TELEIOS

FP7-257662

Deliverable

D2.1

A data model and query language for an extension of RDF with time and space

Manolis Koubarakis, Kostis Kyzirakos, Babis Nikolaou, Michael Sioutis, Stavros Vassos,

and

Consortium Members

November 8, 2011

Status: Final
Scheduled Delivery Date: 28 February, 2011

Executive Summary

This deliverable presents the semantics-based data models and language that we will utilize in TELEIOS.

First, we discuss the challenges for semantic data modeling in TELEIOS by concentrating on the two use cases “A Virtual Observatory for TerraSAR-X data" and “Real-time fire monitoring based on continuous acquisitions of EO images and geospatial data".

For each use case we discuss what challenges arise in semantic modeling of the relevant data sets using W3C standards such as RDF, RDFS, OWL 2 and SPARQL. We also discuss interesting queries that cannot be answered using internal data sets only, but become possible when these data sets are combined with geographic information publicly available on the Web.

After an extensive review of the related work, we present a new version of the data model sRDF and the query language sSPARQL proposed recently by our partner NKUA that extend the W3C standards RDF and SPARQL for representing and querying spatial data in the Semantic Web. In the new version of sRDF and sSPARQL, called sRDF++ and sSPARQL++, we opt for a more practical solution to the problem of representing geospatial data and use the OGC standards Well-known Text and GML instead of linear constraints. Our approach resembles OGC standards regarding how to use SQL to query geographic features as they have been adopted by commercial RDBMSs, e.g., PostGIS or Oracle Spatial. Our approach is also closely related to the language GeoSPARQL, a new proposal for a standard currently discussed by OGC.

Finally, we go a step further and show how to extend sRDF and sSPARQL to allow the representation and querying of *qualitative* spatial relations as they have traditionally been studied by the qualitative spatial reasoning community. In the proposed extension of sRDF, called $sRDF^i$, we use a new kind of literals to represent spatial regions about which the known information is *incomplete* or *indefinite*, e.g., region A is inside a known rectangle R but we do not know its exact geographic location, or region A is north of region B and it overlaps region C etc.

We expect the data models and languages defined in this deliverable to have impact beyond TELEIOS: in the representation and querying of linked geospatial data sets on the Web, and in the development of a new generation of spatial reasoners for the Web.

Document Information

Contract Number	FP7-257662	**Acronym**	TELEIOS
Full title	TELEIOS: Virtual Observatory Infrastructure for Earth Observation Data		
Project URL	`http://www.earthobservatory.eu/`		
EU Project Officer	Francesco Barbato		

Deliverable	**Number**	D2.1	**Name**	A data model and query language for an extension of RDF with time and space		
Task	**Number**	T2.1	**Name**	Extending RDF and SPARQL with time and space		
Work package	**Number**			WP2		
Date of delivery	**Contract**	M6 (Sept. 2011)		**Actual**	28 February 2011	
Status	Draft ☐ Final ☑					
Nature	Prototype ☐ Report ☑					
Distribution Type	Public ☑ Restricted ☐					
Responsible Partner	NKUA					
QA Parner	CWI					
Contact Person	Manolis Koubarakis					
	Email	koubarak@di.uoa.gr	**Phone**	+30 210 727 5213	**Fax**	+30 210 727 5214

Project Information

This document is part of a research project funded by the IST Programme of the Commission of the European Communities as project number FP7-257662. The Beneficiaries in this project are:

Partner	Acronym	Contact
National and Kapodistrian University of Athens Department of Informatics and Telecommunications	NKUA	Prof. Manolis Koubarakis National and Kapodistrian University of Athens Department of Informatics and Telecommunications Panepistimiopolis, Ilissia, GR-15784 Athens, Greece Email: (koubarak@di.uoa.gr) Tel: +30 210 7275213, Fax: +30 210 7275214
Fraunhofer Institute for Computer Graphic Research	Fraunhofer	MSc. Thorsten Reitz Fraunhofer Institute for Computer Graphic Research Fraunhofer Strasse 5, D-64283 Darmstadt, Germany Email: (thorsten.reitz@igd.fraunhofer.de) Tel: +49 6151 155 416, Fax: +49 6151 155 444
German Aerospace Center The Remote Sensing Technology Institute Photogrammetry and Image Analysis Department Image Analysis Team	DLR	Prof. Mihai Datcu German Aerospace Center The Remote Sensing Technology Institute Oberpfaffenhofen, D-82234 Wessling, Germany Email: (mihai.datcu@dlr.de) Tel: +49 8153 28 1388, Fax: +49 8153 28 1444
Stichting Centrum voor Wiskunde en Informatica Database Architecture Group	CWI	Prof. Martin Kersten Stichting Centrum voor Wiskunde en Informatica P.O. Box 94097, NL-1090 GB Amsterdam, Netherlands Email: (martin.kersten@cwi.nl) Tel: +31 20 5924066, Fax: +31 20 5924199
National Observatory of Athens Institute for Space Applications and Remote Sensing	NOA	Dr. Charis Kontoes National Observatory of Athens Institute for Space Applications and Remote Sensing Vas. Pavlou and I. Metaxa, GR 152 36 Athens, Greece Email: (kontoes@space.noa.gr) Tel: +30 210 8109186, Fax: +30 210 6138343
Advanced Computer Systems A.C.S S.p.A	ACS	Mr. Ugo Di Giammatteo Advanced Computer Systems A.C.S S.p.A Via Della Bufalotta 378, RM-00139 Rome, Italy Email: (udig@acsys.it) Tel: +39 06 87090944, Fax: +39 06 87201502

Contents

List of Figures

List of Tables

1. Introduction

Work package WP2 of TELEIOS lays the foundations of data models and query languages that will be used in the project. The technical work that is carried out is divided into two tasks:

- Task 2.1: Extending RDF and SPARQL with time and space (with NKUA as lead partner)
- Task 2.2: Array data models and query languages for earth observation image data (with CWI as lead partner)

The research carried out in the two tasks is complementary. Task 2.1 concentrates on high-level semantics-based data models for querying use case data sets that can profitably be represented in such a way (e.g., satellite image annotations, geospatial data etc.). Task 2.2 develops the query language SciQL which is a combination of multi-dimensional arrays with SQL so that we have a query language that is useful to scientific database users in general, and to scientists and engineers working with satellite images in particular. For details of the work in Task 2.2 see Deliverable D2.2 “An array data model and query language for EO image databases".

This deliverable concentrates on Task 2.1 and makes the following contributions:

- We discuss the challenges for semantic data modeling in TELEIOS by concentrating on the two use cases “A Virtual Observatory for TerraSAR-X data" (called “the DLR use case" in the rest of this deliverable), and “Real-time fire monitoring based on continuous acquisitions of EO images and geospatial data" (called “the NOA use case"). The data sets that have been made available to TELEIOS partners by DLR and NOA include standard products that have been produced after processing raw images, metadata files describing raw images and other products and related geospatial data sets. We give details of these data sets and discuss what challenges they pose to semantic data modeling using W3C standards such as RDF, RDFS, OWL 2 and SPARQL. We also discuss interesting queries that cannot be answered using internal data sets only, but become possible when these data sets are combined with geographic information publicly available on the Web such as gazetteers, linked open GIS data [ALH09] etc. W3C standards such as RDF, RDFS, OWL 2 and SPARQL that are championed in TELEIOS enable us to achieve this integration by tapping the wealth of information that is currently available as linked data.[1]
- We present a new version of the data model sRDF and the query language sSPARQL proposed recently by our partner NKUA [KK10] in the context of the FP7 ICT project SemsorGrid4Env[2]. sRDF and sSPARQL are extensions of the W3C standards RDF and SPARQL for representing and querying spatial data in the Semantic Web. sRDF and sSPARQL follow the tradition of constraint databases [KKR90] and use linear constraints for the representation of spatial data. The constraint-based approach of [KK10] has led to a theoretically elegant framework where the semantics of sSPARQL query evaluation can be easily given as an extension of the standard SPARQL semantics of [PAG06].

  We note that [KK10] defines an extension of RDF and SPARQL that also enables the representation of the *valid time* of a triple as in [GHV07]. For this reason, the most general model of [KK10] is called stRDF and its corresponding query language stSPARQL. In this deliverable we do not consider issues related to the valid time of a triple since the issue has not arisen in the two use cases of the project. However, we do allow the presence of time in the object part of a triple (this is called *user-defined time* in the terminology of temporal databases [SA85]) since this is useful for both of our use cases. But user-defined time can be

[1] http://linkeddata.org/
[2] http://www.semsorgrid4env.eu/

expressed in RDF itself so sRDF is the data model of [KK10] that is of interest to us in this deliverable.

In the new version of sRDF and sSPARQL, called *sRDF++* and *sSPARQL++*, we opt for a more practical solution to the problem of representing geospatial data and use the *OGC standard Well-known Text (WKT)* instead of constraints. Our approach now more closely resembles OGC proposals regarding how to use SQL to query geographic features [OGC10d] as they have been adopted by commercial RDBMs e.g., PostGIS[3] or Oracle Spatial[4]. In this way, we expect that sRDF++ and sSPARQL++ will be adopted more easily by applications such as the ones that are represented in TELEIOS where there is a long tradition of using geospatial representations that are closely related to OGC standards such as WKT.

- We show how to extend sRDF and sSPARQL to allow the representation and querying of *qualitative spatial relations* as they have traditionally been studied by the qualitative spatial reasoning community [RN07]. In the proposed extension of sRDF (called $sRDF^i$), we use a new kind of literals to represent spatial regions about which the known information is *incomplete* or *indefinite* (e.g., region A is inside a known rectangle R but we do not know its exact geographic location, or region A is north of region B and it overlaps region C etc.). In the general case, such incomplete or indefinite information is expressed in terms of *possibly disjunctive qualitative spatial constraints* representing size (e.g., “large", “small"), direction (e.g., “right of", “above"), distance (e.g., “far", “near"), shape (e.g., “convex") or topology (e.g., “overlaps", “contains") of spatial objects [RN07]. sRDF^i is essentially an extension of sRDF in the spirit of *indefinite constraint databases* developed by the NKUA group in the past [Kou94c, Kou94b, Kou97].

  In the data sets of the two TELEIOS use cases, qualitative spatial constraints can be exploited for the representation of geospatial information for many reasons: (i) they are closer to the way human users perceive commonsense geospatial information, (ii) some of the products derived from satellite images contain indefinite geospatial information (e.g., due to the impreciseness of available image processing techniques, spatial granularity etc.), (iii) external data sources such as geospatial information available on the Web are relevant to the two use cases but the geospatial information retrieved from these data sources might be available only in qualitative terms etc.

The contributions of this deliverable go beyond TELEIOS. Geospatial data are made available on the Web continously (e.g., see the recent version of the ontology YAGO2 [HSBW10] that describes time and location for its entities). Thus the importance of data models and query languages such as sRDF and sSPARQL will be greater in the very near future. Current work in OGC for developing the language GeoSPARQL [OGC10b] points to the same direction.

Description logic researchers have also been studying the problem of reasoning with spatial knowledge using description logics for some time [HM02, LM07], and interesting extensions of well-known spatial reasoners such as Pellet and Racer have been presented recently [SS09, Wes02]. The proposed model $sRDF^i$ can also be seen as a contribution towards this line of research (but notice that we consider only ABox-like information).

In Task 2.1, we also have two other activities:

- To define the semantics of query evaluation for stSPARQL++ by following the algebraic approach we used for stSPARQL [KK10].
- To study the computational complexity of query answering in sSPARQL, sSPARQL++ and sSPARQL^i. The first question here is to see whether the data complexity results for SPARQL presented in [PAG09] carry over to sSPARQL and sSPARQL++. Then, the same question will be asked for sSPARQL^i. We will then identify cases where query evaluation can be

[3] http://postgis.refractions.net/
[4] http://www.oracle.com/technetwork/database/options/spatial/index.html

efficiently implemented, so that these cases are developed in the prototype system of work package “Scalable storage and query processing for EO image metadata" (WP4).

These two activities of Task 2.1 research will be reported in the forthcoming deliverable D2.3 “Theoretical results on query processing for RDF/SPARQL with time and space”.

The rest of this deliverable is organized as follows. The next chapter discusses the semantic data modeling challenges in TELEIOS. In Chapter 3 we survey related work in the areas most relevant to this deliverable. In Chapter 4 we present a new version of sRDF and sSPARQL using OGC standards. In Chapter 5 we discuss an extension of sRDF and sSPARQL for reasoning with qualitative spatial information.

2. Semantic Data Modeling Challenges in TELEIOS

In this chapter we discuss the challenges for semantic data modeling in TELEIOS by looking into the specific details of each of the two use cases of the project. First, we focus on the DLR use case that deals with building a Virtual Observatory for TerraSAR-X satellite data. Then we examine in detail the NOA use case that deals with real-time fire monitoring based on continuous acquisitions of earth observation images and geospatial data. More details about the use cases can be found in the Deliverable 6.1 "Requirements specification for the VO for TerraSAR-X data and applications" for the DLR use case, and Deliverable 7.1 "Requirements specification for the real-time fire monitoring application" for the NOA use case.

In order to illustrate the use of semantic web technologies in each of the use cases, we shall give some simple examples using sRDF++ and sSPARQL++. Although a detailed introduction to these languages will be done in Chapter 4, the examples given here are an informal introduction to sRDF++ and sSPARQL++ and their use in the two TELEIOS use cases.

2.1 Semantic challenges in the DLR use case

In this section we focus on the DLR use case of TELEIOS. First, we give a brief overview of this use case and the data sets provided by DLR. Then we review the current system that is used for querying and retrieving the data, and discuss the challenges that arise in using semantic web technologies for the same purpose. We sketch how the current functionality can be achieved using these technologies and give some high-level examples. Finally, we discuss some interesting queries that go beyond the current functionality. These queries involve the effective combination of different data products that are available inside DLR, as well as the use of geographic information publicly available on the web as linked data.[1]

2.1.1 Description of the DLR use case

The DLR use case refers to the construction of a virtual observatory for TerraSAR-X satellite data. TerraSAR-X is a German Earth observation satellite developed by the German Aerospace Center (DLR) and EADS Astrium (an aerospace subsidiary of EADS, the European Aeronautic Defense and Space Company). The primary payload of TerraSAR-X is an X-band radar sensor with a range of different modes of operation, allowing it to record images with different swath widths, resolutions and polarizations. TerraSAR-X thus offers space-based observation capabilities that were previously unavailable. Figure 2.1 shows an artist's view of the satellite.

The objective of the mission is to provide value-added synthetic aperture radar (SAR) image data in the X-band, for research and development purposes as well as scientific and commercial applications. However, the current state-of-the-art methods for SAR information extraction and understanding are not yet sufficiently developed to cope with the dual curse of complexity: high resolution and huge data volumes. The purpose of this use case is to demonstrate the advantages of the TELEIOS technologies by achieving the following objectives:

1. Go beyond the existing EOWEB portal that simply offers a hierarchical organization of TerraSAR-X products together with a temporal and geographic selection menu. The goal is to develop a system that enables queries that capture the semantics of the content of the images and combine it with other data products that are available inside DLR as well as external data sources available on the web such as linked data.

[1] `http://linkeddata.org/`

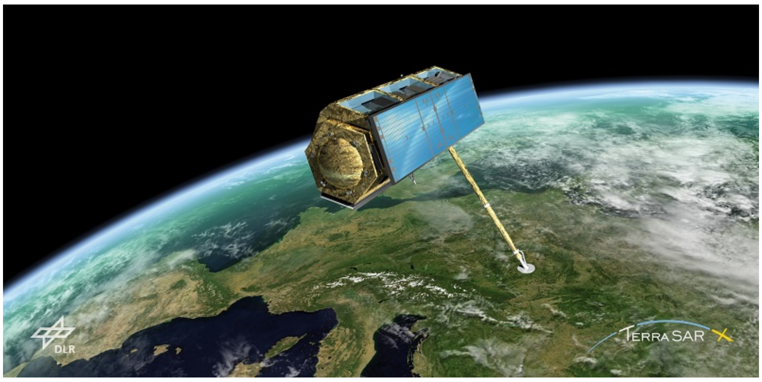


Figure 2.1: TerraSAR-X satellite artist view

2. Build demo rapid mapping applications that make use of the various technologies developed in the project.

2.1.2 Description of the DLR data sets

We now present four data sets that are used so far in the DLR use case, namely the TerraSAR-X products, the TerraSAR-X header files, a classification scheme for annotating image patches, and the feature vectors of image patches. The first two refer to the actual SAR images, while the other two refer to image patches, that is, rectangular parts of the image that have been processed separately.

TerraSAR-X products

The TerraSAR-X instrument is a side-looking X-band SAR based on active phased array antenna technology. The active antenna allows not only the conventional stripmap imaging mode but additionally the spotlight and ScanSAR modes. The instrument timing and pointing of the electronic antenna can be programmed allowing numerous combinations. From the many technical possibilities four imaging modes have been designed to support a variety of applications ranging from medium resolution polarimetric imaging to high resolution mapping. The following imaging modes are defined for the generation of basic products:

- stripmap mode (SM),
- high resolution spotlight mode (HS),
- spotlight mode (SL),
- ScanSAR mode (SC).

The image data can be ordered with selectable processing levels (e.g., prior to or after geometric correction. In our case, we will concentrate on geometrically rectified products in order to get image data that can be superimposed and compared.

Upon delivery, these products are packed in a delivery package which is supplemented by additional administrative information and is then either archived into a tar file or put onto a medium. The packages are named based on a naming scheme that uses constituents that are separated by underscores, of the following form:

```
TSX1_SAR_{type}_{variant}_{mode}_{pol}_{antenna}_{startTime}_{stopTime}
```

The meaning of each constituent is explained in Table 2.1.

Constituent	**Meaning**
`type`	the product variant
`variant`	the resolution variant
`mode`	the imaging mode
`pol`	the polarization mode
`antenna`	the antenna receive configuration
`startTime`	UTC start time
`stopTime`	UTC stop time

Table 2.1: Naming scheme for SAR products

Each package contains various types of information. It includes the full-resolution raster image in TIFF or GeoTIFF format, a quick-look preview of lower resolution, an XML file that holds georeference information, and an XML header file that carries the same name as the product.

For example, Figure 2.2 shows a quick-look preview of a TerraSAR-X satellite spotlight (SL) image taken over taken over Rosenheim, a city in southern Germany, that can be found in a product under the following name:[2]

```
TSX1_SAR__EEC_RE___SL_S_SRA_20080127T051746_20080127T051748
```

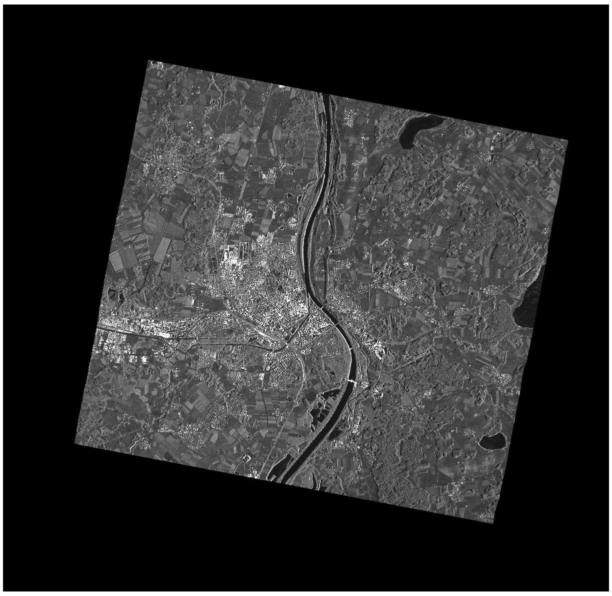


Figure 2.2: A quick-look preview of a TerraSAR-X satellite spotlight (SL) image

The corresponding SAR image is correctly projected onto a latitude/longitude grid on ground in a TIFF file of 9428 by 9142 pixels and 164 MB size. Note that high-resolution images of a different mode may be as large as few GBs.

[2]This is a publicly available TerraSAR-X image that can be downloaded from http://www.infoterra.de/free-sample-data)

TerraSAR-X header files

A TerraSAR-X header XML file contains hundreds of entries that describe the details of a given image including a long list of processing history parameters. One important type of information available is the location of the image on ground: the image center latitude, image center longitude, and the incidence angle of the radar pulses at the image center are given in degrees as floating point numbers.

Moreover, detailed information on the format of the image is contained in element `<imageDataInfo>`:

- `<pixelValueID>` specifies the physical unit of each pixel, e.g., `RADAR BRIGHTNESS`,
- `<imageDataFormat>` specifies the image format, e.g, `GEOTIFF`,
- `<number of layers>` specifies the number of channels the image comprises,
- `<imageDataDepth>` specifies the number of bits per pixel, e.g., `16`,
- `<imageStorageOrder>` specifies the ordering principle of rows and columns, e.g., `ROWBYROW`,
- `<rowContent>` specifies the geographical orientation of the rows, e.g, `EASTING`,
- `<columnContent>` specifies the geographical orientation of the columns, e.g., `NORTHING`,
- `<numberOfRows>` and `<numberOfColumns>` specify the size of the image given as rows and columns,
- `<rowSpacing units="m">` and `<columnSpacing units="m">` that specify the footprint size of a pixel projected on ground in meters.

Classification scheme for annotating image patches

The image mining work carried out by DLR in work package "Knowledge discovery from EO images" (WP3) concentrates on methods that cut an input SAR image into patches, and then apply various image mining techniques with the goal of annotating patches with user-defined semantic categories. Moreover, a classification scheme with semantic categories of interest is currently being developed by DLR as part of the work in Task 6.2: "Developing ontologies for the VO for TerraSAR-X data" of work package WP6.

The scheme is based on the foundational work of Anderson [AHRW76]. The Anderson framework specifies a classification for land use and land cover to be used with remote sensor data. It is hierarchical consisting of four levels, and relies on a uniform categorization at the more generalized first and second levels, while it is intentionally left open-ended so that agencies can have flexibility in developing more detailed land use classifications at the third and fourth levels in order to meet their particular needs.

Feature vector of an image patch

The processing of the SAR images in WP3 is performed by dividing each image into smaller rectangular patches and handling each patch separately. For each patch the following information is available so far:

- Parent product,
- Patch size,

- Patch Index,
- Latitude/Longitude,
- Acquisition Time,
- Applied Algorithms/Parameters,
- Patch Features (array of values),
- Semantic labels (class, code from classification scheme),

The details of the feature vectors will be described in the deliverables of WP3. We foresee additional information about the actual pixel resolution in comparison with pixel spacing as well as the so-called latent models. The annotation of pixel resolution leads to the concept of resolution pyramids where typical scale-dependent details appear and disappear depending on the selected scale. When we represent patch features on different scales, the scale dependency of physical features (especially of man made objects or of vegetation in remote sensing images) allows us to obtain additional feature characteristics and to link them to the classification of image content.

2.1.3 EOWEB portal

The German Remote Sensing Data Center (DFD) is an institute of the German Aerospace Center (DLR). DFD supports science and industry as well as the general public. With its national and international receiving stations DFD offers direct access to data from earth observation missions, derives information products from the raw data, disseminates these products to users, and safeguards data in its National Remote Sensing Data Library for long term use. Its geoscience research is related to the atmosphere, global change and civil security. DFD operates thematic user services, in particular the World Data Center for Remote Sensing of the Atmosphere (WDC-RSAT), and the Center for Satellite-based Crisis Information (ZKI).

The user interface "Earth Observation on the WEB" (EOWEB)[3] provides access to the earth observation data available at the DFD. A user can search for catalogue data, browse images, order catalogue data, specify and order future acquisitions. In particular, the following products are available via the new generation of the interface, EOWEB-NG:

- O3M-SAF products,
- IRS-P5 Cartosat-1 (IRS-P5.PAN.MONO.P and IRS-P5.PAN.STEREO.P),
- TerraSAR-X products (with restricted access to privileged users only).

Also, a series of other products, including thematic maps, atmospheric sensors, and optical images, are available via the standard EOWEB interface.

Figure 2.3 shows the home view of the interface. A user can specify a time period and select a geographical area using the mouse in order to query the catalogue for products. A list of desired products can be selected by clicking on the corresponding check-boxes in a hierarchical menu. For example, the TSX-1 category holds all TerraSAR-X products which are then organized in four categories based on the four available imaging modes (stripmap, high resolution spotlight, spotlight, and ScanSAR).

For example, Figure 2.4 shows the results for a query about SAR images of any imaging mode over Munich in the time period 1/1/2011 - 31/1/2011. As seen in the figure, 10 images are retrieved

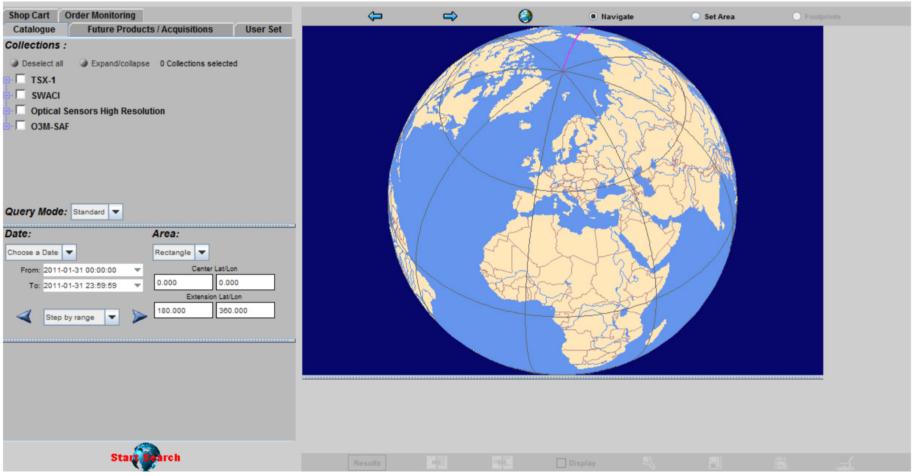


Figure 2.3: The DLR EOWEB portal home view

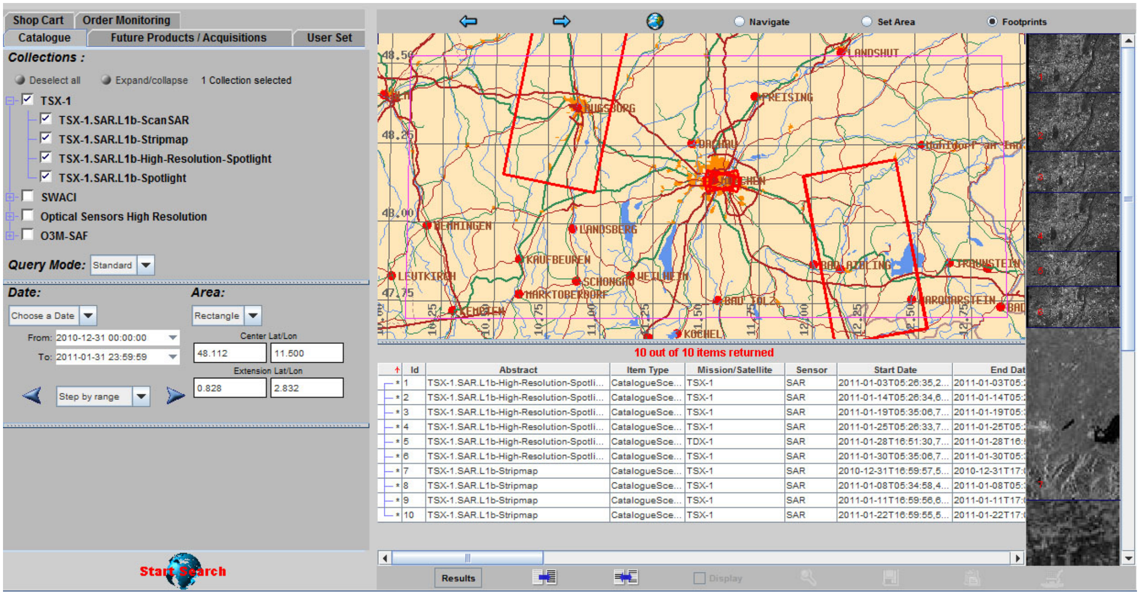


Figure 2.4: Performing a query over Munich on the DLR EOWEB portal

with different characteristics. A preview of these images is seen on the right side of the interface, while the location of each one is denoted in red on the map.

Technically, this type of query capability is realized within EOWEB by extracting key parameters from the image metadata and making them accessible to the user interface. In particular, this includes the image metadata defining the instrument, its operating mode, the date/time range to be searched, the latitude/longitude range to be considered (or an equivalent region of interest defined interactively on a world map), and the ordering of the results (by date, distance, etc.)

In addition, technically demanding instruments such as TerraSAR-X offer additional criteria that have an important impact on image quality. In the case of TerraSAR-X, this includes the beam incidence angle (as it has an impact on image resolution), and the transmitted and received polarization (as an additional target response criterion). In practice, any new instrument has to define which of its metadata parameters shall be made accessible to the EOWEB users.

When a user has completed the list of image selection parameters, the list will be coded internally

[3] `http://eoweb.dlr.de:8080/index.html`

and transmitted to the EOWEB catalogue. A catalogue search will be triggered that returns a list of candidate images that fulfill the criteria of the query. This list appears as an ordered list on the user interface. At present, this is the straightforward way to select candidate images and access the EOWEB catalogue. No additional interfaces are offered to the user community, however, one could think of advanced concepts that could be implemented as future solutions.

2.1.4 Representing the DLR data sets using semantic web technology

In order to make use of semantic web technologies, we need to transform the data sets provided by DLR to some appropriate format such as RDF and its variants. In our case we will use sRDF++ data model.

We now show how a small subset of the provided sets can be represented using this technology. First, we construct an RDFS ontology of concepts and properties as shown in Figure 2.5. Table 2.2 shows the prefixes that we used for the various namespaces that are included in the ontology.

namespace name	prefix
http://www.earthobservatory.eu/ontologies/dlrOntology.owl	dlr
http://harmonisa.uni-klu.ac.at/ontology/corine.owl	clc
http://www.w3.org/2001/XMLSchema	xsd
http://www.w3.org/1999/02/22-rdf-syntax-ns	rdf
http://srdf.di.uoa.gr/ontology	srdf
http://www.w3.org/2000/01/rdf-schema	rdfs

Table 2.2: Namespace names and their prefixes

The ontology has the following classes:

- `dlr:Image`. This class is used to describe TerraSAR-X images.
- `dlr:Region`. This class depicts all areas in Germany along with their land use.
- `clc:CorineArea`. This class imitates the Corine land cover nomenclature, since it contains subclasses that map to all the items of the three levels of the nomenclature. Instances of class `dlr:Region` become instances of class `clc:CorineArea` through the property `dlr:hasCorineLandCoverUse`. CorineArea class belongs to the Europe CORINE ontology, that is available online through the HarmonISA[4] project.

A description of some important properties follows:

- `dlr:hasGeometry`. This property describes the geometry of an area in Well-known Text (WKT) format. Well-known text is a text markup language for representing vector geometry objects on a map, spatial reference systems of spatial objects, and transformations between spatial reference systems.
- `dlr:hasStartDate`, `dlr:hasEndDate`. These properties provide information about the date and time of the acquisition of an image. As the acquisition of SAR images may have a measurable duration, we use both the start and end date/time of the acquisition. The `xsd:dateTime` data type is used represent this information.
- `dlr:hasCorineLandCoverUse`. This property connects areas that are instances of class `dlr:Region`, to instaces of class `clc:CorineArea`. This allows us to obtain all benefits offered by the CORINE ontology for the area of reference.

