/aws-
proxy/service/aws?service=aws&url=https://s3-eu-west-1.amazonaws.com/testbed13-
osm/manhattan/osm-manhattan-roads.osm"/>
  </Input>
  <Input id="INPUT1_DATA_QUALITY">
    <Data mimeType="text/xml">
      <DQ_AbsoluteExternalPositionalAccuracy xmlns="">
        <nameOfMeasure>
          <CharacterString>Test of accuracy of the target data against an
authoritative reference</CharacterString>
        </nameOfMeasure>
        <evaluationMethodType>
          <DQ_EvaluationMethodTypeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#EvaluationMe
thodTypeCode" codeListValue="directExternal">Direct
external</DQ_EvaluationMethodTypeCode>
        </evaluationMethodType>
        <dateTime>
          <DateTime>2006-11-10T00:00:00</DateTime>
        </dateTime>
        <result>
          <DQ_ConformanceResult>
            <specification>
              <CI_Citation>
                <title>
                  <CharacterString>Accuracy of position test</CharacterString>
                </title>
                <date>
                  <CI_Date>
                    <date>
                      <Date>2017-10-04</Date>
                    </date>
                    <dateType>
                      <CI_DateTypeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#CI_DateTypeC
```

```
ode" codeListValue="creation">creation</CI_DateTypeCode>
                    </dateType>
                  </CI_Date>
                </date>
              </CI_Citation>
            </specification>
            <explanation>
              <CharacterString>The mean displacement from the authoritative data is
0.5778691972299712</CharacterString>
            </explanation>
            <pass>
              <Boolean>1</Boolean>
            </pass>
          </DQ_ConformanceResult>
        </result>
      </DQ_AbsoluteExternalPositionalAccuracy>
    </Data>
  </Input>
  <Input id="INPUT1_TRANSLATION">
    <Reference mimeType="text/x-script.phyton"
xlin:href="http://geoprocessing.demo.52north.org:8080/data/TNM_Roads.py"/>
  </Input>
  <Input id="INPUT2_DATA_QUALITY">
    <Data mimeType="text/xml">
      <DQ_AbsoluteExternalPositionalAccuracy xmlns="">
        <nameOfMeasure>
          <CharacterString>Test of accuracy of the target data against an
authoritative reference</CharacterString>
        </nameOfMeasure>
        <evaluationMethodType>
          <DQ_EvaluationMethodTypeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#EvaluationMe
thodTypeCode" codeListValue="directExternal">Direct
external</DQ_EvaluationMethodTypeCode>
        </evaluationMethodType>
        <dateTime>
          <DateTime>2006-11-10T00:00:00</DateTime>
        </dateTime>
        <result>
          <DQ_ConformanceResult>
            <specification>
              <CI_Citation>
                <title>
                  <CharacterString>Accuracy of position test</CharacterString>
                </title>
                <date>
                  <CI_Date>
                    <date>
                      <Date>2017-06-14</Date>
                    </date>
                    <dateType>
```

