GeoTriples: a Tool for Publishing Geospatial Data as RDF Graphs Using R2RML Mappings[*]

Kostis Kyzirakos[1], Ioannis Vlachopoulos[2], Dimitrianos Savva[2], Stefan Manegold[1], and Manolis Koubarakis[2]

[1] Centrum Wiskunde & Informatica, Amsterdam, The Netherlands
{firstname.lastname}@cwi.nl
[2] Department of Informatics and Telecommunications
National and Kapodistrian University of Athens, Greece
{johnvl,dimis,koubarak}@di.uoa.gr

Abstract. A plethora of Earth Observation data that is becoming available at no charge in Europe and the US recently reflects the strong push for more open Earth Observation data. Linked data is a paradigm which studies how one can make data available on the Web, and interconnect it with other data with the aim of making the value of the resulting "Web of data" greater than the sum of its parts. Open Earth Observation data that are currently made available by space agencies such as ESA and NASA are not following the linked data paradigm. Therefore, Earth Observation data and other kinds of geospatial data that are necessary for a user to satisfy her information needs can only be found in different data silos, where each silo may contain only part of the needed data. Publishing the content of these silos as RDF graphs, enables the development of data analytics applications with great environmental and financial value. In this paper we present the tool GeoTriples that allows for the transformation of Earth Observation data and geospatial data into RDF graphs. GeoTriples goes beyond the state of the art by extending the R2RML mapping language to be able to deal with the specificities of geospatial data. GeoTriples is a semi-automated tool that allows the publication of geospatial information into an RDF graph using the state of the art vocabularies like GeoSPARQL and stSPARQL, but at the same time it is not tightly coupled to a specific vocabulary.

Keywords: Linked Geospatial Data, data publishing, GeoSPARQL, stSPARQL

1 Introduction

Lots of Earth Observation (EO) data has become available at no charge in Europe and the US recently and there is a strong push for more open EO data. Linked data is a paradigm which studies how one can make data available on the Web, and interconnect it with other data with the aim of making the value of the resulting "Web of data" greater than the sum of its parts. In the last few years,

[*]This work has been funded in part by the FP7 project LEO (611141).

linked geospatial data has received attention as researchers and practitioners have started tapping the wealth of geospatial information available on the Web. As a result, the linked open data (LOD) cloud has been rapidly populated with geospatial data. For example, Great Britain's national mapping agency, Ordnance Survey, has been the first national mapping agency that has made various kinds of geospatial data from Great Britain available as open linked data.

Recently, the geospatial semantic web has also started to be populated with EO products (e.g., CORINE Land Cover and Urban Atlas published by project TELEIOS[3]). In general, open EO data that are currently made available by space agencies such as ESA and NASA are not following the linked data paradigm. Therefore, EO data and other kinds of geospatial data that are necessary for a user to satisfy her information needs can only be found in different data silos, where each silo may contain only part of the needed data. Publishing the content of these silos as RDF graphs, enables the development of data analytics applications with great environmental and financial value. With the recent emphasis on open government data in many countries, the development of useful Web applications utilizing EO data and geospatial data in general is just a few SPARQL queries away.

Geospatial data in general and EO data in particular, can come in vector or raster form and are usually accompanied by metadata. Vector data, available in formats such as ESRI shapefiles, KML, and GeoJSON documents, can be accessed either directly or via Web Services such as the OGC Web Feature Service or the query language of a geospatial DBMS. Raster data, available in formats such as GeoTIFF, Network Common Data Form (netCDF), Hierarchical Data Format (HDF), can be accessed either directly or via Web Services such as the OGC Web Coverage Processing Service (WCS) or the query language of an array DBMS, e.g., the array-query language SciQL[4]. Metadata about EO data are encoded in various formats ranging from custom XML schemas to domain specific standards like the OGC GML Application schema for EO products and the OGC Metadata Profile of Observations and Measurements.