[4] `https://harmonisa.uni-klu.ac.at/`

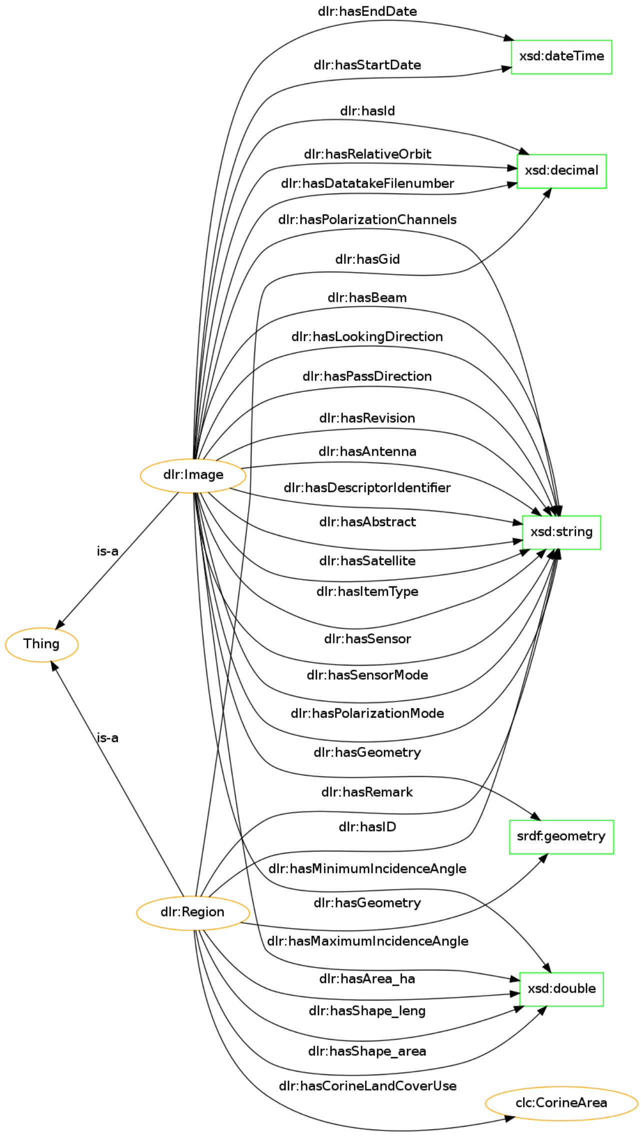


Figure 2.5: Classes in DLR's ontology

Next we represent some illustrative part of the DLR datasets using this vocabulary. In what follows we give an sRDF++ representation (in Turtle notation) of three TerraSAR-X images, and one geographic region. For simplicity, for each image we represent the start and end date of its acquisition, its geometry as a polygon, and a unique id. The geographic region represents a region in Germany along with its land cover user.

Notice that this sample dataset is very simplistic as many properties are omitted. Nonetheless it will serve our purpose of posing some illustrative queries and presenting their results. We use prefix `ex` for the namespace of an example ontology.

The sRDF++ data follows.

```
clc:Forests rdf:type rdfs:Class .
clc:ConiferousForest rdf:type rdfs:Class .
clc:ConiferousForest rdfs:subClassOf clc:Forests .
```

```
ex:image_1
        a dlr:TSX_1;
        dlr:hasStartDate "2011-01-10T05:03:07"^^xsd:dateTime;
        dlr:hasEndDate "2011-01-10T05:03:09"^^xsd:dateTime;
        dlr:hasGeometry "POLYGON((2 3,3 3,3 2,2 2,2 3))"^^srdf:geometry;
        dlr:Id "1"^^xsd:decimal.

ex:image_2
        a dlr:TSX_1;
        dlr:hasStartDate "2011-01-17T16:48:58"^^xsd:dateTime;
        dlr:hasEndDate "2011-01-17T16:49:00"^^xsd:dateTime;
        dlr:hasGeometry "POLYGON((1 3,2 3,2 2,1 2,1 3))"^^srdf:geometry;
        dlr:Id "2"^^xsd:decimal.

ex:image_3
        a dlr:TSX_1;
        dlr:hasStartDate "2011-01-20T05:17:00"^^xsd:dateTime;
        dlr:hasEndDate "2011-01-20T05:17:01"^^xsd:dateTime;
        dlr:hasGeometry "POLYGON((1 2,2 2,2 1,1 1,1 2))"^^srdf:geometry;
        dlr:Id "3"^^xsd:decimal.

ex:Region_1
        a dlr:Region;
        dlr:hasCorineLandCoverUse clc:ConiferousForest;
        dlr:hasGeometry "MULTIPOLYGON(((1 2.5,1 4,3 4,3 2.5,1 2.5)))"^^srdf:geometry;
        dlr:hasGid "1"^^xsd:decimal.
```

2.1.5 Examples of queries using semantic web technology

In this section we show how SPARQL can be used to query the data represented in RDF. We present three queries and their results. The first one essentially captures the current EOWEB functionality. The second one combines TerraSAR-X data with other data that may be available about land cover or land use, and the last one makes use of linked data that is publicly available on the web. Note that the last two queries go beyond the current EOWEB functionality.

1. Find the URIs and the geometries of TerraSAR-X images that overlap with a user specified area, and their acquisition time lies inside a user specified time period.

   ```
   select ?T ?TGEO
   where {
          ?T rdf:type dlr:TSX_1 .
          ?T dlr:hasGeometry ?TGEO .
          ?T dlr:hasStartDate ?TSD .
          ?T dlr:hasEndDate ?TED .
          filter (mbb(buffer(POINT(2.0 2.0), 1.0)) overlap ?TGEO &&
                  "2011-01-01T00:00:00"^^xsd:dateTime < ?TSD &&
                  "2011-01-01T00:00:00"^^xsd:dateTime > ?TED)
         }
   ```

   Essentially this captures the user interaction with the EOWEB interface, that is, the action of choosing a rectangle area, a time period, and performing a search on TerraSAR-X products for this configuration. The rectangle area is chosen similarly to the EOWEB portal by selecting the center point and the size of an orthogonal rectangle using `buffer(POINT(2.0 2.0), 1.0)`.

Query result.

?T	?TGEO
ex:image_1	"POLYGON((2 3,3 3,3 2,2 2,2 3))"
ex:image_2	"POLYGON((1 2,2 2,2 1,1 1,1 2))"
ex:image_3	"POLYGON((1 2,2 2,2 1,1 1,1 2))"

In this very simple case all three images overlap with the specified area, and their acquisition time lies inside the specified time period. Therefore, all three images are returned as the result of the query.

2. Find the URIs and the geometries of images that overlap with a user specified area, have an acquisition time that lies inside a user specified time period, and overlaps with a forest.

```
select ?T ?TGEO
where {

       ?T rdf:type dlr:TSX_1 .
       ?T dlr:hasGeometry ?TGEO .
       ?T dlr:hasStartDate ?TSD .
       ?T dlr:hasEndDate ?TED .
       ?R rdf:type dlr:Region .
       ?R dlr:hasGeometry ?RGEO .
       ?R dlr:hasCorineLandCoverUse clc:Forests .
       filter (mbb(buffer(POINT(2.0 2.0), 1.0)) overlap ?TGEO &&
               "2011-01-01T00:00:00"^^xsd:dateTime < ?TSD &&
               "2011-01-01T00:00:00"^^xsd:dateTime > ?TED &&
               ?TGEO overlap ?RGEO)
      }
```

Notice that this query is very similar to the previous one. The only difference is that we add the requirement that the returned images also overlap with a given forest region.

Query result.

?T	?TGEO
ex:image_1	"POLYGON((2 3,3 3,3 2,2 2,2 3))"
ex:image_2	"POLYGON((1 2,2 2,2 1,1 1,1 2))"

In this case only two of the images qualify as results of the query as `ex:image_3` does not overlap with `ex:Region_1`.

3. Find the URIs and the geometries of images that contain the city of Munich and their acquisition time lies inside a user specified time period.

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos>
PREFIX dbp: <http://dbpedia.org/property/>
```

```
select ?T ?TGEO
where {

        ?T rdf:type dlr:TSX_1 .
        ?T dlr:hasGeometry ?TGEO .
        ?T dlr:hasStartDate ?TSD .
        ?T dlr:hasEndDate ?TED .
        ?TW rdf:type dbo:Town .
        ?TW dbp:location "Munich"^^xsd:string .
        ?TW geo:geometry ?TWGEO .
        filter ("2011-01-01T00:00:00"^^xsd:dateTime < ?TSD &&
                "2011-01-01T00:00:00"^^xsd:dateTime > ?TED &&
                ?TGEO overlaps mbb(buffer(?TWGEO, 1.5)))
      }
```

Query result.

?T	**?TGEO**
ex:image_1	"POLYGON((2 3,3 3,3 2,2 2,2 3))"

This query specifies that the retrieved images should overlap with the city of Munich, *without having to specify the coordinates or the borders of the city.* This information can be retrieved by publicly available datasets such as the DBpedia data set that is available on the web as linked data.

2.2 Semantic Challenges in the NOA use case

In this section we present an outline of the NOA use case of TELEIOS as presented in Deliverable 7.1 "Requirements specification for the real-time fire monitoring application", and focus on the need to develop simpler, more efficient and more adaptive software solutions for dealing with the problems of real time hotspot and active front detection, and burnt area assessments. We describe many aspects of the NOA data sets, providing information about the different file formats that are being used. Then we describe the method that is currently being used for transforming the NOA data sets into sRDF++ metadata. Finally, we present some interesting sSPARQL++ queries on this data.

2.2.1 Description of the NOA use case

European initiatives in the area of Earth Observation such as GMES (Global Monitoring for Environment and Security) have taken an active role in the area of fire monitoring and management in Europe, and supported the development of relevant European pre-operational infrastructures through projects such as LinkER and SAFER supporting the implementation of an operational GMES service in the field of emergency management. Our use case partner NOA participates in SAFER and in the specific activity of SAFER called "Forest Fire Services", assuming the role of provider of fire related services to users (national and regional authorities, civil protection organizations, forestry services, environmental organizations and governmental entities). Moreover, through the LinkER project, NOA assumes the role of the National Focal Point for integrating and disseminating the full set of GMES Emergency Response services throughout Greece. Delivery of operational fire services like Burnt Scar Mapping and Fire Monitoring are the main focus of this activity.

In this use case, NOA concentrates on the development of solutions for two relevant problems: (i) real time hotspot and active front detection, and (ii) burnt area assessments. Technological solutions to both of these cases require integration of multiple, heterogeneous data sources with data of varying quality and varying temporal and spatial scales. Some of the data sources are streaming ones (e.g., streams of EO images, and/or streams of ready to use fire monitoring - fire detection and burnt area products) while others are static like geo-information layers (e.g., land use/land cover maps) providing additional evidence on the underlying characteristics of the affected area.

The current state of the art in NOA's fire monitoring capability is the following. At the premises of NOA, raw images are obtained every 5 to 15 minutes by accessing the infrared imager SEVIRI on board the Meteosat Second Generation (MSG) satellites. The images are processed immediately after they are acquired using in-house developed applications for geo-referencing. They are afterwards projected to the National Geographic System EGSA87, and then analyzed for hotspot and active front detection by invoking specialized image processing software. The results of this process are geo-referenced, raster and vector hotspot detection products which cover the entire territory of Greece. Moreover, fire monitoring and hotspot detection products are generated and archived on a 15-minute basis through the SAFER project by accessing the so called FMM-1 product, and medium resolution burnt area maps covering the entirety of Greece are generated and archived by accessing the so called FMM-2 SAFER product, after processing the daily MODIS acquisitions on board the Terra and Acqua satellites.

An important challenge in NOA use case is to develop advanced semantics-based querying of the existing archives of raw images and ready to use hotspot and burnt area mapping products (FMM-1 and FMM-2), as well as auxiliary information layers (e.g., land use/land cover layers) in order to exclude cases of false alarms in the delivered ERCS products.

2.2.2 Description of the NOA data sets

We now present four data sets (MSG/SEVIRI archives, GMES ERCS FMM-1 archives, GMES ERCS FMM-2 archives and various GIS data) that are used in NOA use case:

- **MSG/SEVIRI archives**. Since 2007, NOA operates an MSG/SEVIRI acquisition station, and receives and archives raw satellite images on a 15-minute basis. The archives of raw MSG/SEVIRI images, which are of the order of 1-2 Terrabytes and refer to the fire periods of recent years, as well as the images to be acquired since day 1 of the project are available for TELEIOS NOA use case purposes. Figure 2.6 shows a preview of the visual channel.

  Currently NOA creates a proccess chain that first uses the program xrit2pic to transform the raw image to some other format, like `.pgm`, `.ppm`, `.bil` or `.hdr`, as soon as a new image is acquired and then makes further use of the products of this transformation.

- **GMES ERCS FMM-1 archives**. The current GMES/SAFER Medium Resolution hotspot detection FMM-1 service provides ready to use fire (hotspot) monitoring products delivered every 15 minutes, based primarily on MSG/SEVIRI satellite acquisitions. These products are used in complement to FMM-2 products. FMM-1 products consist of points representing hotspots accompanied by a raster SEVIRI based background map (one single geo-referenced channel). The data formats are ESRI shapefiles and KML files. The files have been assigned specific attribute tables with information relating to date and time of image acquisition, cartographic X, Y coordinates of detected fire locations, the level of reliability in the observations, the fire radiative power assessed and the observed fire area. The archive time span is from 01/06/2009 to 30/09/2009. The data volume of the existing archives is around 10 MB for the hotspots and more than 10 GB for the raster data. More than 400 hotspot shapefiles, each containing a much bigger number of detected fires all over Greece are available only for the 2009 fire period. Additional archives will be also generated for the fire periods of 2010 and 2011 over the entire Greece.

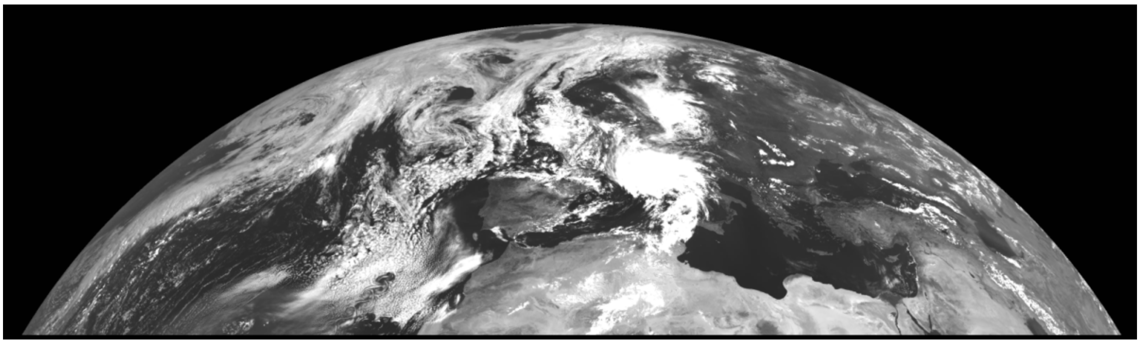


Figure 2.6: Visual channel of a SEVIRI/MSG2 image

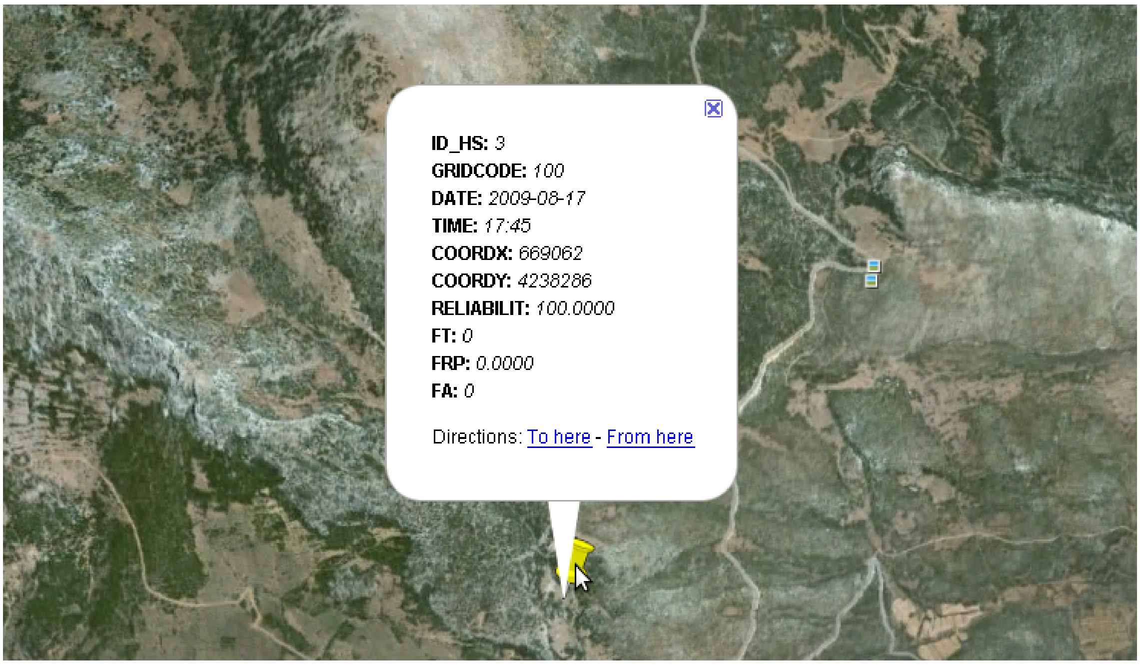


Figure 2.7: Hotspot outside the premises of Athens

An example of a hotspot is shown in Figure 2.7, which was taken on August 12, 2009, at 17:45. Its coordinates place it in an area near the premises of Athens and the reliability level of 100% assures the fact that there was a burning fire at that time. The service was delivered for the first time over entire Europe during the summer of 2009 and will continue for the next summer periods in the framework of GMES.

In Table 2.3 we show some fire monitoring products from the original dataset. Attributes `gid` and `id_hs` uniquely identify a hotspot of reference, attributes `date` and `time` stand for the date and time the image was obtained, attributes `coordx` and `coordy` are the coordinates of the hotspot projected to the HGRS87 reference system. HGRS87, or simply Greek Grid[5], is a projected coordinate reference system describes the area of Greece (onshore and offshore), including Aegean Islands, Ionian Islands, Dodecanese Islands, Crete, and Mount Athos. It has the code 2100 in the EPSG geodetic parameter registry and was last revised in 27/05/2005. HGRS87 serves as a large and medium scale topographic mapping and engineering survey.

- **GMES ERCS FMM-2 archives**. The current GMES/SAFER Medium Resolution burnt area mapping FMM-2 service provides ready to use accumulated burnt area mapping products, in polygon format and projected to the EGSA87 reference system. They are based on

[5] `http://spatialreference.org/ref/epsg/2100/`

No.	gid	id_hs	gridcode	date	time	coordx	coordy	reliability	ft	frp	fa	the_geom
1	1	1	100	2009-08-17	16:15	667808	4233786	100.0000	4978	27.4	89	POINT (667808.30554752 4233786.54964344)
2	2	2	100	2009-08-17	16:15	671374	4234083	100.0000	5073	27.4	82	POINT (671374.88226774 4234083.27742121)
3	3	3	100	2009-08-17	17:45	669062	4238286	100.0000	0	0.0000	0	POINT (669062.839785807 4238286.87450619)

Table 2.3: Fire monitoring products data

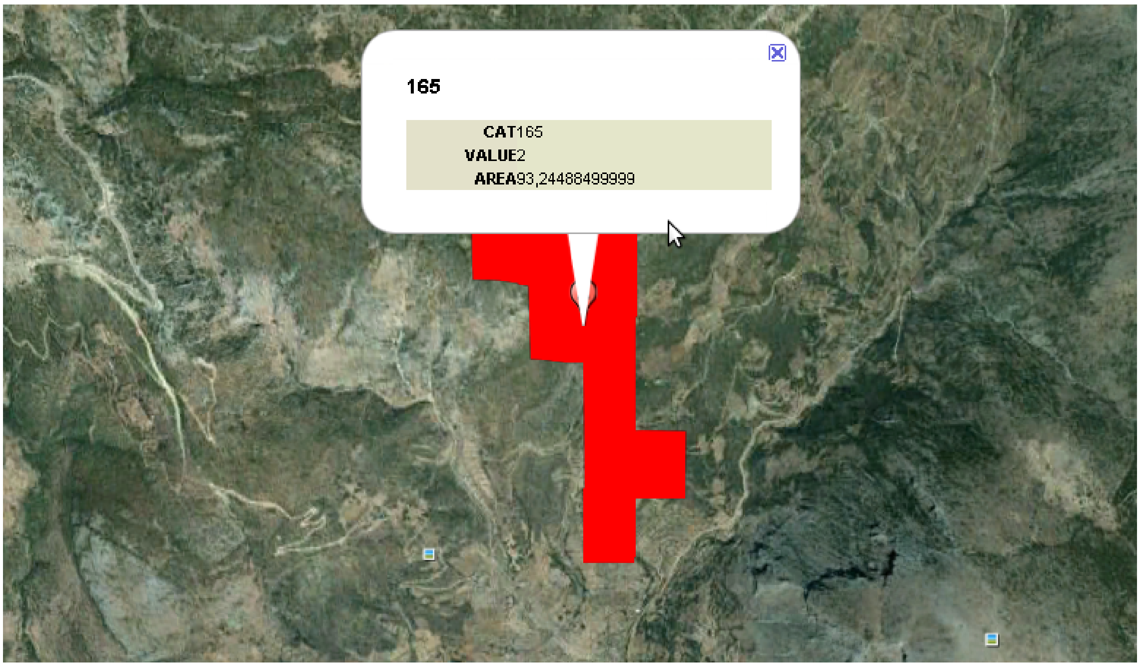


Figure 2.8: Burnt area in the county of Fokida

medium resolution MODIS data. What is more they are derived daily and cover the entire Greek territory. The data formats are ESRI shapefiles and KML files. The files have been assigned specific attribute tables with information relating to date and time of image acquisition, and the mapped fire area. The product time span is from 25/05/2009 to 21/09/2009 hence a total of 120 days. Additional archives will be also generated for the fire periods of 2010 and 2011 over the entire Greece.

An example of a burnt area mapping product is shown in Figure 2.8, where the red polygon stands for a total of approximately 93 sq.deg of burnt land. Like FMM-1, this service was delivered for the first time over the entire Europe during the summer of 2009 and will continue for the next summer periods in the framework of GMES.

In Table 2.4 we show some burnt area mapping products from the original dataset. Attribute `gid` uniquely identifies a burnt area of reference, attribute `area` stands for the total area burnt in square degrees (sq.deg) and attribute `the_geom` describes its geometry projected to the National Geographic System EGSA87 and attribute `the_geom` describes its geometry also according to EGSA87. EGSA87, which is also known as GGRS87[6], is a 2-dimensional geodetic coordinate reference system that that similarly to HGRS87 describes the area of Greece (onshore and offshore), including Aegean Islands, Ionian Islands, Dodecanese Islands, Crete, and Mount Athos. It has the code 4121 in the EPSG geodetic parameter registry and was last revised in 06/01/2004. EGSA87 serves as a geodetic survey.

[6] `http://spatialreference.org/ref/epsg/4121/`

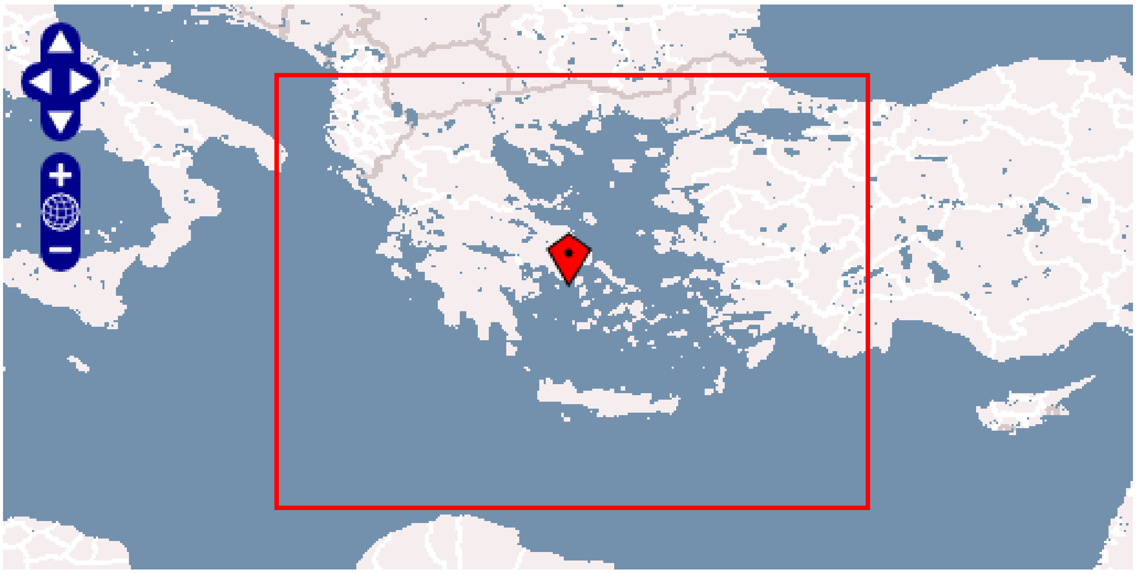


Figure 2.9: WGS84 bounds of Greece

In Figure 2.9 we can see the bounding box of coordinate reference systems EGSA87 and HGRS87. The bounding box is the same for both systems when transformed to a standard reference system for use in cartography, geodesy, and navigation like the World Geodetic System (WGS84), since they describe the same area.

No.	gid	cat	value	label	area	the_geom
1	1	2	2	null	75.274516000000006	MULTIPOLYGON(((26.2614166837823 41.6440201924671,26.2586970809172 41.6440201924671,...)))
2	2	3	2	null	47.971356999999998	MULTIPOLYGON(((26.2750146981076 41.5515536950596,26.2668558895124 41.5515536950596,...)))
3	3	165	2	null	93.248849999999998	MULTIPOLYGON(((26.294051918163 41.5379556807349,26.2886127124329 41.5379556807349,...)))

Table 2.4: Burnt area mapping products data

- **GIS data**
  - **Corine Land Cover data**. NOA has also made available an ESRI shapefile of Corine Land Cover nomenclature that covers greek geological characteristics and provides a location precision of 100 meters.

    In general, such a nomenclature comprises three levels:

    * The first level, consisting of five items indicates the major categories (abstract to a greater or lesser degree) of land cover on the planet. For example forests and semi-natural areas are defined by the code number 3.
    * The second level, consisting of fifteen items is for use on scales of 1:500,000 and 1:1,000,000. For example it can indicate areas like open spaces with little or no vegetation, which are defined by the code number 33 and thus belong to the major category of forests and semi-natural areas, that we used as an example for the first level.

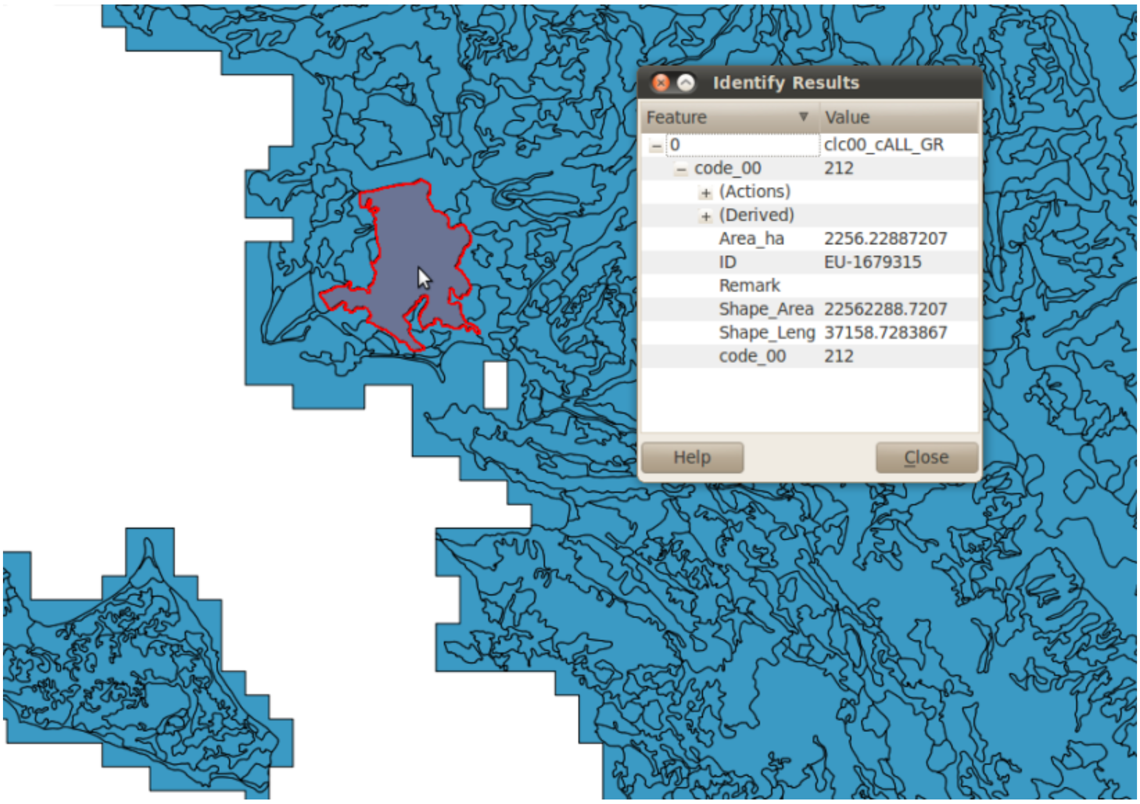


Figure 2.10: Permanently irrigated land in the region of Epirus

- The third level, consisting of forty-four items is for use on a scale of 1:100,000, narrowing down the land use to a very specific geographic characterization. For example it can describe permanently irrigated land and even burnt areas (code number 334) and complete the level chain that starts with forests and semi-natural areas at first level and continues with open spaces with little or no vegetation at second level. Figure 2.10 presents an example of permanently irrigated land.

As it was described above, the index of all three levels combined forms a unique id for the land use of the area of reference. For instance, all burnt areas are tagged with the id 334. Further information about the Corine land cover project can be found at `http://www.eea.europa.eu/publications/COR0-landcover`.

In Table 2.5 we show some areas from the Corine land cover dataset. Attribute `gid` uniquely identifies an area of reference, attribute `code` is the id for the land use of the area according to the Corine land cover nomenclature described previously and attribute `the_geom` describes its geometry projected to the HGRS87 reference system.

- **Coastline data**. This is an ESRI shapefile provided by NOA that describes the geometry of the coastline of Greece. The coastline of Greece is officially given as 13,676 kilometers, while the country covers a total area of 131,940 square kilometers. Thus, this kind of data is very important in the case of hotspot detection that appear near the coastline, since hotspot detection is performed on medium scale satellite images. So during hotspot detection we must consult the coastline geometry and overlay a square product approximation (depending on the spatial analysis of the satellite image) of the hotspot upon the coastline, to find out whether they are wrong readings. The whole coastline of Greece can be seen in Figure 2.11.

  In Table 2.6 we show some Greek coastline data from the original dataset. Attributes `gid` and `id` uniquely identify a coastline of reference and attribute `the_geom` describes its geometry projected to the HGRS87 reference system.

No.	gid	code	id	remark	area_ha	shape_leng	shape_area	the_geom
1	1	111	EU-1584090	null	51.8530917512	4072.195819	518530.9175	MULTIPOLYGON (((5700379.1393 1500342.4196, 5700463.6582 1500258.7894, ...)))
2	2	111	EU-1584091	null	60.6149594705	5837.645074	606149.5947	MULTIPOLYGON (((5597321.7592 1500728.6447, 5597211.1134 1500669.5151, ...)))
3	3	212	EU-1679315	null	2256.22887207	37158.728386	22562288.7202	MULTIPOLYGON (((5598109.7905 1501145.8445, 5598167.0553 1501083.3984, ...)))

Table 2.5: Corine Land Cover data

No.	gid	id	the_geom
1	1	null	MULTIPOLYGON(((693943.875023903 3869464.00023706,693938.875023898 3869464.00023706, ...)))
2	2	null	MULTIPOLYGON(((439110.999999992 3964310.00025417,439070.311999992 3964339.00025417, ...)))
3	3	null	MULTIPOLYGON(((761862.125204979 3914260.00024361,761846.875204895 3914260.00024361, ...)))

Table 2.6: Greek coastline data

2.2.3 Representing the NOA data sets using semantic web technology

In order to make the use of semantics-based query languages possible and obtain all benefits offered by that approach, we need to transform the shapefiles offered by NOA partner to sRDF++ metadata.

Up to this point we have mentioned the term shapefile in our description of NOA's data sets, as a data format related to spatiotemporal attributes. We now proceed with a proper definition of the shapefile. The ESRI Shapefile, or simply a shapefile, is a popular geospatial vector data format for geographic information systems software. Shapefiles spatially describe geometries, like points, polylines, and polygons. These geometries could represent for example coastlines, rivers or lakes. Each geometry may also have attributes that describe it, such as its name or land area.

To represent the information included in these shapefiles as sRDF++ triples, we first constructed an RDFS ontology which is shown in Figure 2.12. In Table 2.7 we show the prefixes used for the various namespaces that are included in our ontology. The ontology has the following classes:

namespace name	prefix
http://www.earthobservatory.eu/ontologies/noaOntology.owl	noa
http://harmonisa.uni-klu.ac.at/ontology/corine.owl	clc
http://www.w3.org/2001/XMLSchema	xsd
http://www.w3.org/1999/02/22-rdf-syntax-ns	rdf
http://srdf.di.uoa.gr/ontology	srdf
http://www.w3.org/2000/01/rdf-schema	rdfs

Table 2.7: Namespace names and their prefixes

- `noa:BurntArea`. This class corresponds to the burnt area mapping products available from FMM-2 service.
- `noa:Hotspot`. This class in its turn has a direct relation with the fire monitoring products from FMM-1 service.

Figure 2.11: Coastline of Greece

- `noa:Coastline`. This class is used to describe a single geometry, the coastline of Greece.
- `noa:Region`. This class depicts all areas in Greece along with their land use.
- `clc:CorineArea`. This class imitates the Corine land cover nomenclature, since it contains subclasses that map to all the items of the three levels of the nomenclature. Instances of class `noa:Region` become instances of class `clc:CorineArea` through the property `noa:hasCorineLandCoverUse`. CorineArea class belongs to the Europe CORINE ontology, that is available online through the HarmonISA[7] project.

A description of some important properties follows:

- `noa:hasGeometry`. This property describes the geometry of an area in Well-known text (WKT) format. Well-known text is a text markup language for representing vector geometry objects on a map, spatial reference systems of spatial objects and transformations between spatial reference systems.
- `noa:hasDateTime`. This property provides information about both the time and date of an acquired hotspot. It uses the `xsd:dateTime` data type to represent this information.

[7] `https://harmonisa.uni-klu.ac.at/`

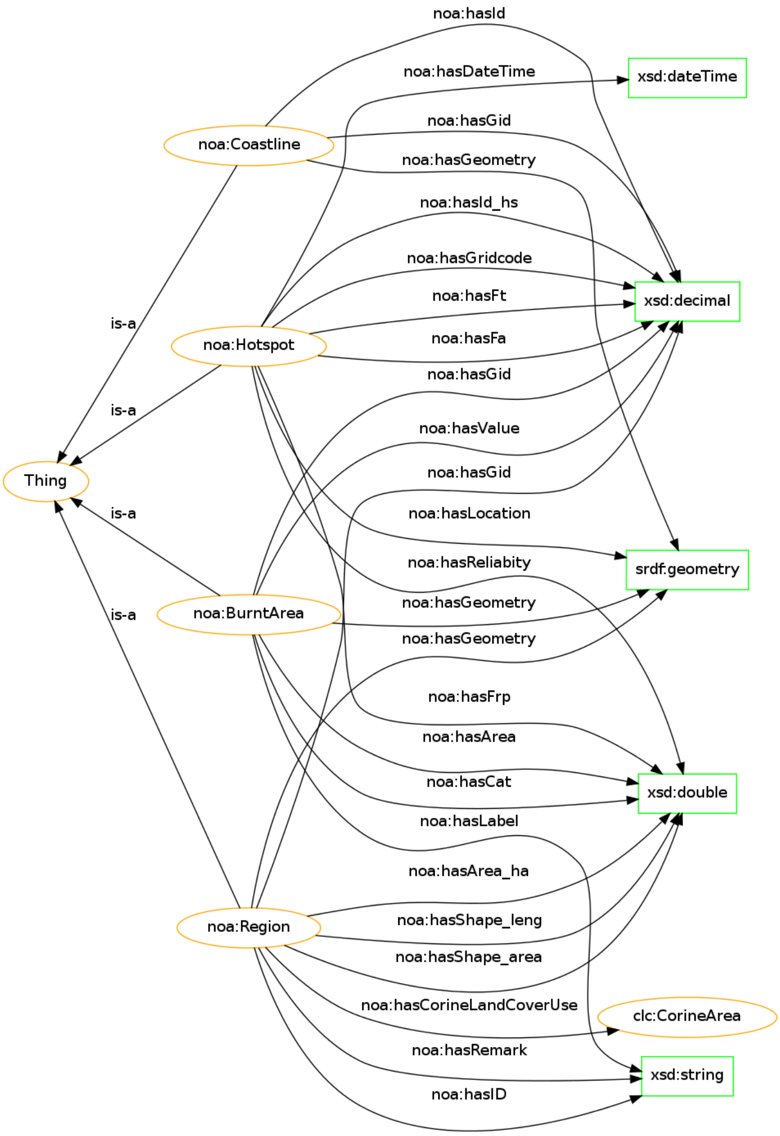


Figure 2.12: Classes in NOA's ontology

- `noa:hasCorineLandCoverUse`. This property connects areas that are instances of class `noa:Region`, to instaces of class `clc:CorineArea`. This allows us to obtain all benefits offered by the CORINE ontology for the area of reference.

sRDF++ triples are created as follows. Each attribute of a shapefile becomes a predicate, each attribute value becomes an object and finally a subject is created, as a unique id that identifies the corresponding resource. For example class `Coastline` has properties `hasGid`, `hasGeometry` and `hasId`. These were originally the attributes `Gid`, `the_geom` and `Id` of the shapefile.

It should be noticed though, that the above procedure is generic and is not the case when we want to combine multiple attributes to create a property. For example, in the case of hotspots where we have attributes `time` and `date`, we create a single property `noa:hasDateTime` which is of type `xsd:dateTime`.

We will use Turtle notation in our examples throughout the rest of the section to represent sRDF++ triples.

Corine land cover data

In Table 2.8 we show an example of Corine land cover data. The rows of the table become instances of class `Region`.

No.	gid	code	id	remark	area_ha	shape_leng	shape_area	the_geom
1	1	111	EU-1584090	null	51.8530917512	4072.195819	518530.9175	MULTIPOLYGON (((5700379.1393 1500342.4196, 5700463.6582 1500258.7894, ...)))

Table 2.8: Corine Land Cover data

In this example the attribute `code` is mapped into property `noa:hasCorineLandCoverUse` and all its values become instances of class `noa:CorineLandCoverUse`. Attribute `gid` becomes property `noa:hasGid`, attribute `id` becomes property `noa:hasId`, attribute `remark` becomes property `noa:hasRemark`, attribute `area_ha` becomes property `noa:hasArea_ha`, attribute `shape_leng` becomes property `noa:hasShape_leng`, attribute `shape_area` becomes property `noa:hasShape_area`, and attribute `the_geom` becomes property `noa:hasGeometry`. The sRDF++ triples that are produced from the row are:

