International Journal of Spatial Data Infrastructures Research, 2014, Vol.9, 107-130

Load testing of HELIDEM geo-portal: an OGC open standards interoperability example integrating WMS, WFS, WCS and WPS*

Massimiliano Cannata [1], Milan Antonovic [1], Monia Elisa Molinari [2]

[1] University of Applied Sciences and Arts of Southern Switzerland, Department for Environment Constructions and Design, Institute of Earth sciences, massimiliano.cannata@supsi.ch , milan.antonovic@supsi.ch

[2] Politecnico di Milano, DICA, Laboratorio di Geomatica, Como Campus, monia.molinari@polimi.it

Abstract

This paper presents a load testing of the HELIDEM geo-portal, which is an example of interoperability between a number of standard geospatial services as defined by the Open Geospatial Consortium. The portal was developed within the European project HELIDEM (www.helidem.eu) with the aim of valorizing the main project output which is a cross-border digital terrain model. The portal aims at fostering its diffusion and usage trough easily accessible tools. The DTM covers the alpine area located between Southern Switzerland (Canton Ticino) and Northern Italy (Lombardy and Piedmont Regions). From a technological point of view, the server-side component of the portal is based on a Service Oriented Architecture implemented using the open source software ZOO-Project, GRASS GIS and Geoserver; the client-side component is a Web interface based on CSS3 and HTML5 trough the usage of the ExtJS framework and the OpenLayers software. The presented solution is a mix of technologies and software, some of which are considered, within the open source for geospatial community, mature and robust while others are considered promising but not sufficiently tested yet. For this reason this research conducted a load test over concurrent users in order to verify

DOI: 10.2902/1725-0463.2014.09.art5

the robustness, quality and performance of the system and to identify eventual bottlenecks. Test results didn't register any runtime exception confirming the good quality of the implemented system and underlying software. Nevertheless, performance and response time exponentially degrades with increasing number of concurrent users, area of analysis and process complexity. Finally, the test confirms that the implemented solution is robust, in fact no system failure was recorded during the analysis.

Keywords: OGC, WMS, WCS, WPS, load testing

1. INTRODUCTION

Digital Terrain Models (DTMs) are a representation of the natural terrain surface by means of terrain altitude expressed in orthometric heights. In practice, these DTM are widely used to extrapolate derived information which is useful in supporting a large number of activities, including land use planning and engineering design (Vieux, 2001; Wilson, 2000). The derivation of contour lines or of profiles are certainly some of the most basic and common operations, but also the evaluation of terrain surface derivatives like slopes, aspects, curvature are very common. Watershed analysis from DTM is a more specific task which is generally performed by hydrologist, but also used for land planning and infrastructure design. These kinds of processing are common features of modern Geographical Information System software (GIS) which, although nowadays available to a larger public if compared with the 90s, still remain accessible to specialized personnel only.

The main scope of the HELIDEM project (Helvetia-Italy Digital Elevation Model) was the creation of a unified digital terrain model and geoid, for the Alpine and Sub-Alpine area on the border between Italy and Switzerland, correctly geo-referenced in the three dimensions. The project was funded in the frame of the European Regional Development Fund within the Italy-Switzerland cooperation program and run from September 2010 to September 2013. More details in Antonovic et al. (2014).

The project leads to two new datasets: a DTM and a geoid of the cross-border region which includes northern Lombardy (Italy), northern Piedmont (Italy), Canton Ticino (Switzerland) and southern parts of the Canton of Graubünden (Switzerland) and the Canton of Valais (Switzerland). The DTM was calculated in the ETRF2000 reference frame in geographic coordinate system with a resolution of $4*10^{-4}$ degrees (about 22 m in latitude and 15 m in longitude) and maintaining the general accuracies of the original data (3m in Switzerland, 10m in Lombardy and 5m in Piedmont) (Biagi et al, 2014). The geoid was derived on the same region combining the Italian geoid ITALGEO2005 and the Swiss geoid CHGeo2004 with the inclusion of GOCE (Barzaghi et al, 2006) data for solving the height datum problem. Both of them are distributed for free under the IODL v2.0 license. Nevertheless, it has to be noted that they are not official datasets: neither for Italy nor for Switzerland.

A secondary, but not less important, goal of the project is the experimentation of the DTM, its diffusion and valorisation trough easily accessible tools. In view of this least aim, the authors conducted research and development to the creation of a geo-spatial portal which could make the DTM and some of the commonly used derived data easily accessible. The requirement analysis, conducted internally within the project team, leads to the identification of the following requisites:

- Accessing data and derived data without the need of any software or components (plug-in) except of a Web Browser;
- Flexibility in the selection of the area of interest to be elaborated;
- Capability of download or visualize the results of an elaboration;
- Capability to derive data in the desired coordinate reference system;
- Accessing at contour, profiles, watershed, derivatives analysis and data extraction tools.

2. OWS: OPEN WEB SERVICES

In the last years, thanks to the large diffusion of Internet and the growing development of open standards related to geospatial sector, today we can access to a number of technologies and standard services which are well established and tested in productive environments. Yet, some of the most recent standards and related software are less verified and applied.

Among the available options, those defined by the Open Geospatial Consortium (OGC, www.opengeospatial.org) are certainly among some of the most used solutions for Web mapping services. Those services use the HTTP protocol, and in particular the GET and POST methods, to communicate with servers and specific XML schemas to encode exchanged information. In few cases, OGC standard services contemplate the usage of the SOAP (Box et al, 2000) protocol and of binary data (also in streaming) as a response.

In the next sections a brief description of a series of standards which have been selected to fulfil the HELIDEM geo-portal requirements are reported.

Web Mapping Service (WMS) (De la Beaujardiere, 2006) enables the access to geospatial data in the form of images throughout Internet. Received images represent cartographic elements rendered according to specific requested setting parameters. WMS is widely used and supported by almost all of the available GIS software and is a mature technology.

Web Feature Service (WFS) (Vretanos, 2010) defines methods and formats to request, also using spatial and semantic filters, and transmit geospatial vector data through the Internet. With respect to the WMS, this standard permits to receive the actual data, and not only its graphical representation on an image. Moreover, this standard enables the capability of the user to remotely modify the data by means of transactional requests. Responses are always in geospatial vector formats like, for example, GML, KML or SHP.

Web Coverage Service (WCS) (Baumann, 2010a) can be considered similar to WFS with the difference that it handles distributed spatial data only. The standard defines rules to distribute, according to user’s requests, raster data. Data could eventually be cropped, re-projected, resampled or converted in one of the supported raster formats like, for example, GeoTiff or ESRI ASCII grid.

Web Processing Service (WPS) (Schut et al, 2008) permits to provide geospatial processing capabilities over the Web. Thanks to this standard predefined processes or algorithms are exposed to Web users similarly to typical desktop GIS modules. The *GetCapabilities* and *DescribeProcess* requests allow to identify offered processes and their inputs and outputs names and types; the *Execute* requests allows to run a selected process.

3. THE HELIDEM SYSTEM

The HELIDEM system implements a Service Oriented Architecture (SOA) based on the OWSs presented in the previous chapter 2. The schematization of the implemented architecture, which is represented in Figure 1, is based on four OGC services that are deployed in the Cloud. Among them, the WPS only interacts directly with the others three by using them as input data provider (WCS) or as output data dispatching services (WMS, WFS). As well as the HELIDEM portal all the services are freely accessible on the Web, and can be consumed as stand-alone service to be integrated in other contexts: for example desktop GIS or specific apps for different devices (tablets, smartphones, etc.).

The implemented OWSs have been deployed using different servers which are physically located at the Institute of Earth Science at SUPSI in Lugano and at the

Geomatics Laboratory at the Polytechnics of Milan in the Como Campus. Figure 2 represents the technological stacks that have been used respectively for the implementation of the OWSs component (server side applications) and the HELIDEM geo-portal (client side application).

The server side component of the system is based on Linux OS (Ubuntu server) and Apache 2 Web server (The Apache Software Foundation, 2013).