```
                      <CI_DateTypeCode
codeList="http://www.isotc211.org/2005/resources/Codelist/gmxCodelist.xml#CI_DateTypeC
ode" codeListValue="creation">creation</CI_DateTypeCode>
                    </dateType>
                  </CI_Date>
                </date>
              </CI_Citation>
            </specification>
            <explanation>
              <CharacterString>The mean displacement from the authoritative
data is 0.5542467979794</CharacterString>
            </explanation>
            <pass>
              <Boolean>0</Boolean>
            </pass>
          </DQ_ConformanceResult>
        </result>
      </DQ_AbsoluteExternalPositionalAccuracy>
    </Data>
  </Input>
  <Output id="CONFLATION_OUTPUT" mimeType="application/x-zipped-shp"
transmission="reference"/>
  <Output id="CONFLATION_REPORT" mimeType="text/plain" transmission="reference"/>
</Execute>
2017-10-04 13:44:47,553 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Checking execute operation.
2017-10-04 13:44:47,569 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got status code 401
2017-10-04 13:44:47,569 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got HTTP error code: 401
2017-10-04 13:44:47,569 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting access token.
2017-10-04 13:44:47,569 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.OAuth2Client: Sending request for access token:
{"client_id":"AhsXezU3...Mzt39mO2ID","client_secret":"LScXw6WHdP...aUx77BvV128roo6gXLG
8gnE4","audience":"http://tb12.dev.52north.org/wps/WebProcessingService","grant_type":
"client_credentials"}
2017-10-04 13:44:47,615 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.OAuth2Client: Received:
{"access_token":"eyJ0eXAiOiJKV1Q..._rKpCyutAHKw1ATK7VEA","scope":"Execute","expires_in
":86400,"token_type":"Bearer"}
2017-10-04 13:44:47,615 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.util.JSONUtil: Trying to parse JSON String:
{"access_token":"eyJ0eXAiOiJKV1Q..._rKpCyutAHKw1ATK7VEA","scope":"Execute","expires_in
":86400,"token_type":"Bearer"}
2017-10-04 13:44:47,615 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Successfully requested access token.
2017-10-04 13:45:05,883 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Response: <wps:Result
xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd" xmlns:wps="http://www.opengis.net/wps/2.0"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlin="http://www.w3.org/1999/xlink">
  <wps:JobID>4aec68d4-c9a6-4426-bdc9-bba7ff6cdc58</wps:JobID>
  <wps:Output id="CONFLATION_OUTPUT">
    <wps:Reference mimeType="application/x-zipped-shp"
xlin:href="https://tb12.dev.52north.org/conflation-wps-
proxy/service/wps?request=GetOutput&amp;version=2.0.0&amp;service=WPS&amp;id=4aec68d4-
c9a6-4426-bdc9-bba7ff6cdc58CONFLATION_OUTPUT.c63470d1-b91e-478d-af1f-5115addba506"/>
  </wps:Output>
  <wps:Output id="CONFLATION_REPORT">
    <wps:Reference mimeType="text/plain"
xlin:href="https://tb12.dev.52north.org/conflation-wps-
proxy/service/wps?request=GetOutput&amp;version=2.0.0&amp;service=WPS&amp;id=4aec68d4-
c9a6-4426-bdc9-bba7ff6cdc58CONFLATION_REPORT.4a783603-527c-42db-be73-58098a1ab8d9"/>
  </wps:Output>
</wps:Result>
```

Appendix B: XML Schema Documents

1.1. InsertProcess operation

A new operation was specified for WPS 2.0 to upload BPMN documents and create new processes. The following two schemas were created:

InsertProcess request

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" xmlns:ows=
"http://www.opengis.net/ows/2.0"
    xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:wps=
"http://www.opengis.net/wps/2.0"
    targetNamespace="http://www.opengis.net/wps/2.0" elementFormDefault="qualified"
    xml:lang="en" version="2.0.0">

    <!-- all-components document include (OGC 06-135r11 s#14) -->
    <include schemaLocation="wps.xsd" />
    <import namespace="http://www.w3.org/1999/xlink" schemaLocation=
"http://www.w3.org/1999/xlink.xsd" />

    <annotation>
        <documentation>
            WPS is an OGC Standard.
            Copyright (c) 2015 Open
            Geospatial Consortium.
            To obtain additional rights of use, visit
            http://www.opengeospatial.org/legal/.
        </documentation>
    </annotation>

    <element name="InsertProcess" type="wps:InsertProcessRequestType" />

    <complexType name="InsertProcessRequestType">
        <complexContent>
            <extension base="wps:RequestBaseType">
                <sequence>
                    <element name="ProcessSpecification" type=
"wps:ProcessSpecificationType"
                        minOccurs="1" maxOccurs="1">
                        <annotation>
                            <documentation>
                                Referenced or inline process specification.
                            </documentation>
                        </annotation>
                    </element>
```

```
                </sequence>
            </extension>
        </complexContent>
    </complexType>

    <complexType name="ProcessSpecificationType">
        <annotation>
            <documentation>
                This structure contains information elements to supply
                a process specification for the process to be inserted.
            </documentation>
        </annotation>
        <sequence>
            <choice>
                <element ref="wps:ProcessSpecificationAsValue" />
                <element ref="wps:ProcessSpecificationAsReference" />
            </choice>
        </sequence>
        <attribute name="id" type="anyURI" use="required">
            <annotation>
                <documentation>
                    Identifier of this process.
                </documentation>
            </annotation>
        </attribute>
    </complexType>

    <element name="ProcessSpecificationAsValue">
        <complexType mixed="true">
            <annotation>
                <documentation>
                    This element is used to embed the process
                    specification in a WPS
                    request.
                    The content can be XML data, plain
                    character
                    data, or specially encoded binary data (i.e. base64).
                </documentation>
            </annotation>
            <complexContent mixed="true">
                <extension base="anyType">
                    <attributeGroup ref="wps:dataEncodingAttributes" />
                </extension>
            </complexContent>
        </complexType>
    </element>

    <element name="ProcessSpecificationAsReference" type=
"wps:ProcessSpecificationAsReferenceType">
        <annotation>
            <documentation>
```

```
                This element is used for web accessible references to
                a process specification
            </documentation>
        </annotation>
    </element>
    <!-- ========================================================== -->
    <complexType name="ProcessSpecificationAsReferenceType">
        <annotation>
            <documentation>
                Reference to a process specification that is a web
                accessible resource.
            </documentation>
        </annotation>
        <attribute ref="xlink:href" use="required">
            <annotation>
                <documentation>
                    HTTP URI that points to the remote resource where the
                    process specification may be retrieved.
                </documentation>
            </annotation>
        </attribute>
        <attributeGroup ref="wps:dataEncodingAttributes" />
    </complexType>

</schema>
```