```
ex:Region_1
        a noa:Region;
        noa:hasArea_ha "51.8530917512"^^xsd:double;
        noa:hasCorineLandCoverUse clc:ContinuousUrbanFabric;
        noa:hasGeometry "MULTIPOLYGON(((5700379.1393 1500342.4196,
        5700463.6582 1500258.7894,...)))"^^srdf:geometry;
        noa:hasGid "1"^^xsd:decimal;
        noa:hasID "EU-1584090"^^xsd:string;
        noa:hasRemark "null"^^xsd:string;
        noa:hasShape_area "518530.9175"^^xsd:double;
        noa:hasShape_leng "4072.195819"^^xsd:double.
```

Coastline data

In Table 2.9 we show an example of Greek coastline data. The rows of the table become instances of class `Coastline`.

No.	gid	id	the_geom
1	1	null	MULTIPOLYGON(((693943.875023903 3869464.00023706,693938.875023898 3869464.00023706, ...)))

Table 2.9: Greek coastline data

The corresponding sRDF++ triples are built in the same manner as with the Corine land cover data example. Attribute `gid` becomes property `noa:hasGid`, attribute `id` becomes property `noa:hasId`, and attribute `the_geom` becomes property `noa:hasGeometry`. The sRDF++ triples that are produced from the row are:

```
ex:CoastLine_1
        a noa:Coastline;
        noa:hasGid "1"^^xsd:decimal;
        noa:hasGeometry "MULTIPOLYGON(((693943.875023903 3869464.00023706,
        693938.875023898 3869464.00023706,...)))"^^srdf:geometry;
        noa:hasId "null"^^xsd:decimal.
```

Fire monitoring products data

In Table 2.10 we show an example of fire monitoring products data. The rows of the table become instances of class Hotspot.

No.	gid	id_hs	gridcode	date	time	coordx	coordy	reliability	ft	frp	fa	the_geom
3	3	3	100	2009-08-17	17:45	669062	4238286	100.0000	0	0.0000	0	POINT (669062.839785807 4238286.87450619)

Table 2.10: Fire monitoring products data

Until now, in all examples we have seen, every attribute became a property of our RDFS ontology and all values became the range of the corresponding property. However, here, there is property noa:hasDateTime that describes the date and the time of the corresponding hotspot, that is defined by attributes date and time. We used two attributes to create a single property. The sRDF++ triples that are produced from the row are:

```
ex:Hotspot_3
        a noa:Hotspot;
        noa:hasFa "0"^^xsd:decimal;
        noa:hasFrp "0.0000"^^xsd:double;
        noa:hasFt "0"^^xsd:decimal;
        noa:hasGridcode "100"^^xsd:decimal;
        noa:hasId_hs "3"^^xsd:decimal;
        noa:hasCoordx "669062"^^xsd:decimal;
        noa:hasCoordy "4238286"^^xsd:decimal;
        noa:hasGeometry "POINT (669062.839785807 4238286.87450619)"^^srdf:geometry;
        noa:hasReliabity "100.0000"^^xsd:double;
        noa:hasDateTime "2009-08-17T17:45:00"^^xsd:dateTime.
```

Burnt area mapping products data

In Table 2.11 we show an example of burnt area mapping products data. The rows of the table become instances of class BurntArea.

No.	gid	cat	value	label	area	the_geom
1	1	2	2	null	75.274516000000006	MULTIPOLYGON(((26.2614166837823 41.6440201924671,26.2586970809172 41.6440201924671,...)))

Table 2.11: Burnt area mapping products data

Attribute gid becomes property noa:hasGid, attribute cat becomes property noa:hasCat, attribute value becomes property noa:hasValue, attribute label becomes property noa:hasLabel and attribute area becomes property noa:hasArea. The sRDF++ triples that are produced from the row are:

```
ex:BurntArea_1
        a noa:BurntArea;
        noa:hasArea "75.274516000000006"^^xsd:double;
        noa:hasCat "2"^^xsd:double;
        noa:hasGeometry "MULTIPOLYGON(((26.2614166837823 41.6440201924671,
        26.2586970809172 41.6440201924671,...)))"^^srdf:geometry;
        noa:hasGid "1"^^xsd:decimal;
        noa:hasLabel "null"^^xsd:string;
        noa:hasValue "2"^^xsd:decimal.
```

2.2.4 Examples of queries using semantic web technology

Consider the following dataset based on NOA use case. We have the description of three hotspots, one that appears in the sea and is most likely a wrong reading, one that is located outside forested areas and may be a false positive and finally one that is almost certainly a fire. We also have the description of a burnt area from recent fires in Greece. Finally we have the description of a part of the Greek coastline. Notice that this dataset with all its values and its geometries is very simple for the sake of the example and certain data properties that are of no use for the queries and the results obtained are omitted. We use prefix `ex` for the namespace of an example ontology.

```
clc:Forests rdf:type rdfs:Class .
clc:ArtificialSurfaces rdf:type rdfs:Class .
clc:ConiferousForest rdf:type rdfs:Class .
clc:ConiferousForest rdfs:subClassOf clc:Forests .

ex:Region_1
        a noa:Region;
        noa:hasCorineLandCoverUse clc:ConiferousForest;
        noa:hasGeometry "MULTIPOLYGON(((1 8,4 8,4 5,1 5,1 8)))"^^srdf:geometry;
        noa:hasGid "1"^^xsd:decimal.

ex:Region_2
        a noa:Region;
        noa:hasCorineLandCoverUse clc:ArtificialSurfaces;
        noa:hasGeometry "MULTIPOLYGON(((4 8,5 8,5 6,4 6,4 8)))"^^srdf:geometry;
        noa:hasGid "1"^^xsd:decimal.

ex:BurntArea_1
        a noa:BurntArea;
        noa:hasGeometry "MULTIPOLYGON(((1 8,4 8,4 6,1 6,1 8)))"^^srdf:geometry;
        noa:hasGid "1"^^xsd:decimal;

ex:Coastline_1
        a noa:Coastline;
        noa:hasGid "1"^^xsd:decimal;
        noa:hasGeometry "MULTIPOLYGON(((1 8,5 8,5 1,1 1,1 8)))"^^srdf:geometry.

ex:Hotspot_1
        a noa:Hotspot;
        noa:hasId_hs "1"^^xsd:decimal;
        noa:hasDateTime "2009-08-17T17:45:00"^^xsd:dateTime;
        noa:hasGeometry "POINT(5.5 3.5)"^^srdf:geometry;
        noa:hasReliabity "40.0000"^^xsd:double.

ex:Hotspot_2
        a noa:Hotspot;
        noa:hasId_hs "2"^^xsd:decimal;
        noa:hasDateTime "2009-08-17T17:45:00"^^xsd:dateTime;
        noa:hasGeometry "POINT(2.5 2.5)"^^srdf:geometry;
        noa:hasReliabity "70.0000"^^xsd:double.

ex:Hotspot_3
        a noa:Hotspot;
        noa:hasId_hs "3"^^xsd:decimal;
        noa:hasDateTime "2009-08-17T17:45:00"^^xsd:dateTime;
```

```
    noa:hasGeometry "POINT(2.5 6.5)"^^srdf:geometry;
    noa:hasReliabity "100.0000"^^xsd:double.
```

We now proceed with the queries:

1. Find the URIs and the geometries of coniferous forest areas.

```
select ?R ?RGEO
where {
       ?R rdf:type noa:Region .
       ?R noa:hasGeometry ?RGEO .
       ?R noa:hasCorineLandCoverUse clc:ConiferousForest .
      }
```

   Query result.

?R	?RGEO
ex:Region_1	"MULTIPOLYGON(((1 8,4 8,4 6,1 6,1 8)))"

   In this query we illustrated how to get the geometries of coniferous forests.

2. Find all the hotspots that overlap the Greek coastline.

```
PREFIX dbpedia: <http://dbpedia.org/resource/>

select ?H ?HGEO
where {
       ?C rdf:type noa:Coastline .
       ?C ex:refersToCountry dbpedia:Greece .
       ?C noa:hasGeometry ?CGEO .
       ?H rdf:type noa:Hotspot .
       ?H noa:hasGeometry ?HGEO.
       filter (?CGEO overlap mbb(buffer(?HGEO, 1.5)))
      }
```

   Query result.

?H	?HGEO
ex:Hotspot_1	"POINT(5.5 3.5)"

   Cases of hotspots like the above, are wrongly represented over inconsistent underlying land use/land cover classes like urban, permanent agriculture or in the sea, due to the rough spatial pixel resolution of MSG/SEVIRI instrument and errors in image georeferencing . This type of error could be easily corrected if seen together with the Greek coastline boundaries and detailed land use/land cover information layers e.g., Corine Land Cover (CLC) data, like in the preceding query.

3. Find all the hotspots that are located outside forested areas.

```
select ?H ?HGEO
where {
       ?R rdf:type noa:Region .
       ?R noa:hasGeometry ?RGEO .
```

```
 ?H rdf:type noa:Hotspot .
 ?H noa:hasGeometry ?HGEO.
 ?R noa:hasCorineLandCoverUse ?F .
 ?F rdfs:subClassOf clc:Forests .
 filter (?AGEO disjoint mbb(buffer(?HGEO, 1.5)))
}
```

Query result.

?H	?HGEO
ex:Hotspot_2	"POINT(2.5 2.5)"

Cases of hotspots like the above, can be false fire detections due to algorithmic drawbacks and/or bad fire/no-fire thresholds applied in the processing chain. They can also be real cases of fires located in big agricultural plain areas put by farmers as part of their agricultural practices. Whichever the case, they are not real forest fires, and they are not emergency situations to be handled. This type of noise in the information could be avoided if combined together with detailed land use/land cover information layers e.g., Corine Land Cover (CLC) data, like in the preceding query.

4. Find burnt areas that were forests in Corine land cover data and are close to cities.

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos>

select ?H ?HGEO
where {
       ?R1 rdf:type noa:Region .
       ?R1 noa:hasGeometry ?R1GEO .
       ?R1 noa:hasCorineLandCoverUse clc:ArtificialSurfaces.
       ?CITY rdf:type dbo:City .
       ?CITY geo:geometry ?CGEO .
       ?R2 rdf:type noa:Region .
       ?R2 noa:hasGeometry ?R2GEO .
       ?R2 noa:hasCorineLandCoverUse ?F .
       ?F rdfs:subClassOf clc:Forests .
       ?B rdf:type noa:BurntArea .
       ?B noa:hasGeometry ?BGEO .
       filter (?R1 contains ?CGEO &&
               ?R2GEO contains ?BGEO &&
               distance(?BGEO, ?CGEO) < 2 )
      }
```

Query result.

?B	?BGEO
ex:BurntArea_1	"MULTIPOLYGON(((1 8,4 8,4 6,1 6,1 8)))"

Cases of burnt areas like the above, may imply malicious intervention by building corporations that want to extend their settlement. This kind of activity, although in most cases is an already accomplished fact, can be seen if combined together with detailed land use/land cover information layers, e.g., Corine Land Cover (CLC) and other structured information such as the DBpedia knowledge base, like in the preceding query.

2.3 Summary

In this chapter we focused on the use cases of the project and discussed the challenges for semantic data modeling. We looked into the specific details of each use case and sketched how the data model sRDF++ and sSPARQL++, to be presented in detail in Chapter 4, can be used to represent the data sets provided, and realize the intended functionality.

3. Related Work

In this chapter we present a survey of related work in the areas most relevant to this deliverable. We start with previous work on querying image annotations from the areas of multimedia and satellite image retrieval. Then, we survey the state-of-the-art in data models and query languages for temporal and spatial information in the following research areas: (i) relational databases, (ii) XML, (iii) RDF, and (iv) description logics and OWL. Since our work will be based on RDF, we pay more attention to the RDF extensions that are currently the state-of-the-art regarding temporal and spatial information. Finally, we discuss some work on geospatial information retrieval which is of interest. Parts of Sections 3.2, 3.3, and 3.4 come verbatim from [KKK09].

3.1 Querying Image Annotations

In this section, we consider the research area of multimedia images and satellite images, and survey work on image annotation and querying. In these approaches images are typically processed in order to extract some low-level features that are linked with specific segments of the original image. These primitive features are then combined to define higher-level semantic labels that are stored as annotations in an appropriate format. This procedure is often based on some sort of supervised or semi-supervised learning and is bootstrapped in an iterative fashion so that the high-level objects that are identified in one step are used to detect others in a more complex processing step. The problem of selecting the low-level features that will be extracted and the approach that will be used for assigning these low-level descriptions to high-level concepts is commonly referred to as the "semantic gap" [BMM96, SWS+00, ZG02].

The querying of the images is based on the derived annotations and other metadata. In the sequel we start with multimedia images and examine the characteristics of the annotations as well as the available methods and languages for querying a large data set of images along with their annotations. Note that the approaches we review are similar to those used in the research area of biomedical images. Indeed, some of the approaches presented, e.g., [HAB08], have been applied to biomedical images. Next, we move to satellite images where the literature is more sparse. One of the main reasons for this is that the characteristics of satellite images make the process of high-level semantic annotation more difficult than multimedia images. Finally, note that the techniques for the extraction of primitive features and the derivation of higher-level semantic labels is beyond the scope of this document.

3.1.1 Multimedia Images

In the era of omnipresent internet availability, community photo collections (e.g. Flickr.com) and widespread personal use of digital still and video cameras, large amounts of audiovisual data are being created everyday. The produced collections contain vast amounts of high-quality multimedia content, which is becoming harder and harder to manage, transmit, and find [MHCW08]. In this setting, text-based search engines give way to more advanced image and video-based, intelligent and context-aware engines which personalize search and delivery. Novel applications are emerging across many industry sectors such as that of home and sports entertainment and new standards such as MPEG-7 are being released that enable the propagation of semantic media adaptation and personalization. In this section we explore the state of the art in the representation languages that are used for image annotation and the methods that are used for querying images in the context of multimedia data.

Semantic web ontologies for the annotation of multimedia images

MPEG-7 [MPE01] is an international standard that specifies how to connect descriptions to parts of a media asset based on XML Schema. The two most important functionalities provided by MPEG-7 are: the decomposition of media assets and the (semantic) annotation of their parts.

- Decomposition: MPEG-7 provides descriptors for spatial, temporal, spatio-temporal and media source decompositions of multimedia content into segments. A segment is the most general abstract concept in MPEG-7 and can refer to a region of an image, a piece of text, a temporal scene of a video or even to a moving object tracked during a period of time.
- Annotation: MPEG-7 defines a very large collection of descriptors that can be used to annotate a segment. These descriptors can be low-level visual features, audio features or more abstract concepts. They allow the annotation of the content of multimedia documents or the media asset itself.

Nonetheless, the XML Schema based nature of MPEG-7 has a number of drawbacks that complicates the direct machine processing of semantic content descriptions. A detailed account of this criticism can be found in [VONH04, NVOH05]. In particular, as MPEG-7 is based on XML Schema rather than on semantic languages such as RDF, the labels used in the annotations are either just plain strings or defined in lightweight MPEG-7-proprietary controlled vocabularies that are often incompatible with Semantic Web ontologies. As a solution, multimedia ontologies based on MPEG-7 have been proposed.

The first approaches essentially modeled parts of MPEG-7 in the vocabulary of RDFS or OWL ontologies such as the work by Hunter [Hun01], Tsinaraki *et al.* [TPC04], and Garcia and Celma [GC05]. As far as static images are concerned Bloehdorn *et al.* [BPS$^+$05] proposed the Visual Descriptor Ontology (VDO) based on the visual part of MPEG-7. Other approaches incorporate features from the Wordnet ontology in order to couple the output of the image analysis and the detected mid-level concepts with semantic labels e.g., Hoogs *et al.* [HRSS03], Hollink *et al.* [HW05], and Snoek *et al.* [SHH$^+$07].

A more recent approach is the Core Ontology for MultiMedia (COMM) [VSS$^+$08, ATSH09] that revisits the structure of the data in MPEG-7 offering conceptual clarity and extensibility towards new annotation requirements. COMM supports the low-level features represented in MPEG-7 including the color descriptors (lay-out, dominant color, etc), motion descriptors (activity, trajectory, etc), shape descriptors (contour, region, etc) and texture descriptors (homogeneous texture, edges histogram, etc.). Moreover, COMM allows to store not only the results of image processing but also the methods that were used of the processing as well as individual parameters set in that particular processing.

Fuzzy thresholds on image annotations

One limitation of the usual ontological technologies based on OWL and RDF is that they cannot represent and reason with uncertainty and imprecision. Several approaches have been proposed in the literature to deal with this by incorporating work from probabilistic and fuzzy reasoning. For example, the probabilistic approach of da Costa *et al.* [dCLL05] proposes to first augment the OWL language to allow additional probabilistic markups and then to convert the probabilistic OWL ontology into the directed acyclic graph of a Bayesian network with translation rules [DPP04]. As the main ontology language OWL is based on description logics, a more practical approach is to deal with uncertainty and imprecision using fuzzy description logics, e.g., [HKS02, Str05, LXL$^+$05, SSSK06, SSP$^+$07, DKS09].

In particular, a common approach is to introduce fuzzy predicates in concrete domains. In description logics, a semantics is associated with concepts, roles and individuals such that a concept represents a set of individuals and a role is a binary relation between objects. Concrete domains give the means to describe real-world properties of objects such as their size, their spatial extension or their color. For instance, a concept may denote the set of persons whose age is lower than or equal to 20. A fuzzy extension can then be obtained by allowing to represent the degree of "youngness" of a person based on her age. In particular for multimedia images, among many works with similar emphasis, Simou *et al.* [SASK08] propose a methodology for the semantic indexing of images utilizing the confidence degree of the derived low-level features that are extracted from image processing. Simou *et al.* also present some preliminary results on using this information for the refinement of mistakenly classified regions.

The capability to express uncertainty by means of fuzziness can also be used in the specification of the segments that hold a particular semantic label. In this case the segments are represented by spatial fuzzy sets [HAB08]. A spatial fuzzy set is a fuzzy set S defined on the image space, so that its membership function μ (defined from S into $[0, 1]$) represents the imprecision on the spatial definition of the segment (its position, size, shape, boundaries, etc.). For each point (x, y) of S, $\mu(x, y)$ represents the degree to which (x, y) belongs to the fuzzy segment. Segments defined as classical "crisp" sets are particular cases for which μ takes only values 0 and 1.

Fuzzy representations of more sophisticated topological relations such as "intersects", "in the interior of", and "exterior to" can be simply defined using fuzzy set theoretical concepts. In particular, the work by Hudelot *et al.* [HAB08] accounts for a fuzzy representation of the RCC mereo-topological relations [CBGG97] as well as directional relations, which are useful to describe the relative position of an object with respect to other ones, such as "to the right of", "to the left of", "above", "below", etc.

Querying image annotations

Typically only the higher-level semantic annotations are stored in a repository so that image retrieval can be utilized. Depending on the data model that is used to represent the image annotations, an appropriate querying mechanism is used to retrieve images from the repository. In particular for the approaches that rely on semantic web technology in order to express the semantic labels, such as the ones presented in the beginning of Section 3.1.1, the querying and retrieval of multimedia images then is typically based on SPARQL, the querying language for the semantic web. Special interfaces have been developed for particular applications in order to assist the users and provide a more user-friendly experience. Two examples are presented next.

SemaPlorer [SSB$^+$09] is an application that allows end users to interactively explore and visualize a very large semantic data set in real-time. This data set is composed from DBpedia[1], GeoNames[2], WordNet[3], personal FOAF files contained in the Swoogle[4] crawl of Semantic Web data, and a partial crawl of Flickr[5].

In SemaPlorer, the users initially state a simple text query to the system. The result list contains different places, people, and tags matching the query. When the user clicks on a city name, the SemaPlorer application updates the interface showing a map of the city. Concurrently, a query is executed filling the map view with interesting places and sights, represented by pins. The users can click on pins to continue the blended browsing and querying. For example, when the map view shows the city of Berlin, one can click on the tag "street art." Instantaneously, the map view is updated and locations of Flickr photos tagged as street art are shown.

[1] `http://dbpedia.org`
[2] `http://geonames.org`
[3] `http://wordnet.princeton.edu`
[4] `http://swoogle.umbc.edu`
[5] `http://flickr.com`

Similarly, MOntoMat-Annotizer [PAS+06] is a tool that has been built in order to allow content providers to annotate visual content without prior expertise in semantic web technologies or multimedia analysis. MOntoMat-Annotizer allows users to extract MPEG-7 visual descriptors from images and store them using an RDF version of the MPEG-7 visual descriptors developed in [ATP+05].

In particular, one of the components of MOntoMat-Annotizer presents a graphical interface for loading and processing visual content, performing visual feature extraction, and linking features with concepts from the domain ontology. The user may draw a region of interest in the image frame and apply the multimedia descriptors extraction procedure only to the specific selected region or alternatively, automatic segmentation of the image can be performed. By selecting a region of interest and a specific concept in the ontology browser the user can extract and link concepts.

3.1.2 Satellite images

Unlike multimedia images, satellite images are typically more difficult to handle. One obvious reason for this is the size of the satellite images that often scale up to a few gigabytes. A more serious reason has to do with the difficulty in identifying objects and features in satellite images. For example, the mining of earth observation (EO) images for content information is very different from mining images of facial characteristics or images of other living things (e.g., lions or tigers) due to the fact there are no features (e.g., ears, stripes, or wings) that have known relationships to help differentiate classes [DDKB07]. Moreover, as far as radar (SAR) images are concerned, additional problems arise from the fact that these images have different properties than optical images. Essentially, SAR products may look like optical images but in reality they are mathematical products that rely on delicate radar measurements.

Nonetheless, setting aside the issue of feature extraction and the semantic gap that we described in the beginning of Section 3.1, the techniques for image annotation and querying are very similar to the ones targeting multimedia images.

In the last few years, semantics-based technologies have been applied to satellite image archives. Durbha and King[DK04] present a content-based retrieval system applied to knowledge discovery in satellite image archives using domain-dependent ontologies. In this paper, a three level processing scheme is adopted for content-based retrieval from remote sensing images. The first level consists of the extraction of primitive features followed by an object ontology level that describes quantitatively the image content. The third level, the more abstract one, is obtained by projecting the object ontology onto keywords and concepts. In this way, relationships between objects comprised in the image provide an accurate content description. The search capabilities are supported by means of an active learning algorithm.

Other system in this category include the KIM system developed in a technology project of ESA by the TELEIOS partners DLR and ACS [DSDM02]. The Knowledge-driven Information Mining (KIM) is an image mining system that employs a Bayesian approach. It consists of three components: a library of algorithms used for the primitive feature extraction, a Bayesian network as the classification component used to generate interactively image classifications, and a database management system for the image metadata and semantics. Another system similar in spirit to KIM is GeoIRIS [SKS+07]. In such systems, user can ask queries of the form "given a query image, show me database satellite images that have similar objects and a lie within a certain radius of a given landmark".

As far as annotations are concerned, more recent work is underway in OGC for the development of a standardized GML application schema for EO products [OGC10c]. This GML application schema is defined in a hierarchical manner capturing general EO products, thematic EO products (for example optical, radar or atmospheric) and mission specific products (for example products of the TerraSAR-X mission). Remotely sensed images are viewed as observations and their specification is captured by the GML construct `gml:Observation`. The intention of this standard is to be used together with relevant OGC catalogue services standards to enable access to EO products.

Finally, project OTEG (Open Access Ontology / Terminology for the GMES Space Component)[6] funded by ESA has developed a system for helping application experts in identifying relevant EO products using familiar semantic terms (e.g., terms from their application domain). The approach of OTEG is based on the following components:

- A multi-domain thesaurus which offers a hierarchical organization of terms from the user application domain (e.g., land monitoring).
- A multi-domain vocabulary which defines all the terms used in the multi-domain thesaurus by offering a natural language definition as well as synonyms and related terms.
- A taxonomy of EO products specific to GMES Space Component Data Access (GSCDA). This taxonomy is essentially an ontology of products, sensors, missions etc.
- A GSCDA vocabulary which defines all the term used in the GSCDA taxonomomy.
- A set of mapping rules that relate terms from the multi-domain thesaurus with products from the GSCDA Taxonomy.

The system developed by OTEG offers a user interface with which users can navigate the multi-domain thesaurus, and see EO products related to the chosen thesaurus terms. The system also offers full-text search capabilities on the multi-domain thesaurus and the GCSDA taxonomy using the well-known Lucene text search engine library.[7]

3.2 Relational Database Models for Spatial and Temporal Information

In this section, we survey related work done in the vast research area of relational database models for spatial and temporal information. We distinguish between three different fields: spatial databases, temporal databases and spatiotemporal databases.

3.2.1 Spatial databases

Spatial database systems offer the underlying database technology for geographic information systems and other related applications, and have gained a lot of attention over the years [PR01]. Research in *spatial databases* has concentrated in the areas of data modeling, query languages, data structures and algorithms, and system architectures. In the area of spatial data modeling, researchers have studied possible representations of geographic data in a database. Work on query languages mainly concentrated on designing query languages for spatial DBMS (e.g., spatial extensions of the relational algebra or SQL) and graphical representations for both the input queries and the output results. Work on data structures and algorithms involved the actual implementation of the proposed spatial algebras and the system's query processing architecture. Finally, the area of system architecture dealt with integrating the spatial extensions to a standard DBMS architecture. For a nice survey of spatial database research (as of that time) the interested reader might refer to [Gut94].

[6] `http://earth.eo.esa.int/rtd/Projects/OTEG/index.html`
[7] `http://lucene.apache.org/java/docs/index.html`

Most of the advanced commercial DBMS available today offer support for spatial data. For example, PostGIS[8] is an open source software which extends PostgreSQL with geographic data types, operators and indexes and follows the standards of SQL and OpenGIS implementation specification. PostGIS offers geometry types for points, polygons, multipoints, multilinestrings, multipolygons and geometrycollections, spatial predicates for determining the interactions of geometries, spatial operators for determining geospatial measurements, spatial operators for determining geospatial set operations and R-tree spatial indexes for faster query processing. Oracle Spatial[9] enables users to manage geographic and location-data in a native type within an Oracle database with providing a schema that prescribes the storage, syntax, and semantics of supported geometric data types, a spatial index system, operators, functions, and procedures for performing area-of-interest queries, spatial join queries, and other spatial analysis operations.