Figure 1: Schematization of the HELIDEM System Architecture

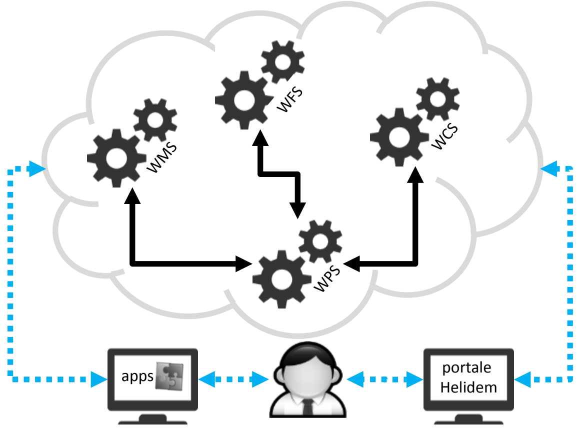
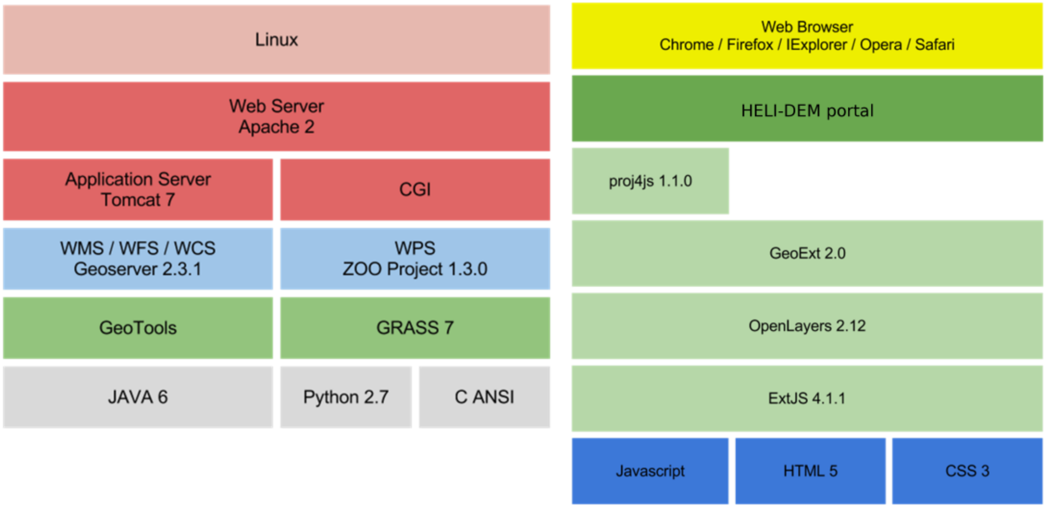


Figure 2: Software and Languages Stack Used to Develop the Server Side (on the left, Figure A) and The Client Side (on the Right, Figure B) of the HELIDEM System.



The data services (WMS, WFS and WCS) rely on Geoserver (Youngblood, 2013) deployed in the Tomcat 7 application server (The Apache Software Foundation, 2013) and served by Apache HTTP server. Geoserver is a mature solution word widely used in large scale projects and in production systems, for example by NOAA or Ordinance Survey. It mostly relies on GeoTools libraries (Turton, 2008) and is developed in JAVA language (Gosling et al, 2013).

The reliability of Geoserver is confirmed by the fact that it has the status of official Open Source Geospatial Foundation (OSGeo) project. One of OSGeo objectives is to group under its umbrella a number of Free and Open Source Software for Geospatial (FOSS4G) projects which have shown a collaborative development community and a high quality code: such a kind of software is promoted as OSGeo project. This status can only be achieved passing the incubation process which carefully reviews code and community (Brovelli et al, 2012)

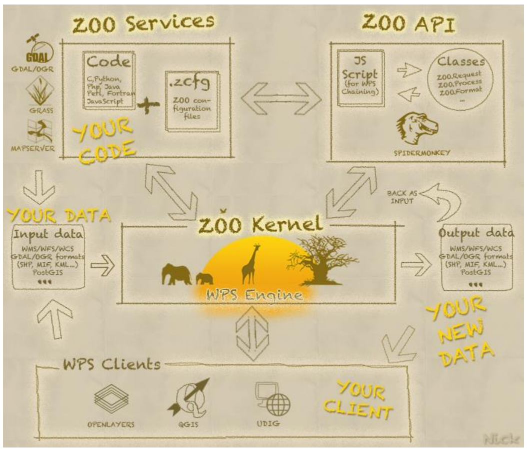
The processing servers (WPS) take advantage of the Zoo-project software (Fenoy et al, 2013) which relies on CGI. The selection of this software to implement the WPS component of the system is because it has already successfully been used in conjunction with GRASS GIS version 7 and because it is with respect to other WPS solution less investigated. In fact, Zoo-project is quite recent and, at the time of writing, is under the OSGeo incubation process.

ZOO-project is a framework to create WPS compliant services; it is composed by three components (see Figure 3): (*i*) the ZOO kernel that is the engine, written in C (International Organization for Standardization and International Electrotechnical Commission, 1999) that enable the managing and chaining of different processes written in different programming languages; (*ii*) the ZOO services that implements the processes by means of a configuration file and function implementing the process algorithms in one of the supported languages (C, Python, Perl, PhP, JavaScript, Java, Fortran); and (*iii*) the ZOO API that is a JavaScript library to easy service chaining and interface development.

In HELIDEM, the WPS processes have been coded in Python (Van Rossum, 2007) taking advantages of the python-requests library and of the modules of the GIS GRASS 7 (Neteler et al, 2012) which are generally coded in C. The processes results are always returned as *simple output* type (link to a resource), so that they are of simple integration in third parties.

The HELIDEM geo-portal, as overviewed in Figure 2 is based on the most modern technologies such as *CSS 3 and HTML 5* (Hogan, 2011) and JavaScript (Flanagan, 2006) through a number of libraries. ExtJS (Zhang et al, 2010) was used to graphically design the portal since this library to easy access advanced graphical elements guaranteeing cross browser compatibility; OpenLayers (Hazzard, 2011) is used to create the map viewer which allows for dynamically navigation of geospatial data; GeoExt (GeoExt Community, 2010) which combining OpenLayers and ExtJS provides additional features and controls; proj4js (Adair et al, 2012) used to provide on-the-fly re-projection of vectors coordinates.

Figure 3: ZOO-Project Components and Interactions. (Source: www.zoo-Project.Com, © Nicolas Bozon)



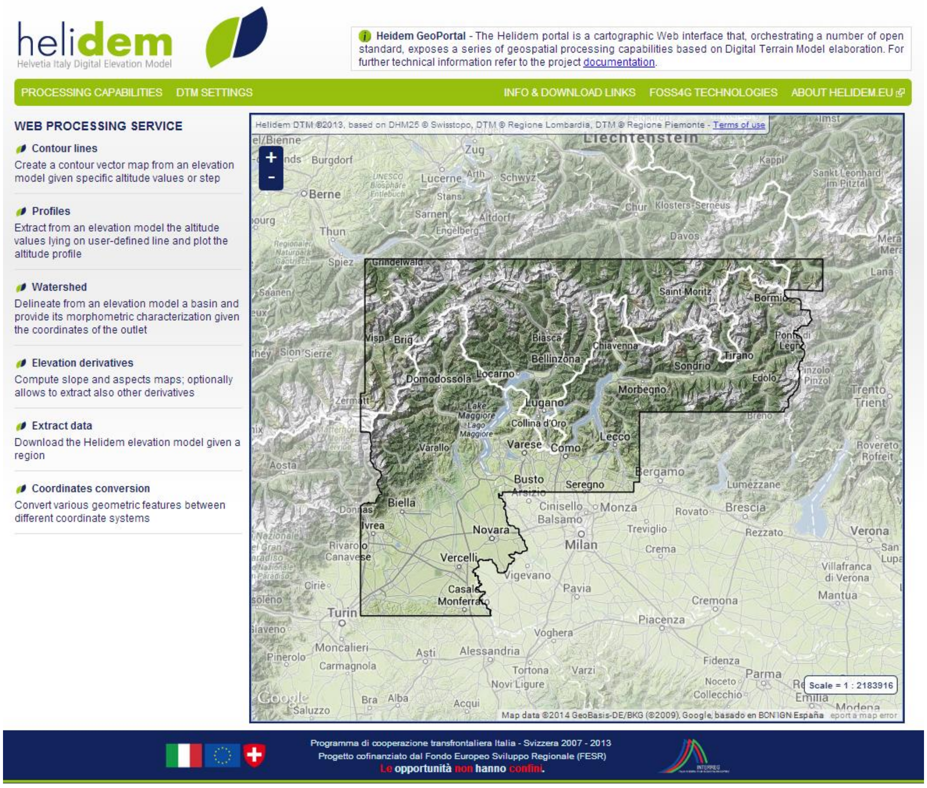
Based on the integration of these libraries and technologies the HELIDEM portal was designed and its graphical interface implemented as shown in Figure 4. The interface is separated in three sections: a left-side panel which presents all the offered processing capabilities, a right-side map panel to represent contextual base map and processing results and an up-side panel which shows general information on the selected process, data accessibility and copyrights.

