FROM BIG COPERNICUS DATA TO BIG INFORMATION AND BIG KNOWLEDGE: THE COPERNICUS APP LAB PROJECT

Konstantina Bereta[1], *Hervé Caumont*[2], *Ulrike Daniels*[5], *Daems Dirk*[3], *Manolis Koubarakis*[1], *Despina-Athanasia Pantazi*[1], *George Stamoulis*[1], *Sam Ubels*[4], *Valentijn Venus*[4], *Firman Wahyudi*[4]

[1]National and Kapodistrian University of Athens, Greece; [2]Terradue Srl, Italy; [3]VITO, Belgium; [4]RAMANI B.V., The Netherlands; [5]AZO Anwendungszentrum GmbH, Germany;

ABSTRACT

We discuss the challenges of big Copernicus data and how our project Copernicus App Lab has dealt with them. Copernicus App Lab takes data from the land monitoring, global land and atmosphere services and makes it available on the Web and the Cloud using semantic technologies to aid its take up by mobile developers. We also discuss lessons learned for information retrieval, database and knowledge management research in the context of Copernicus.

Index Terms— big data, semantic technologies, linked geospatial data, Earth observation, satellite remote sensing

1. INTRODUCTION

Copernicus data is a paradigmatic case of big data which is acquired by the Sentinel satellites and contributing missions, together with in-situ data from sensors on the ground, at sea, or in the air. Copernicus is at the forefront of all big data challenges: *volume, velocity, variety, veracity,* and *value*. The H2020 project Copernicus App Lab (http://www.app-lab.eu/) targets the *volume* and *variety* challenges of Copernicus data, and it follows the path of previous research projects TELEIOS, LEO, and MELODIES, funded by FP7 ICT. Copernicus App Lab goes beyond these projects in the following important ways. First, it develops a software architecture that enables on demand access to big Copernicus data using the well-known OPeNDAP framework and the geospatial ontology-based data access system Ontop-spatial [2]. Now users and application developers do not need to download data or learn the details of sophisticated data formats for EO data. All they need to develop is an ontology describing the data they are interested in and R2RML mappings that capture the correspondence between the ontology and the data sources containing the data. Using traditional approaches, application developers would have to implement different clients/adapters in their applications corresponding to the different file formats their data is in, in order to process the data. Instead of implementing custom code, they can use the functionalities of the Ontop-spatial mapping language for all data sources regardless of their formats.

Secondly, it brings computing resources close to the data by making the Copernicus App Lab tools available as Docker images that are deployed in the Terradue cloud platform as cloud services. The platform allows application developers to access Copernicus data and carry out massively parallel processing without the need to download the data and carry out the processing locally.

Thirdly, it enables search engines like Google to treat datasets produced by Copernicus as "entities" in their own right and store knowledge about them in their internal knowledge graph. In this way, search engines will be able to answer sophisticated users questions which is beyond the reach of modern search engines today.

2. THE COPERNICUS APP LAB ARCHITECTURE

Figure 1 presents the conceptual architecture of the *Copernicus integrated ground segment* and the Copernicus App Lab software architecture.

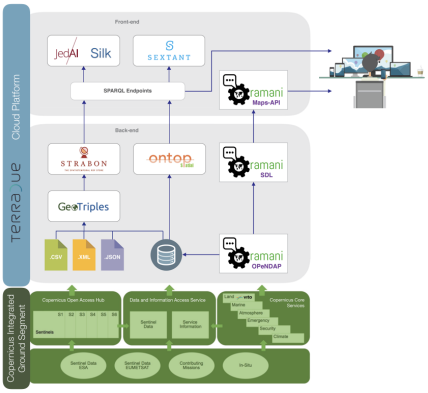

Fig. 1. The Copernicus integrated ground segment and the Copernicus App Lab software architecture

In the lower part of the figure, the Copernicus *data*

This work has received funding from EU Horizon2020, Grant Agreement nr. 730124.

sources are shown. These are Sentinel data from ESA, Sentinel data from the European Organisation for the Exploitation of Meteorological Satellites (EUMETSAT), satellite data from contributing missions and in-situ data. The next layer makes Copernicus data and information available to interested parties in three ways: via the Copernicus Open Access Hub, via the Copernicus Core Services and via the Data and Information Access Service (DIAS).