Automating the process of publishing linked geospatial data has not been addressed yet. For example, in the wildfire monitoring and management application that we developed in TELEIOS, custom Python scripts were used for publishing all necessary data as linked data. For this reason, we designed and implemented the tool GeoTriples in the context of the EU FP7 project LEO[5]. In this paper we present the tool GeoTriples that allows for the transformation of EO data and geospatial data in various formats, such as data stored in spatially-enabled relational databases and raw files, into RDF graphs. GeoTriples goes beyond the state of the art by extending the R2RML mapping language to be able to deal with the specificities of geospatial data. GeoTriples is a semi-automated tool that allows for the publication of geospatial information into an RDF graph using the state of the art vocabularies like GeoSPARQL [9], but at the same

[3]`http://www.earthobservatory.eu/`

[4]`http://www.sciql.org/`

[5]`http://www.linkedeodata.eu/`

time it is not tightly coupled to a specific vocabulary. The publishing process comprises three steps. First, GeoTriples generates automatically R2RML mappings for publishing data that reside in spatially-enabled databases and raw files (e.g., ESRI shapefiles). Afterwards, the user may edit these mappings according to her needs (e.g., utilize a different vocabulary) and finally GeoTriples processes these mappings for producing an RDF graph.

The organization of the paper is as follows. Section 2 discusses related work. In Section 3 we present the architecture of GeoTriples and our extensions to the R2RML language for the geospatial domain. We discuss how GeoTriples generates automatically R2RML mappings and how they are subsequently proccessed for transforming a geospatial data source into an RDF graph. In Section 4 we present an example where we demonstrate how GeoTriples is currently being used for realizing the precision farming application that is being developed in LEO. Finally, in Section 5 we concludes our work.

2 Related Work

In this section we will present related work on methodologies and tools targeting the transformation of existing data sources into RDF graphs. Much work has been done recently on mapping relational data into RDF graphs. An extensive discussion on mapping relational data into RDF can be found in [8]. In this section we will present in detail two prevailing approaches for mapping relational data into RDF: the direct mapping approach and the R2RML mapping language. Then we discuss some relevant tools that implement these approaches and we finish our discussion with tools that convert geospatial information into RDF graphs.

Direct Mapping of Relational Data to RDF. A straightforward mechanism for mapping relational data to the RDF data model is the *Direct Mapping of Relational Data to RDF*[6] approach that became a W3C recommendation in 2012. According to this approach, relational tables are mapped to classes defined by an RDF vocabulary, while the attributes of each table are mapped to RDF properties that represent the relation between subject and object resources. Identifiers, class names, properties, and instances are generated automatically following the respective labels of the input data. For example, given the table `Address`, the class `<Address>` will be generated, and all tuples will be instances of this class. According to this approach, the generation of RDF data is dictated by the schema of the relational database. This mechanism was initially defined in [2], and [11] is an implementation of such a mechanism.

The Mapping Language R2RML. A language for expressing customized mappings from relational databases to RDF graphs is the *R2RML mapping language*[7] that became W3C recommendation in 2012. R2RML mappings provide the user with the ability to transform existing relational data into the RDF data model, following a structure and a target vocabulary that is chosen by the user.

[6] `http://www.w3.org/TR/2012/REC-rdb-direct-mapping-20120927/`

[7] `http://www.w3.org/TR/r2rml/`

R2RML mappings refer to logical tables to retrieve data from an input database. A *logical table* can be a relational table that is explicitly stored in the database, an SQL view or a valid SQL select query. A triples map is defined for each logical table that will be exported into RDF. A *triples map* is a rule that defines how each tuple of the logical table will be mapped to a set of RDF triples. A triples map consists of a subject map and one or more predicate-object maps. A *subject map* is a rule that defines how to generate the URI that will be the subject of each generated RDF triple. Usually, the primary key of the relation is used for this purpose. A *predicate-object* map consists of predicate maps and object maps. A *predicate map* defines the RDF property to be used to relate the subject and the object of the generated triple. An *object map* defines how to generate the object of the triple, the value of which originates from the value of the attribute of the specified logical table.

R2RML processors are applications that take as input a relational database and R2RML mappings and produce RDF graphs. Some R2RML processors provide a virtual SPARQL endpoint that transparently translate SPARQL queries to SQL queries by taking into account the given R2RML mappings. Alternatively, an R2RML processor can choose to export all relational data as an RDF dump according to the given mappings and then offer SPARQL or a linked data interface over the produced data.