Various standards exist for representing and manipulating spatial data in relational systems. The ISO 13249 SQL/MM[10] define a set of types and methods for representing, processing, storing and querying spatial data. The OpenGIS Simple Features Specification for SQL[11] provide a standardized way for storing and accessing spatial data in relational and object-oriented databases. Oracle provides different datatypes for each standard e.g., the `ST_Geometry` datatype is based on the ISO SQL/MM specification while the `BLOB` datatype is used to store spatial data according to the OGC Well-Known Binary representation (fragment of the OGC's Simple Features specification). PostGIS also provides different datatypes for each standard e.g., the `ST_Geometry` datatype is based on the ISO SQL/MM specification while the `Geometry` datatype follows the OpenGIS Simple Features Specification for SQL.

3.2.2 Temporal databases

The field of databases has also seen a lot of work on *temporal databases* including work on foundations, query languages, indexing, optimization, and so on. [SA85] proposed the following classification of DBMS depending on their support for time:

- *Snapshot databases.* All conventional databases belong to this category. The information in these databases corresponds to a snapshot of the world at a particular point in time. Updates are destructive and old information is not retained. The only kind of temporal information supported by such systems is *user-defined time* (e.g., the date of birth of a person can be stored as a string).
- *Rollback databases.* Rollback databases maintain a complete record of the evolution of the database. Intuitively, they can be understood as sequences of snapshot databases indexed by *transaction time* i.e., the time a transaction changing the current database state committed (or the time the system starts believing the new information about the world). In rollback databases, updates are performed with respect to the current state of the database while queries can inquire about past states as well.
- *Historical databases.* Rollback databases keep a record of the changes as registered in the database. Historical databases, on the other hand, keep a record of the changes as they take place in the real world. Intuitively, they can be understood as sequences of snapshot databases indexed by *valid time* i.e., the time a piece of information is true in the real world. In historical databases, the users can pose queries to find out about the world at any point in time. Any errors about the past can be corrected but no record of past database states is kept.

[8] http://postgis.refractions.net/
[9] http://www.oracle.com/technology/products/spatial/index.html
[10] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38651
[11] http://www.opengeospatial.org/standards/sfs/

- *Bitemporal databases.* Bitemporal databases combine the advantages of rollback and historical databases. Intuitively, they are sequences of historical databases indexed by transaction time. The user is allowed to inquire about the world at some point in time according to the information present in the database at some other point in time. The user is also allowed to modify the current state of the database in order to add, delete, or modify historical information about the world. [Sno87] presents some early work on a query language for a bitemporal database management system.

The work on temporal query languages concentrated mostly on the language TSQL2 [AAB+95], a consensus temporal query language developed by a committee of leading temporal database researchers as an extension to SQL-92. TSQL2 could have influenced the SQL3 standard [RTSS98] as discussed in `http://www.cs.arizona.edu/people/rts/sql3.html`, but it does not appear to have influenced implementations of current commercial DBMS. A different approach to introducing time in databases, which is simpler than the TSQL approach, is offered by Date, Darwin and Lorentzos in [DD02]. Some (dated) surveys of work on temporal query languages and temporal database management include [Cho94, JS99].

3.2.3 Spatiotemporal databases

Database researchers have also studied *spatiotemporal databases* which deal with geometries changing over time, e.g., as in the case of *moving objects*. Important research in spatiotemporal databases was carried out in project CHOROCHRONOS whose results have been summarized in [KSF+03]. CHOROCHRONOS achieved results in many subareas of spatiotemporal databases: user interfaces, design methodologies, data models and query languages, query processing algorithms, storage structures and indexes and prototype systems [KSF+03]. One of the highlights of the CHOROCHRONOS work has been work on spatiotemporal data models and query languages. Two competing data models were proposed: one based on abstract data types for moving objects developed by Güting and colleagues [GBE+00] and another based on constraints developed by Grumbach, Koubarakis, Rigaux, Scholl and colleagues [GRS98, RSSG03, GKR+03, KS00]. Both approaches are explained in great depth in the books [GS05, RSV00]. Finally, another interesting approach to a data model and query language for moving objects which also captures uncertainty is presented in [TWHC04]. of

3.3 XML-based Languages for Spatial and Temporal Information

To the best of our knowledge there has not been so far significant work in extending the XML data model and related query languages with spatial or temporal data with the exception of the few papers we survey below. However, since XML is the technology of choice for exchanging information on the Web, there has been significant previous work in *using* XML to represent geospatial features. This work is relevant to this deliverable and it is surveyed here as well.

The Geography Markup Language (GML) [OGC07] is the most common XML-based encoding standard for the representation of geo-information. GML was developed by the OpenGIS Consortium[12] and is based on the OGC Abstract Specification[13], the conceptual foundation for OGC interoperability specifications. GML provides XML schemas for defining a variety of concepts that are of use in geography: geographic features, geometry, coordinate reference systems, topology, time and units of measurement. Initially, the GML abstract model was based on RDF but later the consortium introduced XML schemas in GML structure in order to facilitate the integration of

[12] `http://www.opengeospatial.org/`
[13] `http://www.opengeospatial.org/standards/as/`

existing spatial databases, whose relational schema can be defined more easily with XML Schemas. The GML *profiles* are logical restrictions of GML that might be of use to applications that do not want to use the whole of GML. GML profiles can be expressed through an XML document, an XML schema, or both. Some of the profiles that have been proposed for public use are: (i) Point Profile, (ii) GML Simple Features Profile, (iii) a GML profile for JPG, and (iv) a GML profile for RSS. It should be noted that GML profiles are different from application schemas. The profiles are part of the GML namespaces (Open GIS GML) and define restricted subsets of GML. Applications schemas are XML vocabularies that are application-specific and are valid inside the application-specific namespaces.

GeoRSS [STMD08] is a specification that enables RSS feeds to encode location. GeoRSS-Simple and GeoRSS-GML are two different encodings of GeoRSS. GeoRSS-Simple is a very lightweight format that developers and users can quickly and easily add to their existing feeds with little effort. It supports basic geometries (point, line, box, polygon) and covers the typical use cases when encoding locations. For a more feature-rich option, GeoRSS GML is a formal GML profile, and supports a greater range of features, notably coordinate reference systems other than WGS-84[14] latitude/longitude. Both encodings can be serialized in various flavors of RSS, or RDF, or XHTML.

In [ORS10] the authors present Spatial XPath (SXPath) which is an extension of XPath 1.0 that includes spatial navigation in the query mechanism. The authors extend XPath with spatial axes and spatial position functions. Spatial axes allows the selection of document nodes by taking into account their spatial relationship with the context node. Spatial position functions allows the selection of document nodes by taking into account their spatial position in the plane with respect to the context node. For example, the discovery of visual patterns in a web page can be expressed as a an SXPath query that exploits the relative spatial position of images and text. In [ORS10], the XML document object model is enriched with spatial relationships between minimum bounding rectangles according to the Rectangle Algebra [BCdC99]. The Rectangular Cardinal Relation model [Ren02] is used to express spatial axes that represents directional relations between minimum bounding rectangles. As a result, the SXPath language can express more general queries than pure XPath by exploiting the spatial relations of data items.

Recently there has also been work on temporal XML by database researchers. For example, in [GS03], a temporal query language for XML, called τXQuery, is presented in which the authors add valid time support by minimally extending the syntax and semantics of XQuery. In another relevant paper [RV08], the authors present a data model for tracking historical information in an XML document, a way for summarizing and indexing temporal XML documents, and TXPath, a temporal XML query language that extends XPath 2.0.

3.4 RDF Extensions for Spatial and Temporal Information

Up to now little attention has been paid on extending RDF for spatial and temporal data representation and reasoning. However, there have been more such works than for XML and related research seems to be gaining momentum. In this section, we present the extensions to RDF that have been proposed up to now in order to enable the representation of spatial information, temporal information or both.

3.4.1 Spatial information in RDF

The use of RDF to represent spatial data in the Semantic Web is proposed in [KS07] where the prototype system SPAUK is presented. In SPAUK, geometric attributes of a resource (e.g., location

[14] `http://en.wikipedia.org/wiki/WGS84`

of a gas station) are represented in RDF by introducing a blank node for the geometry, specifying the geometry using GML vocabulary [OGC07], and associating the blank node with the resource using GeoRSS vocabulary [STMD08]. Queries are expressed in the SPARQL query language utilizing appropriate geometric vocabularies and ontologies (e.g., the topological relationships of RCC [CBGG97]). The main assumption of this work is that SPARQL should not be extended with new features for querying spatial data; instead, the existing features of SPARQL together with spatial vocabularies should be utilized.

[KS07] does not specify a semantics for query answering. From the example given in [KS07], we conjecture that such a semantics needs to rely on a model theory (or an axiomatization) which combines the model theory (or axiomatization) of the spatial vocabularies/ontologies used (e.g., RCC) and the RDF model theory (or an equivalent axiomatization). We expect such a semantics to be complicated and will certainly be an extension of the standard semantics for SPARQL query evaluation. Thus, although [KS07] does not change the SPARQL syntax, the semantics of query answering are not the standard ones. This is not said explicitly in [KS07] but it can be deduced from the comments regarding how complicated the spatial query processing part of SPAUK is, and what will happen if the spatial ontologies used are extended in ways that the system does not understand (pages 798-799 of [KS07]).

SPAUK has been implemented by storing RDF triples in Jena and using an in-memory grid file to index the geometries.

Kolas has revisited the problem of defining a Semantic Web data model and query language for spatial data in [Kol08]. This paper assumes the RDF-based spatial data representation of [KS07] and discusses various ways to exploit what is already available in SPARQL to pose queries. The options compared are: (i) to use SPARQL as in [KS07], (ii) to introduce a new PREMISE clause in SPARQL that could be used to introduce spatial geometries that can be used in a query, and (iii) to use some form of the DESCRIBE query form of SPARQL for asking queries about geometries. [Kol08] chooses the last option as the most appropriate. However, it is not clear to us that using something like DESCRIBE which is not given a formal semantics in the SPARQL W3C specification[15] (the answer to a DESCRIBE query depends on the data source; there is no semantics for this kind of query) is a good way to solve the problem of querying spatial data in the Semantic Web.

The only native RDF store that provides some kind of spatial support is AllegroGraph RDFStore[16]. AllegroGraph RDFStore provides a very basic mechanism for the efficient storage and retrieval of geospatial data. It deals with data elements which have coordinates in a two-dimensional region. Support is provided both for Cartesian coordinate systems (i.e., a flat plane) and for spherical coordinate systems (e.g., the surface of the earth or the celestial sphere). Before adding any geospatial data into the store, it is necessary to define the geospatial subtypes the store will use. A geospatial subtype is either spherical or Cartesian, has specific X and Y ranges, and a specific Y strip width. Spatial data and queries are then formed declaratively using AllegroGraph's Prolog facilities (SPARQL currently cannot be used).

GeoSPARQL is a very recent OGC working draft for a query language for geospatial data expressed in RDF [OGC10b]. GeoSPARQL defines a small RDFS ontology that can be used for representing features and geometric objects. In OGC terminology, a feature is an abstraction of real world phenomenon and can have various attributes that describe its thematic and spatial characteristics. The top class of the RDFS ontology proposed by GeoSPARQL is `ogc:AbstractSpatialObject`. This class is a superclass of every geometry or feature that can have a spatial extent. The top class is further specialized into two classes that model features and geometries. The property `ogc:primaryGeometry` is defined for linking a feature with its default geometry, and the property `ogc:hasSerialization` is defined for linking a geometry with its serialization. The serialization of a geometry can be encoded in various formats like GML, KML, WKT etc. In GeoSPARQL,

[15] 2009, from `http://www.w3.org/TR/rdf-sparql-query/`
[16] `http://agraph.franz.com/allegrograph/`

the functions that are defined in the Simple Features Access standard [OGC10d] for accessing geometric objects are mapped to RDF properties like `ogc:dimension`, `ogc:geometryType`, `ogc:srid`, `ogc:isEmpty` and `ogc:is3D`. Additional functions that construct new geometric objects from existing geometric objects like `ogc:buffer` and `ogc:union` are defined and they may be used in the filter part of a query.

Similarly, spatial predicates are also mapped to RDF properties like `ogc:equals`, `ogc:intersects` and `ogc:contains`. An interesting proposal of [OGC10b] is to have triple patterns with such predicates (e.g.,`?x opc:intersects ?y`) instead of allowing these predicates only in the filter part of the query as done in other proposals such as [Per08, KK10]. Given that RDF data can also contain triples using these predicates e.g., `ex:regionA ogc:overlaps ex:regionB` then [Lop10] envision the combination of RDF data and OWL 2 axioms to express more complicated reasoning patterns. For example, one could express the transitivity rule of RCC8 "If region A touches region B and region B is inside region C then region A intersects region C" [Lop10].

The data model sRDF++ and the query language sSPARQL++ that we will present in Chapter 4 of this deliverable are closely related to the language GeoSPARQL. We discuss this relationship in more detail in the concluding section of Chapter 4.

3.4.2 Temporal information in RDF

The first works that proposed to introduce temporal features in RDF were by Gutierrez and colleagues [GHV07, GHV05, HV06]. In their proposal, a framework to incorporate valid time in RDF is introduced. Extending the concept of RDF triple, a *temporal triple* is an RDF triple with an additional temporal label (a natural number). For example, $(s, p, o)[t]$ is a temporal triple which denotes the fact that the triple (s, p, o) is valid at time t. Triples valid at time intervals are then defined by sets of triples valid at time points. Finally, a *temporal RDF graph* is defined as a set of temporal RDF triples. [GHV07, GHV05, HV06] study the semantics of the proposed extension to RDF, define appropriate query languages for the extension and present results on the complexity of query answering.

Another related paper presented lately is the work by Pugliese et. al [PUS08]. In contrast to the work by Gutierrez and colleagues [GHV07, GHV05], this work mainly focuses on the indexing of temporal RDF graphs using the tGRIN index. tGRIN is a specialized index for temporal RDF that is physically stored in an RDBMS. tGRIN can be used to index temporal RDF graphs as defined by Gutierrez et al. in [GHV07, GHV05] but also some case of indeterminate temporal information. The results of this paper show that the tGRIN index is superior in terms of performance than augmenting well-known RDF stores (e.g., Jena, Sesame) with temporal indexes like R+ trees, SR-trees, ST-index, and MAP21.

Available RDF storage and query systems have only recently started to consider ways to introduce temporal RDF in their implementations. The only native RDF store which currently supports some form of temporal RDF is the AllegroGraph RDFStore[16]. AllegroGraph RDFStore supports the storage and retrieval of temporal data including datetimes, time points, and time intervals. Once data has been encoded, applications can perform queries involving a broad range of temporal constraints on data, including relations between points and datetimes, intervals and datetimes, two points, two intervals, and points and intervals. AllegroGraph RDFStore allows users to query temporal RDF graphs using its Prolog querying facilities while support for SPARQL is left for the future.

3.4.3 Spatiotemporal data in RDF

Recently, interesting research on spatial and temporal information in the Semantic Web has been carried out by Amit Sheth's group [Per08, PSHJ07, PHS06, HAmPS06, SP08, ASR+06]. In their more recent work [Per08], an extension of SPARQL, called SPARQL-ST, is defined that allows one to query spatial and non-spatial data with a time dimension. The main idea of [Per08] is to incorporate spatial features in the temporal RDF graphs of [GHV07]. These spatial features are modeled with a spatial ontology based on the GeoRSS GML specification [STMD08]. The main new concept of SPARQL-ST is the introduction of two new types of variables namely spatial variables and temporal variables. Spatial variables (denoted by a % prefix) represent complex spatial features rather than a simple URI, and the concept of SPARQL mappings has been extended to map a spatial variable to a set of triples that represent the required spatial information. Similarly, temporal variables (denoted by a # prefix) are mapped to time intervals and can appear in the fourth position of a temporally extended triple in the style of [GHV07]. Furthermore, in SPARQL-ST two special filters are introduced: `SPATIAL FILTER` and `TEMPORAL FILTER`. These filters are used to filter the results with spatial and temporal constraints (e.g., relations from the RCC calculus [CBGG97] for the spatial part and Allen's interval calculus [All83] for the temporal part). In order to enable the realization of this query language, the used spatial and temporal operators need to be implemented. Both spatial and temporal operators are implemented using Oracle's extensibility framework, while the strictly RDF concepts are implemented using Oracle's RDF storage and inferencing capabilities. For more details on the implementation please see [Per08, PSHJ07].

Yago2 [HSBW10] is a knowledge base where entities, facts, and events are augmented with relevant temporal and spatial information. Yago2 contains approximately 80 millions facts for 10 million entities that are automatically harvested from Wikipedia[17], GeoNames[18], and WordNet[19]. In [HSBW10] the authors present the methodology of harvesting the data, augmenting them with spatial and temporal information and representing the produced knowledge base in SPOTL. SPOTL is a 5-tuple that is defined as a standard RDF triple augmented by time and location. In Yago2, entities are assigned a time span to denote their existence in time, while facts may be assigned a time point or a time span. Spatial information is restricted to latitude/longitude points.

3.5 Geospatial Information in Linked Open Data

Recently, various kinds of geospatial information became available on the Web as linked data. Here we review these efforts since they are closely related to TELEIOS.

In [ALH09] OpenStreetMap (OSM)[20] data is transformed and represented adhering to the RDF data model. A mixed approach in which part of the data is stored in relations and another part is stored according to the RDF data model was followed, due to the billions of triples of OSM data that were difficult to handle by existing triple stores. An ontology was constructed, LinkedGeoData[21], that was derived mainly from OSM tags, i.e. attribute-value annotations to nodes, ways and relations, counting up to 500 classes, 50 object properties and ca. 15,000 datatype properties. The authors employed Triplify [ADL$^+$09] to publish the Linked Data along with a spatial extension to create the neighborhood around a particular point. Points are the type of geometry used by OSM to represent places, cities, etc, defined by their longitude and latitude in the WGS84 coordinate reference system. A circular area is created by using the Haversine formula[22] and a radius that matches the distance between the center of the neighborhood and a point right at the limit of it. Furthermore, an effort was made to match DBpedia resources with LinkedGeoData, taking into account the common classes between the two ontologies, for which the `owl:sameAs` link was used to connect them. Finally, a facet-based browser and editor for linked geographical data[23] was implemented in order to showcase the benefits of revealing the structured information in OSM. The authors conclude that spatial data can be retrieved and interlinked on an unprecedented level of

[17] `http://www.wikipedia.org/`
[18] `http://www.geonames.org/`
[19] `http://wordnet.princeton.edu/`
[20] `http://openstreetmap.org`
[21] `http://linkedgeodata.org/vocabulary`
[22] `http://en.wikipedia.org/wiki/Haversine_formula`
[23] `http://linkedgeodata.org/browser`

granularity and they plan to extend the mapping approach that they currently use for interlinking a knowledge base with other data sources.

[dLSV+10] and [VBVTS+10] describe the process that was followed for the development of an application that makes use of several heterogeneous Spanish public datasets that are related to administrative, hydrographic and statistical domains. In contrast to [ALH09] that just manages every resource as a point (represented by a coordinate of latitude and longitude), Alexander de León et al. deal with these coordinate types and more complex geometry, such as LineStrings. GeoLinked Data[24] is an open initiative whose aim is to enrich the Web of Data with Spanish geospatial data, available from the National Geographic Institute of Spain[25] (IGN-E), and the National Statistic Institute in Spain[26], in order to bring into surface possible relations in the Spanish coastal area and different statistical variables such as unemployment, population, dwelling, industry, and building trade. After identifying the data sources, an ontology was constructed according to the NeOn methodology [SFGP09], by reusing existing ontologies and vocabularies. Statistical Core Vocabulary (SCOVO) [HHR+09] was chosen for describing complex statistics and FAO Geopolitical Ontology[27], hydrOntology [hyd09], GML Ontology[28] and the WSG84 Vocabulary[29] were chosen regarding geospatial vocabulary. Regarding time information and temporal concepts, Time Ontology[30] was also chosen. For generating the RDF data, NOR_2O[31] software library and integrated framework R_2O+ and ODEMapster+[32] were employed. The authors developed a software library, GEOMETRYtoRDF, to create RDF triples from geometrical information in GML[33] and WKT[34] format and as a final step they use Jena[35] to generate the integrated geospatial RDF. Alignment of datasets is carried out by identifying `owl:sameAs` relationships between administrative units and statistical information, in like manner with [ALH09]. Data is published and visualised with the aid of Universal Server[36] and Pubby[37] respectively, and a faceted browser[38] that works on top of those two systems, implemented by the authors. Future work will focus on identifying and interlinking with other knowledge bases belonging to the Linking Open Initiative, mainly DBpedia[39] and GeoNames[40] and coverage of complex geometrical information, like polygons and other geometrical representation types is planned.

[SGD10] discusses how solutions for spatiotemporal data management, specifically Spatial Data Infrastructures (SDI) [Neb04], can be augmented with Linked Data principles and analyses two common scenarios in Linked Data provision and consumption. In scenario one, an agnostic format to codify links instead of RDF is used and in the second scenario the authors elaborate on a complete Linked Data augmentation (with RDF). Focusing on the second scenario, which is the one that actually serves the wider Linked Data community, the realisation of an in-depth integration of Linked Data and SDI is described. OGC identifiers are treated as http URIs[41] and a direct mapping from geospatial data encoded in GML to RDF [SC10] is provided. The basic mappings between GML and RDF used are simple: `xlink:href` is mapped to `rdf:resource` and `gml:identifier` is mapped to `rdf:about`. As metadata is concerned, standards such as ISO 19115[ISO03] and ISO 19119[Per02] provide a core vocabulary that can be exploited. New relation types are defined, if none of the standards relation types fit the requirements, based for instance on the ongoing work of

[24] `http://geo.linkeddata.es`
[25] `http://www.ign.es`
[26] `http://www.ine.es`
[27] `http://www.fao.org/countryprofiles/geoinfo.asp?lang=en`
[28] `http://loki.cae.drexel.edu/~wbs/ontology/2004/09/ogc-gml.owl`
[29] `http://www.w3.org/2003/01/geo/wgs84_pos`
[30] `http://www.w3.org/TR/owl-time/`
[31] `http://code.google.com/p/nor2o/`
[32] `http://neon-toolkit.org/wiki/ODEMapster`
[33] `http://www.opengeospatial.org/standards/gml`
[34] Well-Know Text is a text markup language for representing vector geometry objects on a map, spatial reference systems of spatial objects and transformations between spatial reference systems.
[35] `http://jena.sourceforge.net/`
[36] `http://virtuoso.openlinksw.com/`
[37] `http://virtuoso.openlinksw.com/`
[38] `http://geo.linkeddata.es/browser`
[39] `http://dbpedia.org/`
[40] `http://www.geonames.org/`
[41] `http://portal.opengeospatial.org/files/?artifact_id=39467`

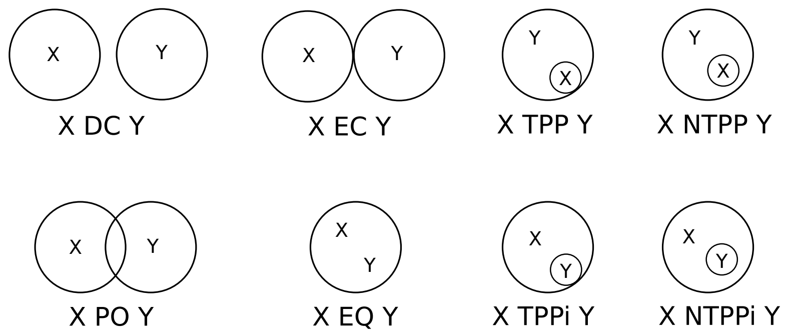


Figure 3.1: The relations of RCC-8.

the NeoGeo Semantic Web Vocabularies Group[42]. Finally the impacts on existing OGC standards and relevance for recent SDI developments are discussed and the authors conclude that only minor changes to current SDI standards are required for implementation, identifying the development of a prototype for a Linked Data augmented SDI followed by a best practice implementation, as actions for future work.

Similarly with [SGD10], [SC10] claims that current SDIs can easily be put on the Linked Data track, by combining GML with the concept of content-negotiation, since Contemporary Linked Data can be directly projected to SDI. The authors use a series of examples in the form of graph and RDF/XML representation to illustrate that GML is easily transformed into a conventional RDF/XML, and native GML is essentially equivalent to RDF. As a consequence, Sven Schade et al. conclude that SDI concepts and the notion of Linked Data do not exclude, but complement each other and linking (geospatial) data is a philosophy of usage and not a technical matter. Thus, the decision of using a GML or RDF representation of geospatial data depends on the intended use.

3.6 Spatial and Temporal Information in Description Logics and OWL

We close our presentation of related work by surveying spatial and temporal extensions of description logics. We survey spatial extensions of DLs and OWL in more detail since they are more relevant to our work in TELEIOS.

3.6.1 Spatial Information in Description Logics and OWL

Research towards representing spatial information in description logics and OWL has focused on the representation of qualitative and quantitative spatial information, and on reasoning with it. To represent qualitative spatial information, relevant works have adopted almost exclusively the Region Connection Calculus (RCC), which is an axiomatization of topological relations in first order logic [RCC92a]. Most of the works deal with a subset of RCC, namely, RCC-8. RCC-8 defines 8 binary spatial relations depicted in Figure 3.6.1. Table 3.1 shows the definitions of the RCC relations in terms of the basic relation C. The relations of RCC-8 are the following: disconnected (DC), externally connected (EC), equal (EQ), partially overlapping (PO), tangential proper part (TPP), tangential proper part inverse (TPPi), non-tangential proper part (NTPP), and non-tangential proper part inverse (NTPPi). These relations capture all possible topological relations between two regions and form a jointly exhaustive and pairwise disjoint (JEPD) set.

[42] `http://sites.google.com/site/neogswvocs/`

$$
\begin{array}{rcl}
\mathbf{DC}(\mathbf{x},\mathbf{y}) & \equiv_{def} & \neg C(x,y) \\
P(x,y) & \equiv_{def} & \forall z[C(z,x) \rightarrow C(z,y)] \\
PP(x,y) & \equiv_{def} & P(x,y) \wedge \neg P(y,x) \\
\mathbf{EQ}(\mathbf{x},\mathbf{y}) & \equiv_{def} & P(x,y) \wedge P(y,x) \\
O(x,y) & \equiv_{def} & \exists z[P(z,x) \wedge P(z,y)] \\
\mathbf{PO}(\mathbf{x},\mathbf{y}) & \equiv_{def} & O(x,y) \wedge \neg P(x,y) \wedge \neg P(y,x) \\
DR(x,y) & \equiv_{def} & \neg O(x,y) \\
\mathbf{TPP}(\mathbf{x},\mathbf{y}) & \equiv_{def} & PP(x,y) \wedge \exists z[EC(z,x) \wedge EC(z,y)] \\
\mathbf{EC}(\mathbf{x},\mathbf{y}) & \equiv_{def} & C(x,y) \wedge \neg O(x,y) \\
\mathbf{NTPP}(x,y) & \equiv_{def} & PP(x,y) \wedge \neg \exists z[EC(z,x) \wedge EC(z,y)] \\
Pi(x,y) & \equiv_{def} & P(y,x) \\
PPi(x,y) & \equiv_{def} & PP(y,x) \\
\mathbf{TPPi}(\mathbf{x},\mathbf{y}) & \equiv_{def} & TPP(y,x) \\
\mathbf{NTPPi}(\mathbf{x},\mathbf{y}) & \equiv_{def} & NTPP(y,x)
\end{array}
$$

Table 3.1: Definition of the various relations of RCC. Relations in bold are included in RCC-8.

To represent quantitative spatial information, relevant works have studied spatial objects with a known geometry (e.g., polygons represented using linear constraints over the domain of rationals, or reals). Depending on the kind of spatial information represented (qualitative or quantitative), different reasoning techniques have been employed.

In the following, we survey most of the relevant works existing in the literature. We distinguish five categories of works: those which are based on description logics with concrete domains, those which axiomatize RCC using TBox axioms, those which solve the relevant representation and reasoning problems at the level of the architecture, those which employ the Semantic Web Rule Language (SWRL) to deliver reasoning capabilities, and last, those which model geospatial information using a specific geospatial ontology.

Concrete Domains Approaches The approach of modeling with concrete domains is a nice way of integrating knowledge of specific domains in a description logic, because the reasoning task for each domain is casted in its respective theory. In the case of spatial information, the idea is to introduce a concrete domain (e.g., real numbers or regions of a topological space) to model space together with appropriate concepts, roles, and features [HLM99, HM02, LM07, CG09]. Work here has concentrated mostly on issues of semantics and reasoning in these logics.

One of the first works in this category is [HLM99] in which the description logic $\mathcal{ALCRP}(\mathcal{D})$ is introduced which supports concrete domains and a role-forming predicate operator. $\mathcal{ALCRP}(\mathcal{D})$, which extends $\mathcal{ALC}(\mathcal{D})$, can effectively be used for reasoning about spatial objects and their qualitative spatial relationships, providing also an appropriate concrete domain for spatial objects. The significance of $\mathcal{ALCRP}(\mathcal{D})$ lies also on its ability for augmenting spatial and terminological reasoning with temporal reasoning in a combined concrete domain. The authors define a concrete domain for polygons ($\mathcal{D}_p$) and a role-forming operator which allows the definition of roles with very complex properties and provides a strong coupling of roles with concrete domains. They also provide a criterion for structurally restricting the form of terminological axioms in order to guarantee decidability for the inference problem of checking satisfiability of $\mathcal{ALRP}(\mathcal{D})$.

In [LM07] the authors identify a general property for concrete domains, called ω-admissibility, that is sufficient for proving decidability of description logics equipped with concrete domains. For defining ω-admissibility they concentrate on a particular kind of concrete domains, *constraint systems*. Such a concrete domain is one that has only binary predicates, which are interpreted as JEPD relations. They consider two examples of constraint systems and prove that are ω-admissible: a temporal one based on the Allen relations [All83], and a spatial one based on the real plane and the RCC-8 relations. Further, in [LM07] a tableau algorithm for description logics with concrete domains and general TBoxes is presented, which establishes a general result about the decidability of $\mathcal{ALC}$ extended with any ω-admissible concrete domain and general TBoxes ($\mathcal{ALC}(\mathcal{C})$). As a consequence, $\mathcal{ALC}(\mathcal{C})$ when extended with the Allen relations or the RCC-8 relations is decidable. Inspired by the approach of [LM07], [CG09] considers the Cardinal Direction Calculus [Fra91] and introduces the corresponding description logic $\mathcal{ALC}(\mathcal{CDC})$.

The tableau algorithm employed in [LM07] is similar to the one that had been implemented and incorporated into the Racer reasoner [HMW01, HM02]. Racer and its most recent version RacerPro[43] supports TBox and ABox reasoning for the description logic $\mathcal{ALCQHI}_{\mathcal{R}^+}(\mathcal{D})^-$ using a default concrete domain for linear inequalities. In particular, Racer's standard concrete domain supports reasoning for linear inequations between rational numbers and interval reasoning (min/max) for integers. Apart from providing standard reasoning functionalities (subsumption, instance retrieval, instance realization, consistency checking, etc.) common to most description logic systems, Racer provides also an expressive query language called nRQL [HMW04] and a publish-subscribe mechanism for subscribing instance retrieval queries and be notified when "relevant" instance changes happen.

The query language nRQL [HMW04] allows for the formulation of conjunctive queries and it can be used to query the ABox and TBox of a description logic, as well as RDF(S) and OWL documents. From the perspective of concrete domains, it supports binary concrete domains predicates, so for example queries like "retrieve all pairs of persons having the same age"[44] are easily expressible. Moreover, it supports negation as failure (NAF) and the projection operator `project-to`, which can be used in combination to express closed world universal quantification.

