OPENEO – A STANDARDISED CONNECTION TO AND BETWEEN EARTH OBSERVATION SERVICE PROVIDERS

Matthias Schramm[1], Edzer Pebesma[2], Wolfgang Wagner[1], Jan Verbesselt[3], Jeroen Dries[4], Christian Briese[5], Alexander Jacob[6], Matthias Mohr[2], Markus Neteler[7], Thomas Mistelbauer[5], Tomasz Miksa[1], Sören Gebbert[2,4], Bernhard Gößwein[1], Miha Kadunc[8], Pieter Kempeneers[9], Noel Gorelick[10]

[1]Vienna University of Technology (TU Wien), Vienna, Austria; [2]University of Münster, Münster, Germany; [3]Wageningen University and Research, Wageningen, The Netherlands; [4]Vlaamse Interstelling Voor Technologisch Onderzoek N.V., Boeretang, Belgium; [5]Earth Observation Data Centre for Water Resources Monitoring GmbH, Vienna, Austria; [6]Eurac Research, Bozen, Italy; [7]Mundialis GmbH & Co. KG, Bonn, Germany; [8]Sinergise Laboratorij Za Geografske Informacijske Sisteme Doo, Ljubljana, Slovenia; [9]Joint Research Centre, Ispra, Italy; [10]Google Switzerland GmbH, Zurich, Switzerland

ABSTRACT

Developments in Big Data technology during the last decade led to the parallel rise of several independent cloud service providers for Earth Observation (EO) data. The resulting variety of customized solutions of the back-end providers forces users to choose between very different, incompatible interfaces.

openEO offers an alternative, presenting an API which connects to EO service providers and provides standardised access points to users via programming languages as Python, R, and JavaScript. In this ongoing project process catalogues are being built up, covering all aspects of the EO data life cycle and serving as a template for interested service providers and front-end users to connect to the API. Several EO service providers can already be accessed via the API. Use cases, based on openEO will be developed for pilot users to proof its advantages and usability.

Index Terms— Earth Observation, Cloud service providers, Standardisation, Process catalogue

1. INTRODUCTION

Over the last decade important innovations in EO data utilization became possible due to novel sensors which combine a fine geometric with a high temporal resolution. The availability of an unprecedented variety and vastly increased volume of EO data led to a paradigm shift in data managing and processing procedures [1] towards cloud computing approaches that bring the users and their software to the data. New Big Data technologies, capable of dealing with the increased volume, variety, velocity and veracity of the EO data [2], are emerging, paving the way for new market players, service offerings and new user groups. For example, users may access raw and value-added Copernicus and other EO data by connecting to Infrastructure as a Service (IaaS) providers as the ESA's Copernicus Open Access Hub [3]. On the other side, advanced Platform as a Service (PaaS) solutions provide on-demand EO data processing through platform-specific API calls. These new platforms allow the user community to process Big Data decentralized at EO service providers, saving time and reducing internet traffic.

Being a rather recent development, EO platform service providers still offer solutions, customised to their user communities; speed and momentum of this new development prevented service providers from developing generally accepted standards. As a consequence, different data formats, process catalogues and available processing capabilities lead to an insufficient interoperability between the EO platform [4]. A switch between available service providers therefore always also means that users must re-develop already existing process chains.

openEO is an ongoing, user driven API development project to form a standardised access point to various EO service providers, allowing users EO data processing at different cloud platforms via Python, R, or JavaScript [5, 6]. Being language neutral, the openEO API can further be enhanced by implementing APIs for e.g. other programming languages and applications including Quantum GIS or GRASS GIS development, which is under way. The openEO API is currently capable to connect to EO service providers from the EODC / Austria, VITO / Belgium, Eurac Research / Italy, Mundialis / Germany, Sinergise / Slovenia, JRC / Italy, and Google Earth Engine / Switzerland. Also the connection to DIAS platforms is planned. Representing diverse infrastructures, these cloud providers are serving as templates for other back-ends to also connect to openEO.