Let us now present some tools that are able to translate raw data sources into RDF following the direct mapping approach, R2RML or a custom mapping language. OpenLink Virtuoso [6], Morph[8] [10], Ultrawrap[9] and D2RQ platform [3, 5] are capable of processing R2RML mappings. They support most of the features of R2RML and can process R2RML mappings both for publishing input data as an RDF graph and for querying the input data by translating a SPARQL query into an SQL query. Triplify [1] is a popular tool that follows a light-weight approach for mapping HTTP-URI requests onto relational database queries, and translating the result into RDF statements.

However, little attention has been paid to the problem of publishing geospatial information in the Semantic Web. Most datasets that contain such information are either generated manually or by semi-automated processes.

LinkedGeoData[10] project focuses on publishing OpenStreetMap[11] data as linked data. Sparqlify[12], developed in the context of this project, is employed for this task, using a proprietary mapping language. It creates mappings of spatial datatypes into RDF but until now, it does not discuss on how to deal with non-relational data.

[8] `https://github.com/jpcik/morph`
[9] `http://capsenta.com/ultrawrap/`
[10] `http://linkedgeodata.org/`
[11] `http://www.openstreetmap.org/`
[12] `http://sparqlify.org/`

The tool Geometry2RDF[13] was the first tool that allows for the conversion of geospatial information that resides in a spatially-enabled relational database into an RDF graph. It takes as input data stored in a spatially enabled DBMS like Oracle Spatial or MySQL, and utilizes the libraries Jena and GeoTools to produce an RDF graph. Geometry2RDF follows the direct mapping approach, and allows the user to configure the properties that connect a URI to the serialization of a geometry and allows for the conversion of the coordinates to the desired coordinate reference system. Geometry2RDF follows the direct mapping approach that is not expressive enough to deal with the specificities of the geospatial domain. For example, since this work has preceded the proposal of GeoSPARQL and stSPARQL, it does not produce RDF graphs according to these vocabularies. For this purpose, the tool TripleGeo[14] was recently developed. TripleGeo is based on Geometry2RDF and allows the generation of RDF graphs that follow the GeoSPARQL vocabulary. However, this tool has the same shortcomings with Geometry2RDF since the mapping process of the input data into an RDF graph that follows the GeoSPARQL vocabulary is hard-coded in the implementation. Thus, significant effort is required for modifying such tools in order to support other vocabularies.

A different approach was followed by [4] where the authors present how R2RML can be combined with a spatially enabled relational database in order to transform geospatial information into RDF. For the manipulation of the geometric information prior to its transformation into RDF, the authors create several logical tables that are based on ad-hoc SQL queries that perform the appropriate pre-processing (e.g., requesting the serialization of a geometry according to the WKT standard). This approach demonstrates the power of utilizing a general-purpose mapping language like R2RML, which is in contrast to other approaches discussed earlier in this section. However, in this work, no automated method for publishing geospatial datasets into RDF is discussed, and dealing with different types of data sources was out of scope.

3 The Tool GeoTriples

In this section we will present in detail the tool GeoTriples that we developed for transforming geospatial data sources into RDF. GeoTriples[15] is an open-source tool that is distributed freely according to the Mozilla Public License v2.0. In this section we will present the architecture of GeoTriples and we will discuss our implementation choices. We will describe in detail how GeoTriples generates automatically R2RML mappings for publishing data that reside in spatially-enabled databases and raw files (e.g., ESRI shapefiles), and how it can process such mappings for producing an RDF graph that follows the GeoSPARQL or any other vocabulary.

[13] `http://mayor2.dia.fi.upm.es/oeg-upm/index.php/en/technologies/151-geometry2rdf`