[Wes02] is concerned with several extensions of $\mathcal{ALC}$, which are suitable for qualitative spatial reasoning and in particular are appropriate for capturing the RCC composition tables. Such extensions should support inverse and disjoint roles as well as composition-based role inclusion axioms of the form $S \circ T \sqsubseteq R_1 \sqcup R_2 \sqcup \cdots \sqcup R_n$. The author groups such extensions under the family of description logic $\mathcal{ALCI}_{\mathcal{RCC}}$, which is undecidable for arbitrary forms of role boxes. [Wes02] shows that inverse relationships are necessary, since some spatial inferences cannot be drawn otherwise. Finally, it studies the decidability of different RCC subsets, like RCC-1, RCC-2, RCC-3, RCC-5, and RCC-8. He concludes that $\mathcal{ALCI}_{RCC1}$ and $\mathcal{ALCI}_{RCC2}$ are decidable, $\mathcal{ALCI}_{RCC3}$ is decidable under the weak equality semantics, while $\mathcal{ALCI}_{RCC5}$ and $\mathcal{ALCI}_{RCC8}$ are not decidable, but can be made decidable if, for example, universal quantification is allowed only for propositional concepts.

Axiomatic Modeling of RCC Relations in TBox In [KG05] the authors suggest a translation of the RCC-8 calculus into OWL-DL by adapting some of the known results on the translation of qualitative spatial formalisms into modal logics. In particular, it is a known result that the description logic $\mathcal{S}$ ($\mathcal{ALC}$ extended with transitive roles), augmented with a reflexive accessibility relation, yields the modal logic $S4$, in which it is possible to translate the RCC-8 calculus and some of its extensions. Moreover, every $S4$ formula can be represented in OWL-DL provided that OWL-DL is extended with the ability to define reflexive roles. Indeed, OWL-DL, which is based on the logic $\mathcal{SROIQ}$, supports reflexivity.

Regarding representation, the authors of [KG05] express regions as concepts. Such a region (concept) must follow the *regularity condition*: it must be a non-empty, closed set (i.e., it must contain at least one individual and contain all of its interior points). To translate the RCC-8 calculus into OWL-DL the authors introduce a TBox axiom for each one of the RCC-8 relations. This, together with the fact that some of the RCC-8 axioms require additional TBox axioms specifying the non-emptiness of some regions, lead to an undesired explosion of the TBox size [HFK10].

The authors of [SS09] are led to the same conclusion, that is, the impracticality of the approach in [KG05]. They implement the PelletSpatial reasoner, which provides RCC-8 and RDF/OWL

[43] User guide (v.1.9.2): `http://www.racer-systems.com/products/racerpro/users-guide-1-9-2-beta.pdf`
Reference Manual (v.1.9.2): `http://www.racer-systems.com/products/racerpro/reference-manual-1-9-2-beta.pdf`

[44] This query can be expressed in nRQL using the following constraint query atom: $(?x\ ?y\ (\mathit{constraint\ age\ age} =))$

reasoning and querying capabilities. The authors chose to implement two approaches for spatial reasoning, the one of which is the approach mentioned in [KG05]. They show that this approach is impractical even for small datasets, which is due to the enforcement of the regularity condition.

Modeling at the Architecture Level The approach of [GBM07a, GBM07b] takes a different direction in comparison with the aforementioned ones. Their approach, which is motivated by the observation that RCC cannot be expressed in OWL without a major revision of the latter, deals with spatial reasoning at the level of the system architecture and not at the level of formalisms: they propose a hybrid knowledge representation system architecture which integrates terminological and spatial aspects of the application domain and provides support for reasoning with RCC and OWL. This implies that the architecture of a knowledge representation system based on description logics is extended with RCC specific components. In particular, they separate spatial from application domain information, introducing a RCCBox (similar to the role box, RBox, in $\mathcal{SROIQ}$), in which spatial relations and composition tables are specified. In terms of modeling, the authors interpret a region as a polygon in the integral plane, and thus avoid interpreting regions as sets, which arise problems when combining RCC with description logics that are related to the type separation requirement of OWL-DL. The authors also note that their hybrid approach does not work well with OWL-DL, because OWL-DL, as opposed to OWL 2, does not support negation of roles, which are needed to express connectivity of two regions. The expressivity of their formalism is that of $\mathcal{ALHI}$ [GMH08].

The same direction is taken by PelletSpatial [SS09], a hybrid RCC-8 reasoner based on the Pellet RDF/OWL-DL reasoner [PS04, SPG$^+$07]. In [SS09] the spatial relations are separated from the RDF/OWL-DL relations providing a hybrid reasoner for both spatial and thematic data. Hence, the set spatial relations is disjoint from the set of RDF/OWL-DL relations. Spatial relations are managed as an RCC constraint network that provides functionality to check its consistency (employing a path-consistency algorithm that uses the RCC-8 composition table) and querying. In such a setting, conjunctive query answering requires two phases: first, evaluating spatial query atoms over the constraint network, and second further constraining the set of bindings, such that the non-spatial query atoms are satisfied. Experiments showed that the implementation is non-practical for real datasets, but exposes reasonable performance for small ones [SS09].

A more complete and formal approach to modeling spatial information at the level of architecture is that of [WM09]. In [WM09] the authors present a software framework architecture for ontology-based information systems and use a case study dealing with spatial data to demonstrate its flexibility. For achieving flexibility and extensibility, the authors propose an abstracted graph-based data model and query language, the *substrate data model* and *substrate query language*, with which any subset of first-order predicate logic (FOPL), e.g., modal logic, description logic, propositional logic, first-order logic, etc., can be associated. Especially for the case of spatial information, a substrate can play the role of a geometric substrate, called SBox (Space Box). The SBox deals with spatial datatypes (e.g., polygons) the geometry of which can be described using an appropriate FOPL, inheriting also its formal semantics for satisfiability, entailment, etc. The authors investigate four options for representing and querying spatial information: use (i) an ABox, (ii) a map substrate, (iii) a spatial ABox, (iv) an ABox and RCC substrate.

Using SWRL Another approach is that of [Bis08] that proposes to encode spatial inferences in the Semantic Web Rule Language (SWRL) [HPSB$^+$04]. SWRL uses Horn-like rules which are combined with OWL-DL and OWL-Lite. Again, this approach is limited, because of the fact that Horn rules do not allow disjunctive heads, which are required in order to formalize the RCC composition axioms. The same approach is followed in [BP10]. The authors propose SOWL, an extension of OWL, to represent spatial qualitative and quantitative information employing the RCC-8 topological relations, the eight direction relations — North (N), North East (NE), East (E), South East (SE), South (S), South West (SW), West (W), and North West (NW) —, and distance relations (e.g., 3Km away from city A) stored in the ontology as n-ary relations.

Further, they use a Location object for spatial objects (e.g., region), which can be optionally connected with a footprint class with subclasses: Point, Line, Polyline, and MBR. To reason about spatial relations they introduce a set of SWRL rules, which can be implemented in the Pellet reasoner [PS04, SPG$^+$07] and additional spatial relations can be inferred from existing ones using composition tables which are defined both for topological and direction spatial relations.

Geospatial Ontologies and Geospatial Semantic Web Finally, there have been recently various papers presenting the vision of a Semantic Web enriched with geospatial information, called the Geospatial Semantic Web [Ege02, BLOR05, ST07]. In [KDH05], the authors propose several kinds of geospatial ontologies that could be used for the Geospatial Semantic Web, while in a most recent paper [MGVO08], a system is presented which provides a spatio-temporal ontology modeling and semantic query environment compatible with OWL-DL. [SAEJ07] et al. present a framework for the representation of geo-ontologies and reasoning over them using OWL. Spatial reasoning and integrity rules are represented using a spatial rule engine extension to the reasoning tools associated with OWL.

3.6.2 Temporal Information in Description Logics and OWL

The main idea here is the same with modeling spatial information, i.e., time is modeled using a concrete domain approach together with appropriate concepts, roles and features [LM07]. Work here has concentrated mostly on issues of semantics and reasoning in these logics. See for example the papers [AT08, Lut04] for recent work in this area, and [AF01] for a somewhat outdated survey of the area.

Recently, works on temporal ontologies using OWL have also started to appear. [HP04] presents OWL-Time[45], an ontology of temporal concepts designed for representing temporal knowledge on the Web. This ontology provides a vocabulary for expressing knowledge about qualitative relations among instants and intervals, information about durations, and information about dates and times. Another relevant effort in this area is the language TOWL [MFKN07] which is an extension of OWL with a concrete interval domain and appropriate classes and properties for the representation of interval knowledge in the Web .

3.7 Summary

In this chapter, we presented a survey of related work in data models and query languages in the areas most relevant to this deliverable. As our work will be based on RDF, and due to the particular requirements of the project, we focused more on RDF extensions that are currently the state-of-the-art regarding temporal and spatial information.

[45] `http://www.w3.org/TR/owl-time/`

4. An extension of RDF and SPARQL for the representation of Geospatial Data Using OGC standards

This and the following chapter present the core of our new, developments presented in this deliverable. In this chapter we present sRDF++ and sSPARQL++, which are new versions of the data model sRDF and the query language sSPARQL that we have developed in the context of the EU project SemsorGrid4Env [KK10]. In sRDF++ and sSPARQL++ we use Open Geospatial Consortium (OGC)[1] standards for the representation of geospatial data (instead of just linear constraints). In the new version of sRDF and sSPARQL, we opt for a more practical solution to the problem of representing geospatial data. This need arose clearly in the project SemsorGrid4Env where the main use case had data that were expressed in WKT. To be able to deal with this real user requirement, we took two steps:

(i) we enabled our implemented system Strabon to read sRDF with objects that are geometries expressed in WKT [KKK10].

(ii) We added syntactic constructs to sSPARQL so that queries could involve geometries expressed with WKT [KKKK09],

The clear definition of sRDF++ and sSPARQL++ that we undertake in this deliverable is the next logical step given our earlier Semsorgrid4Env work. Our current approach resembles OGC proposals regarding the use of SQL to query geospatial data [OGC10d] as they have been adopted by commercial RDBMs e.g., PostGIS[2], MySQL[3] or Oracle Spatial[4]. In this way, we expect that sRDF++ and sSPARQL++ will be adopted more easily by applications such as the ones represented in TELEIOS where there is a long tradition of using geospatial representations that are closely related to OGC standards such as Well Known Text (WKT).

Our work on sSPARQL has commonalities with the recent OGC work on GeoSPARQL, and, in fact, the design of sSPARQL has also been inspired by the presentation [Lop10] which was kindly made available to us by Xavier Lopez of ORACLE who leads the relevant OGC working group after he attended the presentation of [KK10] at ESWC 2010 (of course, the main inspiration has been our independent work in SemsorGrid4Env reported above). NKUA has recently become a member of OGC with the intention of participating in the work on GeoSPARQL and disseminating relevant TELEIOS results to the OGC community and vice versa.

The organization of this chapter is the following. In Section 4.1 we present some background information about the OGC standards that we will bring into sRDF++. In Section 4.2 we will present sRDF++ which is the new version of the data model sRDF that uses these OGC standards. In Section 4.3 we present sSPARQL++ which is the new version of the query language sSPARQL for querying sRDF++ graphs.

4.1 Background on OGC Standards

In this section we will present two OGC standards that can be used for encoding geographic features. In Section 4.1.1 we will present the Well-Known Text OGC standard that can be used

[1] Open Geospatial Consortium, `http://www.opengeospatial.org/`.
[2] `http://postgis.refractions.net/`
[3] `http://www.mysql.com/`
[4] `http://www.oracle.com/technetwork/database/options/spatial/index.html`

for representing geometric objects, spatial reference systems and transformations between spatial reference systems. In Section 4.1.2 we will present various spatial reference systems that are used to relate the coordinates of a geometric object to real locations on the surface of Earth. In Section 4.1.3 we will present the Geography Markup Language which is a more recent standard defined by OGC for representing geographical features in XML.

4.1.1 Well Known Text

WKT is a widely accepted OGC standard for representing spatial objects. WKT can be used for representing geometric objects, spatial reference systems and transformations between spatial reference systems. WKT is described in the OpenGIS Simple Feature Access specification [OGC10e]. This standard establishes a common architecture for representing and accessing simple features. In OGC terminology, a *feature* is an abstraction of real world phenomena and can have various attributes that describe its thematic and spatial characteristics (e.g., a feature can represent a parcel of land with its use and geographic location). A *simple feature* is a feature with all geometric attributes described piecewise by straight line or planar interpolation between sets of points.

Geometries in WKT are restricted to 0-, 1- and 2- dimensional geometric objects that exist in $\mathbb{R}^2$, $\mathbb{R}^3$ or $\mathbb{R}^4$. Geometries that exist in $\mathbb{R}^2$ consist of points with coordinates x and y e.g., `POINT(1,2)`. Geometries that exist in $\mathbb{R}^3$ consist of points with coordinates x, y and z or x, y and m where m is a measurement. For example, the following point represents the temperature of the city of Athens that was measured in Celcius degrees: `POINT(37.96, 23.71, 27)`. Geometries that exist in $\mathbb{R}^4$ have points with coordinates x, y, z and m e.g., `POINT(1,1,2,27)`.

Geometries represented using WKT have the following properties:

- All geometries are topologically closed which means that all the points that comprise the boundary of the geometry are assumed to belong to the geometry, even though they may not be explicitly represented in the geometric object.
- All coordinates within a geometry object are in the same coordinate reference system.
- For geometric objecs that exist in $\mathbb{R}^3$ and $\mathbb{R}^4$, spatial operations work in the "map geometry" of the spatial data. The "map geometry" of a geometric object is the projection of the geometric object on $\mathbb{R}^2$. Therefore, the z and m values are not reflected in calculations (e.g., equals, intersects, touches) or in generation of new geometry values (e.g., buffer, minimum bounding box). However, the WKT specifications provide functions to access the z and m coordinates.

For the convenience of the reader we will present the part of the WKT standard that defines how to represent vector geometries. The WKT representation of geometric objects will be incorporated in the sRDF++ data model. In Figure 4.1 we present the class hierarchy for simple feature geometry as defined in [OGC10e].

The top Geometry class has subclasses for Point, Curve, Surface and Geometry Collection. The Geometry Collection is further specialized to classes of 0-, 1- and 2- dimensional geometric objects named MultiPoint, MultiLineString and MultiPolygon respectively. Each geometric object is linked to a specific Spatial Reference System and optionally to a Measure Reference System. A Measure Reference System may be used to interpret the third or fourth dimension of geometric objects that exist in $\mathbb{R}^3$ or $\mathbb{R}^4$. For example, a hotspot can be modeled as a point that has x, y and m coordinates, where the coordinates x and y are used to represent the location of the hotspot, while the m coordinate represents its reliability.

Let us provide some more information for each class. The WKT specification defines the following classes that may be instantiated:

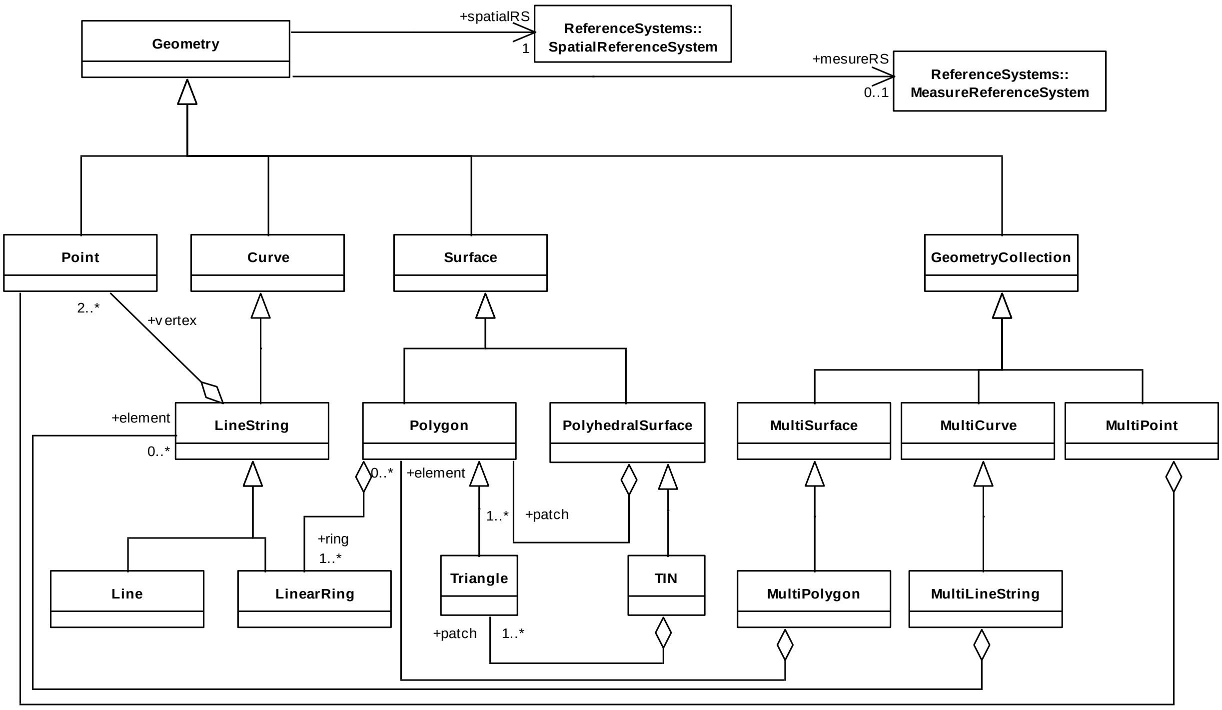


Figure 4.1: The classes of geometric objects in WKT

- **Point**. A Point represents a single location in coordinate space. A Point has x and y coordinate values and may have z and m depending on the associated spatial reference system.
- **Curve**. A Curve is a 1-dimensional geometric object. The subtypes of a Curve define the type of interpolation that is used between points.
- **LineString**. A LineString is a subtype of Curve that uses linear interpolation between points. A LineString is closed if its start point is equal to its end point. A LineString is simple if it has no self-intersections.
- **Line**. A Line is a LineString with exactly two points.
- **LinearRing**. A LinearRing is a LineString that is both closed and simple.
- **Surface**. A Surface is a 2-dimensional geometric object. This geometry is abstract (i.e., it may not be instantiated). A simple surface may consist of a single “patch” that has one “exterior” boundary and 0 or more “interior” boundaries (i.e. a polygon with holes).
- **Polygon**. A Polygon is a simple Surface that is planar. It has exactly one exterior boundary and may have several non-intersecting internal boundaries. Each Polygon is topologically closed and no two boundaries (interior or exterior) cross. However, two boundaries may intersect at a Point, but only as a tangent. The interior of a Polygon is a connected point-set while the exterior of a Polygon with holes is not connected.
- **Triangle**. A Triangle is a Polygon with 3 distinct, non-collinear vertices and no “interior” boundary.
- **Polyhedral Surface**. A Polyhedral Surface is a contiguous collection of Polygons, which share common boundary segments. Each pair of polygons that touch, have a common boundary that is expressed as a finite collection of LineStrings. Each such Linestring is a part of the boundary of at most 2 Polygon patches.

- **Triangulated Irregular Network (TIN)**. A TIN is a Polyhedral Surface consisting only of Triangle patches.
- **Geometry Collection**. A Geometry Collection is a geometric object that is a collection of some number of geometric objects.
- **MultiPoint**. This is a Geometry Collection whose elements are Points that are not connected.
- **MultiCurve**. A MultiCurve is a Geometry Collection whose elements are Curves.
- **MultiLineString**. A MultiLineString is a Geometry Collection whose elements are LineStrings.
- **MultiSurface**. A MultiSurface is a 2-d Geometry Collection whose elements are Surfaces. The geometric interiors of any two Surfaces may not intersect. The boundaries of any two surfaces may not cross but may touch at a finite number of Points.
- **MultiPolygon**. A MultiPolygon is a Multi Surface Collection whose elements are Polygons. The boundaries of each Polygon may not intersect.

The interpretation of the coordinates of a geometry depends on the coordinate reference system that is associated with the geometry. In Section 4.1.2 we will discuss about coordinate reference systems. Note that according to the WKT standard, the coordinate reference system that is associated to a geometric object is never embedded in the object's representation.

The syntax of the WKT representation of a geometric object is presented in [OGC10e]. Examples of geometric objects represented in WKT are shown in Table 4.1.

4.1.2 Coordinate reference systems

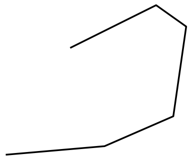
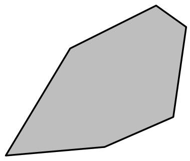
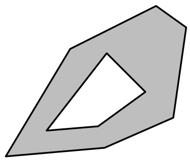
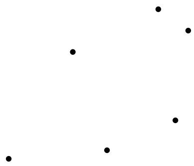
The interpretation of the coordinates of a geometry depends on the coordinate reference system that is associated with the geometry. A *coordinate reference system* defines how to relate the coordinates of a geometric object to real locations on the surface of Earth. A coordinate reference system is a coordinate system that is related to the real world by a datum. A *coordinate system* is a set of mathematical rules for specifying how coordinates are to be assigned to each point. A *datum* is defined as any quantity or set of such quantities that may serve as a referent or basis for calculation of other quantities[5]. For example, a geodetic datum is a set of reference points against which position measurements are made while a geocentric datum is a geodetic datum in which the Earth's center of mass is involved in specifying the coordinate system. In order to define a geodetic datum, one needs to define the origin and orientation of the coordinate system and the dimensions of the reference ellipsoid. In geodesy, a *reference ellipsoid* is a mathematically-defined surface that approximates the irregular shape of the Earth. The planar sections of an ellipsoid are either ellipses or circles.

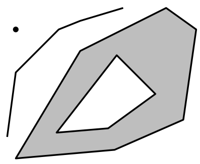
In the literature, a coordinate reference system is also referred to as a *spatial reference system*. Various types of spatial reference systems exist. For example, planar systems have x- and y-axes (or even z if they are three-dimensional), terrestrial systems utilize longitude and latitude coordinates and polar systems utilize polar coordinates. The WKT standard defines a syntax for representing three types of spatial reference systems: geographic (latitude-longitude), geocentric (X, Y, Z) and projected (X, Y).

A *geographic coordinate reference system*, is a three-dimensional system that utilizes latitude, longitude and optionally ellipsoidal height. In a geographic coordinate reference system, the earth's three-dimensional ellipsoid is mapped using a series of horizontal (longitude lines or parallels)

[5] Geodetic Glossary, `http://www.ngs.noaa.gov/CORS-Proxy/Glossary/xml/NGS_Glossary.xml`.

Table 4.1: Examples of geometric objects represented in WKT

Geometry type	**WKT representation**	**Geometric object**
Point	`Point(5 5)`	
LineString	`LineString(5 5,28 7,44 14,47 35,40 40,20 30)`	
Polygon	`Polygon((5 5,28 7,44 14,47 35,40 40,20 30,5 5))`	
Polygon	`Polygon((5 5,28 7,44 14,47 35,40 40,20 30,5 5),` `(28 29,14.5 11,26.5 12,37.5 20,28 29))`	
MultiPoint	`MultiPoint((5 5),(28 7),(44 14),(47 35),(40 40),(20 30))`	

Geometry Collection	`GeometryCollection(`
	`Point(5 35),`
	`LineString(3 10,5 25,15 35,20 37,30 40),`
	`Polygon((5 5,28 7,44 14,47 35,40 40,20 30,5 5),`
	`(28 29,14.5 11,26.5 12,37.5 20,28 29))`
	`)`
		

and vertical (latitude lines or meridians) reference lines. The geographic coordinate reference system uses geodetic longitude and latitude to form the geographic coordinates and additionally it provides a quantity for geodetic height (elevation) that is defined as the distance from the reference ellipsoid in a direction normal to the ellipsoid. The Word Geodetic System (WGS) is a well-known geographic coordinate reference system. The latest revision is WGS 84 and it is the reference coordinate system used by the Global Positioning System. The coordinate origin for WGS 84 is the mass center of the Earth and the WGS 84 datum surface is a pole-flattened spheroid.

A *geocentric coordinate reference system* is a three-dimensional cartesian system. Geocentric coordinate reference systems have a specific point of origin that is the common intersection point for all planes. For example, the earth's mass center is the origin point for the "Earth-center, earth-fixed" (ECEF) coordinate system. The x axis of the ECEF coordinate reference system passes through the equator at the prime meridian, the z axis passes through the north pole and the y axis passes through the equator at 90^o longitude.

A *projected coordinate reference system*, is a system that transforms a three-dimensional world into a two-dimensional system. The reference surface is usually ellipsoid but a sphere can be used for small-scale mapping. Projection coordinate systems are always associated with a geographic coordinate system. A projection always makes compromises, since it either preserves one projection property or makes a compromise between projection properties. The following are projections properties: angles (preserve angles and shapes), areas (preserve the relative size of regions), distance (partially preserve distance relationships), direction (preserve certain lines of direction). It is important to notice that every projection creates distortions and that geographic information from different projections should not be combined directly.

Two examples of projected coordinate reference systems that are used in the NOA use case, are the Greek Geodetic Reference System 1987 and the Universal Transverse Mercator. The Greek Geodetic Reference System 1987 (GGRS87) is a projection system commonly used in Greece. GGRS87 uses a local geographic reference system and a fine-tuned transverse Mercator cartographic projection. The local coordinate reference system is the Geodetic Reference System 1980 (GRS80) that is non-geocentric and consists of a global reference ellipsoid and a gravity field model. The origin of the GGRS87 is slightly different from the center of the GRS80 datum, so that the ellipsoid is optimal for Greece. The Universal Transverse Mercator (UTM) is a well known global map projection. UTM uses the WGS 84 ellipsoid as the underlying geographic coordinate system. UTM is not a single projection system. Instead, it is set upon a zoned grid which divides the earth into sixty zones of equal width. Each zone uses a fine-tuned transverse Mercator projection that is capable of mapping a region of large north-south extent with a low amount of distortion.

Various authorities may provide predefined spatial reference systems. The European Petroleum Survey Group (EPSG) is an authority that provides a huge collection of spatial reference systems. Each reference system is assigned a unique identifier, and this identifier is used in applications to reference a specific spatial reference system. For example, if a spatial dataset is encoded according to the spatial reference system with identification number 4326 or by the urn identifier `urn:epsg:4326`, we can assume that every coordinate in the dataset is encoded according to the WGS84 geographic coordinate reference system.

4.1.3 Geography Markup Language

The Geography Markup Language (GML) [OGC07] is an OGC standard that defines an XML grammar written in XML Schema for modeling, exchanging and storing geographic information. GML defines a grammar for representing features, spatial reference systems, geometric objects, topologies, time and units of measurement. The GML standard defines a large number of XML elements and attributes in order to allow the encoding of features, spatial and temporal topologies, complex geometric property types and coverages. As a result, various profiles like the Point Profile, the Simple Features Profile, the GMJP2 Profile and the Profile for RSS are defined. These profiles deal only with a subset of GML and are intended to simplify the adoption of GML.

The GML Simple Features specification (GML-SF) [OGC10a] and the Simple Features for SQL standard [OGC10e] that we will present in Section 4.3.1 have similar structure and describe similar geometries. However, GML Simple Features can have geometries in three dimensions while Simple Features for SQL can have geometries with only two dimensions. In GML-SF, a feature can have any number of geometric properties, and every geometry should be referenced to a spatial reference system that has 1, 2 or 3 dimensions. Additionally, GML-SF defines a richer mechanism for representing feature metadata.

The base type for every geometric object that can be represented in GML is `gml:AbstractGeometryType`. All geometry elements are derived from this abstract datatype. A geometry should have an identifier (defined by an element with name `gml:id`), may have one or more names (defined by the elements `gml:name` or `gml:identifier`), may have a description (defined by the elements `gml:description` and `gml:descriptionReference`) and may be associated with a spatial reference system (defined by the `gml:SRSReferenceGroup` attribute group). The GML standard defines that all direct positions should be directly or indirectly associated with a spatial reference system. This means that when a geometric object is aggregated in another geometric object (for example the `gml:MultiPoint` contains `gml:Point` elements) that is associated with a spatial reference system, then the aggregated elements are assumed to be associated with the same spatial reference system unless otherwise specified. Additionally, a `gml:AbstractGeometry` element is declared. Any geometry element is directly or indirectly in the substitution group of `gml:AbstractGeometry`.

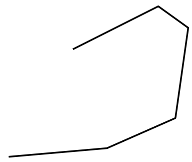
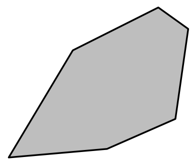
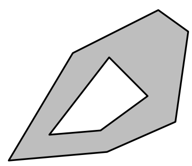
The GML Simple Features Profile supports the following geometry property types that are defined in the GML standard with some additional restrictions: `gml:PointPropertyType`, `gml:CurvePropertyType`, `gml:SurfacePropertyType`, `gml:GeometryPropertyType`, `gml:MultiPointPropertyType`, `gml:MultiCurvePropertyType`, `gml:MultiSurfacePropertyType`, `gml:MultiGeometryPropertyType`. These property types reference the following geometric objects:

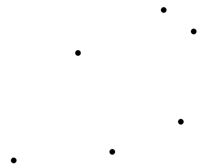
- **Point**. A `gml:Point` element represents a point and is defined by a single coordinate tuple (a list of coordinates).
- **LineString**. The element `gml:LineString` is a curve that consists of a single segment defined by two or more coordinate tuples with linear interpolation between them.
- **Curve with LineString segments**. A `gml:Curve` with `gml:LineStringSegment` segments is a continuous and connected curve that can be composed of one or more `gml:LineStringSegment` segments. A `gml:LineStringSegment` is a curve segment that is similar to a `gml:LineString`.
- **Polygon**. The `gml:Polygon` is a surface that consists of a single surface patch that has a co-planar boundary and uses planar interpolation in its interior.
- **Surface with Polygon patches**. A `gml:Surface` with `gml:PolygonPatch` patches is composed of one or more `gml:PolygonPatch` patches that are connected to one another. A `gml:PolygonPatch` is a surface patch that is similar to a `gml:Polygon`.
- **MultiPoint**. The element `gml:MultiPoint` is a geometric aggregate and consists of one or more `gml:Point` elements.
- **MultiCurve**. The element `gml:MultiCurve` is a geometric aggregate and consists of one or more `gml:AbstractCurve` elements. The GML-SF restricts the supported members to `gml:LineString` and `gml:Curve` with `gml:LineStringSegment` segments.
- **MultiSurface**. The element `gml:MultiSurface` is a geometric aggregate and consists of one or more `gml:AbstractSurface` elements. The GML-SF restricts the supported members to `gml:Polygon` and `gml:Surface` with `gml:PolygonPatch` patches.

- **MultiGeometry**. This property is a geometric aggregate. The GML-SF restricts the supported elements to `gml:Point`, `gml:LineString`, `gml:Curve`, `gml:Polygon`, `gml:Surface`, `gml:MultiPoint`, `gml:MultiCurve` and `gml:MultiSurface`.

The syntax of the GML representation of a geometric object is presented in [OGC07]. Examples of geometric objects represented in GML are shown in Table 4.2.

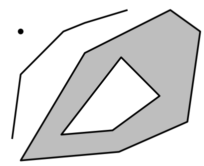
Table 4.2: Examples of geometric objects represented in GML.