4. HELIDEM PROCESSES

In Figure 4 the left-side panel shows a number of processing capabilities that are offered by the HELIDEM portal. Those processes have been implemented in the ZOO-project and offered by means of the WPS standard. The implemented Python code takes advantage of newly developed classes which enables the interaction with the GIS GRASS 7 and Geoserver.

The processes are thus orchestrated by the ZOO-project process core while the core of the processing is designated to GRASS and then the results are pushed into Geoserver for easy access and map navigation by means of WMS and WFS capabilities. To successfully connect Python, or other languages, with GRASS the environment variable of GRASS and the integrity of concurrent operation shall be guaranteed (Cannata et al, 2012). To publish the geospatial data resulting from the process we took advantage of the RESTful API (Diaz et al, 2011) of Geoserver and of the python-requests (Reitz, 2012) library.

Figure 4: HELIDEM Geoportal. Base Maps from Google.



Heidem GeoPortal - The Helidem portal is a cartographic Web interface that, orchestrating a number of open standard, exposes a series of geospatial processing capabilities based on Digital Terrain Model elaboration. For further technical information refer to the project documentation.

WEB PROCESSING SERVICE

Contour lines
Create a contour vector map from an elevation model given specific altitude values or step

Profiles
Extract from an elevation model the altitude values lying on user-defined line and plot the altitude profile

Watershed
Delineate from an elevation model a basin and provide its morphometric characterization given the coordinates of the outlet

Elevation derivatives
Compute slope and aspects maps; optionally allows to extract also other derivatives

Extract data
Download the Helidem elevation model given a region

Coordinates conversion
Convert various geometric features between different coordinate systems

Helidem DTM ©2013, based on DHM25 © Swisstopo, DTM © Regione Lombardia, DTM © Regione Piemonte - Terms of use

Map data ©2014 GeoBasis-DE/BKG (©2009), Google, basado en BCN IGN España

Programma di cooperazione transfrontaliera Italia - Svizzera 2007 - 2013
Progetto cofinanziato dal Fondo Europeo Sviluppo Regionale (FESR)
Le opportunità non hanno confini.

In order to achieve the automatic publishing of results, each implemented process includes a sub-process that:

- Export the output data in a specific formats (shapefiles for vectors and GeoTiff for raster) in a directory accessible by Geoserver.
- Add the data to Geoserver creating the store and the coverage with the RESTful API.
- Create the Style Layer Descriptor (SLD) file using specifically developed commands to export the GRASS styles in SLD.
- Upload and assign the style to the coverage with Geoserver RESTful API.

Albeit the used version of ZOO-Project supports the publication of results of a process as WMS, WFS or WCS service, this feature is available statically setting the ZOO configuration file (zcfg). Dynamic styling with SLD is not supported.

Together with the desire of testing the Geoserver RESTful API this is the reason why we decided not to use this ZOO-Project feature.

In the next sections the processes accessible through the portal are shortly described relevant modules, inputs and outputs.

4.1. Data Extraction

This feature has been implemented taking direct advantage of the WCS capability, in fact, once defined the bounding box and the coordinate system, the JavaScript compose a *getCoverage* request to the WCS server hosting the DTM. The result is available to the user as download.

4.2. Coordinate Conversion

This capability is intended to convert a user-drawn vector feature in different coordinate system. This could be useful, for example, to get the bounding box or the polygon of an area of interest in Well Known Text (WKT) format. The derived coordinates are indicative, in the sense that this feature uses general conversion parameters as defined in EPSG definition and therefore is not suitable for high precision transformations. No server processing is involved in this module and all the jobs are performed at browser level taking advantage of the proj4js library.

4.3. Contour lines

This feature is actually implemented in two WPS processes: given a defined elevation model the first allows for the extraction of contour lines at a predefined list of altitudes (Table 1 and Table 2) while the second at defined intervals among two extremes. The processes are written in Python and take advantage of *r.contour* module of GRASS 7.

Table 1: Contour Levels at Defined Altitudes: inputs and outputs. (Type: i/s = input type string, i/d = input type decimal, o/U = output type URI, o/s = output type string, o/dt = output type datetime; M = mandatory, N = multiplicity).

Parameter	Type	Description	M	N
covermap	i/s	Link to a DTM in geoTiff	Y	1
coverSRS	i/s	EPSG code (EPSG:XXXX)	Y	1
bbox	i/s	BBOX (EWKT)	Y	1
levels	i/d	CSV list of values	Y	n
oSRS	i/s	EPSG code	Y	1
oformat	i/s	output format: GML, KML or Shapefile (default)	N	1
odata	o/U	Link to the contour level zipped file	Y	1
WMS_URL	o/U	WMS address to access the results	Y	1
WFS_URL	o/U	WFS address to access the results	Y	1

layerName	o/s	Laye name to be used in WMS e WFS	Y	1
Expiration Time	o/dt	Exipration date of the results	Y	1
message	o/s	Additional process information	Y	1

Table 2: Contour Levels at Defined Intervals: inputs and outputs. (Type: i/s = input type string, i/d = input type decimal, o/U = output type URI, o/s = output type string, o/dt = output type datetime; M = mandatory, N = multiplicity).

Parameter	Type	Description	M	N
covermap	i/s	Link to a DTM in geoTiff	Y	1
coverSRS	i/s	EPSG code (EPSG:XXXX)	Y	1
bbox	i/s	BBOX (EWKT)	Y	1
step	i/d	interval	Y	n
minLevel	i/d	Minimum altitude	N	1
maxLevel	i/d	Maximus altitude	N	1
oSRS	i/s	EPSG code	Y	1
oformat	i/s	out format: GML, KML or SHP (default)	N	1
odata	o/U	Link to the contour level zipped file	Y	1
WMS_URL	o/U	WMS address to access the results	Y	1
WFS_URL	o/U	WFS address to access the results	Y	1
layerName	o/s	Layer name to be used in WMS e WFS	Y	1
Expiration Time	o/dt	Expiration date of the results	Y	1
message	o/s	Additional process information	Y	1

4.4. Profiles

Provided a linestring this process allows to extract the altimetric profile along the path as CSV file of 3D coordinates and PNG image. This feature is based on the *r.profile* module of GRASS combined with the Python library matplotlib (Hunter, 2007). Process parameters are listed in Table 3.

Table 3: Profile extraction process: inputs and outputs. (Type: i/s = input type string, i/d = input type decimal, o/U = output type URI, o/s = output type string, o/dt = output type datetime; M = mandatory, N = multiplicity).

Parameter	Type	Description	M	N
covermap	i/s	Link to a DTM in geoTiff	Y	1
coverSRS	i/s	EPSG code (EPSG:XXXX)	Y	1
bbox	i/s	BBOX (EWKT)	Y	1
coord	i/d	Linestring in EWKT format	Y	n
oSRS	i/s	EPSG code	Y	1
oformat	i/s	output format: GML, KML or Shapefile (default)	N	1
outImage	o/U	PNG of the profile	Y	1
outText	o/U	CSV of 3D coordinates	Y	1

Expiration Time	o/dt	Expiration date of the results	Y	1
Message	o/s	Additional information	Y	1

4.5. Elevation Derivatives

Using the module *r.slope.aspect* this process generates raster maps of slope, aspect, curvatures and partial derivatives from an elevation raster map. Aspect is calculated counter clockwise from east. Process parameters are listed in Table 4.