InsertProcess response

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
    xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:ows="http://www.opengis.net/ows/2.0"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:wps="http://www.opengis.net/wps/2.0"
    targetNamespace="http://www.opengis.net/wps/2.0"
    elementFormDefault="qualified"
    xml:lang="en"
    version="2.0.0">

    <!-- all-components document include (OGC 06-135r11 s#14) -->
    <include schemaLocation="http://schemas.opengis.net/wps/2.0/wps.xsd"/>

    <annotation>
        <documentation>
            WPS is an OGC Standard.
            Copyright (c) 2015 Open Geospatial Consortium.
            To obtain additional rights of use, visit
http://www.opengeospatial.org/legal/.
        </documentation>
    </annotation>

        <element name="InsertProcessInfo">
        <annotation>
            <documentation>
                InsertProcessInfo document containing information about inserted
processes.
            </documentation>
        </annotation>
        <complexType>
            <sequence>
                <!-- reference to JobID -->
                <element ref="wps:processID"/>
            </sequence>
        </complexType>
    </element>

        <element name="processID" type="string">
        <annotation>
            <documentation>
                Identifier of the new process.
            </documentation>
        </annotation>
    </element>

</schema>
```

1.2. Extension of the WPS 2.0 StatusInfo document

The WPS 2.0 StatusInfo document was extended to be able to add more information.

Excerpt from wpsCommon, modified StatusInfo section

```
    <!-- ========================================================== -->
    <!-- StatusInfo elements and types -->
    <!-- ========================================================== -->
    <element name="StatusInfo">
        <annotation>
            <documentation>
                StatusInfo document containing information about executed processes.
            </documentation>
        </annotation>
        <complexType>
            <sequence>
                <!-- reference to JobID -->
                <element ref="wps:JobID"/>
                <!-- definition of Status element -->
                <element name="Status">
                    <annotation>
                        <documentation>
                            This element is used to communicate basic status
information about executed processes.
                        </documentation>
                    </annotation>
                    <simpleType>
                        <annotation>
                            <documentation>
                                Basic status set to communicate the status of a
server-side job to the client.
                                Extensions of this specification may introduce
additional states for fine-grained
                                monitoring or domain-specific purposes.
                            </documentation>
                        </annotation>
                        <union>
                            <simpleType>
                                <restriction base="string">
                                    <enumeration value="Succeeded">
                                        <annotation>
                                            <documentation>
                                                The job has finished with no errors.
                                            </documentation>
                                        </annotation>
                                    </enumeration>
                                    <enumeration value="Failed">
                                        <annotation>
                                            <documentation>
                                                The job has finished with errors.
```

```
                                    </documentation>
                                </annotation>
                            </enumeration>
                            <enumeration value="Accepted">
                                <annotation>
                                    <documentation>
                                        The job is queued for execution.
                                    </documentation>
                                </annotation>
                            </enumeration>
                            <enumeration value="Running">
                                <annotation>
                                    <documentation>
                                        The job is running.
                                    </documentation>
                                </annotation>
                            </enumeration>
                        </restriction>
                    </simpleType>
                    <simpleType>
                        <restriction base="string"/>
                    </simpleType>
                </union>
            </simpleType>
        </element>
        <element ref="wps:Input" minOccurs="0"/>
        <element ref="wps:Message" minOccurs="0"/>
        <!-- reference to job expiration date -->
        <element ref="wps:ExpirationDate" minOccurs="0"/>
        <element name="EstimatedCompletion" type="dateTime" minOccurs="0">
            <annotation>
                <documentation>
                    Estimated date and time by which the job will be
completed. Use if available.
                    The time of estimated completion lies significantly before
the expiration date of this job.
                </documentation>
            </annotation>
        </element>
        <element name="NextPoll" type="dateTime" minOccurs="0">
            <annotation>
                <documentation>
                    Suggested date and time for the next status poll
(GetStatus) for this job. Use if appropriate.
                    The time of the next poll shall lie significantly before
the expiration date of this job.
                    If this element is provided but an expiration date for the
job is not given, clients are expected to check
                    the job status on time to eventually receive an update on
the expiration date and avoid missing the results.
                </documentation>
```

```
                    </annotation>
                </element>
                <element name="PercentCompleted" minOccurs="0">
                    <annotation>
                        <documentation>
                            Use as a progress indicator if appropriate. Like most
progress bars the value is an estimate without accuracy guarantees.
                        </documentation>
                    </annotation>
                    <simpleType>
                        <restriction base="integer">
                            <minInclusive value="0"/>
                            <maxInclusive value="100"/>
                        </restriction>
                    </simpleType>
                </element>
            </sequence>
        </complexType>
    </element>

    <element name="Input" type="string">
        <annotation>
            <documentation>
                If the status concerns a specific input.
            </documentation>
        </annotation>
    </element>