<table>
<tr><th>Geometry type</th><th>GML representation</th><th>Geometric object</th></tr>
<tr><td>Point</td><td><pre>
<gml:Point gml:id="p1"
     srsName="urn:ogc:def:crs:EPSG:6.6:4326">
   <gml:coordinates>5, 5</gml:coordinates>
</gml:Point>
</pre></td><td></td></tr>
<tr><td>LineString</td><td><pre>
<gml:LineString gml:id="p2"
     srsName="urn:ogc:def:crs:EPSG:6.6:4326">
    <gml:coordinates>
             5,5 28,7 44,14 47,35 40,40 20,30
    </gml:coordinates>
</gml:LineString >
</pre></td><td></td></tr>
<tr><td>Polygon</td><td><pre>
<gml:Polygon gml:id="p3"
     srsName="urn:ogc:def:crs:EPSG:6.6:4326">
  <gml:exterior>
    <gml:LinearRing>
      <gml:coordinates>
        5,5 28,7 44,14 47,35 40,40 20,30 5,5
      </gml:coordinates>
    </gml:LinearRing>
  </gml:exterior>
</gml:Polygon>
</pre></td><td></td></tr>
<tr><td>Polygon</td><td><pre>
<gml:Polygon gml:id="p4"
     srsName="urn:ogc:def:crs:EPSG:6.6:4326">
  <gml:exterior>
    <gml:LinearRing>
      <gml:coordinates>
        5,5 28,7 44,14 47,35 40,40 20,30 5,5
      </gml:coordinates>
    </gml:LinearRing>
  </gml:exterior>
  <gml:interior>
    <gml:LinearRing>
      <gml:coordinates>
        28,29 14.5,11 26.5,12 37.5,20 28,29
      </gml:coordinates>
    </gml:LinearRing>
  </gml:interior>
</gml:Polygon>
</pre></td><td></td></tr>
</table>

MultiPoint	<gml:MultiPoint gml:id="p5" srsName="urn:ogc:def:crs:EPSG:6.6:4326"> <gml:pointMember> <gml:Point> <gml:coordinates>5,5</gml:coordinates> </gml:Point> </gml:pointMember> <gml:pointMember> <gml:Point> <gml:coordinates>28,7</gml:coordinates> </gml:Point> </gml:pointMember> <gml:pointMember> <gml:Point> <gml:coordinates>44,14</gml:coordinates> </gml:Point> </gml:pointMember> <gml:pointMember> <gml:Point> <gml:coordinates>47,35</gml:coordinates> </gml:Point> </gml:pointMember> <gml:pointMember> <gml:Point> <gml:coordinates>40,40</gml:coordinates> </gml:Point> </gml:pointMember> <gml:pointMember> <gml:Point> <gml:coordinates>20,30</gml:coordinates> </gml:Point> </gml:pointMember> </gml:MultiPoint>	

Geometry Collection

```
<gml:MultiGeometry gml:id="p6"
     srsName="urn:ogc:def:crs:EPSG:6.6:4326">
  <gml:geometryMember>
    <gml:Point>
      <gml:coordinates>5,35</gml:coordinates>
    </gml:Point>
  </gml:geometryMember>
  <gml:geometryMember>
    <gml:LineString>
      <gml:coordinates>
        3,10 5,25 15,35 20,37 30,40
      </gml:coordinates>
    </gml:LineString >
  </gml:geometryMember>
  <gml:geometryMember>
    <gml:Polygon>
      <gml:exterior>
        <gml:LinearRing>
          <gml:coordinates>
            5,5 28,7 44,14 47,35 40,40 20,30 5,5
          </gml:coordinates>
        </gml:LinearRing>
      </gml:exterior>
      <gml:interior>
        <gml:LinearRing>
          <gml:coordinates>
            28,29 14.5,11 26.5,12 37.5,20 28,29
          </gml:coordinates>
        </gml:LinearRing>
      </gml:interior>
    </gml:Polygon>
  </gml:geometryMember>
</gml:MultiGeometry>
```



4.2 The data model sRDF++

sRDF [KK10] is a constraint data model that extends RDF with the ability to represent spatial and temporal data using linear constraints. In sRDF, we followed the main ideas of constraint databases [KKR90, RSV00, Rev02, Kou94d] and we represented spatial objects as quantifier-free formulas in the first-order theory of linear inequalities over the rational numbers. In this section we present a new version of sRDF that uses OGC standards to represent geospatial information[6].

In sRDF++ we introduce the new data type `srdf:geometry` for modelling geometric objects. The values of this datatype are typed literals that encode geometric objects using the WKT format. Literals of the datatype `srdf:geometry` will often be called *spatial literals*.

[6] sRDF++ can actually be defined as an extension of sRDF defined in [KK10] by keeping linear constraints as an alternative abstract version of representing spatial geometries. But since we do not plan to explore this representation at all in TELEIOS, we no longer discuss this issue here.

In RDF datatypes are defined as in XML[7]. A datatype consists of a value space, a lexical space and a lexical-to-value mapping and is identified by one or more URI references. The *value space* is the set of values for a given datatype and is denoted by one or more literals in its lexical space. The *lexical space* of a datatype is a set of strings and the *lexical-to-value mapping* of a datatype is a set of pairs whose first element belongs to the lexical space of the datatype, and the second element belongs to the value space of the datatype.

The new datatype `srdf:geometry` is defined as follows. The value space of the datatype `srdf:geometry` is the geometry values defined in [OGC10e] and [OGC10a], that is the power-set of $\mathbb{R}^2$, $\mathbb{R}^3$ and $\mathbb{R}^4$. The lexical space of the datatype `srdf:geometry` consists of finite-length sequences of characters that can be produced from the WKT representation for geometry grammar defined in [OGC10e], accompanied by a semi-colon and a URI that identifies the corresponding spatial reference system. Additionally, the lexical space contains the finite-length sequences of characters that can be produced from that subset of the GML grammar that produces the geometric objects defined in [OGC10a]. The lexical-to-value mapping for the datatype `srdf:geometry` maps elements of the lexical space to elements of the value space in the obvious way by taking into account that the vector-based model is used for representing geometries.

Example 1. *The following are RDF triples that utilize the new datatype.*

```
ex:hotspot1 rdf:type noa:Hotspot .
ex:hotspot1 noa:hasReliability "100"^^xsd:decimal .
ex:hotspot1 noa:hasLocation
            "POINT(669062 4238286);urn:epsg:ggrs87"^^srdf:geometry .
```

The above triples define a hotspot, its reliability and its 2-dimensional geometry. The latter is a typed literal of the new data type `srdf:geometry` *and is encoded in WKT. Notice that a urn is used to denote that the coordinates that give the point of the hotspot are projected to the Greek Geodetic Reference System 1987 (GGRS87).*

Example 2. *The following triples are from Example 1. But now we use GML to represent the hotspot's location.*

```
ex:hotspot1 rdf:type noa:Hotspot .
ex:hotspot1 noa:hasReliability "100"^^xsd:decimal .
ex:hotspot1 noa:hasLocation
                          """<gml:Point gml:id="p1" srsName="urn:epsg:ggrs87">
                               <gml:coordinates>669062 4238286</gml:coordinates>
                             </gml:Point>"""^^srdf:geometry .
```

sRDF++ imposes very minimal new requirements to Semantic Web developers that want to use our approach: all they have to do is utilize a new literal datatype. The new literal datatype can be used together with spatial ontologies expressed in RDFS to give the same kind of class-based modeling capabilities offered by other approaches like [Per08].

Representing time in sRDF++

Database researchers have differentiated among user-defined time, valid time and transaction time (see Section 3.2.2). The data model stRDF [KK10] allows the representation of the *valid time* of a triple (i.e., the time that the triple was valid in reality) using the approach of Gutierrez et al.

[7] Resource Description Framework (RDF): Concepts and Abstract Syntax, `http://www.w3.org/TR/rdf-concepts/#section-Datatypes`.

[GHV07] where a fourth component is added to each triple. According to our current understanding of the NOA and DLR use cases, only the concept of user-defined time is required.

RDF (and therefore sRDF++) supports user-defined time since triples are allowed to have as objects literals of the following XML Schema datatypes: `xsd:dateTime`, `xsd:time`, `xsd:date`, `xsd:gYearMonth`, `xsd:gYear`, `xsd:gMonthDay`, `xsd:gDay`, `xsd:gMonth`.

Example 3. *The following triples are from Example 1. But now we use an additional triple to represent the time that the hostspot was detected.*

```
ex:hotspot1 rdf:type noa:Hotspot .
ex:hotspot1 noa:hasReliability "100"^^xsd:decimal .
ex:hotspot1 noa:hasLocation
            "POINT(669062 4238286);urn:epsg:ggrs87"^^srdf:geometry  .
ex:hotspot1 noa:hasDetectionTime "2009-08-17T17:30:01"^^xsd:dateTime .
```

4.3 The query language sSPARQL++

In this section we will present the query language sSPARQL++ that is a new version of sSPARQL to query sRDF++ graphs. First, we will present the OpenGIS Simple Features Access for SQL specification that is a well-established OGC standard for managing sets of simple features using SQL. Then we will present the query language sSPARQL++ by means of examples.

4.3.1 OpenGIS Simple Features Access

The OpenGIS Simple Features Access standard [OGC10d] (also known as ISO 19125) defines a standard SQL schema that supports storage, retrieval, query and update of collections of simple features using SQL. Simple features have both spatial and non-spatial attributes. The spatial attributes are geometries of the types described in Section 4.1.1. Sets of simple features are stored as relational tables and each feature is a row in the table. The spatial attributes of the features are represented as geometry-valued columns, while non-spatial attributes are represented as columns whose types are the standard SQL data types. The standard describes schemas for two types of feature tables implementations: implementations using only the SQL predefined data types and SQL with Geometry types.

The former approach is based on the classical SQL relational model and geometric objects are stored in a geometry-valued column that is implemented using a “geometry id” reference into a geometry table. A geometry table is a relational table where geometric objects are stored (possibly by using more than one rows) and may be implemented using standard SQL types or SQL binary types. When using predefined types, a geometry will be stored as an array of coordinate values using predefined SQL numeric types. When using SQL binary types, the well known binary representation of a geometric object will be used for storing geometries.

The latter approach uses the types presented earlier in Section 4.1.1 to define new geometric data types for SQL. In both cases, additional tables are used to store information about features and spatial reference systems. However, the standard does not provide SQL functions for accessing geometries in an implementation based on predefined SQL data types. As a result, we focus on the implementation based on SQL with geometry types, as the standard defines a simple geometric profile for the definition of geometric attributes, and describes a set of SQL Geometry Types together with SQL functions on those types.

The SQL Geometry Types are the types we presented earlier in Section 4.1.1. In this section we will present the functions that are defined in the OpenGIS Simple Features Access standard for all Geometry Types. These functions have been widely adopted by the GIS community and our aim is to incorporate them in sSPARQL++ so that sRDF++ graphs will be queried using a user-friendly language in a similar way that SQL is used to query geographic features in spatially enabled relational databases.

The following functions have been defined for requesting the desired representation of a geometry, checking whether some condition holds for a geometry and returning some properties of the geometry:

1. `ST_Dimension(A:Geometry):Integer`, returns the inherent dimension of the geometric object `A`, which must be less than or equal to the coordinate dimension.
2. `ST_GeometryType(A:Geometry):String`, returns the name of the instantiable subtype of Geometry as defined in [OGC10e], of which the geometric object `A` is an instantiable member.
3. `ST_AsText(A:Geometry):String`, exports the geometric object `A` to a specific WKT Representation of Geometry.
4. `ST_AsBinary(A:Geometry):Binary`, exports the geometric object `A` to a specific Well-known Binary Representation of Geometry.
5. `ST_SRID(A:Geometry):Integer`, returns the Spatial Reference System ID for the geometric object `A`.
6. `ST_IsEmpty(A:Geometry):Boolean`, returns true if the geometric object `A` is the empty Geometry. Otherwise, it returns false.
7. `ST_IsSimple(A:Geometry):Boolean`, returns true if the geometric object `A` has no anomalous geometric points, such as self intersection or self tangency. Otherwise, it returns false.

The following functions have been defined for testing named spatial relationships between two geometric objects.

1. `ST_Equals(A:Geometry, B:Geometry):Boolean`, returns true if the geometric object `A` is "spatially equal" to the geometric object `B`. Otherwise it returns false.
2. `ST_Disjoint(A:Geometry, B:Geometry):Boolean`, returns true if the geometric object `A` is "spatially disjoint" to the geometric object `B`. Otherwise it returns false.
3. `ST_Intersects(A:Geometry, B:Geometry):Boolean`, returns true if the geometric object `A` "spatially intersects" the geometric object `B`. Otherwise it returns false.
4. `ST_Touches(A:Geometry, B:Geometry):Boolean`, returns true if the geometric object `A` "spatially touches" the geometric object `B`. Otherwise it returns false.
5. `ST_Crosses(A:Geometry, B:Geometry):Boolean`, returns true if the geometric object `A` "spatially crosses" the geometric object `B`. Otherwise it returns false.
6. `ST_Within(A:Geometry, B:Geometry):Boolean`, returns true if the geometric object `A` is "spatially within" to the geometric object `B`. Otherwise it returns false.
7. `ST_Contains(A:Geometry, B:Geometry):Boolean`, returns true if the geometric object `A` "spatially contains" the geometric object `B`. Otherwise it returns false.
8. `ST_Overlaps(A:Geometry, B:Geometry):Boolean`, returns true if the geometric object `A` "spatially overlaps" the geometric object `B`. Otherwise it returns false.

9. `ST_Relate(A:Geometry, B:Geometry, intersectionPatternMatrix: String):Boolean`, returns true if the geometric object `A` is "spatially related" to the geometric object `B` by testing for intersections between the interior, boundary and exterior of the two geometric objects as specified by the values in the `intersectionPatternMatrix`.

The following functions have been defined for constructing new geometric objects from existing geometric objects.

1. `ST_Boundary(A:Geometry):Geometry`, returns a geometric object that is the boundary of the geometric object `A`.
2. `ST_Envelope(A:Geometry):Geometry`, returns a geometric object that is the minimum bounding box for the input Geometry `A`.
3. `ST_Intersection(A:Geometry, B:Geometry):Geometry`, returns a geometric object that represents the Point set intersection of the geometric objects `A` and `B`.
4. `ST_Union(A:Geometry, B:Geometry):Geometry`, returns a geometric object that represents the Point set union of the geometric objects `A` and `B`.
5. `ST_Difference(A:Geometry, B:Geometry):Geometry`, returns a geometric object that represents the Point set difference of the geometric objects `A` and `B`.
6. `ST_SymDifference(A:Geometry, B:Geometry):Geometry`, returns a geometric object that represents the Point set symmetric difference of the geometric objects `A` and `B`.
7. `ST_ConvexHull(A:Geometry):Geometry`, returns a geometric object that represents the convex hull of the geometric object `A`.
8. `ST_Buffer(A:Geometry, distance:Double):Geometry`, returns a geometric object that represents all Points whose distance from the geometric object `A` is less than or equal to `distance`. Calculations are in the spatial reference system of this geometric object.

The `ST_Distance(A:Geometry, B: Geometry):Double` function is defined for calculating the shortest distance between two geometric objects. This operation is calculated in the spatial reference system of the geometric objects.

4.3.2 Design choices

Let us now explain how we have chosen to introduce the above functions into sSPARQL++. Our approach is to define a URI for each of the functions defined in the OpenGIS Simple Feature Access standard that were described in the previous section and use them in SPARQL queries. For example, we assign the URI `srdf:Contains` to the function `ST_Contains`.

In sSPARQL++ we allow these functions to be used either in the select or the filter part of a query. The SPARQL W3C standard allows the usage of functions inside the filter part of a query. These functions can be either built-in functions like `isIRI` or extension functions[8]. Extension functions are named by an IRI, can have various RDF terms as arguments and return an RDF term as a result. For example, in order to use the `ST_Contains` function that is not part of the core SPARQL specification, we introduce the function:

```
xsd:boolean srdf:Contains(srdf:geometry g1, srdf:geometry g2)
```

[8] SPARQL Query Language for RDF, `http://www.w3.org/TR/rdf-sparql-query/#extensionFunctions`.

that take as arguments two spatial literals `g1` and `g2`, checks whether `g1` "spatially contains" `g2` and returns the appropriate result encoded as an `xsd:boolean` typed literal.

Naturally, the arguments of the functions can be variables, spatial literals encoded in WKT or GML or other extension functions. For example, to check whether the variable `?GEO` is bound to a spatial literal that contains the point `"POINT(669062 4238286);urn:epsg:ggrs87"^^srdf:geometry`, we could use the following expression inside the filter clause of a standard SPARQL query:

```
srdf:Contains(?GEO, "POINT(669062 4238286);urn:epsg:ggrs87"^^srdf:geometry)
```

As we mentioned earlier, in sSPARQL++ we allow spatial functions to be used in the select part of a SPARQL query as well. The current specification of SPARQL 1.0 does not provide this functionality. However, the current working draft on SPARQL 1.1, define that select expressions[9] can be either variables, built-in functions, extension functions, RDF literals, aggregate functions or complex expressions. As a result, new spatial literals can be generated on the fly during query time based on pre-existing spatial literals. For example, to obtain the buffer of a spatial literal that is bound to the variable `?GEO`, we would use the following select expression:

```
select srdf:Buffer(?GEO) as ?GEOBUFFER
```

In sSPARQL++ we also define a function that transforms the coordinates of a geometry to another spatial reference system. The function `srdf:Transform` takes as an argument a spatial literal and a URI and returns a new spatial literal with its coordinates transformed to the spatial reference system referenced by the URI parameter:

```
srdf:geometry srdf:Transform(srdf:geometry g1, URI srid)
```

We also define the following functions that transform the representation of a geometric object:

- `ST_AsText(A:Geometry):String`, returns the Well-Known Text (WKT) representation of the geometric object `A`.
- `ST_AsBinary(A:Geometry):String`, returns the Well-Known Binary (WKB) representation of the geometric object `A`.
- `ST_AsGML(A:Geometry):String`, returns the Geographic Markup Language (GML) representation of the geometric object `A`.

4.3.3 sSPARQL++ by examples

Let us now consider examples of sSPARQL++. We will use some queries that we presented earlier in Chapter 2 where we discussed the semantic challenges for the NOA use case, but in this case we will present in detail the new features introduced by sSPARQL++.

We will consider a dataset from the NOA use case that is similar to the examples presented in Section 2.2. We will consider the bounding box of the coastline of Greece, land cover information about Greece based on the Corine Land Cover dataset, FMM-1 products that describe hotspots and FFM-2 products that describe burned areas. For the sake of the example, we will use very simple geometries. The sRDF++ description of two hotspots that have a location in space, a reliability factor and the time that they were detected are the following (`noa` is the namespace of the vocabulary presented in Chapter 2 and `ex` is the namespace of an example namespace):

[9] SPARQL 1.1 Query Language, `http://www.w3.org/TR/sparql11-query/#select_expressions`.

```
ex:Hotspot1 rdf:type noa:Hotspot .
ex:Hotspot1 noa:hasLocation "POINT(38.17 23.73);urn:epsg:wgs84"^^srdf:geometry.
ex:Hotspot1 noa:hasReliability "40.0000"^^xsd:double .
ex:Hotspot1 noa:hasDetectionTime "2009-08-17T17:45:00"^^xsd:dateTime .

ex:Hotspot2 rdf:type noa:Hotspot .
ex:Hotspot2 noa:hasLocation "POINT(38.18 23.77);urn:epsg:wgs84"^^srdf:geometry.
ex:Hotspot2 noa:hasReliability "70.0000"^^xsd:double .
ex:Hotspot2 noa:hasDetectionTime "2009-08-17T17:45:00"^^xsd:dateTime .

ex:Hotspot3 rdf:type noa:Hotspot .
ex:Hotspot3 noa:hasLocation "POINT(38.24 23.75);urn:epsg:wgs84"^^srdf:geometry.
ex:Hotspot3 noa:hasReliability "20.0000"^^xsd:double .
ex:Hotspot3 noa:hasDetectionTime "2009-08-17T17:45:00"^^xsd:dateTime .
```

Notice the last triples in the descriptions of `Hotspot1` and `Hotspot2` that capture temporal information.

The sRDF++ description of an individual burned area is the following:

```
ex:BurntArea1 rdf:type noa:BurntArea .
ex:BurntArea1 noa:hasGeometry
              "POLYGON((38.17 23.72,38.19 23.72,38.19 23.78,38.19 23.72,
                        38.17 23.72));urn:epsg:wgs84"^^srdf:geometry .
```

The sRDF++ description of the minimum bound box of the coastline of Greece is the following:

```
ex:coastlinembb rdf:type noa:GeographicBound .
ex:coastlinembb ex:refersToCountry dbpedia:Greece .
ex:coastlinembb ex:hasGeometry
                "POLYGON((33.23 18.27,33.23 29.97,41.77 29.97,41.77 18.27,
                          33.23 18.27));urn:epsg:wgs84"^^srdf:geometry .
```

We assume a very simple class hierarchy that describes forests according to the Corine Land Cover nomenclature. For this example, we have forests, that are specialized to broad-leaved forests, coniferous forests and mixed forests:

```
clc:Forest            rdf:type rdfs:Class .
clc:BroadLeavedForest rdf:type rdfs:Class .
clc:ConiferousForest  rdf:type rdfs:Class .
clc:MixedForest       rdf:type rdfs:Class .

clc:BroadLeavedForest rdfs:subClassOf clc:Forest .
clc:ConiferousForest  rdfs:subClassOf clc:Forest .
clc:MixedForest       rdfs:subClassOf clc:Forest .
```

The sRDF++ description of three individual forests according to the Corine Land Cover dataset is the following:

```
ex:ForestArea1 rdf:type noa:Area .
ex:ForestArea1 noa:hasGeometry
               "POLYGON((38.16 23.7,38.18 23.7,38.18 23.8,38.16 23.8,
```

```
                     38.16 23.7));urn:epsg:wgs84"^^srdf:geometry .
ex:ForestArea1 noa:hasCorineLandCover clc:ConiferousForest .

ex:ForestArea2 rdf:type noa:Area .
ex:ForestArea2 noa:hasGeometry
             "POLYGON((38.18 23.7,38.19 23.7,38.19 23.8,38.18 23.8,
                     38.18 23.7));urn:epsg:wgs84"^^srdf:geometry .
ex:ForestArea2 noa:hasCorineLandCover clc:MixedForest .

ex:ForestArea3 rdf:type noa:Area .
ex:ForestArea3 noa:hasGeometry
             "POLYGON((38.19 23.7,38.21 23.7,38.21 23.8,38.19 23.8,
                     38.19 23.7));urn:epsg:wgs84"^^srdf:geometry .
ex:ForestArea3 noa:hasCorineLandCover clc:MixedForest .
```

Example 4. Spatial join. *Find the URIs of the burned areas that are located in Greece.*

```
select ?BA
where {?BA rdf:type noa:BurntArea .
       ?BA noa:hasGeometry ?BAGEO .
       ?C  rdf:type noa:GeographicBound .
       ?C  ex:refersToCountry dbpedia:Greece .
       ?C  ex:hasGeometry ?CGEO .
       filter(srdf:Contains(?CGEO, ?BAGEO)) }
```

Query result. The result of this query is displayed below.

?BA
ex:BurntArea1

Let us now explain in some detail the features of sSPARQL++ by referring to the above example. As in standard SPARQL, variables can be used in basic graph patterns to refer to spatial literals denoting point sets (e.g., variable `?BAGEO` in `?BA noa:hasGeometry ?BAGEO`). These variables can also be used in the functions we introduced in Section 4.3.2. These functions can be used in filter expressions to compare *spatial terms* using spatial predicates. Spatial terms include spatial literals (e.g., `"POINT(0 0)"^^srdf:geometry`), spatial variables and complex spatial terms (e.g. `srdf:Intersection(?GEO,POLYGON((0 0,1 0,1 1,0 1,0 0))"^^srdf:geometry)` which denotes the intersection of the value of spatial variable `?GEO` and the polygon `"POLYGON((0 0, 1 0, 1 1, 0 1, 0 0))"^^srdf:geometry`). There are several types of spatial predicates such as topological, distance, directional, etc. that one could introduce in a user-friendly spatial query language. In the current version of sSPARQL++ only the topological relations of [CCR93, EF91] that were presented earlier in Section 4.3.1 can be used as predicates in a spatial filter expression, e.g., `filter(srdf:Inside(?GEO1, ?GEO2))`.

Example 5. Query with OPTIONAL and spatial function application. *Find the URIs and the geometries of all forests. Optionally, if a forest was burned in the past, return the geometry of its burned part as well.*

```
select ?F srdf:AsText(?FGEO) AS ?FORESTGEO
       srdf:AsText(srdf:Intersection(?FGEO,?BAGEO)) as ?BURNEDFOREST
where {?F rdf:type noa:Area .
       ?F noa:hasCorineLandCover ?FLC .
       ?FLC rdfs:subClassOf clc:Forest .
       ?F noa:hasGeometry ?FGEO .
       optional {?BA rdf:type noa:BurntArea .
                 ?BA noa:hasGeometry ?BAGEO .
                 filter(srdf:Overlaps(?BAGEO, ?FGEO)) } }
```

Query result. *The result of this query is displayed below.*

?F	?FORESTGEO	?BURNEDFOREST
`ex:ForestArea1`	`"POLYGON((38.16 23.7, 38.18 23.7,38.18 23.8, 38.16 23.8,38.16 23.7)); urn:epsg:wgs84"^^srdf:geometry`	`"POLYGON((38.17 23.72, 38.18 23.72,38.18 23.78, 38.17 23.78,38.17 23.72)); urn:epsg:wgs84"^^srdf:geometry`
`ex:ForestArea2`	`"POLYGON((38.18 23.7, 38.19 23.7,38.19 23.8, 38.18 23.8,38.18 23.7)); urn:epsg:wgs84"^^srdf:geometry`	`"POLYGON((38.18 23.72, 38.19 23.72,38.19 23.78, 38.18 23.78,38.18 23.72)); urn:epsg:wgs84"^^srdf:geometry`
`ex:ForestArea3`	`"POLYGON((38.19 23.7, 38.21 23.7,38.21 23.8, 38.19 23.8,38.19 23.7)); urn:epsg:wgs84"^^srdf:geometry`	

The above is an example of an optional graph pattern in sSPARQL++. As in SPARQL, optional graph patterns allow information to be added to what has already been collected if this information is available, instead of rejecting solutions if some part of a triple pattern does not match. If an optional graph pattern matches a graph, new bindings are defined and added to one or more solutions. If an optional graph pattern does not match a graph, the solution is left unchanged. Unlike SQL left outer joins, optional graph patterns in SPARQL may be complex graph patterns with many joins, may contain new variables or references to variables previously declared and may contain multiple `filter` expressions.

In the select clause of an sSPARQL++ query we allow expressions like `srdf:Intersection(GEO1, GEO2)` or `srdf:Union(GEO1, GEO2)` where `GEO1`, `GEO2` are spatial variables, spatial literals or other extension functions. These expressions compute new spatial objects, e.g., the intersection or the union of two regions respectively.

The previous query first finds forests and their geometries. If there is a burned area with a geometry that overlaps a forest's geometry, then the intersection of these two geometries will be added to the solution. In this example, only two forests were (partially) burned (`ex:ForestArea1` and `ex:ForestArea2`), so the answer variable `?BURNEDFOREST` is not bound for `ex:ForestArea3`.

Example 6. Spatial function application. *Find the URIs of the forests that were burned and compute the burned area in square meters. Use the GGRS87 spatial reference system for projecting the geometries.*

```
select ?F srdf:Area(srdf:Transform(srdf:Intersection(?FGEO,?BAGEO)),
                                   "urn:epsg:ggrs87") as ?BURNEDAREA
where {?F rdf:type noa:Area .
       ?F noa:hasCorineLandCover ?FLC .
       ?FLC rdfs:subClassOf clc:Forest .
       ?F noa:hasGeometry ?FGEO
       ?BA rdf:type noa:BurntArea .
       ?BA noa:hasGeometry ?BAGEO .
       filter(srdf:Overlaps(?BAGEO, ?FGEO)) }
```

Query result. *The result of this query is displayed below.*

?F	?BURNEDAREA
ex:ForestArea1	"7129357"^^xsd:double
ex:ForestArea2	"7129882"^^xsd:double

In the `select` clause of an sSPARQL query we allow arbitrary spatial terms such as `srdf:Area(srdf:Transformsrdf:Intersection(?FGEO,?BAGEO)), ‘‘urn:epsg:ggrs87’’` above. This expression returns the area of the spatial term that is computed after projecting the intersection of the geometries `?FGEO` and `?BAGEO` to the GGRS87 spatail reference system. Since the units of measurment for the projection GGR87 are meters, the result is expressed in square meters. The function `srdf:Transform` is used for projecting a geometric object to another spatial reference system. The function `srdf:Transform` takes as an argument a spatial literal and a urn and returns a new spatial literal with its coordinates transformed to the spatial reference system referenced by the urn parameter.

Example 7. Spatial join and temporal selection. *Find the hot spots that were detected during the 17th of August 2009 in Greece and are in either broad-leaved or coniferous forests.*

```
select ?HS
where {?C rdf:type noa:GeographicBound .
       ?C ex:refersToCountry dbpedia:Greece .
       ?C ex:hasGeometry ?CGEO .
       ?HS rdf:type noa:Hotspot .
       ?HS noa:hasLocation ?HSGEO .
       ?HS noa:hasDetectionTime ?T .
       filter (srdf:Contains(?CGEO, ?HSGEO)) .
       ?F rdf:type noa:Area .
       ?F noa:hasGeometry ?FGEO
       optional(?F noa:hasCorineLandCover ?FLC1 .
                ?FLC1 rdf:type clc:BroadLeavedForest) .
       optional(?F noa:hasCorineLandCover ?FLC2 .
                ?FLC2 rdf:type clc:ConiferousForest) .
       filter ((bound(?FLC1) || bound(?FLC2)) && srdf:Contains(?FGEO, ?HSGEO)).
       filter ( ?T >= "2009-08-17T00:00:00"^^xsd:dateTime &&
                ?T <= "2009-08-17T23:59:59"^^xsd:dateTime)}
```

Query result. *The result of this query is displayed below.*

?HS
ex:Hotspot1

In the above example, we ask for the minimum bounding box of the coastline of Greece and its geometry. Then, we ask for hotspots that were detected in the past and are located inside Greece. Afterwards we ask for forests that are either broad-leaved or coniferous and contain the detected hotspots. Finally, we do a temporal selection where we ask that the hotspot was detected during August 17, 2009. Notice that the result contains neither `ex:Hotspot2` that is located at a mixed forest nor `ex:Hotspot3` that is located outside the minimum bounding box of the coastline of Greece.

Example 8. Spatial join and spatial function application. *Select the URIs of the hotspots that have a geometry that is considered to be the center of a square whose sides have length 1 km and is located in the sea. For all calculations, project the geometries to the GGRS87 spatial reference system.*

```
select ?HS
where {?HS rdf:type noa:Hotspot .
       ?HS noa:hasLocation ?HSGEO .
       ?C rdf:type noa:Coastline .
       ?C ex:refersToCountry dbpedia:Greece .
       ?C ex:hasGeometry ?CGEO .
       filter(srdf:Overlaps(srdf:Transform(?CGEO,"urn:epsg:ggrs87"),
                             srdf:Envelope(srdf:Buffer(srdf:Transform(
                                            ?HSGEO,"urn:epsg:ggrs87"), 500))))}
```

Query result. *The result of this query is displayed below.*

?HS
ex:Hotspot3

This query uses the `srdf:Buffer` spatial operation which involves a spatial object and a distance. The result of this spatial operation is a new geometry that contains all the points at the specified maximum distance along the spatial object that was given as an input. The distance is expressed in the units of measurement of the spatial reference system that is associated to the geometries. We project the geometries to the GGRS87 spatial reference system that is an appropriate projection system for the area of interest, and we calculate the buffer of the hotspot's location using as a distance 500 meters. The `srdf:Envelope` spatial operation has as a result a new geometry, that is the minimum bounding box of the input geometry.

Example 9. Spatial join. *Select URIs of the hotspots that lies into a burnt area and do not have reliability 100.*

```
select ?HS
where {?HS rdf:type noa:Hotspot .
       ?HS noa:hasLocation ?HSGEO .
       ?HS noa:hasReliability ?REL .
       ?BA rdf:type noa:BurntArea .
       ?BA noa:hasGeometry ?BAGEO .
       filter(srdf:Contains(?BAGEO, ?HSGEO) &&
              ?REL <> "100"^^xsd:double)}
```

Query result. *The result of this query is displayed below.*

?HS
ex:Hotspot1
ex:Hotspot2

This query uses the `srdf:Contains` spatial operation to select the hotspots that are located inside a burned area. Afterwards, a standard SPARQL filter is used to select the hotspots that are not 'certain' based on their reliability.