Table 4: Elevation derivative process: inputs and outputs. (Type: i/s = input type string, i/d = input type decimal, o/U = output type URI, o/s = output type string, o/dt = output type datetime; M = mandatory, N = multiplicity).

Parameter	Type	Description	M	N
covermap	i/s	Link to a DTM in geoTiff	Y	1
coverSRS	i/s	EPSG code (EPSG:XXXX)	Y	1
bbox	i/s	BBOX (EWKT)	Y	1
oSRS	i/s	EPSG code	Y	1
slope	i/s	Create slope map (yes/no)	N	1
format	i/s	Slope format (degrees/perc.)	N	1
aspect	i/s	Create slope map (yes/no)	N	1
pcurv	i/s	Create slope map (yes/no)	N	1
tcurv	i/s	Create tangential map (yes/no)	N	1
dx, dy, dxx, dyy, dxy	i/s	Create derivative map (yes/no)	N	1
odata	o/U	Zip of output maps	Y	1
slopeLayer	o/U	Layer name	Y	1
aspectLayer	o/U	Layer name	Y	1
pcurvLayer	o/U	Layer name	Y	1
tcurvLayer	o/U	Layer name	Y	1
dxLayer,	o/U	Layer name	Y	1
dyLayer,	o/U	Layer name	Y	1
dxxLayer	o/U	Layer name	Y	1
dyyLayer	o/U	Layer name	Y	1
dxylayer	o/U	Layer name	Y	1
WMS_URL	o/U	WMS address to results	Y	1
WFS_URL	o/U	WFS address to results	Y	1
Expiration Time	o/dt	Expiration date of the results	Y	1
message	o/s	Additional process information	Y	1

4.6. Watershed Analysis

This process allows to conduct a number of analysis of a basin using the *r.basin* command (Di Leo and Di Stefano 2013) specifically ported during this work at the version 7 of GRASS and integrated with a the newly developed modules named *r.nearest.coord*, which find the coordinates of the nearest cell having value higher of a threshold, and *r.out.sld,* which produce valid style descriptor.

Given a DTM the process evaluate the flow accumulation and flow direction maps (Tarboton and Ames, 2001) using one of the possible approaches: the Single Flow Direction (SFD) which allows the water to flow in one cell only using the maximum slope criterion or the Multiple Flow Direction (MFD) which allows the water to flow in all the neighbour lower cells at the same time proportionally to the slope. SFD approach is further subdivided in two algorithms named D8 if all the neighbour cells are considered or D4 if only the 4 neighbour cells along the cardinal directions are considered.

Combining the calculated flow accumulation and flow direction maps with the coordinate of the closing section of the watershed and a threshold value of the accumulation, the process calculates the basin and the hydrographic network. These maps are further used in the process to perform a number of analyses and elaborations that leads to the morphological and hydrological characterization of the watershed through a number of outputs including raster maps, plots and reports.

The produced raster maps are: hierarchical classification of the hydrographic network according to Horton, Strahler, Hack and Shreve (Gangodagamage et al, 2011), distance to outlet and the length of the slopes. The process, using the *r.out.sld* module, produces for each map a SLD (Lupp, 2007) which defines symbolization and colouring of layers.

The created plots are: hypsographic and hypsometric curves and width function (Strahler, 1957). The reports are: CSV file and PDF report with a number of morphometric parameters, including centre of gravity, area, perimeter, mean slope, length of the directing vector, prevalent orientation, characteristic altitudes, shape factors, topological diameter, magnitude, Horton ratios and concentration time, drainage density. The PDF have been produced using the pyUNO library (OpenOffice.org, 2010) which allows to produce documents using the libreoffice API. Process parameters are listed in Table 5.

Table 5: Watershed analysis process: inputs and output. (Type: i/s = input type string, i/d = input type decimal, o/U = output type URI, o/s = output type string, o/dt = output type datetime; M = mandatory, N = multiplicity).

Parameter	Type	Description	M	N
covermap	i/s	Link to a DTM in geoTiff	Y	1
coverSRS	i/s	EPSG code (EPSG:XXXX)	Y	1
bbox	i/s	BBOX (EWKT)	Y	1
coord	i/d	Point in EWKT format	Y	1
threshold	i/s	Flow accumulation threshold	Y	1
method	i/s	Flow direction algorithm	Y	1
flat_area	i/s	Beatify of flat area	Y	1
oSRS	i/s	EPSG code	Y	1

parameters	o/U	CSV report	Y	1
pdf_report	o/U	PDF report	Y	1
outmaps	o/U	Zip of output raster maps	Y	1
ipsographic_curve	o/U	PNG of hypsographic curve	Y	1
ipsometric_curve	o/U	PNG of hypsometric curve	Y	1
width_function	o/U	PNG of width function	Y	1
WMS_URL	o/U	WMS address to access the results	Y	1
WFS_URL	o/U	WFS address to access the results	Y	1
networkLayer	o/s	WMS/WFS River network layer name	Y	1
dist2outLayer	o/s	WMS/WFS distance to outlet layer name	Y	1
accumulationLayer	o/s	WMS/WFS flow accumulation layer name	Y	1
hillslope_distanceLayer	o/s	WMS/WFS hillslope length layer name	Y	1
hackLayer	o/s	Hack classification layer name	Y	1
hortonLayer	o/s	Horton classification layer name	Y	1
shreveLayer	o/s	Shreve classification layer name	Y	1
strahlerLayer	o/s	Strahler classification layer name	Y	1
Expiration Time	o/dt	Expiration date of the results	Y	1
message	o/s	Additional process information	Y	1

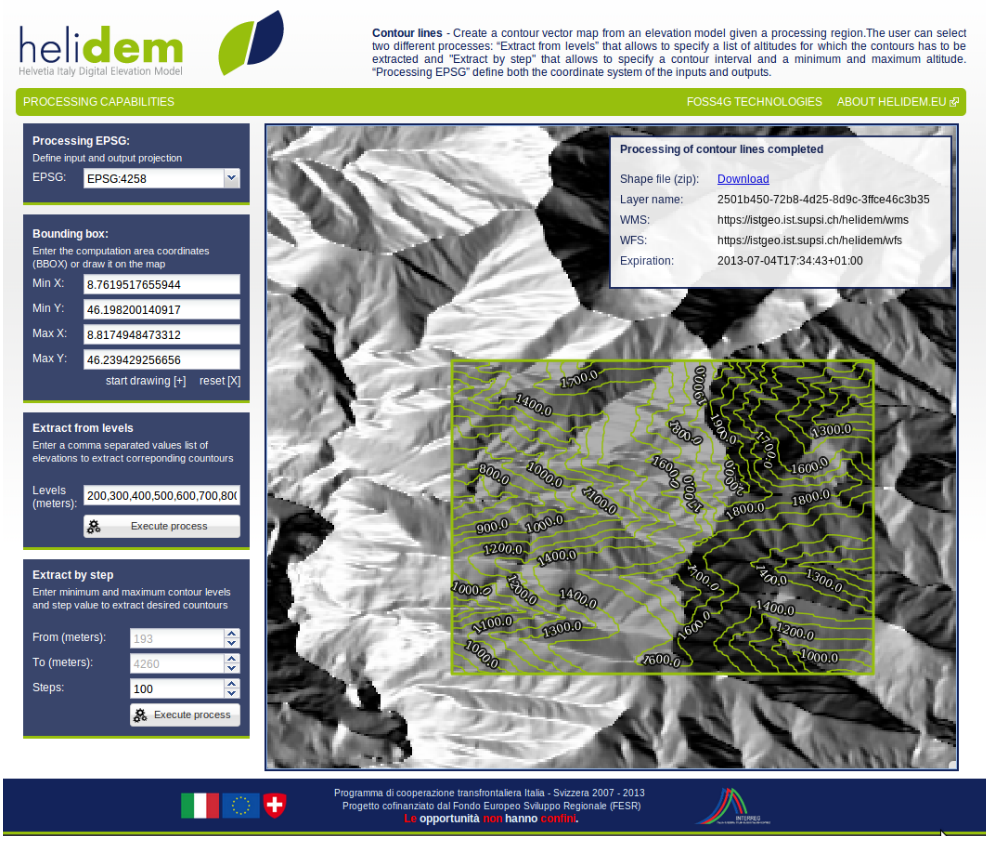
5. HELIDEM GEOPORTAL

The HELIDEM geo-portal (http://geoservice.ist.supsi.ch/helidem) is the official access point of the project to the produced data. Moreover, it provides access to the previously described processes, which even if available as stand-alone services, are best suited to be chained by means of the portal business logic.