All the software components of the project run in the Terradue cloud platform (https://www.terradue.com/portal/). The platform allows cloud orchestration, storage virtualisation, and virtual machine provisioning, as well as application burst-loading and scaling on third-party cloud infrastructures. Within the Terradue cloud platform, the developer cloud sandbox service provides a platform-as-a-service (PaaS) environment to prepare data and processors. It has been designed with the goal to automate the deployment of the resulting EO applications to any cloud computing facility that can offer storage and computing resources (e.g., AWS).

In Copernicus App Lab, access to Copernicus data and information can be achieved in two ways: (i) by downloading the data via the Copernicus Open Access Hub or the Web sites of individual Copernicus services, and (ii) via the popular OPeNDAP framework (https://www.opendap.org/) for accessing scientific data. In the first case (workflow on the left part of the two top layers of Figure 1), the downloaded data should then transform into RDF using the tool GeoTriples [6] or scripts written especially for this task. GeoTriples enables the transformation of geospatial data stored in raw files (shapefiles, CSV, KML, XML, GML and GeoJSON) and spatially-enabled RDBMS (PostGIS and MonetDB) into RDF graphs using well-known geospatial vocabularies such as the Open Geospatial Consortium (OGC) standard GeoSPARQL [9]. The performance of GeoTriples has been studied experimentally [6] using large publicly available geospatial datasets. It has been shown that GeoTriples is very efficient especially when its mapping processor is implemented using Apache Hadoop.

After Copernicus data has been transformed into RDF, it can be stored in the spatiotemporal RDF store Strabon [5, 3]. Strabon can store and query linked geospatial data that changes over time. It has been shown to be the most efficient spatiotemporal RDF store available today using the benchmark Geographica in [4, 3]. Copernicus data stored in Strabon may also be interlinked with other relevant data. To do this in Copernicus App Lab, we use the interlinking tools JedAI and Silk. JedAI is a toolkit for entity resolution and its multi-core version has been shown to be scalable to large datasets [8]. Silk is a well-known framework for interlinking RDF datasets which we have extended to deal with geospatial and temporal relations [10].

The *novel way of accessing Copernicus data and information* in Copernicus App Lab is captured by the workflow on the right part of the two top layers of Figure 1, and it is based on the popular *OPeNDAP framework* for accessing scientific data. The *streaming data library (SDL)* implemented by RAMANI communicates with the OPeNDAP server and receives Copernicus services data as *streams.* In this way, SDL enables on-the-fly computation of spatial and temporal aggregations (e.g., a longterm moving average that is often of interest to EO applications). The SDL is accessible through a list of APIs that are enhanced with an API ontology, which directly links to a function ontology that describes the offered functionality and analytics. This ontology describes calls and responses of the API and assists users in determining valid functions over different data types. The API responses are provided as JSON-LD with direct references to the semantics of the returned variables, allowing easier interpretation. OPeNDAP and SDL are installed and configured by VITO on a virtual machine running on the VITO hosted PROBA-V mission exploitation platform (https://proba-v-mep.esa.int), which has direct access to the data archives of the Copernicus global land service. The installation of OPeNDAP was done using Docker and access to the Copernicus global land and PROBA-V datasets via OPeNDAP is realised by mounting the necessary disks on the virtual machine.

One of the main contributions of Copernicus App Lab is the extension of the ontology-based data access system Ontop-spatial [1] with OPeNDAP support. Ontop-spatial is a system that connects to existing geospatial databases and creates virtual semantic graphs on top of them using ontologies and mappings, without downloading files and transforming them into RDF. Mappings encode how we map relational data to RDF terms. As we describe in [2], the new version of Ontop-spatial is able to connect to non-relational external data sources (e.g., APIs like OPeNDAP) and enable users to pose GeoSPARQL queries on top of them without the need of importing the data in relational databases.

