

Title:

<b>Title:</b>
Title: Processing Components General Model
<b>Author(s)/Organisation(s):</b>
Jan Jezek, Help Service – Remote Sensing
<b>Working Group:</b>
WP5
<b>References:</b>
A5.2-D1 1.1 Mediator Service Component Specification

<b>Short Description:</b>
<p>This paper is focused on description on possibilities how to implement particular processing component. By processing component we mean the software that is capable to perform some kind of data precessing during harmonization.</p>
<b>Keywords:</b>
<p>WPS, SOAP, Transformer, Harmonization</p>

<b>History:</b>			
Version	Author(s)	Status	Comment
001	Jan Jezek	new	

Title:

## Table of contents

Title:

# 1 Introduction

## 1.1 Purpose of this document

This document is a specification of the *General Model of Processing Components* of the HUMBOLDT Framework (part of Deliverable WP05 V2.0). It's main aim is to establish common architectural elements for processing services to be used in the HUMBOLDT Framework, especially for harmonisation processing services. This architecture joins existing and well known approaches that are often very general with specific needs of the HUMBOLDT framework.

Title:

## 2 Processing component specification

As the harmonization process is very complex task, several common rules need to be defined for processing services which are part of it. These common rules include the requirements that need to be fulfilled by a certain processing component to be used by the HUMBOLDT framework in a harmonization processing workflow and are described in a later section.

Furthermore, this document specifies three main approaches to provide a particular processing functionality for the HUMBOLDT framework. These three possibilities represent different levels of generality. These three possibilities are:

- Direct implementation of the **Transformer** interface on the platform for which the [Mediator Service](#) to be used has been implemented. In the case of the reference implementation, this is the Java platform. This approach is especially suitable for specific tasks where it is for some reasons not convenient to transfer data through web service.
- WPS - OGC Web Processing Service is gaining traction as an interface for the description and execution of transformation capabilities. It provides Web Service advantages like low coupling and is considered the preferred way to implement particular processes in HUMBOLDT.
- WS-\*/SOAP - the most complex approach that should especially be taken into account when working with non-geo-specific services. However, it also becomes clear that OGC services will more commonly offer such interfaces in the future. Using this approach fits the needs of most general processing tasks for which the WPS interface is not suitable.

One of the aims of this document is to provide examples on how to easily implement all these possibilities by reusing existing open source implementations.

### 2.1 Common Requirements towards HUMBOLDT Transformation Services

As far as we are thinking about complex framework that will includes not just particular WPS services but will be able also to chain them and catalogue them we have to formalized the informations that are provided about such services.

Humboldt transformation services have to provide enough metadata that describes particular WPS. Also particular metadata that will be added to resulting transformed data are necessary.

The WPS as such has to be described using ProcessDescription. Except mandatory elements of DeccribedProcess we have to use also the Metadata optional element that contains OWS:Metadata.

**TODO:** specify metadata profile for ProcessDescription Metadata element.

Humboldt WPS also has to add proper metadata to process results. According to Mediator service specifications the Lineage elements of ISO 19115 should be used for this propose. In this way the data generated by Transformer should have formalized metadata in same way as Humboldt WPS. For more see Mediator service specification.

Title:

## 2.2 HUMBOLDT Transformer

The HUMBOLDT *Transformer* is the core interface used by the [Mediator Service](#) in the orchestration of data provision services and transformation services. It's specification is part of the main HUMBOLDT specification. Furthermore, a subtype variant of this interface with additional capabilities is used by the [Workflow Service](#) during workflow design and construction. The description of this specific interface can be found in the Information Viewpoint of the [Workflow Service](#).

If you plan to implement the Transformer interface directly, the processing capability can be used by all HUMBOLDT components that make use of the Transformers. This includes the [Mediator Service](#) and the [HUMBOLDT Alignment Editor](#). However, you will not be able to use the transformation capability on it's own, as would be with a web service.

In any way, the general Transformer interface needs to be implemented:

```
import eu.esdihumboldt.workflow.repository.Transformer;  
public class MyTransformer implements Transformer {  
    ...  
}
```

Title:

## 3 WPS

(Short description of each implementation with example of extending it by own processes).

- 52North WPS

52North is providing Java implementation of WPS 1.0.0 that is easily extensible by additional functionality. This tool is distributed under GNU General Public License (GPL).

The possibilities to extend this tool is possible through extending provided super class `org.n52.wps.server.AbstractAlgorithm`. The only method that has to be implemented is:  
`public java.util.Map run(java.util.Map layers, java.util.Map parameters);`

The `java.util.Map` of layers is containing in general the GeoTools data model (GeoTools FeatureCollection or Coverage).

Detail developer guide for implementing you own processes is available here:

[https://52north.org/twiki/bin/viewfile/Processing/52nWebProcessingService?rev=2;filename=WPS\\_Tutorial.pdf](https://52north.org/twiki/bin/viewfile/Processing/52nWebProcessingService?rev=2;filename=WPS_Tutorial.pdf)

- pyWPS

pyWPS is python based implementation of WPS. Current version 3.0.0 supports OGC WPS 1.0.0. Detailed how to for implementing new service is available here

<http://pywps.wald.intevation.org/doc/3.0.0/pywps-howto/pywps-howto.html#SECTION00600000000000000000>

It is worth to mention that pyWPS is cooperating well with GRASS functionality so in the case that GRASS contains the capability of required process it should be easy to configure such process as WPS.