When the user selects a desired process (see Figure 5), the left-side panel allows to set the required inputs parameters by means of selecting available options, inserting values and text, or interacting with the map to derive geospatial features like bounding boxes, point or polylines; it worth to be noted that all the geospatial features are also editable as text box, so that expert users could define these parameters by means of known coordinates.

When the process is executed, it runs asynchronously and the server keep updated the execute response document indicating the current status of the process; the client periodically check for process status description, so that the user is aware if the process is running, what is the approximate percentage of execution and eventual fails.

Figure 5: Example of Processing Interface with User Inputs on the Left-Side Panel And Results In The Top-Right Frame And On Map.



Contour lines - Create a contour vector map from an elevation model given a processing region.The user can select two different processes: "Extract from levels" that allows to specify a list of altitudes for which the contours has to be extracted and "Extract by step" that allows to specify a contour interval and a minimum and maximum altitude. "Processing EPSG" define both the coordinate system of the inputs and outputs.

Processing EPSG:
Define input and output projection
EPSG: EPSG:4258

Bounding box:
Enter the computation area coordinates (BBOX) or draw it on the map
Min X: 8.7619517655944
Min Y: 46.198200140917
Max X: 8.8174948473312
Max Y: 46.239429256656

Extract from levels
Enter a comma separated values list of elevations to extract correponding contours
Levels (meters): 200,300,400,500,600,700,80(

Extract by step
Enter minimum and maximum contour levels and step value to extract desired countours
From (meters): 193
To (meters): 4260
Steps: 100

Processing of contour lines completed

Shape file (zip): Download
Layer name: 2501b450-72b8-4d25-8d9c-3ffce46c3b35
WMS: https://istgeo.ist.supsi.ch/helidem/wms
WFS: https://istgeo.ist.supsi.ch/helidem/wfs
Expiration: 2013-07-04T17:34:43+01:00

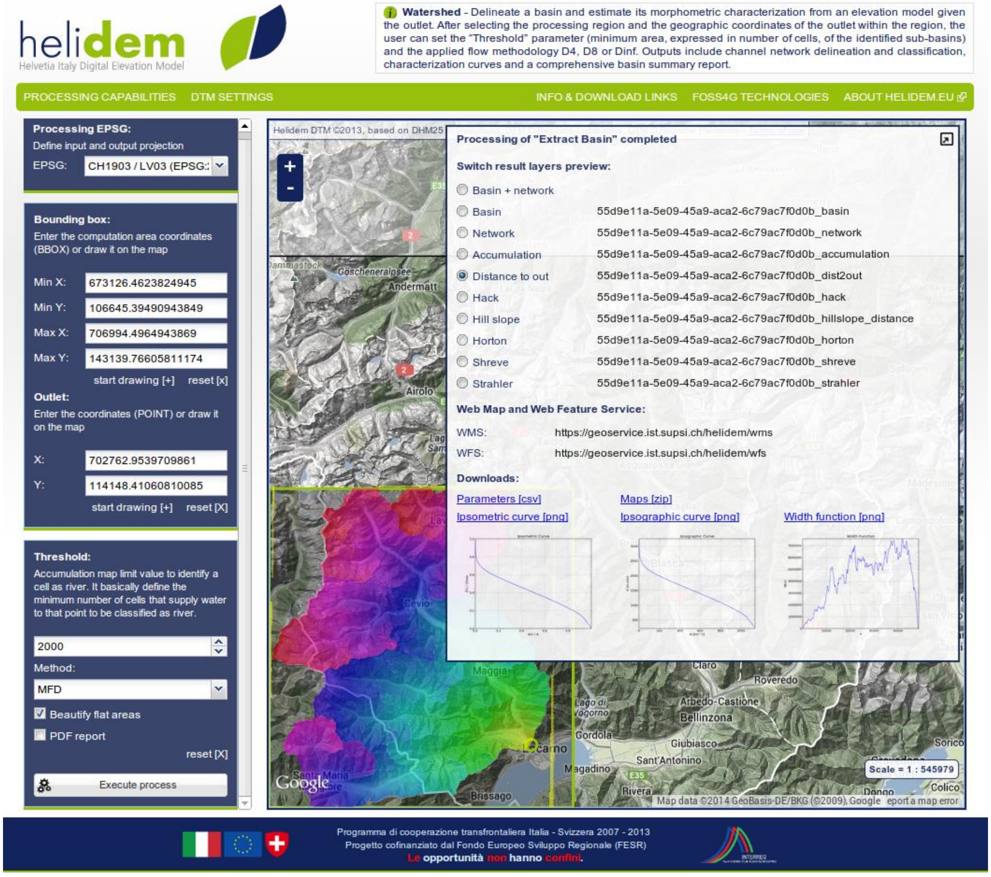
Programma di cooperazione transfrontaliera Italia - Svizzera 2007 - 2013
Progetto cofinanziato dal Fondo Europeo Sviluppo Regionale (FESR)
Le opportunità non hanno confini.

When the process execution ends, results are made available within a collapsible frame on top right of the map: link to zipped layers, layer names, WMS and WFS URLS, images and expiration date are opportunely reported. At the same time, derived geospatial data are represented in the map as additional WMS layer; in case of multiple layers in the outputs (see Figure 6), the user has the ability to select one of them for its visualization.

6. LOAD TESTING

Load testing is performed to determine a system's behaviour under both normal and anticipated peak load conditions. In the next paragraphs system configuration, test settings and results are presented.

Figure 6: Example of Processing Interface with Multiple Output Layers and Different Output Types.


helidem
Helvetia Italy Digital Elevation Model
PROCESSING CAPABILITIES
DTM SETTINGS
INFO & DOWNLOAD LINKS
FOSS4G TECHNOLOGIES
ABOUT HELIDEM.EU
Helidem DTM ©2013, based on DHM25
Google
Scale = 1 : 545979
Map data ©2014 GeoBasis-DE/BKG (©2009), Google
Programma di cooperazione transfrontaliera Italia - Svizzera 2007 - 2013
Progetto cofinanziato dal Fondo Europeo Sviluppo Regionale (FESR)
Le opportunità non hanno confini.

Watershed - Delineate a basin and estimate its morphometric characterization from an elevation model given the outlet. After selecting the processing region and the geographic coordinates of the outlet within the region, the user can set the "Threshold" parameter (minimum area, expressed in number of cells, of the identified sub-basins) and the applied flow methodology D4, D8 or Dinf. Outputs include channel network delineation and classification, characterization curves and a comprehensive basin summary report.

Processing EPSG:
Define input and output projection
EPSG: CH1903 / LV03 (EPSG:

Bounding box:
Enter the computation area coordinates (BBOX) or draw it on the map

Min X:	673126.4623824945
Min Y:	106645.39490943849
Max X:	706994.4964943869
Max Y:	143139.76605811174

start drawing [+] reset [x]

Outlet:
Enter the coordinates (POINT) or draw it on the map

X:	702762.9539709861
Y:	114148.41060810085

start drawing [+] reset [X]

Threshold:
Accumulation map limit value to identify a cell as river. It basically define the minimum number of cells that supply water to that point to be classified as river.

2000

Method:
MFD

☑ Beautify flat areas
□ PDF report

reset [X]

Execute process

Processing of "Extract Basin" completed

Switch result layers preview:

○ Basin + network	
○ Basin	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_basin
○ Network	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_network
○ Accumulation	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_accumulation
● Distance to out	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_dist2out
○ Hack	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_hack
○ Hill slope	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_hillslope_distance
○ Horton	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_horton
○ Shreve	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_shreve
○ Strahler	55d9e11a-5e09-45a9-aca2-6c79ac7f0d0b_strahler

Web Map and Web Feature Service:

WMS:	https://geoservice.ist.supsi.ch/helidem/wms
WFS:	https://geoservice.ist.supsi.ch/helidem/wfs

Downloads:

Parameters [csv] Maps [zip]
Ipsometric curve [png] Ipsographic curve [png] Width function [png]

As previously described the HELIDEM geo-portal relies on a server-side component composed of several OWS services which are located on two different servers: a server for data processing and results dispatch implementing the WPS, WMS and WFS (herein after referred to as "WPS server") and a server for DTM serving implementing the WCS (herein after referred to as "WCS server"). Both are virtual machines set up using Oracle VM VirtualBox (Romero, 2010) whose characteristics are summarized in Table 6.