Finally, data can be visualized using the tools Sextant [7] or Maps-API (https://ramani.ujuizi.com/maps/index.html). Sextant is essentially a GIS for linked geospatial data. It enables users to build layered maps consisting of geospatial data made available in various formats (e.g., KML, GML etc.) and SPARQL or GeoSPARQL endpoints. The Maps-API is similar to Sextant in terms of visualization functionality, but it takes its data from SDL and it cannot deal with linked geospatial data sources accessed by SPARQL or GeoSPARQL.

All tools are open source and they are available on the following Web page: http://kr.di.uoa.gr/#systems

3. A COPERNICUS APP LAB CASE STUDY

A simple case study, which demonstrates the functionality of the Copernicus App Lab software, involves studying the "greenness" of Paris. This can be done by relating "greenness" features of Paris using geospatial data sources such as OpenStreetMap and relevant Copernicus datasets from the land monitoring service of Copernicus, which are the leaf-

area index dataset (global), the CORINE land cover dataset (pan-European) and the Urban Atlas dataset (local).

Leaf area index (LAI) is a dimensionless quantity that characterizes plant canopies and it is defined as the one-sided green leaf area per unit ground surface area in broadleaf canopies (https://en.wikipedia.org/wiki/Leaf_area_index). The *CORINE land cover dataset* covers 39 EU countries (https://land.copernicus.eu/pan-european/corine-land-cover). Land cover is characterized using a 3-level hierarchy of classes with 44 classes in total at the 3rd level. The *Urban Atlas dataset* (https://land.copernicus.eu/local/urban-atlas/view) provides land use and land cover data for European urban areas, and it covers 800 urban areas in 28 EU countries.

In addition to the above datasets, our case study utilizes data from OpenStreetMap and the global administrative divisions dataset GADM. OpenStreetMap is an open and free map of the whole world constructed by volunteers. GADM (https://gadm.org/) is an open and free dataset giving us the geometries of administrative divisions of various countries.

The first task of any case study using the Copernicus App Lab software is to develop INSPIRE-compliant ontologies for the selected Copernicus data. The *INSPIRE directive* (https://inspire.ec.europa.eu/) aims to create an interoperable spatial data infrastructure for the EU, to enable the sharing of spatial information among public sector organizations and better facilitate public access to spatial information across Europe.

Once all the ontologies are defined, we can easily translate them into RDF using a custom script. Then, they can be stored in Strabon and be queried jointly in interesting ways. For example, assuming appropriate PREFIX definitions, the following GeoSPARQL query asks for the LAI values of the area occupied by the Bois de Boulogne park in Paris.

```
SELECT DISTINCT ?geoA ?geoB ?lai WHERE {
  ?areaA osm:poiType osm:park.
  ?areaA geo:hasGeometry ?geomA . ?geomA geo:asWKT ?geoA .
  ?areaA osm:hasName  "Bois de Boulogne"^^xsd:string .
  ?areaB lai:lai ?lai .
  ?areaB geo:hasGeometry ?geomB . ?geomB geo:asWKT ?geoB .
  FILTER(geof:sfIntersects(?geoA, ?geoB))}
```

Similarly, in Figure 2, we have used Sextant to build a temporal map that shows the "greenness" of Paris, using the datasets LAI, GADM, CORINE land cover, Urban Atlas and OpenStreetMap. We show how the LAI values (small circles) change over time in each administrative area of Paris (administrative areas are delineated by magenta lines) and correlate these readings with the land cover of each area (taken from the CORINE land cover dataset or Urban Atlas).

All RDF datasets and ontologies that have been discussed above are freely available at: http://kr.di.uoa.gr/#datasets.

The "greenness of Paris" case study can also be developed using the workflow on the right in the Copernicus App Lab software architecture of Figure 1. In this case, the datasets can be queried using Ontop-spatial and visualized in Sextant without transforming any datasets into RDF. In this case, the developer has to write R2RML mappings expressing the correspondence between a data source and classes/properties in the corresponding ontology. An example of such a mapping is provided below (in the native mapping language of Ontop-spatial which is less verbose than R2RML).