- Deegree WPS

Deegree provide Java based implementation of OGC WPS 0.4.0.

To add your own functionality it is necessary to extend abstract class `org.deegree.ogcwebservices.wps.execute.Process`.

Process offers one public method with the signature  
`public abstract ProcessOutputs execute( Map<String, IOValue> inputs, OutputDefinitions outputDefinitions )` throws `OGCWebServiceException`;

More information is available here:

[http://download.deegree.org/deegree2.2/docs/html/docu\\_wps/deegree\\_wps\\_documentation\\_en.html](http://download.deegree.org/deegree2.2/docs/html/docu_wps/deegree_wps_documentation_en.html)

- GeoServer WPS (??)

GeoServer development team is working on WPS extensions. This module is actually work in progress. It build on top of the GeoTools library and the plan is to use Sextante library for processing algorithms.

Geoserver and Sextante are both based on Java.

## 4 SOAP

Processing component can be also implemented as a SOAP service by binding SOAP wrapper for Transformer interface. SOAP can be used in the case of problems, that WPS interface doesn't fit the requirements of particular process.

Title:

Advantage of SOAP is that available software tools makes it easy to build the Server side as well as client side. Also chaining and configuring it using BPEL is very well implemented in various kinds of software tools.

Examples and use cases:

- SOAP can be used as particular process step in SDI - Spatial Data Integrator. In this way we can use GUI to chain all available functionality with newly developed SOAP service
- SDI - Spatial Data Integrator let users to export the prepared process chain as a SOAP service. Such service is having just on method 'runJob' that simply executes predefined Job on the server.

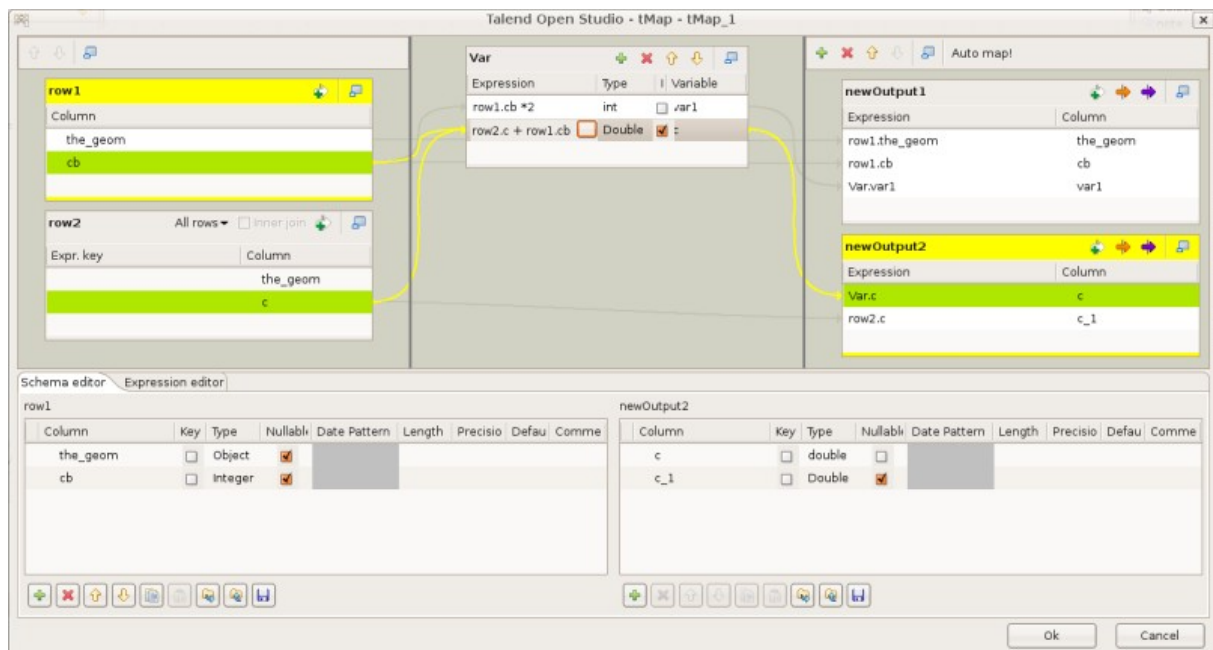


Illustration 1: Spatial Data Integrator Tmap component

## 5 Enterprise Viewpoint

Processing component is one of the core part of Humboldt framework. On one side it must be accessible and understandable by advanced GIS users that should be able to configure, set up, customize and publish their component. It must be also open enough to be possible to integrate it with other software libraries.

On other side the interfaces of such component must be specific enough so that the other parts of Humboldt framework (Mediator, Work flow service) can handle it in general way.

## 6 Actors in this component

Actors in this component are:

- Advance GIS specialist (not programmer) who understands expected results and particular harmonization process. This actor is able to follow the guide to set up his own processes and publish them into the Humboldt Framework.

Title:

- Programmer who is able to extend functionality of the off-the-self software by special processes that might be required. This programmer should be able to implement Java interface or make web service following WPS specification.
- Mediator and Work flow service – components that are able to manage particular processes.