Geospatial meteorological and climatic data services application programming interface

A G Titov
Federal Research Center for Information and Computational Technologies, Tomsk, Russia
titov@scert.ru

Abstract. At present, it becomes necessary to develop software for Earth science data repositories instead of downloading data to computational servers and workstations. This is especially important taking into account the upcoming volume of big environmental datasets. WPS (Web Processing Service) is an OGC standard for distributed geospatial data processing services. The WPS services provide access to geospatial data processing packages including statistical processing software tools. The main goal of this paper is to present an approach that provides a unified access to the web processing services developed for climate and meteorological research based on the latest technological and scientific achievements. Based on the "OGC API - Processes" standard, a Web API Climate is proposed to provide a uniform access to the “Climate” platform (http://climate.scert.ru/climate/environment/v3.0a-demo/) geospatial services. The Climate Web API provides a software basis for an advanced development and use of the “Climate” platform web processing services aiming at climate change studies at regional and global levels. A correct implementation of the specification will form a solid foundation for further development of desktop client applications in the form of GIS plugins, as well as universal Web-GIS clients easily embedded into thematic virtual research environments.

1. Introduction
At present, bringing software to the Earth science data repositories instead of downloading data to the computational servers and workstations is becoming increasingly necessary, taking into account the upcoming volume of big environmental datasets. The modern distributed information-computational infrastructure aimed at complex use of heterogeneous geospatial datasets is based on the spatial data infrastructure concepts [1], which represent a combination of geospatial data, metadata, approved standards, as well as geospatial web services for access, computational processing, and cartographic visualization [2]. There are two main architectural styles most commonly used during the web service development: SOAP (Simple Object Access Protocol) and REST (Representational State Transfer) [3]. The SOAP services perform function invocations on remote systems by using remote procedure calls. The corresponding application programming interface (API) used is described by using the Web Services Description Language (WSDL, https://www.w3.org/TR/wsd1.html) including data types used, requests allowed, network protocol used, and web service name and location (URL). The REST, in turn, is an alternative architectural model where REST interfaces rely exclusively on Uniform Resource Identifiers (URI) for resource detection and interaction. In the REST, the interaction with web services is based upon stateless transfer (no client context is stored at the server between

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd.

IOP Conf. Series: Earth and Environmental Science 611 (2020) 012059 doi:10.1088/1755-1315/611/1/012059
requests). It should be noted that RESTful web services do not require additional configuration related to firewall settings due to HTTP use. Thus, the REST web service follows four basic design principles [4]:

- Explicit usage of HTTP methods
- Stateless interaction
- Exposing directory structure-like URIs
- Transferring XML and JavaScript Object Notation (JSON)

These basic REST design principles establish a one-to-one mapping between create, read, update, and delete (CRUD) basic operations and HTTP methods:

- To create a resource on the server: POST.
- To retrieve a resource: GET.
- To change/update the state of a resource: PUT.
- To remove or delete a resource: DELETE.

Thus, the RESTful Web services allow the requesting systems to access and manipulate with textual representations of Web resources by using a uniform and predefined set of stateless operations. An essential benefit is that the REST API user can easily explore self-describing resources while developing without having to refer to API documentation. It also means that the API user is not required to manually construct the URIs to be called, which causes a great number of software faults.

Turning to geospatial processing, according to the OGC (Open Geospatial Consortium), a process is defined as an algorithm, calculation, or model which either produces new data or transforms input data into output data, while running a process is defined as a job. A service provides a collection of processes implementing scientific algorithms. A complex process as a combination of processes represents a workflow, and a collection of related software compartments is a framework. The WPS (Web Processing Service) is an OGC standard for distributed geospatial data processing services providing access to software packages including statistical processing tools. The advantages of using the WPS for data processing and analysis are as follows:

- Output data of one process can be input data of another one, thus presenting a data analysis task as a workflow.
- Output data comes back as reference URLs or encoded within the response.
- Processes could be designed independently.

The main goal of this work is to present an approach that provides unified access to web processing services (WPSs) developed for climate and meteorological research using a “Climate” platform [5] based on the latest technological and scientific achievements. As a result of the state-of-the-art analysis, the REST architectural style was chosen for further development.

2. Methodology
A Web API represents an application programming interface for either a web server (server side) or a web browser (client side), where the server-side Web API is a programmatic interface of the defined request–response message system (web service), typically expressed in JSON or XML. To apply the Web API approach to the RESTful web services, an OpenAPI Specification (http://spec.openapis.org/oas/v3.0.3) is used, which specifies machine-readable interface files for describing, using, and visualizing RESTful web services.