[14] `https://github.com/GeoKnow/TripleGeo`

[15] `http://sourceforge.net/projects/geotriples/`

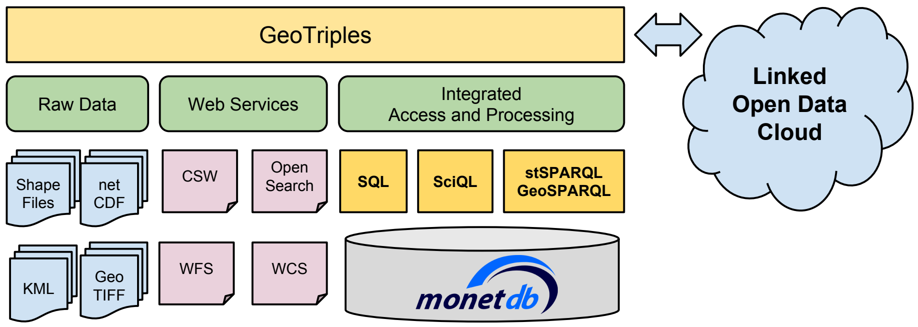

Fig. 1: The abstract architecture of the system GeoTriples

3.1 System Architecture

The abstract architecture of GeoTriples, from a data perspective is shown in Figure 1. GeoTriples takes as input data that are stored in a spatially-enabled database, data that reside in raw files (e.g., ESRI shapefiles), or the results that derive from processing the aforementioned data (e.g., the result of a SciQL query over raster or array data). Regarding the system architecture at a lower level, GeoTriples uses a connector for each type of input data in order to transparently access and process the input data. As Figure 2 depicts, GeoTriples comprises two main components: the mapping generator and the R2RML processor. The mapping generator takes as input a data source and creates automatically an R2RML mapping that transforms it into an RDF graph. The generated mapping is enriched with subject and predicate-object maps, in order to take into account the specifities of geospatial data and cater for all transformations that are needed to produce an RDF graph that is compliant with the GeoSPARQL vocabulary. To accomplish this task, we extend R2RML mappings to allow the representation of a transformation function over input data. Afterwards, the user may edit the generated R2RML mapping document to comply with her requirements (e.g., use a different vocabulary).

Implementation Choices. One of the main choices we had to make during the design of GeoTriples was to choose which RDB2RDF framework to extend. We chose to extend the D2RQ platform which seemed to be the most mature system for publishing relational data into RDF. It provides a mechanism for generating and processing R2RML mappings for a variety of relational databases that are accessible via JDBC, like MonetDB, PostgreSQL and Oracle. D2RQ provides preliminary support for translating SPARQL queries into SQL queries given some R2RML mappings. However, given the recent results of the system Morph [10], we will consider utilizing Morph as the R2RML processor of GeoTriples in the future.

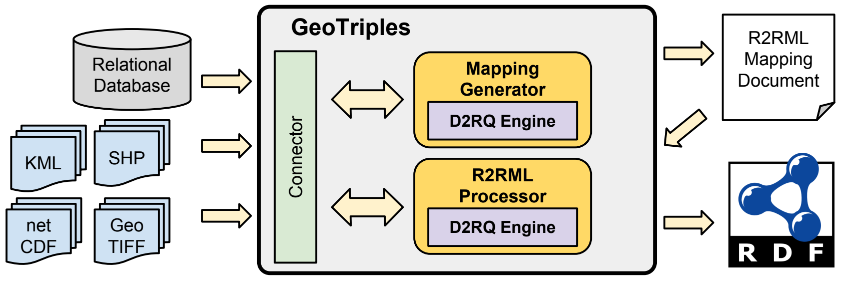

Fig. 2: The architecture of the system GeoTriples

3.2 Transforming Geospatial Data into RDF graphs using GeoTriples

In this section we will present the main functionality of GeoTriples. First we will show the extended R2RML mappings that are produced automatically by GeoTriples that take into account the spatial dimension of the input data, and then we will show how these mappings are processed subsequently by GeoTriples in order to generate an RDF graph.

Automatic Generation of R2RML Mappings. Much work has been done recently on extending RDF to represent and query geospatial information. The most mature results of this work are the data model stRDF and the query language stSPARQL [7] and the OGC standard GeoSPARQL. GeoSPARQL is an OGC standard for the representation and querying of geospatial linked data. GeoSPARQL defines much of what is required for such a query language by providing vocabulary (classes, properties, and functions) that can be used in RDF graphs and SPARQL queries to represent and query geospatial data. The top level classes defined in GeoSPARQL are `geo:SpatialObject` that has as instances everything that can have a spatial representation and `geo:Feature` that represents all features and is the superclass of all classes of features that the users might want to define. To represent geometric objects, the class `geo:Geometry` is introduced by GeoSPARQL. Additional vocabulary is also defined by GeoSPARQL for asserting and querying information about geometries.