We will now consider the dataset from the DLR use case that was presented in Section 2.1. We will consider three TerraSAR-X images, and one geographic region. For simplicity, for each image we represent the start and end date of its acquisition, the geographic coverage of the image , and a unique identifier. The geographic region represents a region in Germany along with its land cover. Notice that this sample dataset is very simplistic as many properties are omitted and a cartesian coordinate system is used for encoding the coordinates of the geometric objects. We use the prefix `ex` for the namespace of an example ontology.

```
clc:Forest rdf:type rdfs:Class .
clc:ConiferousForest rdf:type rdfs:Class .
clc:ConiferousForest rdfs:subClassOf clc:Forest .

ex:image1
        a dlr:TSX1;
        dlr:hasStartDate "2011-01-10T05:03:07"^^xsd:dateTime;
        dlr:hasEndDate "2011-01-10T05:03:09"^^xsd:dateTime;
        dlr:hasGeometry
            "POLYGON((2 3,3 3,3 2,2 2,2 3));urn:cartesian"^^srdf:geometry;
        dlr:Id "1"^^xsd:decimal .

ex:image2
        a dlr:TSX1;
        dlr:hasStartDate "2011-01-17T16:48:58"^^xsd:dateTime;
        dlr:hasEndDate "2011-01-17T16:49:00"^^xsd:dateTime;
        dlr:hasGeometry
            "POLYGON((1 3,2 3,2 2,1 2,1 3));urn:cartesian"^^srdf:geometry;
        dlr:Id "2"^^xsd:decimal .

ex:image3
        a dlr:TSX1;
        dlr:hasStartDate "2011-01-20T05:17:00"^^xsd:dateTime;
        dlr:hasEndDate "2011-01-20T05:17:01"^^xsd:dateTime;
        dlr:hasGeometry
            "POLYGON((1 2,2 2,2 1,1 1,1 2));urn:cartesian"^^srdf:geometry;
        dlr:Id "3"^^xsd:decimal .

ex:Region1
        a dlr:Region;
        dlr:hasCorineLandCoverUse clc:ConiferousForest;
        dlr:hasGeometry
            "POLYGON((1 2.5,1 4,3 4,3 2.5,12.5));urn:cartesian"^^srdf:geometry;
        dlr:hasGid "1"^^xsd:decimal .
```

Example 10. Find the URIs and the geometries of TerraSAR-X images that overlap with a user specified area, and their acquisition time lies inside a user specified time period.

```
select ?H srdf:AsText(?HGEO) as ?RGEO
where {?T rdf:type dlr:TSX1 .
       ?T dlr:hasGeometry ?TGEO .
       ?T dlr:hasStartDate ?TSD .
       ?T dlr:hasEndDate ?TED .
       filter (srdf:Overlaps(srdf:Envelope(srdf:Buffer(
               "POINT(2 2);urn:cartesian"^^srdf:geometry,1)), ?TGEO) &&
               "2011-01-01T00:00:00"^^xsd:dateTime < ?TSD &&
               "2011-01-01T00:00:00"^^xsd:dateTime > ?TED)
      }
```

Query result. *The result of this query is displayed below.*

Essentially this captures the user interaction with the EOWEB interface, that is, the action of choosing a rectangle area, a time period, and performing a search on TerraSAR-X products for this configuration.

?R	?RGEO
ex:image1	"POINT(2.5 2.5);urn:cartesian"^^srdf:geometry
ex:image2	"POINT(1.5 2.5);urn:cartesian"^^srdf:geometry
ex:image3	"POINT(1.5 1.5);urn:cartesian"^^srdf:geometry

This query uses the `srdf:Buffer` spatial operation which involves a spatial object and a distance. The result of this spatial operation is a new geometry that contains all the points at the specified maximum distance along the spatial object that was given as an input. The `srdf:Envelope` spatial operation has as a result a new geometry, that is the minimum bounding box of the input geometry. Finally, the `srdf:Overlaps` functions checks whether the geometry constructed by the `srdf:Buffer` and `srdf:Envelope` functions "spatially overlaps" the image geometry.

Example 11. Find the URIs and the geometries of images that overlap a user specified area, have an acquisition time that lies inside a user specified time period, and overlap a forest.

```
select ?H ?HGEO
where {?T rdf:type dlr:TSX1 .
       ?T dlr:hasGeometry ?TGEO .
       ?T dlr:hasStartDate ?TSD .
       ?T dlr:hasEndDate ?TED .
       ?R rdf:type dlr:Region .
       ?R dlr:hasGeometry ?RGEO .
       ?R dlr:hasCorineLandCoverUse clc:Forests .
       filter (srdf:Overlaps(srdf:Envelope(
               srdf:Buffer("POINT(2 2);urn:cartesian"^^srdf:geometry,
                             1)), ?TGEO) &&
               "2011-01-01T00:00:00"^^xsd:dateTime < ?TSD &&
               "2011-01-01T00:00:00"^^xsd:dateTime > ?TED &&
               srdf:Overlaps(?RGEO, ?TGEO))
      }
```

Query result. *The result of this query is displayed below.*

?R	?RGEO
ex:image1	"POINT(2.5 2.5);urn:cartesian"^^srdf:geometry
ex:image2	"POINT(1.5 2.5);urn:cartesian"^^srdf:geometry

This query is similar to the query we presented in Example 10. In this query, we additionally request that the geometry of the image overlaps the geometry of a region that is a forest.

4.4 Summary

In this chapter we presented the data model sRDF++ and the query language sSPARQL++ which are new versions of the data model sRDF and the query language sSPARQL. We extended RDF and SPARQL so that OGC standards such as WKT and GML can also be used for the representation of geospatial data (instead of just linear constraints). Our approach resembles OGC proposals regarding how to use SQL to query geographic features [OGC10d] as they have been adopted by commercial RDBMs. In this way, we expect that sRDF++ and sSPARQL++ will be adopted more easily by applications such as the ones that are represented in TELEIOS.

Our work also resembles the recent OGC proposal GeoSPARQL discussed in [OGC10b]. Both approaches represent geometric objects as spatial literals and may be encoded in various formats like GML, KML, WKT etc. In our approach, basic spatial functions (e.g., `isEmpty`), spatial predicates (e.g., `Overlaps`), and functions that support spatial analysis (e.g., `Buffer`) are all mapped to extension functions (as defined in standard SPARQL) that may be used in the select part or the filter part of a query. In GeoSPARQL, basic spatial functions and spatial predicates are mapped to RDF properties. Spatial predicates and functions that support spatial analysis are mapped to extension functions that may be used in the filter part of a query. The ability to have basic spatial predicates as RDF properties in GeoSPARQL allows GeoSPARQL to be easily combined with other representational frameworks for example OWL2 rules that formalize qualitative spatial reasoning so that more expressive applications can be written (see the discussion of GeoSPARQL in Section 3.4). We plan to add similar functionality to our language in the future but not before our work on qualitative spatial reasoning presented in the following chapter has been completed. The results of this work will determine the details. For this reason, We plan to participate in the activities of the relevant OGC working group so that TELEIOS results are disseminated to this community and vice versa.

5. Going Beyond sRDF and sSPARQL: Reasoning with Qualitative Spatial Information

In this chapter we show how to extend the sRDF and sSPARQL framework that we have developed in [KK10] so that we have the ability to represent, query and reason with *qualitative spatial relationships represented by constraints.* The representation of qualitative spatial relationships using *constraints* has been studied for quite some time in the area of Spatial Reasoning [RN07]. In this chapter we show how to integrate spatial information expressed by constraints together with sRDF data, and query it using an extension of sSPARQL.

In the proposed extension of sRDF, which we call sRDF^i, we use a new kind of literals to represent spatial regions about which the known information is *incomplete* or *indefinite* (e.g., region A is inside a known rectangle R but we do not know its exact geographic location, or region A is north of region B and it overlaps region C, but we do not know anything else about it, etc.). In the general case, such incomplete or indefinite information is expressed in terms of *disjunctive qualitative spatial constraints* representing size (e.g., "large", "small"), direction (e.g., "right of", "above"), distance (e.g., "far", "near"), shape (e.g., "convex") or topology (e.g., "overlaps", "contains") of spatial objects [RN07]. sRDF^i is essentially an extension of sRDF in the spirit of *indefinite constraint databases* developed by the NKUA group in the past [Kou94c, Kou94b, Kou97].

We choose to extend sRDF and sSPARQL instead of sRDF++ and sSPARQL++, since we already have a formal framework for the former, and this makes most of the theoretical developments of this chapter easier. Carrying out the same developments in the context of sRDF++ and sSPARQL++ is then straightforward.

Let us recall that the only kind of spatial data that can be represented in the model sRDF presented earlier is exact geometries (e.g., a hotspot is represented as a point in $\mathbb{Q}^2$ or the extension of a burnt area is represented as a polygon). However, EO applications such as the ones we consider in the two TELEIOS use cases often need the representation of qualitative spatial information as well. This issue has not been discussed in Chapter 2, so we will now give some concrete examples to motivate our developments.

5.1 Qualitative spatial information in the two use cases

The examples of this section allow us to introduce some features of sRDF^i. Detailed developments start in Section 5.2. We start with an example from the NOA use case.

As we have explained in Section 2.2, the MSG SEVIRI instrument has medium resolution, therefore each image pixel representing a hotspot in the FMM-1 product corresponds to a 3km by 3km rectangle in geographic space. In Section 2.2.3, we have explained that the FMM-1 product represents hotspots as points in geographic space using the center of the corresponding rectangle. This has led us to write queries accordingly so that the "3km by 3km" resolution is taken into account (e.g., Query 2 in Section 2.2.4).

In this case, another useful representation of the real world situation that corresponds to a hotspot would be to state that there is a geographic region with unknown exact coordinates where a fire is taking place, and that region is included in a known 3km by 3km rectangle. This is captured by the following triples and constraints in sRDF^i that introduce the hotspot, the fire corresponding to it and the region corresponding to the fire. This region (`_region1`) is a new kind of literal, called an *unknown literal*, which is asserted to be inside the rectangle formed by the points $(0, 0)$ and $(3000, 3000)$ on $\mathbb{Q}^2$ (this specific rectangle is chosen for the sake of the example).

```
noa:hotspot1 rdf:type noa:Hotspot .
noa:fire1 rdf:type noa:Fire .
noa:hotspot1 noa:correspondsTo noa:fire1 .
noa:fire1 noa:occuredIn _region1 .

_region1 NTPP "x ≥ 0 ∧ x ≤ 3000 ∧ y ≥ 0 ∧ y ≤ 3000" .
```

Unknown literals are like existentially quantified variables in first-order logic. However, we have chosen to distinguish them from blank nodes to emphasize their "literal nature". By convention, identifiers for unknown literals in sRDF^i *always start with an underscore*. In the above example, NTPP is the "non-tangential-proper-part" relation of RCC8 [CCR93].

Imagine that, later on, when fire noa:fire1 is validated, NOA would like to annotate the relevant hotspot, validated fire and burnt area with information from news sources available on the Web that have reported the corresponding fire (such news information can actually be one way to validate the fire). As an example, information related to noa:fire1 obtained from a regional Greek newspaper available on the Web might say that "there was a fire *north of* the village of Zoniana in the Prefecture of Rethymno, Crete". In this case NOA might choose to produce the following annotation which mixes the qualitative spatial information discovered from the newspaper with information that corresponds to the relevant administrative regions of Greece.[1]

```
noa:fire1 rdf:type noa:ValidatedFire .
noa:fire1 noa:hasBurnedArea _region2 .


kal:Zoniana rdf:type kal:Community .
kal:Mylopotamos rdf:type kal:Municipality .
kal:Rethymno rdf:type kal:Prefecture .


kal:Zoniana kal:occupies _region3 .
kal:Mylopotamos kal:occupies _region4 .
kal:Rethymno kal:occupies _region5 .

kal:Zoniana kal:partOf kal:Mylopotamos .
kal:Mylopotamos kal:partOf kal:Rethymno .

_region3 NTPP _region4 .
_region4 NTPP _region5 .

_region1 northOf kal:Zoniana .
_region2 northOf kal:Zoniana .
```

The above triples introduce the burnt area corresponding to noa:fire1 and some details related to the administrative geography of Greece as defined by the recent "Kallikratis Plan"[2]. Direction relationships such as northOf are surveyed in [RN07]. Since there is already work in encoding the administrative geography of countries e.g., the UK [GDH08] in terms of qualitative spatial constraints such as the ones we used above, we expect that such annotations can be a useful source of information in both TELEIOS use cases. This is stressed by the fact that currently much of

[1] Developing software that harvests spatial information from the Web is not part of the TELEIOS work programme. See the paper [PCJ+07] for relevant work in Geographic Information Retrieval.

[2] http://en.wikipedia.org/wiki/Administrative_divisions_of_Greece

this information is or will become available as public open data in portals of the relevant European governments (e.g., see the geodata portal of the Government of Greece[3]).

Now let us consider an example from the DLR use case. Imagine that during an environmental emergency, the Center for Satellite-based Crisis Information (ZKI) of DLR has processed various TerraSAR-X images using the image mining methods of work package "Knowledge discovery from EO images" (WP3) of TELEIOS, and determined a large flooded area that extends over two adjacent administrative regions of a certain country (e.g., Romania). ZKI might want to represent this qualitative spatial information as an annotation to the damage assessment map that will be produced by encoding it in sRDF^{i} as follows:

```
_region1 rdf:type dlr:FloodedArea .

_region2 rdf:type ex:AdministrativeRegion .
_region3 rdf:type ex:AdministrativeRegion .

_region2 EC _region3 .
_region1 PO _region2 .
_region1 NTPPi _region3 .
```

The above triples introduce three unknown regions `_region1`, `_region2` and `_region3` that correspond to the flooded area and the two administrative regions of Romania respectively. The qualitative spatial relations `EC`, `PO`, `NTPPi` used in the above constraints are the relationships "externally connects", "partially overlaps" and "non-tangential proper part inverse" of RCC8 [CCR93]. Naturally, more information about the administrative geography of Romania could also be added as in the previous example from the NOA use case.

Let us now continue with our technical developments. The next two sections define constraint languages that allow us to express some kinds of qualitative spatial information that we envision to be useful in the two use cases, and define some properties of these languages that are important for defining sRDF^{i}.

5.2 Constraint Languages

In this chapter, we will consider (many-sorted) first-order languages, structures and theories [End72] that allow us to capture qualitative spatial relations. Every language $\mathcal{L}$ will be interpreted over a *fixed* structure, called the *intended structure*, which will usually be denoted by $\mathbf{M}_{\mathcal{L}}$. If $\mathbf{M}_{\mathcal{L}}$ is a structure then $Th(\mathbf{M}_{\mathcal{L}})$ will denote the theory of $\mathbf{M}_{\mathcal{L}}$, i.e., the set of sentences of $\mathcal{L}$ which are true in $\mathbf{M}_{\mathcal{L}}$. For every language $\mathcal{L}$, we will distinguish a class of quantifier free formulas called $\mathcal{L}$*-constraints*. The atomic formulas of $\mathcal{L}$ will be included in the class of $\mathcal{L}$-constraints. There will also be two distinguished $\mathcal{L}$-constraints *true* and *false* with obvious semantics. Similar assumptions have been made in the context of related frameworks such as the constraint logic programming [Mah93], constraint databases [KKR90] and the indefinite constraints [Kou94c, Kou94b, Kou97]. A set of $\mathcal{L}$-constraints will be the algebraic counterpart of the logical conjunction of its members. Thus we will freely mix the terms "set of $\mathcal{L}$-constraints" and "conjunction of $\mathcal{L}$-constraints". We will assume that the reader is familiar with the notions of *solution, consistency* and *equivalence* of sets of constraints [Mah93].

Let us now give some examples of spatial constraint languages that we would like to study in TELEIOS.

[3] `http://geodata.gov.gr`

The language *TCL*. The language *TCL* (*T*opological *C*onstraint *L*anguage) allows us to represent topological properties of non-empty, regular closed subsets of $\mathbb{Q}^2$ (we will call these subsets *regions* for brevity). We prefer to have $\mathbb{Q}^2$ instead of $\mathbb{R}^2$ as our set, since in an implementation, relevant information will be represented by floating point numbers. Thus, rational numbers are the closest abstraction. TCL is a first-order language with the following 6 binary predicate symbols: DC, EC, PO, EQ, TPP and $NTPP$. An *atomic formula* of TCL is a formula of the form $r_1\ R\ r_2$, where r_1, r_2 are variables and R is one of the above predicates. A *TCL-constraint* is a disjunction of atomic formulas of TCL involving the same two variables. For example, the following are TCL-constraints:

$$r_1\ NTPP\ r_2,\quad r_2\ PO\ r_3 \vee r_2\ EQ\ r_3$$

In TCL and all other languages of this section, we also assume the existence of constraints *true* and *false* with obvious semantics.

The intended structure for TCL, denoted by $\mathbf{M}_{TCL}$, has the set of regions as its domain, and interprets each of the predicate symbols given above by the corresponding topological relation of RCC-8 [RCC92b] shown in Table 3.1 of Section 3.6. Note that relations $NTPPi$ and $TPPi$ of RCC-8 are not included in the vocabulary of TCL since they can be expressed by interchanging the arguments of $NTPP$ and TPP.

The language TCL allows us to capture the topology of regions of interest to an application but makes no commitment regarding other non-topological properties of these regions, e.g., shape. The languages PCL and RCL considered below deal with regions with specific shapes (polygons and rectangles respectively) that are useful for encoding the geospatial data sets considered in TELEIOS.

The language *PCL*. The language *PCL* (*P*olygon *C*onstraint *L*anguage) allows us to represent topological properties of polygons in $\mathbb{Q}^2$. TCL is a first-order language with the same 6 predicate symbols as TCL, but also constant symbols representing polygons in $\mathbb{Q}^2$. From the two standard notations for polygons (sequence of vertices or conjunctions of linear constraints), we choose to write polygons using the constraint notation and include these constraints in quotes[4] (e.g., "$x \leq 1, x \geq 0, y \leq 1, y \geq 0''$ is a rather long constant that denotes the unit rectangle with lower-left vertex $(0, 0)$). The terms and atomic formulas of PCL are defined as follows. Constants and variables are *terms*. An *atomic formula* of PCL is a formula of the form $t_1\ R\ t_2$ where t_1, t_2 are terms and R is one of the above predicates. A *PCL-constraint* is a disjunction of atomic formulas of PCL involving the same pair of terms. For example, the following are PCL-constraints:

$$r_1\ NTPP\ r_2 \vee r_1\ TPP\ r_2,\quad r_2\ NTPP\ \text{“}x \leq 1, x \geq 0, y \leq 1, y \geq 0''$$

The intended structure for PCL, denoted by $\mathbf{M}_{PCL}$, has the set of polygons in $\mathbb{Q}^2$ as its domain. $\mathbf{M}_{PCL}$ interprets each constant symbol by the corresponding polygon in $\mathbb{Q}^2$, and each of the predicate symbols by the corresponding topological relation of RCC8 [RCC92b].

The language *RCL*. The language *RCL* (*R*ectangle *C*onstraint *L*anguage) allows us to capture spatial constraints (e.g., topological or directional) involving rectangles with sides parallel to the axes in $\mathbb{Q}^2$ (we will call them *boxes*). RCL is useful not only for modeling regions of space with such rectangular shapes but also for modeling *minimum bounding rectangles* that are typically used as approximations of spatial objects, e.g., in spatial data structures and elsewhere.

RCL is a first-order language with equality and 2 sorts: the sort $\mathcal{Q}$ for rational constants, and the sort $\mathcal{R}$ for boxes. The set of non-logical symbols of RCL includes: all rational constants of sort

[4] We choose to live with this non-standard way of writing formulas in quotes without going into the details of their syntax. This will not create any technical problems for us for the development of this chapter.

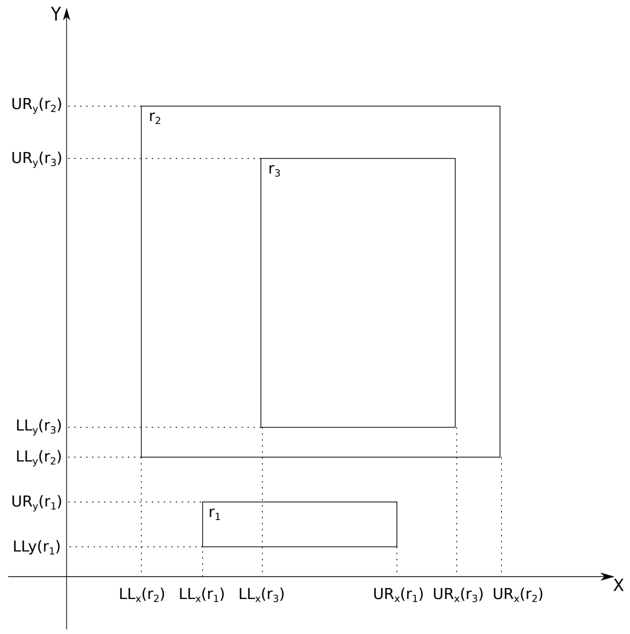


Figure 5.1: Three boxes

$\mathcal{Q}$, function symbols $LL_x(\cdot), LL_y(\cdot), UR_x(\cdot), UR_y(\cdot)$ of sort $(\mathcal{R}, \mathcal{Q})$, and predicate symbol $<$ of sort $(\mathcal{Q}, \mathcal{Q})$.

The terms and atomic formulas of RCL are defined as follows. Constants of sort $\mathcal{Q}$ and variables of sort $\mathcal{R}$ are terms. If r is a variable of sort $\mathcal{R}$ then $LL_x(r), LL_y(r), UR_x(r)$ and $UR_y(r)$ is a term of sort $\mathcal{Q}$. An *atomic formula* of RCL is a formula of the form $t_1 \sim t_2$ where $\sim$ is $<$ or $=$ and t_1, t_2 are terms involving function symbols with the same subscript (x or y) or rational constants. $=$ is the equality predicate for sort $\mathcal{Q}$; we will not use the equality predicate for sort $\mathcal{R}$ in our formulas.

The intended structure for RCL, denoted by $\mathbf{M}_{RCL}$, interprets each non-logical symbol as follows. Each rational constant is interpreted by its corresponding rational number. The function symbols $LL_x(\cdot), LL_y(\cdot), UR_x(\cdot)$ and $UR_y(\cdot)$ are interpreted by the easily-defined functions that given a box in $\mathbb{Q}^2$, return the x- and y-coordinate of its lower-left and lower-right vertex respectively. Predicate $<$ is interpreted by the relation "less than" over $\mathbb{Q}$.

A *RCL-constraint* is a RCL formula of the form $t_1 \sim t_2$ where $\sim$ is $=$, $<$, $>$, $\leq$ or $\geq$ and t_1, t_2 are terms (the predicates $<$, $\leq$, and $\geq$ are defined as usual).

Example 12. The formulas

$$LL_x(r_2) < LL_x(r_1),\ UR_y(r_1) < LL_y(r_2),\ UR_x(r_1) < UR_x(r_2)$$

$$LL_x(r_2) < LL_x(r_3),\ LL_y(r_2) < LL_y(r_3),\ UR_x(r_3) < UR_x(r_2),\ UR_y(r_3) < UR_y(r_2)$$

are RCL-constraints. The conjunction of these constraints tells us that box r_1 is to the south of box r_2 (first line) and box r_3 is a non-tangential proper part of r_2 (second line). Three boxes satisfying the constraints are depicted in Figure 5.1.

The language *TRCL*. We also define the language TRCL which essentially extends TCL with the ability to express a topological relation of a region with a box in $\mathbb{Q}^2$. TRCL has 2 sorts: the sort of regions $\mathcal{R}$ and the sort of boxes $\mathcal{B}$. In TRCL, we only allow constants of sort $\mathcal{B}$. These constants are written using order constraints of the form $x \leq c, x \geq c, y \leq c$ and $y \geq c$, where c

is a rational constant. An example of such a constant is "$x \geq 0 \wedge x \leq 3000 \wedge y \geq 0 \wedge y \leq 3000$". We assume that the order constraints defining these constants are in *normal form*, i.e., there are exactly four constraints, two on variable x and two on variable y defining a box in $\mathbb{Q}^2$ in the obvious way.

The atomic formulas of TRCL are the ones allowed by TCL but we also allow constants of sort $\mathcal{B}$ to appear in these formulas, e.g., $x\ NTPP$ "$x \geq 0 \wedge x \leq 3000 \wedge y \geq 0 \wedge y \leq 3000''$. The rest of the formal definition of TRCL is obvious, so we omit it for brevity.

The above four languages have been defined so that the reader can appreciate the scope of modeling possibilities for geospatial applications like the ones studied in TELEIOS. Obviously, many more possible languages can be defined depending on the richness of spatial information that one would like to represent. There is a well-known trade-off among the expressivity of a given language and its computational properties. This is something that we explore briefly in the following section.

5.3 Satisfiability and Validity

In the literature of constraint satisfaction and spatial reasoning, the most important decision problem studied for languages such as TCL, PCL, RCL, and TRCL is deciding whether a formula in these languages is satisfiable or valid. The most important known results in this area are the following.

Results for TCL. The validity problem for arbitrary formulas of TCL is undecidable [Grz51, Dor98]. The satisfiability problem for conjunctions of TCL-constraints is NP-complete but there are tractable subcases [GPP95, RN99]. There have been implemented algorithms that determine the satisfiability of conjunctions of TCL-constraints. For the general case, these are backtracking algorithms [RN01], while for the tractable cases these are algorithms relying on path consistency for the corresponding constraint networks [RN99, RN07].

There are also interesting results on related spatial logics that consider more expressive languages with connectedness predicates, Boolean algebra operators for forming new regions, etc. [KPHZ10].

Results for PCL. The validity problem for arbitrary formulas of PCL is also undecidable [Dor98]. The satisfiability problem for conjunctions of PCL-constraints is also NP-hard since restricting our attention to polygons does not change the difficulty of the problem. This particular restriction of regions to be polygons has not been studied in the literature and no detailed complexity results or algorithms are known. [LL10] is a recent paper that studies binary topological relations among convex planar regions.

Results for RCL. Reasoning in RCL is easier than in the previous two languages. The restricted syntax of RCL allows us to reason about boxes by considering the constraints involving functions with a subscript x and functions with a subscript y independently. It is easy to see that the validity problem for arbitrary formulas of RCL is in PSPACE. The result follows from the results of Koubarakis on similar language of points and intervals [Kou94a]. The satisfiability problem for conjunctions of RCL-constraints can be solved in PTIME using well-known results from the Point Algebra [VKvB89, Kou06].

One could similarly define another language, let us call it RCL*, that has rectangles in $\mathbb{Q}^2$ as constants, variables ranging over boxes and the 13×13 basic binary relations from the Rectangle Algebra studied by Balbiani and others [BCdC99, PT97] as predicates. The validity problem in RCL* can similarly be proved to be in PSPACE. The satisfiability problem for conjunctions of RCL*-constraints is NP-complete but there are tractable subcases [BCdC99].

Results for TRCL. The validity problem for TRCL is undecidable given the undecidability of TCL. It is not difficult to say that the satisfiability problem for conjunctions of TRCL constraints is NP-complete. No detailed complexity results or algorithms are known for this language.

From the above, it is easy to conclude that RCL and RCL* are first-order languages with better computational properties than TCL, PCL, and TRCL. Computational issues will be revisited in the conclusions of this chapter; further computational properties of TCL, PCL, RCL, and TRCL will be studied in Deliverable D2.3 "Theoretical results on query processing for RDF/SPARQL with time and space".

5.4 Variable and Quantifier Elimination

We now define two operations that will be useful for the rest of our developments: variable elimination and quantifier elimination. *Variable elimination* is an algebraic operation; its logical counterpart is *quantifier elimination.* We will now define these two notions and then use the one which is more appropriate in each case. We will always assume that we have to deal with formulas of finite length.

Notation 5.4.1. The vector of symbols $(o_1, \ldots, o_n)$ will be denoted by $\overline{o}$. The natural number n will be called the *size* of $\overline{o}$ and will be denoted by $|\overline{o}|$. This notation will be used for vectors of variables but also for vectors of domain elements. Variables will be denoted by x, y, z, t, etc. and vectors of variables by $\overline{x}, \overline{y}, \overline{z}, \overline{t}$, etc. If $\overline{x}$ and $\overline{y}$ are vectors of variables then $\overline{x} \setminus \overline{y}$ will denote the vector obtained from $\overline{x}$ by deleting the variables in $\overline{y}$. If $\overline{x}$ and $\overline{y}$ are vectors of variables and every variable in $\overline{x}$ is also contained in $\overline{y}$ then we will write $\overline{x} \subseteq \overline{y}$. If $\overline{x}$ is a vector of variables then $\overline{x}^0$ will be a vector of constants of the same size. The notation $\overline{x}^0 \setminus \overline{y}^0$ is similarly defined for vectors of constants.

Definition 1. Let $\mathcal{L}$ be a many-sorted first-order language. The class of $\mathcal{L}$-constraints *admits variable elimination* iff for every boolean combination ϕ of $\mathcal{L}$-constraints in variables $\overline{x}$, and every vector of variables $\overline{z} \subseteq \overline{x}$, there exists a disjunction ϕ' of conjunctions of $\mathcal{L}$-constraints in variables $\overline{x} \setminus \overline{z}$ such that

1. If $\overline{x}^0$ is a solution of ϕ then $\overline{x}^0 \setminus \overline{z}^0$ is a solution of ϕ'.
2. If $(\overline{x} \setminus \overline{z})^0$ is a solution of ϕ' then this solution can be extended to a solution $\overline{x}^0$ of ϕ.

The following definition will also be useful.

Definition 2. Let $\mathcal{L}$ be a many-sorted first-order language. The class of $\mathcal{L}$-constraints is *weakly closed under negation* if the negation of every $\mathcal{L}$-constraint is equivalent to a disjunction of $\mathcal{L}$-constraints.

The following proposition, whose proof can easily be done by induction, shows that if the class of $\mathcal{L}$-constraints is weakly closed under negation then we can determine whether it also admits variable elimination by considering only conjunctions of $\mathcal{L}$-constraints.

Proposition 5.4.1. *Let $\mathcal{L}$ be a many-sorted first-order language. The class of $\mathcal{L}$-constraints admits variable elimination if*

- *it is weakly closed under negation, and*
- *for every conjunction θ of $\mathcal{L}$-constraints in variables $\overline{x}$ and every vector of variables $\overline{z} \subseteq \overline{x}$, there exists a disjunction θ' of conjunctions of $\mathcal{L}$-constraints in variables $\overline{x} \setminus \overline{z}$ such that*

1. *If* $\overline{x}^0$ *is a solution of* θ *then* $\overline{x}^0 \setminus \overline{z}^0$ *is a solution of* θ'.
2. *If* $(\overline{x} \setminus \overline{z})^0$ *is a solution of* θ' *then this solution can be extended to a solution* $\overline{x}^0$ *of* θ.

The importance of this proposition is that it provides us with the following algorithm for eliminating variables $\overline{z}$ from a Boolean combination ϕ of $\mathcal{L}$-constraints in variables $\overline{x}$:

1. Transform ϕ into a formula where negation applies only to $\mathcal{L}$-constraints (by applying De Morgan's laws and laws for negation).
2. Substitute every negated $\mathcal{L}$-constraint by its equivalent disjunction of $\mathcal{L}$-constraints.
3. Transform ϕ into disjunctive normal form $\theta_1 \lor \cdots \lor \theta_m$ where each θ_i is a conjunction of $\mathcal{L}$-constraints.
4. Perform variable elimination in each θ_i, i.e., substitute each disjunct θ_i of ϕ by the equivalent disjunction of conjunctions of $\mathcal{L}$-constraints in variables $\overline{x} \setminus \overline{z}$.

For this algorithm to be effective, we must know how to perform Steps 2 and 4. For most languages of interest (e.g., TCL, PCL, RCL, or TRCL) step 2 will be obvious. Step 4 will be the challenging step.

Let us give an example of a variable elimination algorithm which is well-known. Consider a set (conjunction) of linear inequalities from the first-order language $\mathcal{L}_{LIN}$ of linear inequalities over the rational numbers considered among others in our sRDF/sSPARQL paper [KK10]. Variable elimination for linear inequalities can be performed using the well-known Fourier's algorithm which can be summarized as follows [Sch86]. Any weak linear inequality involving a variable x can be written in the form $x \leq r_u$ or $x \geq r_l$, i.e., it gives an upper or a lower bound on x. Thus if we are given two linear inequalities, one of the form $x \leq r_u$ and the other of the form $x \geq r_l$, we can eliminate x and obtain the inequality $r_l \leq r_u$. Obviously, $r_l \leq r_u$ is a logical consequence of the given inequalities. In addition, any solution of $r_l \leq r_u$ can be extended to a solution of the given inequalities (simply by choosing for x any value between the values of r_l and r_u). Following this observation, Fourier's elimination algorithm forms all pairs $x \leq r_u$ and $x \geq r_l$, eliminates x and returns the resulting inequalities together with the inequalities that do not involve variable x. If pairs of the form $x \leq r_u$ and $x \geq r_l$ cannot be formed (e.g., because all inequalities involving x are of the form $x \leq r_u$), then all inequalities involving x are discarded and the remaining inequalities are retained. The generalization of this algorithm to the case where we also have equalities and strict inequalities is obvious.

