ontopEO: VKGs over Earth Observation Data (Extended Abstract)

Albulen Pano[1,*], Davide Lanti[1], Diego Calvanese[1], Piero Campalani[2], Alexander Jacob[2] and Alessandro Mosca[3]

[1]Free University of Bozen-Bolzano, Italy
[2]Eurac Research, Viale Druso 1, 39100, Bozen-Bolzano, Italy
[3]ISTC Laboratory for Applied Ontology, National Research Council (CNR), Italy

Abstract

Earth Observation (EO) data publication and dissemination continues to grow driven by the need to consistently monitor the earth in times of climate change. openEO is one of the most well known APIs to query and process EO data and is operationally available on many EO cloud platforms such as the Copernicus Data Space Ecosystem or Destination Earth. To be truly valuable, EO data often needs to be combined with other data sources, notably relational databases. Knowledge graphs offer a way to bridge the semantic gap between EO data and these other sources. An unsolved issue with this solution is that the execution of integrated queries can be costly due to the enormous volume of EO data. In this paper, we tackle this challenge through the use of Virtual Knowledge Graphs — a paradigm that presents data to end-users as a knowledge graph, while being kept its original sources and formats. Our approach is prototyped and evaluated against multiple real-world openEO examples.

Keywords

Virtual knowledge graph, Earth observation, Ontology-based data access, Raster data, openEO

1. Introduction

The steady growth of satellite launches for Earth Observation (EO) has increased the range and volume of Earth imagery coverage. Satellite imagery is often represented as *raster datacubes* (or simply *raster data*) with multiple dimensions including space and time, as well as a feature vector (composed of *bands*) measuring different parts of the electromagnetic spectrum. While the spectral dimensionality is limited by the number of bands that satellites support (typically less than 10), the temporal and spatial dimensions can easily lead to an explosion in volume. Moreover, not all entities producing the data need to abide by the same dissemination strategies; as a result, different datasets and formats may arise not only from direct satellite observations but also from interpolation processes. This heterogeneity poses challenges in terms of data variety.

Several software solutions support EO data analysis, including relational extensions (e.g., PostGIS), Python libraries (e.g., rasterio, xarray), and array databases such as Rasdaman [1] and SciDB [2]. However, these approaches do not provide a unified, interoperable interface across systems, and each comes with specific assumptions about data storage and access.

A prominent solution to these challenges is openEO [3, 4, 5], a web API designed to provide a standardized interface for querying EO data backends, regardless of their underlying storage formats. Notably, openEO is an OGC Community Standard [6] supported by the European Space Agency (ESA). Central to openEO is the notion of a *process graph* — a directed acyclic graph that represents a sequence of processing steps to be applied to EO data. Each node in the graph corresponds to an *openEO process* (e.g., loading data, filtering by time, applying a function), and edges define data dependencies between

DL 2026: 39th International Workshop on Description Logics, July 17–19, 2026, Lisbon, Portugal
alpano@unibz.it (A. Pano); lanti@inf.unibz.it (D. Lanti); diego.calvanese@unibz.it (D. Calvanese); Piero.Campalani@eurac.edu (P. Campalani); Alexander.Jacob@eurac.edu (A. Jacob); alessandro.mosca@cnr.it (A. Mosca)
0000-0002-0905-9004 (A. Pano); 0000-0003-1097-2965 (D. Lanti); 0000-0001-5174-9693 (D. Calvanese); 0000-0002-1032-9825 (P. Campalani); 0000-0003-4434-7244 (A. Jacob); 0000-0003-2323-3344 (A. Mosca)

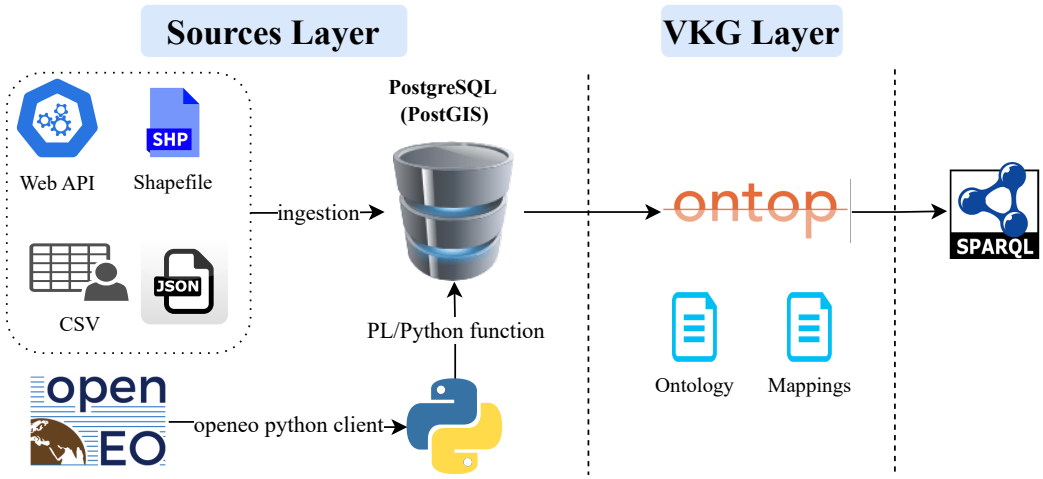

Figure 1: Architecture of ontopEO

these processes. This structure enables users to build flexible, modular, and backend-independent EO analysis workflows.