Given a spatially-enabled database or a raw file that contains geometric information, GeoTriples generates an R2RML mapping document. Let us take for example the Natura 2000 dataset of Germany that contains information about protected areas in Germany and is distributed in the form of an ESRI shapefile. More details on this dataset can be found in Section 4. Conceptually, each geometric object stored in the ESRI shapefile should be an instance of the class `geo:Geometry`, and all non-geometric attributes that characterize each geometry are thematic attributes of the corresponding feature. Following this modeling approach, we generate an instance of the class `geo:Feature` and an instance of the class `geo:Geometry` for each geometric object stored in the ESRI shapefile. Geometric and non-geometric attributes that appear at the ESRI shapefile are assigned accordingly to the appropriate instances. Part of the R2RML mapping that is generated automatically by GeoTriples to represent the features stored in the Natura 2000 ESRI shapefile is the following:

```
_:natura
    rr:logicalTable [ rr:tableName "‘natura‘"; ];
    rr:subjectMap [
            rr:class geo:Feature;
            rr:template "http://data.example.com/natura/Feature/id/{‘gid‘}"; ];

    rr:predicateObjectMap [
            rr:predicate nato:has_SITECODE;
            rr:objectMap [ rr:datatype xsd:string;
                       rr:column "‘SITECODE‘"; ];
    ];
    rr:predicateObjectMap [
            rr:predicate geo:hasGeometry ;
            rr:objectMap [
                rr:parentTriplesMap _:natura_geometry;
                rr:joinCondition [
                        rr:child  "gid";
                        rr:parent "gid"; ]; ]; ].
```

This mapping document contains a triples map that describes how instances of the class `geo:Feature` are constructed. All thematic information that is stored in the ESRI shapefile is assigned to the generated feature and a link between the feature and its geometry is also generated. However, the notion of a primary key is not defined for ESRI shapefiles. Each ESRI shapefile though is accompanied by a relational table in dBASE format that stores infromation about the geometries, so we define as a unique identifier for each geometric object the respective row identifier in the dBASE table. In the example above we represent this information as an extra attribute with name ’gid’.

Part of the R2RML mapping that is generated automatically by GeoTriples to represent the geometries stored in the Natura 2000 ESRI shapefile is the following:

```
_:naturaGeometry
    rr:logicalTable [ rr:tableName "‘natura‘"; ];
    rr:subjectMap [
        rr:class geo:Geometry;
        rr:template "http://data.example.com/natura/Geometry/id/{‘gid‘}"; ];

    rr:predicateObjectMap [
        rr:predicate geo:dimension;
        rr:objectMap [
            rrx:transformation [
                rrx:function geof:dimension;
                 rrx:argumentMap (
                    [rr:column "‘Geom‘"] ); ]  ]; ].
```

This mapping document contains a triples map that describes how instances of the class `geo:Geometry` are constructed. All geometric information that is stored in the ESRI shapefile is assigned to the generated geometry. In addition, object maps define that geospatial functions like `geof:dimension` are applied to the serialization of each geometric object in order to produce the values that will appear at the object part of the corresponding triple.

Notice that in the above example we extended the definition of an object map by allowing it to be the RDF term obtained by applying a transformation on the source data. Each transformation defines the SPARQL built-in function or the SPARQL extension function to be invoked, using as an argument the sequence of RDF terms that are produced by the respective term maps.

Processing of R2RML mappings for producing RDF graphs. R2RML mappings usually consist of two triples maps; one for handling thematic information and one for geospatial information. The triples map that handles thematic information defines a logical table that contains the thematic attributes and a unique identifier for the generated instances. The latter could be either the primary key of the table in case the input data is a relational database or a row number in the dBASE table of an ESRI shapefile. Combined with a URI template, the unique identifier is used to produce the URI that serve as subjects of the produced triples. The predicate object maps are also processed accordingly in order to define RDF properties the value of which originate from the value of the column of the thematic logical table.

The triples map that handles the geospatial information of the input data source, defines a logical table with unique identifier similar to the thematic one. However, according to the type of the data source, the definition of this logical table may vary. For instance, in the case of a relational database, it is defined by providing an appropriate SQL query that uses spatial functions provided by spatially enabled relational backend (e.g. utilize the function `ST_Dimension`). If the input source is an ESRI shapefile, then GeoTriples will perform such transformations on the fly by evaluating the SPARQL extension function using the JTS Topology Suite.