    <element name="Message" type="string">
        <annotation>
            <documentation>
                Additional information.
            </documentation>
        </annotation>
    </element>
```

Appendix C: TIE results

Additional information about the Technology Integration Experiments. Please refer to Technology Integration Experiments for the used abbreviations and a TIE table.

1.1. TIE between WEN and CTP - Use case: Dominating Priviledges

All services require a client certificate and are protected by OAuth2. Therefore, this use case is covered by the general communication in the workflow.

The following excerpt from the workflow engine log shows the communication between the workflow engine and a WPS (see Workflow engine log for the full log):

```
2017-10-04 13:44:45,229 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting WPS: https://tb12.dev.52north.org/data-
quality-wps-proxy/service/wps with process:
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy
2017-10-04 13:44:45,229 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Checking access to wps capabilities:
https://tb12.dev.52north.org/data-quality-wps-
proxy/service/wps?request=GetCapabilities&service=WPS&acceptVersions=2.0.0
2017-10-04 13:44:45,244 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got status code 400
2017-10-04 13:44:45,244 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Cannot access wps, trying to set client
certificate: ogc-tb-13-x509-test-client.p12
2017-10-04 13:44:45,244 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting capabilities from:
https://tb12.dev.52north.org/data-quality-wps-proxy/service/wps
2017-10-04 13:44:45,291 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting process description for process:
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy from:
https://tb12.dev.52north.org/data-quality-wps-proxy/service/wps
2017-10-04 13:44:45,307 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got process: Id:
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy
Title: iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy
            Abstract:

2017-10-04 13:44:45,307 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Creating execute request for process
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy
2017-10-04 13:44:45,307 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got result from previous process:
http://ows.dev.52north.org:8080/SecurityProxy/service/wps?request=GetOutput&version=2.
0.0&service=WPS&id=998b0f75-07ec-4a85-93cd-d480ff97da0a
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
```

```
org.n52.testbed13.workflow.LogUtil: <Execute version="2.0.0" service="WPS"
response="document" mode="sync" xmlns="http://www.opengis.net/wps/2.0"
xmlns:ns="http://www.opengis.net/ows/2.0" xmlns:xlin="http://www.w3.org/1999/xlink">
....
</Execute>
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Checking execute operation.
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got status code 401
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Got HTTP error code: 401
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Requesting access token.
2017-10-04 13:44:45,322 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.OAuth2Client: Sending request for access token:
{"client_id":"AhsXe....YquMzt39mO2ID","client_secret":"LScXw6WHdP....8roo6gXLG8gnE4","
audience":"http://tb12.dev.52north.org/data-quality-
wps/WebProcessingService","grant_type":"client_credentials"}
2017-10-04 13:44:45,353 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.OAuth2Client: Received:
{"access_token":"eyJ0eXAiOiJKV....3o1DF-
tI7syTmw","scope":"Execute","expires_in":86400,"token_type":"Bearer"}
2017-10-04 13:44:45,353 [http-nio-8081-exec-4] INFO
org.n52.geoprocessing.oauth2.util.JSONUtil: Trying to parse JSON String:
{"access_token":"eyJ0eXAiOiJKV....3o1DF-
tI7syTmw","scope":"Execute","expires_in":86400,"token_type":"Bearer"}
2017-10-04 13:44:45,353 [http-nio-8081-exec-4] INFO
org.n52.testbed13.workflow.LogUtil: Successfully requested access token.
```

1.2. TIE between WEN and DQP - Use case: Tunneling Proxies

As mentioned before, all services require a client certificate. However, as the services are running behind a proxy, the certificate will be digested before it reaches the downstream service. Therefore, a SOAP request was created and an additional certificate was added to the SOAP header. The original request is stored in the SOAP body. The proxy extracts the certificate from the SOAP header and can treat it accordingly, e.g. forward it to the downstream service or decode it and check an access control list.

The following listing shows a request to the data quality WPS process wrapped in a SOAP request (the certificate was shortened):

```
<soapenv:Envelope xmlns:soapenv="http://www.w3.org/2003/05/soap-envelope">
    <soapenv:Header xmlns:wsa="http://www.w3.org/2005/08/addressing">
        <wsa:To>https://tb12.dev.52north.org/SecurityProxy/services/soap
        </wsa:To>
        <wsa:Action>http://www.opengis.net/wps/requests#Execute
        </wsa:Action>
```

```
        <wsa:MessageID>urn:uuid:3aa0ed95-d6d7-40c8-8c71-4ab8b380d789
        </wsa:MessageID>
        <wsse:Security
            xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-secext-1.0.xsd"
            soapenv:mustUnderstand="1">
            <wsse:BinarySecurityToken
                EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
soap-message-security-1.0#Base64Binary"
                xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
wssecurity-utility-1.0.xsd"
                wsu:Id="CertID-1e8c009b-a7aa-4f65-8cac-a3394cb278c7"
                ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
x509-token-profile-1.0#X509v3">