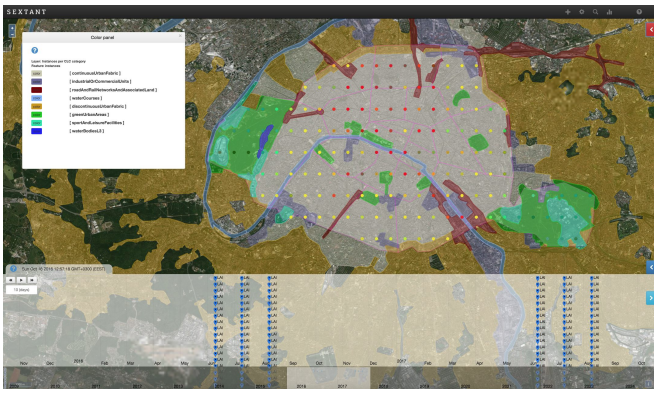

Fig. 2. The "greenness" of Paris

```
mappingId opendap_mapping
target lai:{id} rdf:type lai:Observation .
       lai:{id} lai:lai {LAI}^^xsd:float;
                time:hasTime {ts}^^xsd:dateTime .
       lai:{id} geo:hasGeometry _:g .
       _:g geo:asWKT {loc}^^geo:wktLiteral .
source SELECT id, LAI, ts, loc FROM (ordered opendap
       url:https://analytics.ramani.ujuizi.com/
       thredds/dodsC/Copernicus-Land-timeseries-global
          -LAI%29/readdods/LAI/)  WHERE LAI > 0
```

In this mapping, the `source` is the LAI dataset, provided through the RAMANI OPeNDAP server of the Copernicus App Lab software stack. The dataset contains observations that are LAI values, the time and location for each observation. Operator `Opendap` retrieves this data and populates a virtual SQL table with schema `(id,LAI,ts,loc)`. Because of the fact that the `Opendap` operator is implemented as an SQL user-defined operator, it can be embedded into any SQL query. In the above mapping, we also refine the data we want to be translated into virtual RDF terms by adding a filter to the query to eliminate negative or zero LAI values. The `target` part of the mapping encodes how the relational data is mapped into RDF terms.

4. LESSONS LEARNED AND FUTURE CHALLENGES

The use of OPeNDAP offers better data access capabilities specifically for application developers that are not experts in EO, and thus it a clear benefit. OPeNDAP and SDL provide streaming data to the user and have some significant advantages over the OGC Web Coverage Service standard which is already offered by VITO. First of all, from a data provider perspective, OPeNDAP is easier to use, as it is able to deal with a wider variety of grid types. Furthermore, OPeNDAP

can be easily extended with different conventions, allowing for easier integration of different datasets and without overhead like file conversion. Also, OPeNDAP enables the loose coupling of different Copernicus data sources into one data model, providing the user easy access through a single access point that uses this data model. Finally, when using the Web Coverage Service, there is limited possibility to obtain client-specific parts of the datasets (one is limited to, for example, a bounding-box). In contrast, OPeNDAP allows for the caching of datasets by serialization based on internal array indices.

The most innovative aspect of using Ontop-spatial in Copernicus App Lab is its ability to give access to Copernicus data through the OPeNDAP framework. When data is stored in a database connected with Ontop-spatial, DBMS optimisations and database constraints are applied and query plans are optimized. This does not happen in the case where Ontop-spatial retrieves data on-the-fly from OPeNDAP, since data is preprocessed before it gets translated into virtual triples using Ontop-spatial. However, if we want to access Copernicus data that gets frequently updated, the virtual RDF graphs approach is useful as it avoids the repeated translation steps that have to be done by the data provider. For costly operations (e.g., spatial joins of complex geometries), it is better to materialize the data. To improve performance, we have implemented a caching mechanism so that queries that result in the same API calls for a time window w, whose length is a configurable parameter, can get cached data. We also extended our system with the ability to integrate other kinds of data e.g., HTML tables and social media data (e.g., twitter, foursquare). In our current work, we are developing further optimisation techniques to improve performance.