At present, the OGC API family of standards (https://ogcapi.ogc.org/) is being developed based on the OGC Web Service standards (WMS, WFS, WPS) to facilitate web access to geospatial data and
tools. An "OGC API - Processes" standard draft was created based on the WPS 2.0 specification. It focuses on a simple RESTful core specified as reusable OpenAPI components. The "OGC API - Processes" enables the execution of computing processes and the retrieval of metadata describing their purpose and functionality. In Table 1, basic OGC API HTTP requests are presented with the corresponding WPS requests:

| OGC API HTTP request | Description | WPS Request |
|----------------------|-------------|-------------|
| GET /processes       | Lists the processes this API offers. | GetCapabilities |
| GET /processes/{process-id} | Returns a detailed description of a process. | DescribeProcess |
| POST /processes/{process-id}/jobs | Executes a process creating a new job. Inputs and outputs have to be specified in a JSON document to be sent in the POST body. | Execute |

To avoid, in general, the business logic coding in the web client code and, therefore, re-implementing the workflow transition rules for different types of clients (desktop, web, etc.), it is supposed to represent each workflow transition as a subresource and add hyperlinks in the JSON response to let the client know what workflow transitions are possible in the current state [6].

3. Climate Web API
Based on the "OGC API - Processes" standard, the Climate Web API is proposed to provide uniform access to the “Climate” platform geospatial services. Table 2 presents typical computational and utility processes corresponding to relevant HTTP requests required for platform functioning:

| Climate API HTTP request | Description |
|--------------------------|-------------|
| POST /processes/classes  | Lists the processing classes providing statistical analysis of geospatial data |
| POST /processes/classes/Average | Class calculating average value of the variable provided |
| POST /processes/classes/Minimum | Calculates minimum value of the variable provided |
| POST /processes/classes/Maximum | Calculates maximum value of the variable provided |
| POST /processes/classes/NumberofIcingDays | Calculates the number of icing days for the period provided |
| ...                        | ...         |

| POST /processes/MeteorologicalParameters | Lists meteorological parameters available for processing class (Average, etc.) |
| POST /processes/Collections | Lists data collections available for the processing class and meteorological parameter |
| POST /processes/Scenarios | Lists modeling scenarios for the data collection specified |
| POST /processes/SpatialResolutions | Lists spatial resolutions for the collection and scenario specified |
| POST /processes/TimeSteps | Lists time steps for the collection, scenario and spatial resolution specified |
The computational processes correspond to the “Climate” platform data processing modules implemented by using GDL/Python languages and realizing various statistical and climatological procedures, such as Number of icing days for the year, hydrothermal coefficient of Selyaninov, etc. It should be noted that the utility processes mentioned are based on the corresponding SQL queries to the metadata database described in [7] and developed for the “Climate” platform. They provide the metadata required by the end user to compile the proper XML/JSON configuration for his computational task as a consistent workflow: lists of meteorological parameters, datasets, spatial resolutions, and so on. Figure 1 presents a Climate API UML diagram displaying the basic web services.

**Figure 1.** Climate API UML diagram.

As an example, when applying the workflow transition approach mentioned earlier, we could add a subresource “availableTransitions” with potential transition changes to /processes/MeteorologicalParameters/{id}/resource:

```
GET /processes/MeteorologicalParameters/{id}/availableTransitions
```

For a meteorological parameter with selected id, this request will return the JSON code similar to this:
Thus, the client application will be aware of the next possible steps to be displayed to the end user (Figure 2). Once the “XML Configuration” object is done, it is used as input for the computational process. When the computational process is finished, the user is provided with the results in conventional formats (NetCDF, GeoTIFF, etc.). Figure 2 displays the current version of the “Climate” platform Web-GIS client implementing the business logic on the client side. The platform will be used as a prototype for its new version based on the REST principles.

Figure 2. Possible workflow transitions (marked orange) after selection of the meteorological parameter.

4. Conclusions
The Climate Web API presented above provides a software basis for an advanced development and use of the “Climate” platform web processing services aiming at climate change studies at regional and global levels. A correct implementation of the above-proposed specification will form a solid foundation for further development of the desktop client applications in the form of GIS plugins, as well as universal Web-GIS clients easily embedded into the thematic virtual research environments.

Acknowledgments
This study was supported under project AAAA-A17-117120670141-7 of a RAS Basic Research Program.

References
[1] Steiniger S and Hunter A J S 2012 Free and open source GIS software for building a spatial data infrastructure Geospatial Free and Open Source Software in the 21st Century ed Bocher E, Neteler M (Heidelberg: LNGC, Springer) 247-61
[2] Koshkarev A V 2008 Geoportal as a tool to control spatial data and services Spatial data 2 6-14
[3] Vitolo C, Elkhatib Y, Reusser D, Macleod C and Buytaert W 2015 Web technologies for environmental Big Data Environmental Modelling & Software 63 185–198
[4] Rodriguez A 2008 RESTful Web services: The basics IBM developerWorks 33 18
[5] Gordov E P, Okladnikov I G, Titov A G, Voropay N N, Ryazanova A A and Lykosov V N 2018 Development of Information-computational Infrastructure for Modern Climatology Russian Meteorology and Hydrology 43 11 722-28

[6] Lange K 2016 The Little Book on REST Services (Copenhagen) p 19

[7] Okladnikov I G, Gordov E P and Titov A G 2016 Development of climate data storage and processing model IOP Conf. Series: Earth and Environmental Science 48 012033 doi:10.1088/1755-1315/48/1/012033