While the Geoserver instance has 2 GB of RAM assigned the ZOO-project does not have any pre-allocated memory and thus the limit is that of the operating system (4.5 GB including SWAP memory).

The load testing has been conducted using an open Source framework named Locust (Heyman et al, 2011). Locust is a scalable and distributed framework developed in Python and available under the MIT-license. This framework enable the set up of a load test under different scenarios identified by the number of concurrent users and the *hatch-ratio* which indicate the user spawned for seconds.

Table 6: Characteristics of the Servers Used For the Load Testing.

parameter	WPS server	WCS server
OS	Ubuntu server version 12.04, 32 bit	Ubuntu server version 12.04, 32 bit
RAM	4GB	4GB
CPU	Intel(R) Xeon(R) CPU E5-2650 0 @ 2.00GHz	Intel(R) Xeon(R) CPU E5-2650 0 @ 2.00GHz
N° of processors	6	4
Disk size	100GB	100GB

Locust test requires a “locustfile” which is a normal python file that overrides the *TaskSet* and *Locust* classes. The extension of the *TaskSet* class defines the user behaviors by mean of a series of tasks the simulated user would perform: generally those tasks are HTTP POST and GET requests. The extension of the *Locust* class represents one user which is defined trough few attributes: the *task_set* which specifies the operations to be considered (the implemented extension of the TaskSet class), the *min_wait* and *max_wait* which are respectively the minimum and maximum time, in milliseconds, that a simulated user will wait between executing each task. Beyond the standard HTTP error status codes (400-499) automatically detected by Locust, exception responses which are returned with a success HTTP status code (e.g. 200) can be opportunely caught and reported as task failure.

Operationally, when a test is started, each instance of the *Locust* classes (each concurrent user) start calling its *task_set* which pick one of the tasks and call it. It will then wait a random number of milliseconds, between *min_wait* and *max_wait*, and then launch a new task, and so on.

For the scope of the study, because the HELIDEM geo-portal is essentially an interface to a number of geo-processing features offered through the WPS that orchestrate other services, the load testing is focused on the implemented WPS processes. Thus, the load test was conducted setting up a single user type – HELIDEM geo-portal user – performing four different tasks on four different processing areas. Tasks are contour extraction, profile calculation, basin analysis and derivative elaboration; areas as reported in Table 7 were sampled in order to include large, medium, small and very small basins with respect to the Alpine morphology. The combination of task and area for each request is selected randomly.

Table 7: Processing Areas Used in Load Testing; Bbox Is Expressed As [minx, miny, maxx, maxy] in CH1903 Coordinate System and areas in sqkm.

Area name	Area	bbox	Basin type
Maggia	1236	673126,106645, 706994,143139	Large
Verzasca	488	694828,114091, 712831,141198	Medium
Breggia	210	719542,76682, 733188,92109	Small
Cama	43	733520,120819, 740044,127433	Very small

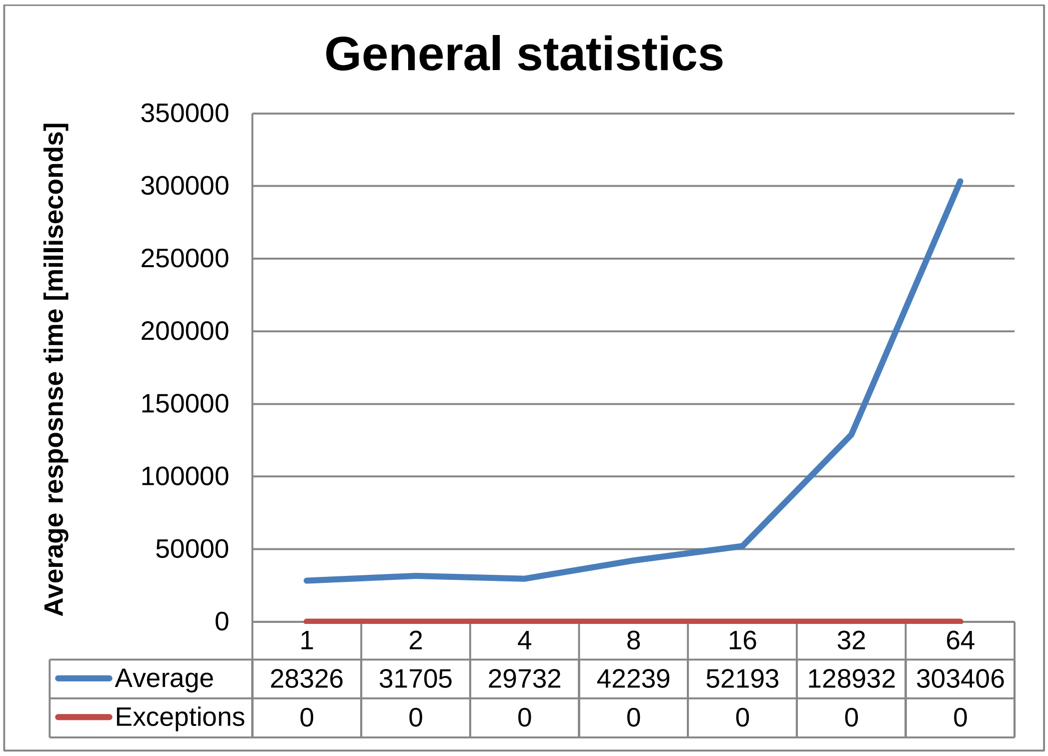
Different load tests were conducted simulating respectively 1, 2, 4, 8, 16, 32 and 64 concurrent users; each test was run for approximately 2 hours. The low number of simulated concurrent users depends on the fact that this is a specific portal expected to be used by a limited number of users from specific sectors and therefore with a very limited traffic with respect to other services like those offered by social network where millions of concurrent users are easily registered. Because of the expected duration of the processes (several tenths of a second), the *max_wait* was set to 2 minutes while the *min_wait* was set to 5 seconds; used *hatch-ratio* value was 1.

Figure 7 shows the average response time for the different scenarios of concurrent users without discrimination of request type. This plot clearly shows that the quality of the system is very high in fact no exception were raised over a total number of 3380 requests. Nevertheless, the performance of the system quickly degrades with increasing number of concurrent users, particularly evident with more than 16 users.

Plots A, B, C and D in Figure 8, illustrate the response time in milliseconds for each analysed process (Locust's task) over the growing number of concurrent users. The fastest process is the extraction of profile with response time that varies from 2.5 seconds (one users and very small basin) to almost 2 minutes (64 users and large basin) with an average of 18 seconds. The Contours extraction registered a response time between a minimum of 3.6 seconds and a maximum of 3 minutes with an average of 26.9 seconds. The time required by the user to get a response from the server in case of calculation of elevation derivative ranges from 8.4 seconds to 17.4 minutes with an average of 2.2 minutes. The longest process is the hydro-morphological analysis of watershed where a minimum of 24 seconds, a maximum of 23.5 minutes and an average of 3.15 minutes were recorded. WCS service response time (see plot E of Figure 8) vary smoothly with increasing concurrent users with respect to WPS processes. In plot F of Figure 8, the cost of data retrieval from WCS with respect to the total processing time for the watershed analysis is represented. The plot F shows that the WCS process is almost negligible in terms of response time.