Utilizing openEO, EO data platforms are able to provide services to the user, reflecting examples from all stages of EO

doi:10.2760/848593

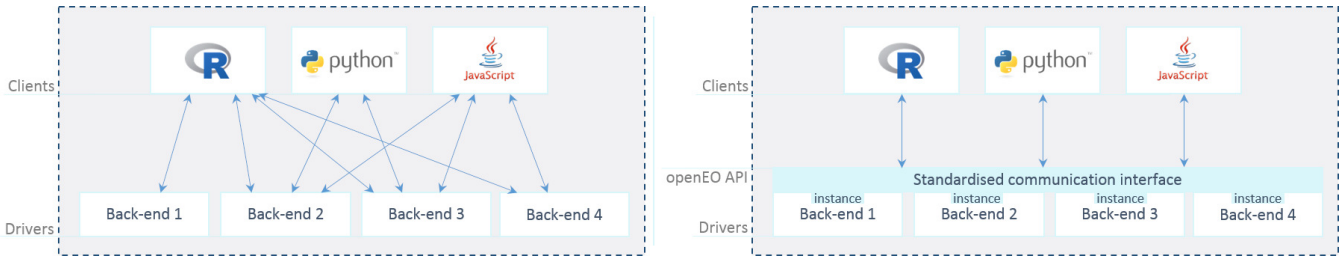

Fig. 1. Communication between clients and back-end drivers. Left: common many–to–many communication; Right: many–to–one communication, using the openEO API.

data processing, such as EO data query, image enhancement, band math operations, image mosaicking and layer stacking, or visualisation and download. Researchers will benefit by having a uniform way of connecting to different back-ends that allow them easily switch between compatible environments with little or no effort. Back-end operators do not need to modify the way back-end work, but need to provide additional interface that translates openEO requests to the software running in the back-end.

Compared to OGC WCPS, which currently sees rather limited uptake in the Earth Observation community, openEO is thought of as more user-oriented, more flexible in functionality, and less restrictive to back-end data representation: it allows for instance image collections that have not been mosaicked to a coverage. WCPS is explored as one of the possible back-ends.

Being a user-oriented open source project, the openEO consortium aims for an extended involvement of the user community while defining and developing relevant EO data processes and user-defined functions (UDFs). Connecting them to workflows for five already defined use cases will help ensuring and validating openEO's usability.

2. OPENEO STRUCTURE

openEO represents a set of standardised contracts between various clients and one well–defined API. Instances of the API can be installed at specific back-end drivers, which are deployed by EO service providers to implement these contracts and thus to establish openEO compatible access points. Contracts, formerly implemented separately for each client and service providers, following a many–to–many communication strategy can now be realised as many–to–one communication (see Fig. 1).

Client APIs are realised by software libraries specific to a given programming language, e.g. Python modules or R packages. This gives the user a way to interact with the openEO API without having to take care of the complexity of building back-end specific HTTP requests. By implementing an openEO instance as access point to a back-end provider, it will be able to process user commands at the back-ends infrastructures and return its results. The process graphs, transferred via JSON requests can be executed at the back-ends in three different ways.

1. A batch job can be submitted, which stays inactive until processing is requested. It will run only once and stores its results after execution.

2. Secondary web services allow web-based access using different protocols such as OGC WMS, OGC WCS or XYZ tiles. The computation runs on demand to allow users to change e.g. the result's viewing extent or level of detail.

3. Lightweight process graphs (e.g. small previews) can be executed synchronously. More costly processes have to expect timeouts for long-polling HTTP requests.

To enable the use of higher level EO process graphs, user-defined functions (UDFs) can be executed within openEO. Meeting extremely specialised demands, their implementation is an ongoing process that will need further input from potential users. While the standardisation of the available processes within openEO is still ongoing, its overall architecture was already specified / developed preliminarily, containing following aspects.

- Query of back-end capabilities regarding e.g. authentication method or UDF compatibility,
- Query of available EO data and processes, depending on metadata,
- User management: e.g. token based user registration / authentication, retrieval of user credits, billing,
- Synchronous / asynchronous job management,
- Data download in different output formats

3. API FUNCTIONALITY

A process catalogue was developed, describing a set of functionalities to be implemented within the project, their I/O data

doi:10.2760/848593

and their exact workflow. Its user-driven development will enable openEO to form widely accepted and used standards, and to build a consistent syntax. Furthermore, the back-ends will be able to translate the commands, standardised to the openEO's syntax to well defined internal processes, guaranteeing a compatibility between the EO service providers. 3rd-party processing platforms are able to use the process catalogue as a guideline to access to openEO.