Participants of the ESA Space App Camp (www.app-camp.eu/) that was organised in September 2017 and 2018 had the opportunity to use the Copernicus App Lab technologies to implement demo applications. The objective was to make EO data, particularly from Copernicus, accessible to a wide range of businesses and citizens. The developers of the winning teams AiR and URBANSAT used Copernicus App Lab tools to access and integrate data from different sources.

It is important to point out that an approach very similar to our projects TELEIOS, LEO, Melodies and Copernicus App Lab is currently been taken by the CREODIAS platform, a cloud-based one-stop shop for all Copernicus satellite data and imagery, as well as the Copernicus services information (https://creodias.eu/). The CREODIAS approach is limited though since only *metadata* of Copernicus datasets are available as linked data and can be queried by relevant discovery tools. We, on the other hand, allow users to also make information and knowledge extracted from Copernicus data available as linked data. In this way it can be combined with other linked datasets (public or private) and enable the development of applications by mobile developers easily. In this way we contribute to the *value* dimension of big Copernicus data.

Google has recently activated the beta version of its dataset search (https://toolbox.google.com/datasetsearch), where the datasets that are indexed using *schema.org* (https://schema.org/), as proposed by Google, show up. We have followed these guidelines and annotated all the datasets used in the use case of Section 3, and made them available at the following link: http://kr.di.uoa.gr/#datasets. We have also recommended that the same practice is followed by the Copernicus services we have worked with (land monitoring, global land and atmosphere services). Our current work focuses on designing an extension to the community vocabulary *schema.org* appropriate for annotating EO data in general and Copernicus data in particular, by extending the class *Dataset* with subclasses and properties which cover the EO dataset metadata defined in relevant OGC standards.

5. SUMMARY

The Copernicus App Lab project targets the variety and volume challenges, and has developed a novel software stack that can be used to develop applications using Copernicus data even by developers that are not experts in EO. We presented a case study developed using the Copernicus App Lab software stack and discussed lessons learned and future plans.

REFERENCES

[1] K. Bereta and M. Koubarakis. Ontop of geospatial databases. In *ISWC*, 2016.

[2] K. Bereta and M. Koubarakis. Creating virtual semantic graphs ontop of big data from space. In *BiDS*, 2017.

[3] K. Bereta, P. Smeros, and M. Koubarakis. Representation and querying of valid time of triples in linked geospatial data. In *ESWC*, 2013.

[4] G. Garbis, K. Kyzirakos, and M. Koubarakis. Geographica: A benchmark for geospatial RDF stores. In *ISWC*, 2013.

[5] K. Kyzirakos, M. Karpathiotakis, and M. Koubarakis. Strabon: A semantic geospatial DBMS. In *ISWC*, 2012.

[6] K. Kyzirakos, D. Savva, I. Vlachopoulos, A. Vasileiou, N. Karalis, M. Koubarakis, and S. Manegold. GeoTriples: Transforming Geospatial Data into RDF Graphs Using R2RML and RML Mappings. *Journal of Web Semantics*, 2018.

[7] C. Nikolaou, K. Dogani, K. Bereta, G. Garbis, M. Karpathiotakis, K. Kyzirakos, and M. Koubarakis. Sextant: Visualizing time-evolving linked geospatial data. *Journal of Web Semantics*, 35 (1), 2015.

[8] G. Papadakis, K. Bereta, T. Palpanas, and M. Koubarakis. Multi-core meta-blocking for big linked data. In *SEMANTICS*, 2017.

[9] M. Perry and J. Herring. GeoSPARQL - a geographic query language for RDF data. OGC Implementation Standard, 2012.

[10] P. Smeros and M. Koubarakis. Discovering spatial and temporal links among RDF data. In *LDOW*, 2016.