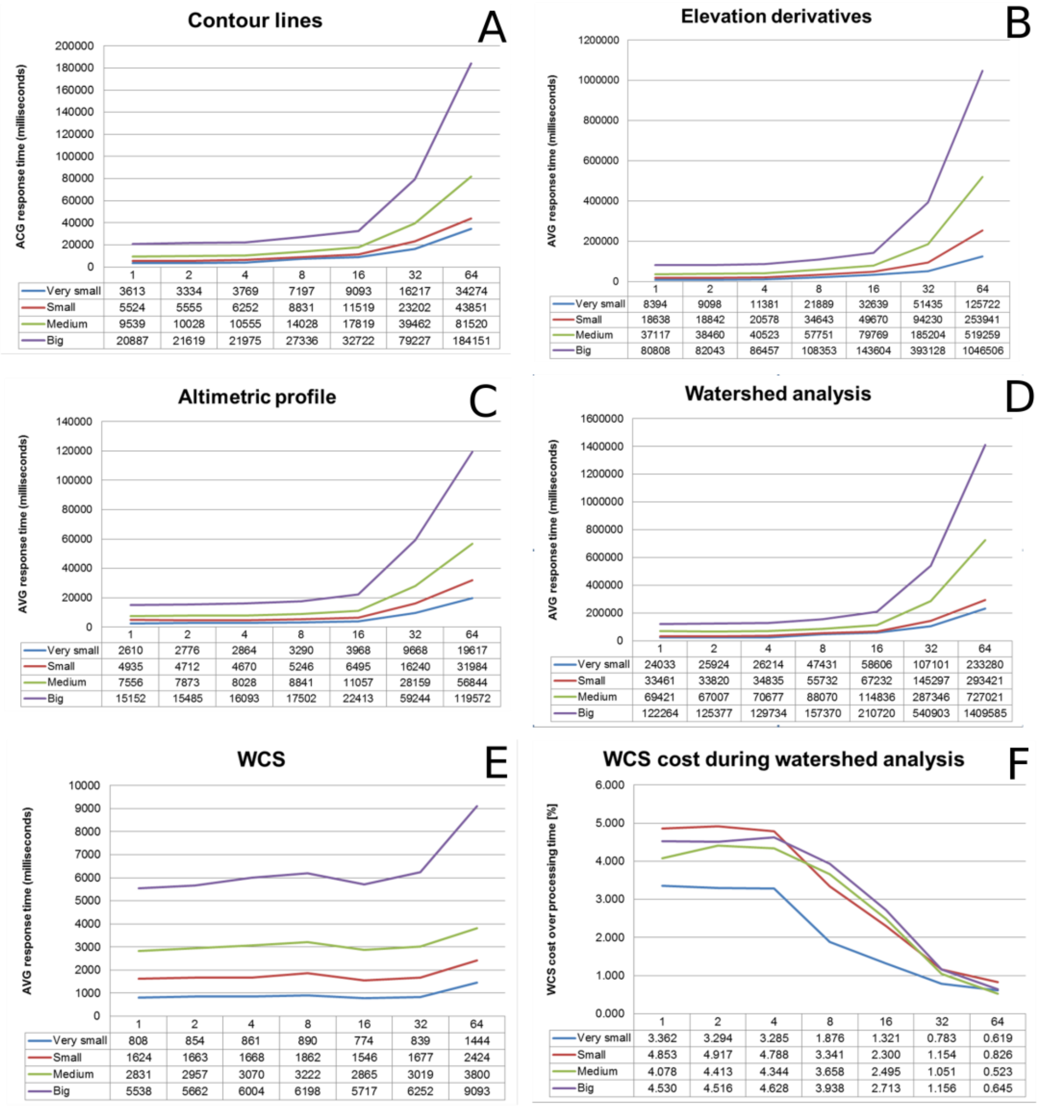
It worth to be noted that during the first phase of test settings we have experienced some issues related to disk space. In fact, results are stored for 24 hours and tend to exhaust disk space as the number of requests increase. Nevertheless, even if exception was raised, the system showed a good robustness to cope with errors during execution and to continue to operate despite anomalies: no crash and no downtime were recorded.

Figure 7: Average Response Time during the Whole Test.



	1	2	4	8	16	32	64
Average	28326	31705	29732	42239	52193	128932	303406
Exceptions	0	0	0	0	0	0	0

Figure 8. Load Testing Results. Variation of Response Time with Increasing Concurrent Users for Different Implemented Processes (A-D) and For the WCS Service (E). In Plot F The Cost Of WCS With Respect To Total Response Time For The Case Of Watershed Analysis.



	1	2	4	8	16	32	64
Very small	3613	3334	3769	7197	9093	16217	34274
Small	5524	5555	6252	8831	11519	23202	43851
Medium	9539	10028	10555	14028	17819	39462	81520
Big	20887	21619	21975	27336	32722	79227	184151

	1	2	4	8	16	32	64
Very small	8394	9098	11381	21889	32639	51435	125722
Small	18638	18842	20578	34643	49670	94230	253941
Medium	37117	38460	40523	57751	79769	185204	519259
Big	80808	82043	86457	108353	143604	393128	1046506

	1	2	4	8	16	32	64
Very small	2610	2776	2864	3290	3968	9668	19617
Small	4935	4712	4670	5246	6495	16240	31984
Medium	7556	7873	8028	8841	11057	28159	56844
Big	15152	15485	16093	17502	22413	59244	119572

	1	2	4	8	16	32	64
Very small	24033	25924	26214	47431	58606	107101	233280
Small	33461	33820	34835	55732	67232	145297	293421
Medium	69421	67007	70677	88070	114836	287346	727021
Big	122264	125377	129734	157370	210720	540903	1409585

	1	2	4	8	16	32	64
Very small	808	854	861	890	774	839	1444
Small	1624	1663	1668	1862	1546	1677	2424
Medium	2831	2957	3070	3222	2865	3019	3800
Big	5538	5662	6004	6198	5717	6252	9093

	1	2	4	8	16	32	64
Very small	3.362	3.294	3.285	1.876	1.321	0.783	0.619
Small	4.853	4.917	4.788	3.341	2.300	1.154	0.826
Medium	4.078	4.413	4.344	3.658	2.495	1.051	0.523
Big	4.530	4.516	4.628	3.938	2.713	1.156	0.645

7. CONCLUSIONS AND DISCUSSIONS

This work has presented the realization of a geospatial portal based on OWSs which uses WPS as the main component orchestrating WCS for data gathering and WFS and WMS for result dispatch and visualization. Load testing has been performed in order to understand the system behaviour under real case scenarios and concurrent access.

Test results show that the system has a good robustness and good quality of service; in fact no system failure and no exception response were registered. Performances are relatively good if compared with desktop processing and considering data loading times. Nevertheless, system response time exponentially degrades with process complexity, increasing number of users and target areas size. As a result, response time of WPS vary from about 2.5 seconds (to satisfy a single user requesting an altimetric profile over a very small basin area) to more than 20 minutes (to provide a watershed analysis over a large basin when 64 concurrent users are accessing the application).

Response time, as expected, is dependent on process algorithms complexity and increases with it. This is confirmed by tests, in fact higher latency can be observed from profile calculation to contour line extrapolation, to elevation derivatives creation, to watershed analysis. Looking at a single process the causes of service performance degradation are the presence of concurrent users and the size of the elaboration area.

The size of the elaboration area affects the response time because it set the area to be processed and / or because it defines the size of the data to be extracted and downloaded. To better understand the influence of remote access to data in a distributed system like this a WCS test was conducted. Results show that response time is sensitive to the number of concurrent user, but it also shows that if compared with WPS response time, it is slowly degrading. This means that, when a higher number of concurrent users are operating its impact is reduced. In the case of watershed analysis the WCS relative cost in term of time, decrease from about 5% when a single user is accessing the system to about 0.7% with 64 concurrent users. Moreover, the relative impact of data gathering is depending on the complexity of the process: in simpler processes like the profile extraction it accounts in average for about the 25% of the whole process, in more complex processes like basin analysis it is almost negligible counting for less than 3%.

From the previously presented considerations on data gathering cost, it can be deduced that the most important cause of HELIDEM service performance degradation is the concurrent processing. In particular, the bottleneck of this system is the CPUs load. The CPU is also quickly exhausted: during the tests starting from 16 users the CPUs usage was around 80% and from 32 users a usage rate of 100% was observed.