MIIQGQIBAzCCD98GCSqGSIb3DQEHAaCCD9AEgg/MMIIPyDCCCn8GCSqGSIb3DQEHBqCCCnAwggpsAgEAMIIKZQ
YJKoZIhvcNAQcBMBwGCiqGSIb3DQEMAQYwDgQI2aSdUV6iC5gCAggAgIIKOMCvjcCPhuOr3AFEMp2cSsolGvyY
fwSG7Q/QMILgzwA/WdcbtsWm7FxTFWN1tZIMo+h4QQDHN8OM/yhMUB6WAU/7HCesRwP+UCHu3jwfS/I6UmKugW
3cjeYk43oliho24vt5oHWFHE9tRW9Djs0ueYCYKG16aydKbXsNKh1wc8bYWLN/VOitYMIUdjKJcyFqvZPXJoG5
wPou+X7vgbgz3jlH7568AzY/uHCoMq0TAOZsDWKi+PAMa3k1Q7Sr4YHtLrfSgjp9iXL6v5d/JtL01YOAqAYn2R
0dC/48EK+CJS7Ia8mwzgbmZQXenbi2qQkry6B4QKUrypdR08yCv56Kr5oxEsE8WtB9t0qCZI/nryCk9nbNVKmJ
AlBqluxH5+QvybQT5DGMiF+6lkws/1nDewq87J10q5bqKDWhnZthpxKzlbOi+4ghe/
                ....
                dmhbkAQIXMCcNzbZdpwCAggA
            </wsse:BinarySecurityToken>
        </wsse:Security>
    </soapenv:Header>
    <soapenv:Body>
<wps:Execute
  xmlns:wps="http://www.opengis.net/wps/2.0"
  xmlns:ows="http://www.opengis.net/ows/2.0"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.opengis.net/wps/2.0
http://schemas.opengis.net/wps/2.0/wps.xsd"
  service="WPS" version="2.0.0" response="document" mode="sync">
  <ows:Identifier>
iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy</ows:Identifier>
  <wps:Input id="inputTargetDataset">
  <wps:Reference schema="http://schemas.opengis.net/gml/3.1.0/base/feature.xsd"
mimeType="text/xml; subtype=gml/3.1.0" xlink:href=
"https://tb12.dev.52north.org/security-proxy/service/wfs?service=WFS&amp;
version=1.0.0&amp;request=GetFeature&amp;typeName=tb13:tnm-manhattan-streets
&amp;outputFormat=gml3&amp;maxFeatures=50"/>
</wps:Input>
   <wps:Input id="inputReferenceDataset">
  <wps:Reference schema="http://schemas.opengis.net/gml/3.1.0/base/feature.xsd"
mimeType="text/xml; subtype=gml/3.1.0" xlink:href=
"https://tb12.dev.52north.org/security-proxy/service/wfs?service=WFS&amp;
version=1.0.0&amp;request=GetFeature&amp;typeName=tb13:manhattan-streets-
reference&amp;outputFormat=gml3&amp;maxFeatures=50"/>
```

```
</wps:Input>
  <wps:Input id="threshold">
    <wps:Data>
    <wps:LiteralValue>10</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Input id="inputTargetField">
    <wps:Data>
    <wps:LiteralValue>OBJECTID_1</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Input id="inputReferenceField">
    <wps:Data>
    <wps:LiteralValue>OBJECTID_1</wps:LiteralValue>
      </wps:Data>
  </wps:Input>
  <wps:Output id="outputMetadataChunk" transmission="value"/>
</wps:Execute>
    </soapenv:Body>
</soapenv:Envelope>
```

The common name is extracted and checked against an access control list.

SimplePermission policy allowing a user to execute the data quality WPS process:

```
<SimplePermissions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://www.52north.org/security/simple-permission/1.0">
    <PermissionSet name="Data Quality WPS Permission">
        <ResourceDomain value="http://127.0.0.1:8080/data-quality-
wps/WebProcessingService" />
        <ActionDomain value="http://127.0.0.1:8080/data-quality-
wps/WebProcessingService" />
        <SubjectDomain value="urn:n52:security:subject:role" />
        <Permission name="e.h.juerrens_Execute">
            <Resource value=
"iso19157.DQ_PositionalAccuracy.DQ_AbsoluteExternalPositionalAccuracy" />
            <Action value="Execute" />
            <Subject value="e.h.juerrens" />
        </Permission>
    </PermissionSet>
</SimplePermissions>
```

The service is running on localhost and only accessible via the proxy.

1.3. TIE between COP and ABU - Use case: Identity Mediation

An OpenStreetMap (OSM) dataset used as input for the conflation WPS process was stored on a

Amazon Web Services (AWS) S3 bucket. Access was set to private and a policy was created granting access only to a certain user registered in the Amazon AWS Identity and Access Management (IAM). A proxy was set up requiring a client certificate to be sent along with each request.

Proxy URL for accessing the OSM dataset:

```
https://tb12.dev.52north.org/aws-proxy/service/aws?service=aws&url=https://s3-eu-west-
1.amazonaws.com/testbed13-osm/manhattan/osm-manhattan-roads.osm
```

The proxy extracts the common name out of the certificate and checks it against a access control list.

1.4. TIE between IBR CTP - Oauth Authorization Code Grant Flow

The OAuth Client Credentials Grant Flow was implemented for most of the involved components. This works if the clients and resource servers are known to the workflow composer and respective authorizations can be granted before executing the workflow. If a client has not been granted authorization to a resource, the Authorization Code Grant Flow can be used to grant access at the time of execution. Please refer to section 9 in the OGC Testbed-13: Security ER (OGC 17-021) for detailed information about the implementation. A proxy was used in front of the WPS to handle the authorization of inputs dynamically. After sending an execute request to the proxy, it checks, whether the inputs is protected. The GetStatus document of the WPS 2.0 standard was extended to add more information.

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:StatusInfo xmlns:wps="http://www.opengis.net/wps/2.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
"http://www.opengis.net/wps/2.0 http://schemas.opengis.net/wps/2.0/wps.xsd">
  <wps:JobID>ca3c699c-e32f-4655-800d-7ba277732a9d</wps:JobID>
  <wps:Status>Running</wps:Status>
  <wps:Input>data</wps:Input>
  <wps:Message>Checking input.</wps:Message>
</wps:StatusInfo>
```

If the input is protected by OAuth, first a request for an access code using the Client Credentials is sent to the authorization server. If the access is denied, the proxy creates an authorization-URL and updates the GetStatus document accordingly.