4 Linked geospatial data in the precision farming application of LEO

Let us now give an example that demonstrates how GeoTriples is being used in LEO for the development of a precision farming application. The aim is to develop a precision farming application that combines traditional geospatial data with linked geospatial data for enhancing the quality of precision farming activities. Precision farming is a concept based on observing the heterogeneity within an agricultural field and providing information about cultivating each part of the field according to the characteristics of the soil. Precision farming aims to solve numerous problems for farmers such as the minimization of the environmental pollution by fertilizers. For dealing with this issue, the farmers have to comply with many legal and technical guidelines that require the combination of information that resides in diverse information sources. In this section we present how linked geospatial data can form the knowledge base for providing solutions for this problem. We published the following datasets as RDF graphs using GeoTriples in order to use them in the precision farming application.

Talking Fields[16] is a precision farming project aiming to increase the efficiency of agricultural production via precision farming by means of geo-information services integrating space and ground-based assets. Currently, TalkingFields produces products for improved soil probing using satellite-based zone maps, and provide services for monitoring crop development through provision of biomass maps and yield estimates.

Natura 2000[17] is an ecological network designated under the Birds Directive and the Habitats Directive which form the cornerstone of Europe's nature conservation policy. National authorities submit a standard data form that describes each site and its ecology in order to be characterized as a Natura site.

OpenStreetMaps (OSM)[18] is a collaborative project for publishing free maps of the world. OSM maintains a community-driven global editable map that gathers map data in a crowdsourcing fashion. LinkedGeoData (LGD)[19], is a project focused on publishing OSM data as linked data.

LGD data is the only dataset that is already published as linked geospatial data. For the rest datasets, we design an appropriate ontology that follows the GeoSPARQL vocabulary. Afterwards, we use GeoTriples in order to produce the R2RML mappings that dictate the process of generating the desired RDF output. Finally, we use the R2RML processor of GeoTriples for translating the input data into RDF graphs. The produced data are then stored into an appropriate geospatial RDF store. We chose to utilize our own geospatial RDF store Strabon[20] [7].

Let us now see an example query that can provide a precision farming application with information about the parts of agricultural fields that are either close to a river or are located within a protected area. This information allows the precision farming application to take into account legal restrictions regarding distance requirements when preparing the prescription maps that the farmers will utilize afterwards.