A final consideration to be taken into account is the disk space. If results are made available to users for a period, trough standard services and/or compressed archive, then the disk may run out of space. This is particularly important when long data availability periods are used and processes produce large data: derivatives process in the case study outputs up to 10 raster maps and about 30 Mb of data for large basin. A combination between average size of process outputs, length of the period of output availability and maximum number of request expected in the same period should be used to define the required disk space.

To improve the performance of the system under intensive concurrent accesses, the increase of computational power and disk space seems to be the natural solution to override these limitations: a scalable cloud computing service, like the Amazon Elastic Compute Cloud (Amazon EC2), could resolve the issue.

Nevertheless more research on process optimization should be conducted. For example, it would be interesting to verify the impact of asynchronous programming over response time; in fact it is a well-known approach to reduce the user waiting time by freeing resources when are not needed (generally during I/O operations like data download or file reading and writing) and making them available to other requests in accordance with the non-blocking paradigm. Another interesting option that could be investigated is the optimization of data storage, access and serving, like the Rasdaman array database (Owonibi and Baumann, 2012). Finally, the usage of Web Coverage Processing Service (WCPS) (Baumann, 2010b) instead of the WCS shall also be tested to verify if accessing pre-processed data instead of the raw DTM could increase the system efficiency.

In summary this research shows that the used software stack is robust and of good quality and that response time for processing digital terrain model services, also when complex operations are required, is reasonable when processing power is balanced with the number of concurrent users.

8. ACKNOWLEDGMENTS

Our thanks to the HELIDEM project team for the collaborative work, the Geodesy Division of swisstopo and to the UMUG office of the Canton Ticino for supporting this project.

9. REFERENCES

Biagi, L., Caldera, S., Carcano, L., Lucchese, A., Negretti, M., Sansò, F. and M. G. Visconti (2014). The HELI-DEM model estimation. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 1: 15-20.

Antonovic, A., Belotti, P., Brovelli, M.A., Caldera, S., Campus, S., Cannata, M. and M. G. Visconti (2014). "HELI-DEM: integrazione dei dati di altezza transalpini fra Italia e Svizzera", In SIFET 5/2013 Ludovico Biagi, Ambrogio

Maria Manzino, Fernando Sansò (Eds.) *SIFET Società Italiana di Fotogrammetria e Topografia.*

Barzaghi, R., Conte, R., Falletti, G., Maggi, A., Martino, M., Migliaccio, F. and N. Tselfes (2006). "Exploitation of GOCE data for a local estimate of gravity field and geoid in the Piemonte area (northern Italy)". In *Atti del 3rd International GOCE User Workshop* (pp. 6-8).

Baumann, P. (2010a). *OGC WCS 2.0 interface standard—core*. Open Geospatial Consortium Inc., Wayland, MA, USA, OpenGIS® Interface Standard OGC.

Baumann, P. (2010b). The OGC web coverage processing service (WCPS) standard. *Geoinformatica*, 14(4): 447-479.

Box, D., Ehnebuske, D., Kakivaya, G., Layman, A., Mendelsohn, N., Nielsen, H. F. and D. Winer (2000). *Simple object access protocol (SOAP) 1.1*.

Brovelli, M. A., Mitasova, H., Neteler, M. and V. Raghavan (2012). Free and open source desktop and Web GIS solutions. *Applied Geomatics*, 4(2): 65-66.

Cannata, M., Molinari, M. E., Luan, T. X. and N. H. Long (2012). Web Processing Services for Shallow Landslide. *International Journal of Geoinformatics*, 8(1): 25-34.

De la Beaujardiere, J. (2006). *OpenGIS® web map server implementation specification*. Standard Specification, 06-042.

Di Leo, M. and M. Di Stefano (2013). An Open-Source Approach for Catchment's Physiographic Characterization. In *AGU Fall Meeting Abstracts* (Vol. 1, p. 06).

Díaz, L., Granell, C., Gould, M. and J. Huerta (2011). Managing user-generated information in geospatial cyberinfrastructures. *Future Generation Computer Systems*, 27(3): 304-314.

Fenoy, G., Bozon, N. and V. Raghavan (2013). ZOO-Project: the open WPS platform. *Applied Geomatics*, 5(1): 19-24.

Flanagan, D. (2006). *JavaScript: the definitive guide*. O'Reilly Media, Inc.

Gangodagamage, C., Belmont, P. and E. Foufoula ‐ Georgiou (2011). Revisiting scaling laws in river basins: New considerations across hillslope and fluvial regimes. *Water Resources Research*, 47(7).

GeoExt Community (2010). *GeoExt Documentation*, available at: http://www.geoext.org/docs.html [accessed 24 November 2014].

Gosling J., Joy B., Steele G., Bracha G. and A. Buckley (2013). *The Java Language Specification, Java SE 7 Edition*. Addison-Wesley Professional.

Hazzard, E. (2011). *OpenLayers 2.10 Beginner's Guide*. Packt Publishing Ltd.

Heyman, J., Byström, C. and J. Hamrén (2011). *User load testing tool for web services*, ESN Social Software, 2011-11-05, available at: http://locust.io and version 0.4 of Locust, https://github.com/locustio/locust [accessed 24 November 2014]

Hogan, B. P. (2011). *HTML5 and CSS3: Develop with Tomorrow's Standards Today*. Pragmatic Bookshelf.

Hunter, J.D. (2007). Matplotlib: A 2D Graphics Environment, *Computing in Science & Engineering*, 9(3): 90-95

International Organization for Standardization and International Electrotechnical Commission, ISO/IEC 9899 (1999), *Programming languages—C*

Lupp, M. (2007). *Styled layer descriptor profile of the web map service implementation specification*. Open Geospatial Consortium Inc. OGC, 1(0).

Neteler, M., Bowman, M. H., Landa, M. and M. Metz (2012). GRASS GIS: A multi-purpose open source GIS. *Environmental Modelling & Software*, 31: 124-130.

OpenOffice.org (2010). *Python-UNO bridge*, available at: http://www.openoffice.org/udk /python/python-bridge.html [accessed 24 November 2014].

Owonibi, M. and P. Baumann (2012). “D-WCPS: A Framework for Service Based Distributed Processing of Coverages”. *Proc. GEOProcessing 2012, The Fourth International Conference on Advanced Geographic Information Systems, Applications, and Services, Valencia, Spain*, January 30, 2012, pp. 215 – 221.

Reitz, K. (2012). *Requests: HTTP for Humans*, available at: http://docs.python-requests.org/en/latest [accessed 24 November 2014].

Romero, A. V. (2010). *VirtualBox 3.1: Beginner's Guide*. Packt Publishing Ltd. Chicago

Schut, P. and A. Whiteside (2007). *OpenGIS web processing service*. OGC project document.

Strahler, A. N. (1957). *Quantitative analysis of watershed geomorphology*. Civ. Eng, 101, 1258-1262.

Tarboton, D. G., and D. P. Ames (2001). “Advances in the mapping of flow networks from digital elevation data”. *World water and environmental resources congress USA*: Am. Soc Civil Engrs. pp. 20-24.

The Apache Software Foundation (2013). *Apache HTTP Server Version 2.2 Documentation*. Available at: http://httpd.apache.org/docs/2.2 [accessed 24 November 2014].

The Apache Software Foundation (2013). A*pache Tomcat 7 – Documentation Index*. Available at: http://tomcat.apache.org/tomcat-7.0-doc/index.html [accessed 24 November 2014].

Turton, I. (2008). “Geo tools” In *Open source approaches in spatial data handling* (pp. 153-169). Springer Berlin Heidelberg.

Van Rossum, G. (2007). “Python Programming Language”. In *USENIX Annual Technical Conference*.

Vieux, B. E. (2001). *Distributed hydrologic modeling using GIS*. Springer Netherlands.

Vretanos, P. A. 2010. *OpenGIS Web Feature Service 2.0 Interface Standard*. Open Geospatial Consortium Inc, Version, 2(0).

Wilson, J. P., and Gallant, J. C. (2000). “Digital terrain analysis”, *Terrain analysis: Principles and applications*, 1-27.

Youngblood, B. (2013). *GeoServer Beginner's Guide*. Packt Publishing Ltd..

Zhang, J. F., Wang, J. X., X. R. Jia (2010). *Design and implementation of Data Maintenance System based on ExtJS* [J]. Railway Computer Application, 1, 014.