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:StatusInfo xmlns:wps="http://www.opengis.net/wps/2.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
"http://www.opengis.net/wps/2.0 http://schemas.opengis.net/wps/2.0/wps.xsd">
  <wps:JobID>f3ef65da-1dce-491f-ae4c-10ae899becb3</wps:JobID>
  <wps:Status>Authorization needed.</wps:Status>
  <wps:Input>data</wps:Input>
  <wps:Message>https://bpross-52n.eu.auth0.com/authorize?scope=GetFeature
GetFeature/TypeName=tb13:tnm-manhattan-streets GetFeature/TypeName=tb13:tnm-manhattan-
streets-wgs84&amp;audience=http://tb12.dev.52north.org/security-proxy/service/wfs
&amp;response_type=code&amp;client_id=DVS5...pBG4H&amp;redirect_uri=http://127.0.0.1:8
080/secured-input-proxy/oauth2callback&amp;state=f3ef65da-1dce-491f-ae4c-
10ae899becb3:data</wps:Message>
</wps:StatusInfo>
```

A request to this URL will bring up the following Web page:

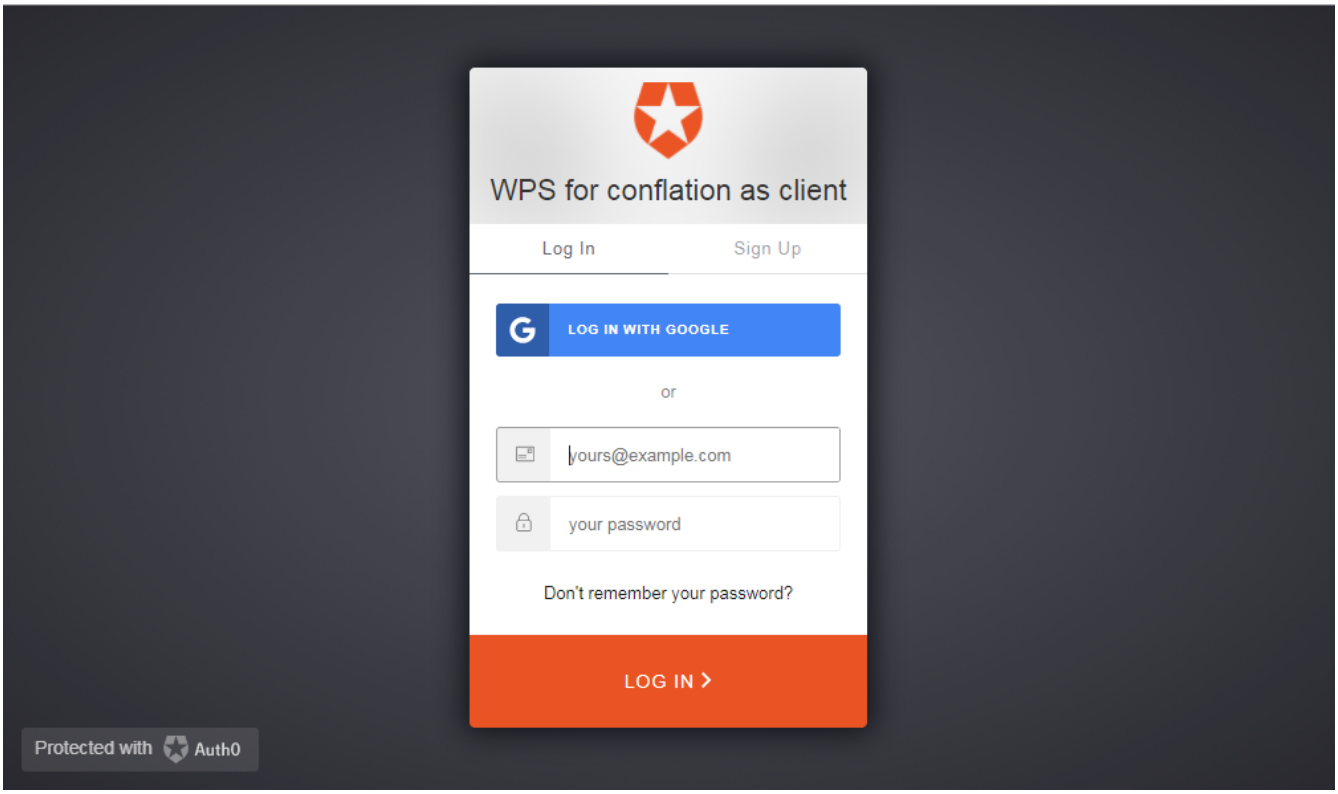

Figure 39. Auth0 login screen

After login, the client application can be authorized to access the resources/requested scopes:

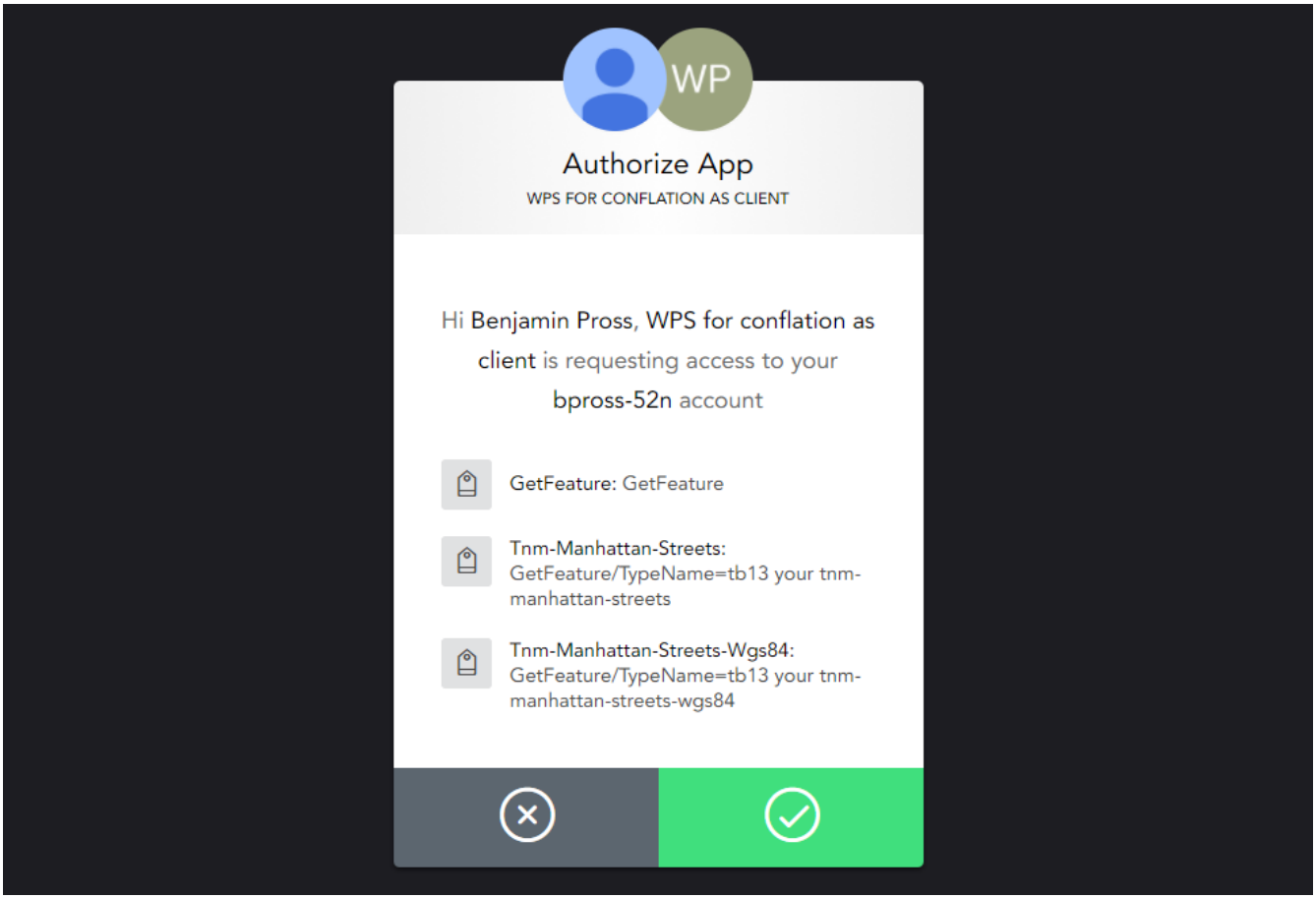

Figure 40. Auth0 authorize screen

If the authorization was successful, the redirect-URL specified in the authorization-URL is called with the authorization code. The GetStatus document is also updated.