[16]http://www.talkingfields.de/

[17]http://ec.europa.eu/environment/nature/natura2000/access_data/index_en.htm

[18]http://www.openstreetmap.org

[19]http://linkedgeodata.org

[20]http://www.strabon.di.uoa.gr/

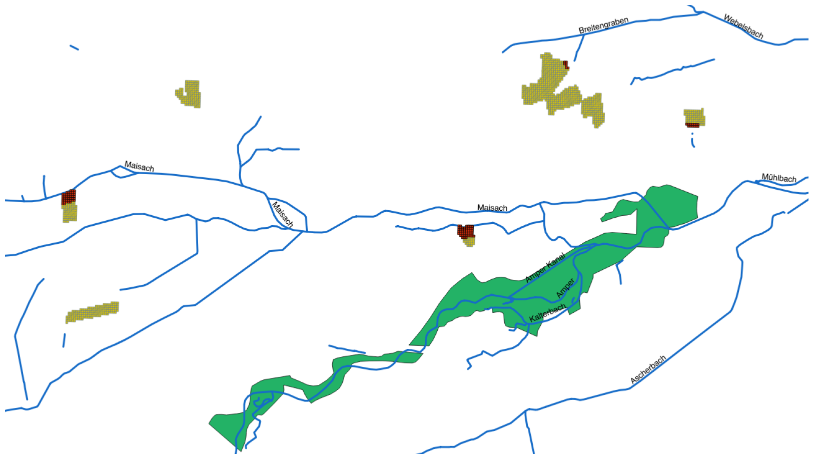

Fig. 3: Visualization of query results

Example. Select all parts of agricultural fields that are close to a river or are located inside a protected area.

```
SELECT  distinct ?field_name ?river_name ?cell_wkt
WHERE {{?river     rdf:type osmo:River ;
                   osmo:hasName ?river_name ;
                   geo:hasGeometry ?river_geo .
        ?river_geo geo:asWKT ?river_wkt .
        ?field     rdf:type tf:Field ;
                   tfo:hasFieldName ?field_name ;
                   tf:hasRasterCell ?cell .
        ?cell      geo:hasGeometry ?cell_geo ;
        ?cell_geo  geo:asWKT ?cell_wkt .
        FILTER(geof:distance(?river_wkt ,
                             ?cell_wkt, uom:meter) < 100) .
       } UNION {
        ?field     rdf:type tf:Field ;
                   tfo:hasFieldName ?field_name ;
                   tf:hasRasterCell ?cell .
        ?cell      geo:hasGeometry ?cell_geo ;
        ?cell_geo  geo:asWKT ?cell_wkt .
        ?nat       rdf:type nat:NaturaArea ;
                   geo:hasGeometry ?nat_geo .
        ?nat_geo   geo:asWKT ?nat_wkt .
        FILTER(geof:contains(?nat_wkt, ?cell_wkt)}}
```

This above query uses the `geo:distance` function whose arguments are the serializations of two geometries and a URI that identifies a unit of measurement. The result of this metric function is the minimum distance between these two geometric objects. This filter identifies the parts of a field that are close to a river, so the precision farming application should adjust the fertilizer/pesticide usage accordingly. The second part of the query identifies the parts of a field that resides inside a protected, thus the precision farming application should adjust the fertilizer/pesticide usage accordingly. Figure 3 depicts protected areas, rivers, agricultural fields and the parts of the agricultural fields that are close to a river.

5 Conclusions

In this paper we presented the tool GeoTriples that transforms geospatial data that reside in a spatially-enabled DBMS or raw files into RDF graphs, by extending the R2RML mapping language. This allows GeoTriples to transform geospatial data into RDF graphs without being tightly coupled to a specific vocabulary. We plan to extend GeoTriples to support numerous types of vector data formats such as KML, as well as raster formats like GeoTIFF and netCDF.

References

1. S. Auer, S. Dietzold, J. Lehmann, S. Hellmann, and D. Aumueller. Triplify: Light-weight Linked Data Publication from Relational Databases. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 621–630, New York, NY, USA, 2009. ACM.
2. T. Berners-Lee. Relational Databases on the Semantic Web. http://www.w3.org/DesignIssues/RDB-RDF.html.
3. C. Bizer and A. Seaborne. D2RQ-treating non-RDF databases as virtual RDF graphs. In *Proceedings of the 3rd international semantic web conference (ISWC2004)*, volume 2004, 2004.
4. K. Chentout and A. A. Vaisman. Adding Spatial Support to R2RML Mappings. In *OTM Workshops*, volume 8186 of *Lecture Notes in Computer Science*. Springer, 2013.
5. R. Cyganiak, C. Bizer, J. Garbers, O. Maresch, and C. Becker. The D2RQ Platform. http://d2rq.org/.
6. O. Erling and I. Mikhailov. RDF Support in the Virtuoso DBMS. In S. Auer, C. Bizer, C. Müller, and A. V. Zhdanova, editors, *CSSW*, volume 113 of *LNI*, pages 59–68. GI, 2007.
7. K. Kyzirakos, M. Karpathiotakis, and M. Koubarakis. Strabon: A semantic geospatial dbms. In P. Cudré-Mauroux, J. Heflin, E. Sirin, T. Tudorache, J. Euzenat, M. Hauswirth, J. X. Parreira, J. Hendler, G. Schreiber, A. Bernstein, and E. Blomqvist, editors, *International Semantic Web Conference (1)*, volume 7649 of *Lecture Notes in Computer Science*, pages 295–311. Springer, 2012.
8. K. Kyzirakos, S. Manegold, I. Vlachopoulos, D. Savva, M. Koubarakis, C. Nikolaou, S. Giannakopoulou, P. Smeros, and B. Valentin. Data models and languages for mapping EO data to RDF. Deliverable D2.1, LEO - FP7 ICT Project.
9. OGC. GeoSPARQL - A geographic query language for RDF data, November 2012.
10. F. Priyatna, Ó. Corcho, and J. Sequeda. Formalisation and experiences of R2RML-based SPARQL to SQL query translation using Morph. In C.-W. Chung, A. Z. Broder, K. Shim, and T. Suel, editors, *WWW*, pages 479–490. ACM, 2014.
11. D. Steer. SquirrelRDF, 2006. `http://jena.sourceforge.net/SquirrelRDF/`.