While using openEO addresses heterogeneity in EO data access, it does not by itself provide a means to integrate EO data with other domains. Virtual Knowledge Graphs (VKGs), a variant of Knowledge Graphs in which new graph datasets are not materialized, provide a lightweight solution for integrating heterogeneous data across diverse sources and formats. In VKGs, data remain in the original sources and are accessed via on-the-fly translations of user queries into source-specific requests, enabled by declarative *mappings* that link data to a target ontology. Among open-source solutions, Ontop [7, 8] is a prominent VKG system supporting most major relational databases.

In recent years, different approaches have been proposed to address the problem of accessing EO data in VKGs. Existing VKG-based approaches for EO [9, 10] rely on transforming raster data into tabular form or Rasdaman arrays before mapping it, introducing partial materialization and data duplication. The Plato system [11] instead accesses raster data directly using xarray over formats such as NetCDF or Zarr, avoiding database preloading. However, none of these approaches interfaces with openEO.

In this paper, we present ontopEO, the first extension of a VKG system (Ontop) to support EO data via the openEO specification. Our approach avoids any raster preloading or transformation: SPARQL queries are reformulated into API requests to openEO, and the retrieved results are combined with tabular data from PostGIS to compute the final answer.

This paper is an extended abstract of a recent publication [12] at the International Semantic Web Conference (ISWC 2025), and is structured as follows: Section 2 introduces the system architecture, Section 3 elaborates on the implementation within Ontop, Section 4 presents an experimental evaluation, and finally Section 5 discusses some limitations and future work.

2. System Architecture

Figure 1 illustrates the architecture of ontopEO, which is organized in two layers: the *Sources Layer*, interfacing with the underlying data sources, and the *VKG Layer*, providing integrated SPARQL access to these sources using the VKG paradigm via Ontop.

Sources Layer. At the basis of our architecture is PostgreSQL 17, a robust relational database, extended with PostGIS 3.5.0 to support geospatial capabilities. PostGIS is the widely adopted geospatial extension for PostgreSQL, enabling the storage, querying, and processing of both vector and raster geospatial data. In our architecture, we use PL/Python to interact with the openEO API via the official openEO

Python client [13], a well-maintained library for accessing openEO services. This integration offers enhanced flexibility by enabling the use of any Python package (or combination of packages) within the database backend.

VKG Layer. Our system extends *Ontop v5.2.1*. Ontop supports the main W3C standards for VKGs, including the RDF data model, the SPARQL query language, the OWL 2 QL [14] profile of OWL 2, and the R2RML [15] mapping language. The system takes as input an ontology that captures the domain knowledge, together with a mapping specification that links the relational tables (e.g., municipalities with geometries) stored in PostGIS to ontology elements. It exposes the underlying relational data to end users as RDF terms within the VKG, which can then be queried using SPARQL.

3. Ontop Implementation

Implementation of openEO Processes as SPARQL Functions. Our extension is based on the idea of mapping SPARQL functions to openEO processes. To this aim, we implemented a query-rewriting mechanism that composes complex function chains into a single process graph. This allows SPARQL queries to emulate process graphs (typically authored through the openEO Python client or web interfaces) by composing functions directly within the query. Currently, support is limited to the openEO processes required by the CRIMA project[1], including common operations such as `apply`, `mask`, and `reduce_dimension`, for a total of 18 processes. The system can be extended with additional processes by implementing a PL/Python function that links the openEO process to its corresponding SPARQL function, along with a JSON template defining the structure of the associated process graph.

SQL Translation. In Ontop, SPARQL queries over the VKG are translated into equivalent SQL queries over the underlying relational data sources. In our case, however, reformulation must go beyond SQL translation: it must also include calls to the openEO API. Moreover, multiple SPARQL function calls related to openEO within a single query need to be grouped and composed into a unified openEO process graph, so as to take advantage of the expressivity of the openEO API and reduce round-trip interactions. In ontopEO, this is handled by the PL/Python function `process_graph_function`, which generates an openEO process graph from a structured JSON representation. This representation is encoded as a flattened SQL array capturing the sequence of SPARQL openEO function calls, where each segment (delimited by identifiers of the form `id_`*n*) contains the function name, arguments, and dependency links. The function scans these segments, maps each to a predefined template for the corresponding openEO process, and constructs a valid JSON process graph that is then sent to the openEO backend for execution.