```
<?xml version="1.0" encoding="UTF-8"?>
<wps:StatusInfo xmlns:wps="http://www.opengis.net/wps/2.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation=
"http://www.opengis.net/wps/2.0 http://schemas.opengis.net/wps/2.0/wps.xsd">
  <wps:JobID>f3ef65da-1dce-491f-ae4c-10ae899becb3</wps:JobID>
  <wps:Status>Running</wps:Status>
  <wps:Input>data</wps:Input>
  <wps:Message>Input authorized.</wps:Message>
</wps:StatusInfo>
```

The input is then forwarded to the WPS, where it can be processed.

Appendix D: Revision History

Table 3. Revision History

Date	Release	Editor	Primary clauses modified	Descriptions
May 2, 2017	.1	C. Stasch	all	initial version
May 3, 2017	.1	C. Stasch	9	new chapter for client and catalog added
May 30, 2017	.1	B. Pross	1	added summary
May 31, 2017	.1	B. Pross	1	Initial ER
Jun 22, 2017	.1	B. Pross	1	Addressed comments from review
Jul 14, 2017	.1	D. Misev	8	Added rasdaman WCS component description
Jul 18, 2017	.1	C.Stasch, B. Pross	all	Major update
Jul 24, 2017	.1	M. Lawrence	9	Update on chapter for client and catalog
Jul 25, 2017	.1	M. Lawrence	9	Update on chapter for client and catalog
Jul 31, 2017	.1	B. Pross	8	Update on data services
Aug 9, 2017	.1	B. Pross	6	Added InsertProcess request/response
Sep 22, 2017	.1	B. Pross	5	Added TIE table
Oct 6, 2017	.1	B. Pross	all	Draft ER
Oct 9, 2017	.1	B. Pross	6	Minor fixes
Oct 11, 2017	.1	B. Pross	6,10	Minor fixes
Oct 25, 2017	.1	M. Lawrence	1,6,11	Review
Nov 10, 2017	.1	B. Pross	all	Review from T. Idol
Nov 14, 2017	.1	B. Pross	all	Revision 1

Date	Release	Editor	Primary clauses modified	Descriptions
Nov 24, 2017	.1	B. Pross	all	NGA Review
Nov 27, 2017	.1	B. Pross	all	Review from G. Hobona

Appendix E: Bibliography

[1] Baranski, B., Schaeffer, B., Redweik, R.: Geoprocessing in the Clouds. OSGeo Journal. 8, 1, 5 (2010).

[2] Chen, N., Di, L., Yu, G., Gong, J.: Geo-processing workflow driven wildfire hot pixel detection under sensor web environment. Computers & Geosciences, 36, 3, 362-372 (2010).

[3] de Jesus, J., Walker, P., Grant, M., & Groom, S.: WPS orchestration using the Taverna workbench: The eScience approach. Computers & Geosciences. 47, 75-86 (2012).

[4] Di, L.: GeoBrain-a web services based geospatial knowledge building system. Proceedings of NASA earth science technology conference. pp. 22-24. [Palo Alto] (2004).

[5] Hobona, Gobe, Fairbairn, D., Hiden, H., James, P.: Orchestration of grid-enabled geospatial web services in geoscientific workflows. IEEE Transactions on Automation Science and Engineering. 7, 2, 407-411 (2010).

[6] Meek, S.: OGC 16-091 BPMN 2.0 for Orchestrating OGC Services, https://portal.opengeospatial.org/files/?artifact_id=68879&version=1.

[7] Rosser, J., Pourabdollah, A., Brackin, R., Jackson, M., Leibovici, D. G: Full Meta Objects for flexible geoprocessing workflows: profiling WPS or BPMN?, https://www.researchgate.net/profile/Julian_Rosser/publication/303881781_Full_Meta_Objects_for_flexible_geoprocessing_workflows_profiling_WPS_or_BPMN/links/5767941808ae421c448c52b2.pdf.

[8] Schaeffer, B.. Towards a transactional web processing service. Proceedings of the GI-Days. [Münster] (2008).

[9] Shao, Y., Di, L., Bai, Y., Guo, B., & Gong, J.: Geoprocessing on the Amazon cloud computing platform—AWS. 2012 First International Conference on Agro-Geoinformatics. pp. 1-6, [Shanghai] (2012).

[10] Stollberg, B., & Zipf, A.: OGC web processing service interface for web service orchestration aggregating geo-processing services in a bomb threat scenario. Web and wireless geographical information systems. W2GIS 2007: Web&Wireless GIS Conference 2007. [Cardiff] (2007).