Core processes of following topics are momentarily defined within an increasing list to be implemented in a standardised way within openEO.

- User data management: Authentication, EO data upload / storage / download / sharing,
- EO meta-database query,
- EO data masking, filtering by logical expressions / metadata / geometry / time range
- Image enhancement: zonal / pixel based math operators, re-projection, contrast enhancement stretching, kernels, predefined math operators (e.g. vegetation indices), geometric and temporal rescaling, spatial / temporal resampling,
- Math operations: binary arithmetic, statistical operations, Boolean operations, zonal statistics, regression,
- Subsetting, mosaicking, layer stacking, expanding / reducing dimensions,
- Sorting and ordering algorithms, searching for specific elements
- Visualizing / saving / downloading EO data,
- Real time interaction: job management, process monitoring

This list, containing all needed processes to realise 5 well–defined use cases (see Section 4), is meant to serve as a template for future enhancement.

Additional to the possible use of the openEO API via R, Python or JavaScript Syntax, a visual process graph builder serves as a GUI for merging single openEO processes to workflows (see Fig. 2).

4. USE CASES

The defined core processes will allow to process first use cases with openEO. The following use cases were chosen, based on existing demands from the user community, to provide processes for broad topics and to serve as seeding points for user discussions and for future development.

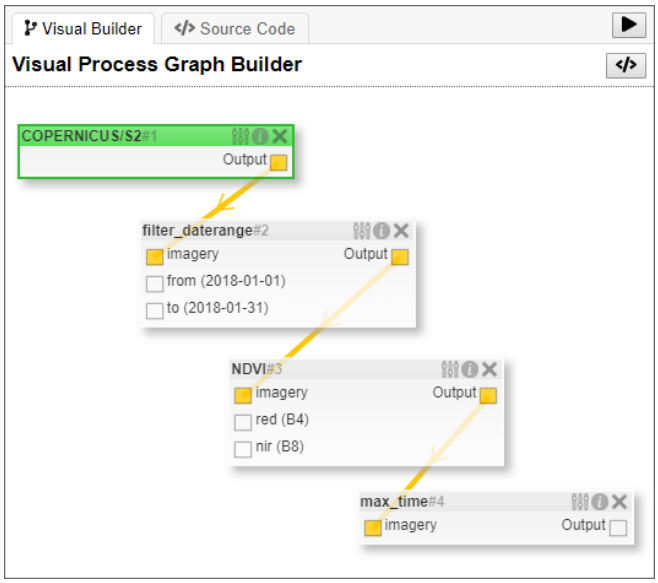

Fig. 2. Example of a visual process graph: NDVI calculation from a Sentinel-2 time series, using openEO.

4.1. Radar image compositing

The process chain produces monthly and seasonal RGB composites of Sentinel-1 backscatter [7]. The composites can be used for further classification and crop monitoring [8, 9] and will be a test case for the basic openEO's functionalities like querying and transforming data, basic statistics, layer stacking and exporting to specific output formats. This work flow will be tested by the Austrian *Federal Ministry Sustainability and Tourism* (BMNT) for Austrian pilot areas.

4.2. Multi-source phenology metrics and data fusion

Already existing data fusion and phenology metrics tools [10] will be ported to openEO to combine Sentinel-2 time series with Sentinel-3 and Proba-V data. Image pre-processing tools will be implemented as well as a product validation with in-situ and other ancillary datasets. These work flows will be used in semi-arid regions in Western Africa for the *Action against Hunger* (ACF) and in the Hindu Kush Region for the *International Centre for Integrated Mountain Development* (ICIMOD).

4.3. Optical-Radar forest monitoring

This use case focuses on the combination of Sentinel-1 and Sentinel-2 time series for a near real-time forest and deforestation monitoring. It provides openEO with various basic statistical and time series algorithms. The work flow will be tested in Latin America for the *Food And Agriculture Organization of the United Nations* (FAO).