4. Experiments and Evaluation

Experiments. To comprehensively evaluate openEO queries using real world phenomena like heat-waves and wildfires, we considered diverse geographic regions, specifically selecting study areas in Italy and the Netherlands. The administrative boundary and population datasets were obtained from official national and regional sources. The idea is to use our system integrate this data with information coming from openEO. The openEO ecosystem consists of several data and infrastructure providers (commonly referred to as *cloud providers*), each hosting its own instance of the openEO Web API. One such provider is the *Copernicus backend*, part of the Copernicus Data Space Ecosystem [16], which we used for evaluating our solution. Copernicus provides a set of Python notebooks [17] that showcase the capabilities of openEO to execute queries for real-world use cases. Our queries, written in SPARQL, reflect those use-cases and we provide below 3 representative queries:

[1] https://crima.eurac.edu/

Q1. Find the NDVI for the municipalities in the Val Venosta district of South Tyrol between 15 April 2025 and 25 April 2025.

Q2. Find heatwave data in neighborhoods of the municipality of Krimpen aan den IJssel in the Netherlands between 1 June 2023 and 30 October 2023.

Q6 Find oil spill data in southern coast of Kuwait near the resort community of Al Khiran (reported in 2017).

Results and Discussion. Execution times show that ontopEO successfully answers all queries, although durations vary significantly — from approximately 2 minutes for **Q6** to about 2.5 hours for **Q1**. The relatively short execution time of **Q6** is due to the simplicity of the query, which returns only a single record defined by a bounding box and does not require integration with administrative boundaries data. In contrast, **Q1** involves 13 municipalities, each described by complex polygon geometries, which increases processing time considerably. Overall, these results demonstrate that ontopEO can effectively support a variety of real-world EO queries, as posed by domain experts, while highlighting the need for further optimizations if the query results require further post-processing.

Limitations. Current limitations apply to both the system and evaluation approach. The prototype supports only a subset of openEO processes, but we are working on extending it in the context of the CRIMA project; all SPARQL functions currently return `xsd:string`; the PL/Python integration introduces overhead while inheriting backend API rate limits. Finally, the evaluation focuses on realistic use cases rather than synthetic scalability benchmarks, and direct comparison with equivalent SPARQL-to-openEO systems is not yet possible.

Query time as illustrated above can differ notably based on the query scenario. If a query is spatially fragmented through many polygons or addressing a long time frame, this might be suboptimal with respect to how most data centers organize their collections of raster imagery. Often these datasets are stored as observed by the sensing system on the satellite, meaning that for each observation a large area is covered for one specific point in time. For example, for Sentinel-2 a standard delivery is a tile of 10000 $\times$ 10000 pixels covering an area of 100 km $\times$ 100 km, typically chunked as 10000 $\times$ 1 rows per spectral band (e.g near-infrared, red, green, etc.). A time series of NDVI for a specific polygon, e.g. a specific agricultural field, or forest plot, might cover only a small subset of this large image scene. If a query request is sent to that subset for e.g. a time series of multiple years over a large area of 100 $\times$ 100 pixels to study agricultural practices this becomes very inefficient in terms of IO performance. The query has to open 100 data chunks per image times two for the two bands and extracts only about 1% of the data that is extracted from those files for the analysis and this times the length of the time series. Current research and development in this field is hence focusing on data storage and chunking optimizations weighted against cost of storage, due to data duplication when rechunking data. These costs can quickly become astronomically high as we speak about data archives in the hundreds of petabytes.

5. Conclusions and Future Work

We introduced ontopEO, a tool that leverages the VKGs paradigm to query EO data via openEO. Our approach reformulates SPARQL queries, enriched with openEO functions, into process graphs that can be executed by openEO providers. ontopEO can answer typical Copernicus-related questions over a higher-level conceptualization enabled by a domain ontology. The tool has been developed and deployed within the context of the EFRE 1078 project CRIMA, which aims at the realization of an ontology-based decision support system for climate risk mitigation and adaptation. For future work, we plan to support additional openEO processes, including provider-specific UDPs.

Acknowledgments

This research has been partially supported by the HEU project CyclOps (GA n. 101135513), by the Province of Bolzano and FWF through project OnTeGra (DOI 10.55776/PIN8884924). by the Province of Bolzano and EU through projects ERDF-FESR 1078 CRIMA, and ERDF-FESR 1047 AI-Lab, and by MUR through the PRIN project 2022XERWK9 S-PIC4CHU. This work has been carried out while Albulen Pano was enrolled in the Italian National Doctorate on Artificial Intelligence run by Sapienza University of Rome in collaboration with Free University of Bozen-Bolzano. This work was partially supported by the Agenzia per la cybersicurezza nazionale under the programme for promotion of XL cycle PhD research in cybersecurity – B83C24005640005.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] P. Baumann, P. Furtado, R. Ritsch, N. Widmann, The RasDaMan approach to multidimensional database management, in: Proc. of the 12th ACM Symp. on Applied Computing (SAC), 1997, pp. 166–173. doi:10.1145/331697.331732.