4.4. Snow monitoring

This use case focuses on algorithms for detecting snow cover and snow status based on the combination of Sentinel-1 and Sentinel-3 time series. It includes basic user functionalities as e.g. user authentication or data management. This work flow will be tested in South Tyrol by the *Hydrologic office of the Province of Bolzano* to enrich their downstream services.

4.5. Agricultural monitoring

A test case on agriculture monitoring, based on Sentinel-1 and -2 time series will be tested on the JRC back-end (JEODPP). The use case includes functionalities as data extraction and reduction, machine learning methods and interactive data view applications, integrating 3rd party reference datasets.

5. USER INVOLVEMENT

openEO is an open source project and as such depends critically on user input – from remote sensing experts and software developers to back-end providers. Therefore the project consortium has established different communication channels:

1. A website was implemented, explaining in detail the ongoing development of the project: http://openeo.org/.
2. The newest version of the openEO source code is freely available at GitHub. The page also entails maintained user forums: https://github.com/Open-EO.
3. A first documentation and user manual is available at https://open-eo.github.io/openeo-api/.
4. Introduction videos and manuals are available on YouTube. Latest news regarding openEO are published via twitter: https://twitter.com/Open_EO.
5. An email hotline was established to provide users with a direct contacting possibility: `openeo@list.tuwien.ac.at`.

Furthermore, the project consortium hosts hackathons and publishes user questionnaires to enable interested users steering the openEO's path during and after the project's period.

6. ACKNOWLEDGEMENT

This project receives funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement n° 776242). This paper reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

REFERENCES

[1] T. Hey, S. Tansley, and K. Tolle, *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, October 2009. [Online]. Available: https://www.microsoft.com/en-us/research/publication/fourth-paradigm-data-intensive-scientific-discovery/

[2] S. Schade, "Big data breaking barriers - first steps on a long trail," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-7/W3, pp. 691–697, apr 2015.

[3] "Copernicus open access hub," https://scihub.copernicus.eu/, accessed: 2018-10-22.

[4] "Communication from the commission to the european parliament, the council, the european economic and social committee and the committee of the regions: European cloud initiative – building a competitive data and knowledge economy in europe," apr 2016. [Online]. Available: http://ec.europa.eu/transparency/regdoc/rep/1/2016/EN/1-2016-178-EN-F1-1.PDF

[5] E. Pebesma, W. Wagner, J. Verbesselt, E. Goor, C. Briese, and M. Neteler, "Openeo: a gdal for earth observation analytics," http://r-spatial.org/2016/11/29/openeo.html, accessed: 2018-10-22.

[6] E. Pebesma, W. Wagner, M. Schramm, A. Von Beringe, C. Paulik, M. Neteler, J. Reiche, J. Verbesselt, J. Dries, E. Goor, T. Mistelbauer, C. Briese, C. Notarnicola, R. Monsorno, C. Marin, A. Jacob, P. Kempeneers, and P. Soille, "Openeo – a common, open source interface between earth observation data infrastructures and front-end applications," *Zenodo*, 2017.

[7] D. Sabel, Z. Bartalis, W. Wagner, M. Doubkova, and J.-P. Klein, "Development of a global backscatter model in support to the sentinel-1 mission design," *Remote Sensing of Environment*, vol. 120, pp. 102–112, may 2012.

[8] D. Nguyen, K. Clauss, S. Cao, V. Naeimi, C. Kuenzer, and W. Wagner, "Mapping rice seasonality in the mekong delta with multi-year envisat ASAR WSM data," *Remote Sensing*, vol. 7, no. 12, pp. 15 868–15 893, nov 2015.

[9] V. Naeimi, S. Hasenauer, S. Cao, B. Bauer-Marschallinger, A. Dostalova, S. Schlaffer, and W. Wagner, "Monitoring water resources using big data from sentinel-1 satellites," in *Proceedings of the 2014 conference on Big Data from Space (BiDS'14)*, Nov. 2014.

[10] J. Reiche, S. de Bruin, D. Hoekman, J. Verbesselt, and M. Herold, "A bayesian approach to combine landsat and ALOS PALSAR time series for near real-time deforestation detection," *Remote Sensing*, vol. 7, no. 5, pp. 4973–4996, apr 2015.

doi:10.2760/848593