[2] P. Cudré-Mauroux, Scidb, in: S. Sakr, A. Y. Zomaya (Eds.), Encyclopedia of Big Data Technologies, Springer, 2019, pp. 1467–1470. doi:10.1007/978-3-319-77525-8.

[3] M. Schramm, E. Pebesma, M. Milenković, L. Foresta, J. Dries, A. Jacob, W. Wagner, M. Mohr, M. Neteler, M. Kadunc, et al., The openEO API–harmonising the use of earth observation cloud services using virtual data cube functionalities, Remote Sensing 13 (2021) 1125.

[4] M. Mohr, E. Pebesma, J. Dries, S. Lippens, B. Janssen, D. Thiex, G. Milcinski, B. Schumacher, C. Briese, M. Claus, et al., Federated and reusable processing of earth observation data, Scientific Data 12 (2025) 194.

[5] OpenEO, OpenEO: An Open Earth Observation Processing Platform, https://openeo.org/, 2024. Accessed: 2024-12-10.

[6] A. Jacob, M. Stoicescu, R. Ahola, P. J. Zellner, R. Conway, C. Iacopino, I. Simonis, Introduction to the OGC geodatacube standard working group, in: IGARSS 2023 - 2023 IEEE International Geoscience and Remote Sensing Symposium, 2023, pp. 2489–2492. doi:10.1109/IGARSS52108.2023.10282998.

[7] G. Xiao, D. Lanti, R. Kontchakov, S. Komla-Ebri, E. Güzel-Kalayci, L. Ding, J. Corman, B. Cogrel, D. Calvanese, E. Botoeva, The virtual knowledge graph system Ontop, in: Proc. of the 19th Int. Semantic Web Conf. (ISWC), volume 12507 of *Lecture Notes in Computer Science*, Springer, 2020, pp. 259–277. doi:10.1007/978-3-030-62466-8_17.

[8] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro, G. Xiao, Ontop: Answering SPARQL queries over relational databases, Semantic Web J. 8 (2017) 471–487. doi:10.3233/SW-160217.

[9] Y. Hamdani, G. Xiao, L. Ding, D. Calvanese, An ontology-based framework for geospatial integration and querying of raster data cube using virtual knowledge graphs, ISPRS Int. J. of Geo-Information 12 (2023) 375. doi:10.3390/ijgi12090375.

[10] A. Ghosh, A. Pano, G. Xiao, D. Calvanese, OntoRaster: Extending VKGs with raster data, in: Proc. of the 8th Int. Joint Conf. on Rules and Reasoning (RuleML+RR), volume 15183 of *Lecture Notes in Computer Science*, Springer, 2024, pp. 108–123. doi:10.1007/978-3-031-72407-7_9.

[11] D. Bilidas, A. Mantas, F. Yfantis, G. Stamoulis, M. Koubarakis, J. M. Tárraga, E. Sevillano Marco, F. Castel, C. Laine, The semantic data cube system Plato and its applications, in: Proc. of the 44th IEEE Int. Symposium Geoscience and Remote Sensing (IGARSS), IEEE Computer Society, 2024, pp. 2514–2518. doi:10.1109/IGARSS53475.2024.10640737.

[12] A. Pano, D. Lanti, D. Calvanese, Virtual knowledge graphs over earth observation data, in: Proc. of the Int. Semantic Web Conf. (ISWC), Lecture Notes in Computer Science, Springer, 2025, pp. 149–166. URL: https://doi.org/10.1007/978-3-032-09530-5_9. doi:10.1007/978-3-032-09530-5_9.
[13] openEO Consortium, openEO Python Client, https://open-eo.github.io/openeo-python-client/, 2025. Accessed: 14-05-2025.
[14] B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue, C. Lutz, OWL 2 Web Ontology Language Profiles (Second Edition), W3C Recommendation, World Wide Web Consortium, 2012. URL: https://www.w3.org/TR/owl2-profiles/.
[15] S. Das, S. Sundara, R. Cyganiak, R2RML: RDB to RDF Mapping Language, W3C Recommendation, World Wide Web Consortium, 2012. URL: https://www.w3.org/TR/r2rml/.
[16] European Space Agency (ESA), Copernicus Data Space - openEO Interface, https://dataspace.copernicus.eu/analyse/openeo, 2025. Accessed: 14-05-2025.
[17] European Space Agency (ESA), Copernicus Data Space – Use Case Documentation, https://documentation.dataspace.copernicus.eu/Usecase.html, 2025. Accessed: 14-05-2025.