The following is an example of variable elimination using Fourier's algorithm.

Example 13. Eliminating variable x_1 from the set of inequalities

$$x_3 \leq x_1,\ x_5 \leq x_1,\ x_1 - 3x_2 + 5x_6 \leq 2,\ x_4 \leq x_5$$

gives

$$x_3 - 3x_2 + 5x_6 \leq 2,\ x_5 - 3x_2 + 5x_6 \leq 2,\ x_4 \leq x_5.$$

If we also eliminate variable x_5, we get

$$x_3 - 3x_2 + 5x_6 \leq 2,\ x_4 - 3x_2 + 5x_6 \leq 2.$$

Let us now consider the languages and classes of constraints defined in Section 5.2.

Proposition 5.4.2. *The class of RCL-constraints admits variable elimination and is weakly closed under negation.*

The result is trivial concerning "weakly closed under negation". Regarding variable elimination, we give an example instead of the easy formal proof.

Example 14. Let us consider the following set of RCL-constraints taken from Example 12:

$$LL_x(r_2) < LL_x(r_1),\ UR_y(r_1) < LL_y(r_2),\ UR_x(r_1) < UR_x(r_2)$$

$$LL_x(r_2) < LL_x(r_3),\ LL_y(r_2) < LL_y(r_3),\ UR_x(r_3) < UR_x(r_2),\ UR_y(r_3) < UR_y(r_2)$$

We would like to eliminate variable r_2. We will proceed in two steps. First, we introduce the implicit constraints that capture the relationships between the x- and y-coordinates of each box. The given set of constraints is now transformed into the following equivalent set of order constraints:

$$LL_x(r_2) < LL_x(r_1),\ UR_y(r_1) < LL_y(r_2),\ UR_x(r_1) < UR_x(r_2)$$

$$LL_x(r_2) < LL_x(r_3),\ LL_y(r_2) < LL_y(r_3),\ UR_x(r_3) < UR_x(r_2),\ UR_y(r_3) < UR_y(r_2)$$

$$LL_x(r_1) < UR_x(r_1), LL_y(r_1) < UR_y(r_1)$$

$$LL_x(r_2) < UR_x(r_2), LL_y(r_2) < UR_y(r_2)$$

$$LL_x(r_3) < UR_x(r_3), LL_y(r_3) < UR_y(r_3)$$

We now treat terms as variables and use Fourier elimination for eliminating the term $LL_y(r_2)$ and get:

$$LL_x(r_2) < LL_x(r_1),\ \cancel{UR_y(r_1) < LL_y(r_2)},\ UR_x(r_1) < UR_x(r_2)$$

$$LL_x(r_2) < LL_x(r_3),\ \cancel{LL_y(r_2) < LL_y(r_3)},\ UR_x(r_3) < UR_x(r_2),\ UR_y(r_3) < UR_y(r_2)$$

$$LL_x(r_1) < UR_x(r_1), LL_y(r_1) < UR_y(r_1)$$

$$LL_x(r_2) < UR_x(r_2), \cancel{LL_y(r_2) < UR_y(r_2)}$$

$$LL_x(r_3) < UR_x(r_3), LL_y(r_3) < UR_y(r_3)$$

$$UR_y(r_1) < LL_y(r_3)$$

$$UR_y(r_1) < UR_y(r_2)$$

If we also eliminate the terms $LL_x(r_2)$, $UR_x(r_2)$, and $UR_y(r_2)$, we get:

$$\cancel{LL_x(r_2) < LL_x(r_1)},\ \cancel{UR_x(r_1) < UR_x(r_2)}$$

$$\cancel{LL_x(r_2) < LL_x(r_3)},\ \cancel{UR_x(r_3) < UR_x(r_2)},\ \cancel{UR_y(r_3) < UR_y(r_2)}$$

$$LL_x(r_1) < UR_x(r_1), LL_y(r_1) < UR_y(r_1)$$

$$\cancel{LL_x(r_2) < UR_x(r_2)}$$

$$LL_x(r_3) < UR_x(r_3), LL_y(r_3) < UR_y(r_3)$$

$$UR_y(r_1) < LL_y(r_3)$$

$$\cancel{UR_y(r_1) < UR_y(r_2)}$$

From this set of constraints and by removing the implicit constraints, we get the single constraint:

$$UR_y(r_1) < LL_y(r_3)$$

Inspecting now the above constraint in relation to Figure 5.1, we observe that all constraints between boxes r_1, r_3 and boxes r_2, r_3 have been ruled out (due to the fact that we chose to eliminate variable r_2), while a new one has been introduced (the above) that preserves the property between r_1 and r_3 (i.e., r_1 is south of r_3).

Let us now consider elimination of quantifiers.

Definition 3. Let $Th(\mathcal{L})$ be the theory of language $\mathcal{L}$. $Th(\mathcal{L})$ *admits elimination of quantifiers* iff for every formula ϕ there is a disjunction ϕ' of conjunctions of $\mathcal{L}$-constraints such that $Th(\mathcal{L}) \models \phi \equiv \phi'$.

This definition is stronger than the traditional one where ϕ' is simply required to be quantifier-free [End72]. We require ϕ' to be in the above form because we do not want to deal with negations of $\mathcal{L}$-constraints.

Proposition 5.4.3. *The language RCL admits quantifier elimination.*

Proof. Let us assume that ϕ is a formula of RCL. The *standard algorithm* to eliminate quantifiers proceeds by computing the prenex normal form form of ϕ and then eliminating quantifiers starting from the innermost one. Eliminating quantifier (Qx) from $(Qx)\theta$, where θ is quantifier-free can be done by transforming θ in disjunctive normal form and then *eliminating variable* x from each disjunct. The result now follows from Proposition 5.4.2. □

Unfortunately, the other three languages we discussed in Section 5.2 do *not* admit quantifier elimination. This is easy to see since they are both undecidable; if they would admit quantifier elimination, then we would immediately have a decision procedure for these languages. Bennett has discussed the issue of quantifier elimination in TCL in [Ben97].

Proposition 5.4.4. *The languages TCL, PCL, and TRCL do not admit quantifier elimination.*

Thus, as with the computational complexity properties discussed in Section 5.3, RCL is more "well-behaved" than TCL and PCL with respect to variable and quantifier elimination.

We have now presented the concepts necessary for the understanding of spatial constraints and spatial constraint languages. In the next section we show how sRDF graphs as defined in [KK10] and constraints from a language such as TCL, PCL, RCL, and TRCL defined above allow us to query qualitative spatial information in the Semantic Web in general and the TELEIOS use cases in particular.

5.5 The model sRDF$^{\text{i}}$

As in theoretical treatments of RDF [PAG06], we assume the existence of pairwise-disjoint countably infinite sets I, B and L that contain IRIs, blank nodes and literals respectively. We also assume the existence of a set C of *spatial* literals. We will not consider arbitrary semi-linear sets as spatial literals, as in [KK10]. Instead, C is the set of all boxes in $\mathbb{Q}^2$ written in the normal form of Section 5.2 (e.g., "$x \geq 0 \wedge x \leq 1 \wedge y \geq 0 \wedge y \leq 1$"). In standard RDF terminology, the elements of C are typed literals with datatype `srdf:Box`. We will call these literals *spatial* to distinguish them from other RDF literals.

Technically, this choice of geometry for spatial literals is necessary since, as you will see below, we will use the language TRCL to express qualitative spatial constraints. The fact that the only constants of TRCL are boxes in $\mathbb{Q}^2$ determines the constants (i.e., spatial literals) that we can allow in triples.

In sRDF$^{\text{i}}$ we assume the existence of one more countably infinite set disjoint from all the above: the set U of *unknown spatial literals*. By convention, the identifiers of unknown spatial literals will start with an underscore, e.g., `_region5`. The set of sRDF$^{\text{i}}$ terms will be denoted by T and is the union $I \cup B \cup L \cup C \cup U$.

We now define the basic concepts of sRDF$^{\mathrm{i}}$: triples, constraint stores, graphs and databases. Triples in sRDF$^{\mathrm{i}}$ are as in sRDF but now unknown spatial literals are also allowed in the object position. Constraint stores are simply Boolean combinations of constraints. The combination of a graph and a constraint store is called database.

Definition 4. *An* sRDF$^{\mathrm{i}}$ triple *is an element of the set* $(I \cup B) \times I \times (I \cup B \cup L \cup C \cup U)$. *If* (s, p, o) *is an* sRDF$^{\mathrm{i}}$ *triple, s will be called the subject, p the predicate and o the object of the triple.*

The theory of sRDF$^{\mathrm{i}}$ can be developed using any of the languages TCL, TPCL, RCL, TRCL. However, the properties of each one of these languages determine how rich the theory of sRDF$^{\mathrm{i}}$ will be. We will use the language TRCL in the rest of this section since this language is expressive enough to cover our examples from the use cases. We will discuss some of the relevant issues regarding the choice of constraint language in the conclusions of this chapter.

Definition 5. *A* constraint store *is a Boolean combination of $TRCL$-constraints.*

Example 15. The following is a constraint store:

$$\_region1 \ NTPP \ \text{“}x \geq 0 \wedge x \leq 3000 \wedge y \geq 0 \wedge y \leq 3000''$$

Definition 6. *An* sRDF$^{\mathrm{i}}$ graph *is a set of* sRDF$^{\mathrm{i}}$ *triples. An* sRDF$^{\mathrm{i}}$ database *D is a pair $D = (G, \phi)$ where G is an* sRDF$^{\mathrm{i}}$ *graph and ϕ is a constraint store.*

Example 16. The following set of triples comprises an sRDF$^{\mathrm{i}}$ graph from the example from the NOA use case in Section 5.1 (namespaces are omitted for brevity):

$$\{(hotspot1, type, Hotspot),\ (fire1, type, Fire),$$

$$(hotspot1, correspondsTo, fire1),\ (fire1, occuredIn, \_region1)\}$$

Together with the constraint store of Example 15, they define an sRDF$^{\mathrm{i}}$ database.

5.6 Evaluating sSPARQL on sRDF$^{\mathrm{i}}$ databases

Let us now discuss how to evaluate sSPARQL queries on sRDF$^{\mathrm{i}}$ databases. We will see that due to the presence of unknown literals, query evaluation now becomes more complicated and is similar to query evaluation in databases with marked null values [IL84, Gra91]. In fact, since the language used to express constraints in sRDF$^{\mathrm{i}}$ is tightly connected to the kind of queries that can be answered precisely, we will consider a *subset* of sSPARQL. The exact details will be given later in this section. We will use the algebraic syntax of sSPARQL used in [KK10]. Once our theoretical development of sRDF$^{\mathrm{i}}$ is complete in Deliverable 2.3, we will introduce in sSPARQL++ appropriate syntax for dealing with indefinite spatial information.

We assume the existence of the following disjoint sets of variables: (i) the set of *non-spatial variables* V_{ns} that will be used to denote IRIs, blank nodes or RDF literals, and (ii) the set of *spatial variables* V_s that will be used to denote spatial literals from the sets C (known) or U (unknown). We use V to denote the set of all variables $V_{ns} \cup V_s$. The set V is assumed to be disjoint from the set of terms T we defined in Section 5.5.

We first define a concept of mapping appropriate for our task by extending the definition of [KK10]. To be able to deal with unknown spatial literals, mappings are now equipped with a constraint part and they become *conditional.*

Definition 7. *A* mapping *μ is a pair (ν, θ) where:*

- *ν is a partial function $\nu : V \to T$ such that $\nu(x) \in I \cup B \cup L$ if $x \in V_{ns}$ and $\nu(x) \in C \cup U$ if $x \in V_s$.*
- *θ is a conjunction of atomic TRCL formulae.*

Example 17. The following are mappings:

$$\mu_1 = (\{?F \to fire1,\ ?S \to \text{“}x \geq 1 \wedge x \leq 2 \wedge y \geq 1 \wedge y \leq 2''\},\ true)$$

$$\mu_2 = (\{?F \to fire1,\ ?S \to \_region1\},\ \_region1\ NTPP\ \text{“}x \geq 0 \wedge x \leq 10 \wedge y \geq 0 \wedge y \leq 10'')$$

$$\mu_3 = (\{?F \to fire1,\ ?S \to \_region1\},$$
$$(\_region1\ NTPP\ \_region2) \wedge (\_region2\ DC\ \text{“}x \geq 0 \wedge x \leq 1 \wedge y \geq 0 \wedge y \leq 1''))$$

The notions of domain and restriction for a mapping are now defined as follows.

Definition 8. *The* domain *of a mapping $\mu = (\nu, \theta)$, denoted by $dom(\mu)$, is the subset of V where the partial function ν is defined.*

Definition 9. *Let $\mu = (\nu, \theta)$ be a mapping with domain S and $W \subseteq S$. The* restriction *of the mapping μ to W denoted by $\mu_{|W}$ is the mapping $(\nu_{|W}, \theta)$ where $\nu_{|W}$ is the restriction of the partial function ν to W.*

We now define what it means for a mapping to be applied to a triple pattern.

Definition 10. *For a triple pattern p, we denote by $var(p)$ the variables appearing in p.*

Definition 11. *Let $\mu = (\nu, \theta)$ be a mapping. For a triple pattern p, we denote by $\mu(p)$ the triple obtained from p by replacing each variable $x \in var(p) \cap dom(\mu)$ by $\nu(x)$.*

We now introduce the notion of compatible mappings as in [PAG06].

Definition 12. *Two mappings μ_1 and μ_2 are* compatible *if for all $x \in dom(\mu_1) \cap dom(\mu_2)$ we have $\nu_1(x) = \nu_2(x)$.*

Example 18. Mappings μ_1 and μ_2 from Example 17 are not compatible, while mappings μ_2 and μ_3 are.

To take into account unknown literals, we also need to define another notion of compatibility of two mappings.

Definition 13. *Two mappings $\mu_1 = (\nu_1, \theta_1)$ and $\mu_2 = (\nu_2, \theta_2)$ are* possibly compatible *if for all $x \in dom(\mu_1) \cap dom(\mu_2)$, we have $\nu_1(x) = \nu_2(x)$ or at least one of $\nu_1(x), \nu_2(x)$ is an unknown literal from U.*

Example 19. Mappings μ_1, μ_2, and μ_3 from Example 17 are pairwise possibly compatible.

If two mappings are possibly compatible, then we can define their join as follows.

Definition 14. *Let $\mu_1 = (\nu_1, \theta_1)$ and $\mu_2 = (\nu_2, \theta_2)$ be possibly compatible mappings. The* join *$\mu_1 \bowtie \mu_2$ is a new mapping (ν_3, θ_3) where:*

i. $\nu_3(x) = \nu_1(x) = \nu_2(x)$ for each $x \in dom(\mu_1) \cap dom(\mu_2)$ such that $\nu_1(x) = \nu_2(x)$.

ii. $\nu_3(x) = \nu_1(x)$ for each $x \in dom(\mu_1) \cap dom(\mu_2)$ such that $\nu_1(x)$ is an unknown literal and $\nu_2(x)$ is a known literal.

iii. $\nu_3(x) = \nu_2(x)$ *for each* $x \in dom(\mu_1) \cap dom(\mu_2)$ *such that* $\nu_2(x)$ *is an unknown literal and* $\nu_1(x)$ *is a known literal.*

iv. $\nu_3(x) = \nu_1(x)$ *for* $x \in dom(\mu_1) \cap dom(\mu_2)$ *such that both* $\nu_1(x)$ *and* $\nu_2(x)$ *are unknown literals.*

v. $\nu_3(x) = \nu_1(x)$ *for* $x \in dom(\mu_1) \setminus dom(\mu_2)$*.*

vi. $\nu_3(x) = \nu_2(x)$ *for* $x \in dom(\mu_2) \setminus dom(\mu_1)$*.*

vii. θ_3 *is* $\xi_1 \wedge \xi_2 \wedge \xi_3$ *where:*

- ξ_1 *is* $\bigwedge_i \_v_i \; EQ \; \_t_i$*, where the* $\_v_i$*'s and* $\_t_i$*'s are all the pairs of unknown literals* $\nu_1(x)$ *and* $\nu_2(x)$ *from Case (iv) above. If there are no such pairs, then* ξ_1 *is true.*
- ξ_2 *is* $\theta_1 \wedge \bigwedge_i \_w_i \; EQ \; l_i$ *where the* $\_w_i$*'s and* l_i*'s are all the pairs of unknown literals* $\nu_1(x)$ *and known literals* $\nu_2(x)$ *from Case (ii) above. If there are no such pairs, then* ξ_2 *is* θ_1*.*
- ξ_3 *is* $\theta_2 \wedge \bigwedge_i \_w_i \; EQ \; l_i$ *where the* $\_w_i$*'s and* l_i*'s are all the pairs of unknown literals* $\nu_2(x)$ *and known literals* $\nu_1(x)$ *from Case (iii) above. If there are no such pairs, then* ξ_3 *is* θ_2*.*

The predicate EQ used in the above definition is the equality relation of RCC-8.

Example 20. If μ_1 and μ_2 are the mappings of Example 17, then:

$$\mu_1 \bowtie \mu_2 = (\{?F \rightarrow fire1, ?S \rightarrow \_region1\},$$
$$true \wedge \_region1 \; NTPP \; \text{“}x \geq 0 \wedge x \leq 10 \wedge y \geq 0 \wedge y \leq 10'' \wedge$$
$$\_region1 \; EQ \; \text{“}x \geq 1 \wedge x \leq 2 \wedge y \geq 1 \wedge y \leq 2'')$$

For two sets of mappings Ω_1 and Ω_2, the operations of join and union are now defined as follows.

$$\Omega_1 \bowtie \Omega_2 = \{\mu_1 \bowtie \mu_2 \mid \mu_1 \in \Omega_1, \mu_2 \in \Omega_2 \text{ are possibly compatible mappings}\}$$

$$\Omega_1 \cup \Omega_2 = \{\mu \mid \mu \in \Omega_1 \text{ or } \mu \in \Omega_2\}$$

The reader is invited to compare these definitions with the ones in [PAG06]. The new thing in sRDF$^{\text{i}}$ is that due to the presence of unknown spatial literals, we have to anticipate the possibility that two mappings from Ω_1 and Ω_2 are compatible. We anticipate these cases by adding relevant constraints to the constraint part of a mapping.

We can now define the result of evaluating a graph pattern over an sRDF$^{\text{i}}$ graph. The definitions are essentially the same as in [KK10] (only the first case is slightly different).

Definition 15. *Let* $D = (G, \phi)$ *be an* sRDF$^{\text{i}}$ *database over* T*,* p *a triple pattern and* P_1, P_2 *graph patterns. Evaluating a graph pattern* P *over database* D *is denoted by* $[[P]]_D$ *and is defined as follows:*

1. $[[p]]_D = \{\mu = (\nu, true) \mid dom(\mu) = var(p) \text{ and } \mu(p) \in G\}$*.*
2. $[[(P_1 \; AND \; P_2)]]_D = [[P_1]]_D \bowtie [[P_2]]_D$
3. $[[(P_1 \; UNION \; P_2)]]_D = [[P_1]]_D \cup [[P_2]]_D$

Example 21. *Let us now give an example of an evaluation of graph pattern P_1 AND P_2 over the database D of Example 16, where P_1, P_2 are the triple patterns $(?F, type, Fire)$ and $(?F, occuredIn, ?R)$ respectively. According to the above definition, we have:*

$$[[(P_1\ AND\ P_2)]]_D = [[P_1]]_D \bowtie [[P_2]]_D =$$

$$[[(?F, type, Fire)]]_D \bowtie [[(?F, occuredIn, ?R)]]_D =$$

$$\{(\{?F \rightarrow fire1\}, true)\} \bowtie \{(\{?F \rightarrow fire1, ?R \rightarrow \_region1\}, true)\} =$$

$$\{(\{?F \rightarrow fire1, ?R \rightarrow \_region1\}, true)\}$$

As we have explained in the introduction of this chapter, we will consider a simpler version of sSPARQL. As you can see from the following definition, we have simplified the notion of spatial term and do not consider any of the complex functions such as MBB, AREA, etc. that sSPARQL has.

Definition 16. *A* spatial term *is a known spatial literal from the set C or an unknown spatial literal from the set U, or a spatial variable from the set V_s.*

The evaluation of filters involving TRCL-constraints can now be defined as follows. Notice that the evaluation imposes extra constraints on the mappings that have already been computed. The evaluation of other kinds of filters involving non-spatial variables is as in standard SPARQL [PAG06].

Definition 17. *Given an* sRDF^i *database $D = (G, \phi)$ over T, a graph pattern P and an atomic TRCL-constraint R, we have:*

$$[[P\ FILTER\ R]]_D = \{\mu' = (\nu, \theta') \mid \mu = (\nu, \theta) \in [[P]]_D \text{ and } \theta' \text{ is } \theta \wedge R'\}$$

In the above formula, R' is an TRCL-constraint obtained from R by substituting each variable x by $\nu(x)$, R does not contain ground atomic formulas that are not satisfied by $\mathbf{M}_{\mathcal{L}_{RCL}}$, and all the ground atomic formulas of R' that are satisfied in $\mathbf{M}_{\mathcal{L}_{RCL}}$ have been eliminated.

Example 22. *Based on the evaluation of the graph pattern of Example 21, the evaluation of the graph pattern $((P_1\ AND\ P_2)\ FILTER\ R)$, where R is the TRCL-constraint $(?R\ NTPP$ "$x \geq 100 \wedge x \leq 1000 \wedge y \geq 100 \wedge y \leq 1000$"$)$, is the following:*

$$[[(P_1\ AND\ P_2)\ FILTER\ R]]_D = [[((?F, type, Fire)\ AND\ (?F, occuredIn, ?R))$$

$$FILTER\ (?R\ NTPP\ \text{“}x \geq 100 \wedge x \leq 1000 \wedge y \geq 100 \wedge y \leq 1000\text{''})]]_D =$$

$$\{(\{?F \rightarrow fire1, ?R \rightarrow \_region1\}, \_region1\ NTPP\ \text{“}x \geq 100 \wedge x \leq 1000 \wedge y \geq 100 \wedge y \leq 1000\text{''})\}$$

The next definition defines the concept of an sSPARQL query.

Definition 18. *An sSPARQL query is a pair (W, P) where W is a set of spatial and non-spatial variables and P is a graph pattern.*

Example 23. Let us consider the following query over the database of Example 16: "Find all fires that have occurred in a region which is a non-tangential proper part of the box defined by the points $(100, 100)$ and $(1000, 1000)$".

In the algebraic syntax of sSPARQL we consider in this chapter, this query can be expressed as follows:

$$(\{?F\},\ \{(?F, type, Fire)\}\ AND\ \{(?F, occuredIn, ?R)\}\ AND$$

$$FILTER\ (?R\ NTPP\ \text{“}x \geq 100 \wedge x \leq 1000 \wedge y \geq 100 \wedge y \leq 1000\text{''}))$$

The next definition defines the notion of answer to an sSPARQL query. Notice that in contrast to sSPARQL queries over sRDF graphs, sSPARQL queries over sRDF^{i} databases have answers that consist of conditional mappings so they might be hard to understand.

Definition 19. *Let $q = (W, P)$ be an sSPARQL query. The* answer *to q over a database $D = (G, \phi)$ is the set of mappings $\{\mu_{|W} \mid \mu \in [[P]]_D\}$.*

Example 24. The answer to the query from Example 23 can be obtained from the evaluation of the respective graph pattern from Example 22. The answer is a set that contains only the following mapping:

$$(\{?F \rightarrow fire1\},$$
$$\_region1\ NTTP\ \text{“}x \geq 100 \wedge x \leq 1000 \wedge y \geq 100 \wedge y \leq 1000''\text{)}$$

This answer is conditional. Because the information in the database of Example 16 is indefinite (the exact geometry of $\_region1$ is not known), we cannot say for sure whether $fire1$ satisfies the requirements of the query. These requirements are satisfied under the condition given in the above mapping.

It is easy to see how we can simplify the answer to a query q over database $D = (G, \phi)$ using the constraints involving unknown literals in the constraint store ϕ. There are two things that we can do. The first is to remove from the answer mappings $\mu = (\nu, \theta)$ where θ is a conjunction which contains a constraint σ such that $\phi \models \neg\sigma$. The second thing that we can do is simplify the conjunction θ of a mapping $\mu = (\nu, \theta)$ by deleting constraints σ such that $\phi \models \sigma$. Of course, this simplification step can also be performed in some intermediate step of query evaluation and not just at the end, when the final answer has been computed.

5.7 Certainty queries

Let us consider the query of Example 23 again. If we rephrase it to “Find fires that have *certainly* occurred in a region which is a non-tangential proper part of the box defined by the points $(100, 100)$ and $(1000, 1000)$”, then $fire1$ does not satisfy the query.

To be able to express such queries the notion of *possible world* [Kri63] is needed. A set of possible worlds can capture all the different states the world can be given the indefinite information in a database. These issues have been studied in detail in the literature of incomplete information in relational databases [IL84, Gra91] and we will take our methodology from these works.

Thus, an sRDF^{i} database $D = (G, \phi)$ represents a set of possible sRDF graphs each one corresponding to a possible state of the real world. This set of possible graphs captures completely the semantics of an sRDF^{i} database. The constraint store ϕ determines the number of possible sRDF graphs represented by D; there is one sRDF graph for each solution of ϕ obtained by considering the unknown literals of ϕ as variables.

The following example illustrates our discussion.

Example 25. *Let $D = (G, \phi)$ be the* sRDF^{i} *database given in Examples 15 and 16. The database D mentions a hotspot, which is located in a region, which is inside but does not intersect with the boundary of the box defined by the points $(0, 0)$ and $(3000, 3000)$. The same knowledge can be represented by an infinite set of possible sRDF graphs, one for each region inside the box defined by the points $(0, 0)$ and $(3000, 3000)$. Two of these graphs are:*

$$G_1 = \{(hotspot1, type, Hotspot),\ (fire1, type, Fire),\ (hotspot1, correspondsTo, fire1)$$
$$(fire1, occuredIn, \text{“}x \geq 1 \wedge x \leq 1000 \wedge y \geq 1 \wedge y \leq 1000''\text{)}\}$$
$$G_2 = \{(hotspot1, type, Hotspot),\ (fire1, type, Fire),\ (hotspot1, correspondsTo, fire1)$$
$$(fire1, occuredIn, \text{“}x \geq 2 \wedge x \leq 2000 \wedge y \geq 2 \wedge y \leq 2000''\text{)}\}$$

The problem of answering a certainty query like the one we gave in the beginning of this section amounts to checking whether all possible sRDF graphs satisfy the query. In the following we give formal definitions and examples for these concepts inspired by [IL84, Gra91].

In order to be able to go from sRDF$^{\text{i}}$ databases to the equivalent set of possible sRDF graphs, the notion of *valuation* is needed. Informally, a valuation maps an unknown literal to a specific box from C.

Definition 20. *A* valuation v *is a function from* U *to* C *assigning to each unknown literal a box from* C.

We will denote by $v(G)$ the application of a valuation v to graph G. $v(G)$ is obtained from G by replacing all unknown literals $\_l$ of G with $v(\_l)$ and leaving all other terms the same.

The set of valuations that satisfy the constraint store of an sRDF$^{\text{i}}$ database can help us to define the set of possible sRDF graphs that correspond to it. This set of graphs is denoted by *Rep*, following the notation of [Gra91].

Definition 21. *Let* $D = (G, \phi)$ *be an* sRDF$^{\text{i}}$ *database. The set of possible sRDF graphs represented by* D *is the following:*

$$Rep(D) = \{g \supseteq v(G) \mid \textit{there exists a valuation } v \textit{ such that } \mathbf{M}_{\mathcal{L}_{RCL}} \models v(\phi)\}$$

Note that RDF (and therefore sRDF$^{\text{i}}$) makes the open world assumption. Thus, we use "$\supseteq$" in the above definition. If we wanted to make the closed world assumption, we should have used $=$ instead.

Example 26. *Going back to Example 25, Rep*(D) *is infinite and contains all super graphs of all graphs formed by each solution of* ϕ *for variable* $\_region1$. G_1 *and* G_2 *are two of these graphs.*

In Section 5.6, we gave a *syntactic* definition of what an answer to a query q over an sRDF$^{\text{i}}$ database D is. Given that now we have given the semantics of an sRDF$^{\text{i}}$ database as a set of possible sRDF graphs, what is an appropriate semantic definition for a certainty query? This is captured by the following definition of *certain answer*.

Definition 22. *Let* q *be a query and* S *a set of sRDF graphs. The* certain answer *to* q *over* S *is the following set:*

$$\{\mu \mid \mu \textit{ is a mapping and } \mu \in \bigcap_{G \in S} q(G)\}$$

Example 27. *Considering again query* q *of Example 23, we have the following certain answer:*

$$\emptyset$$

Indeed, query q *asks for fires that certainly occurred inside the box defined by the points* $(100, 100)$ *and* $(1000, 1000)$. *The set* S *contains graphs* G *representing fires that have occurred outside of the given box, so there is no mapping contained in every* $q(G)$, *and thus, the intersection is the empty set.*

Example 28. Let us consider the following query over the database of Example 16: "Find all fires that have occurred in a region which is a non-tangential proper part of the box defined by the points $(0, 0)$ and $(4000, 4000)$". The certain answer for this query is the following mapping:

$$\{\{?F \rightarrow fire1\}\}$$

Let us now return to the syntactic definitions of query answering we gave in Section 5.6. Since the answer to a query can be quite complex syntactically, a user might be interested in asking whether a particular answer that interests him or her (i.e., technically, particular mappings μ) is in the certain answer to the query. This motivates the definition of the certainty problem originally defined for incomplete relational databases in [Gra91].

Next, we give the definition of the certainty problem.

Definition 23. *Let q be a sSPARQL query. The* certainty problem *for query q is, given a set of mappings M and an* sRDF$^{\mathrm{i}}$ *database D, to decide if $M \subseteq \bigcap q(Rep(D))$.*

Example 29. *Let D be the database of Example 16, q the sSPARQL query of Example 23, $S = Rep(D)$ the set of possible graphs of D as given in Example 25, and $M = \{(?F \to fire1)\}$ a set containing a single mapping. The answer to the certainty problem for q, given M and D, is "no". This is because according to the computation in Example 27 we have that $M \nsubseteq \bigcap q(Rep(D))$.*

5.8 Summary

In this chapter, we presented the model sRDF$^{\mathrm{i}}$ that allows the representation and querying of indefinite spatial information by combining sRDF graphs containing unknown spatial literals, and constraint stores containing qualitative spatial constraints, as they have traditionally been studied by the qualitative spatial reasoning community [RN07]. We defined the syntax and semantics of sRDF$^{\mathrm{i}}$ and how to evaluate sSPARQL queries over sRDF$^{\mathrm{i}}$ databases. We pointed out that answers to queries can be quite complex, and proposed the certainty problem as the technical means of capturing this complexity.

In the continuation of our work, which will be undertaken in Deliverable 2.3, we plan to do the following:

- Study how we can introduce the ideas of [Kou94d, Kou97] in sRDF$^{\mathrm{i}}$. This will allow us to answer certainty queries explicitly by computing *finite representations* of the infinite set of mappings that are answers to queries. The transfer of the results of [Kou94d, Kou97] to our case depends on whether the constraint language involved admits quantifier elimination. Since this is not possible for TRCL, a key question in our work is to determine subsets of TRCL (or other useful related languages) that have this property.
- Study the complexity of answering sSPARQL queries over sRDF$^{\mathrm{i}}$ databases and the associated certainty problem.

At a later stage of the project, based on the above results, we will design an extension of sSPARQL++ that allows us to deal with indefinite spatial information. This will allow sSPARQL++ to go beyond GeoSPARQL, the current OGC proposed standard for a geospatial extension of SPARQL. We plan to contribute these ideas and research directions to OGC through our participation to the GeoSPARQL working group of OGC.

6. Conclusions

In this deliverable we presented the semantics-based data models and languages that we will utilize in TELEIOS. First, we discussed the challenges for semantic data modeling that arise in the two use cases “A Virtual Observatory for TerraSAR-X data" and “Real-time fire monitoring based on continuous acquisitions of EO images and geospatial data". Then, we surveyed related work in the areas of image annotation, and spatial and temporal data models and query languages for XML, RDF, description logics and OWL.

We then presented a new version of the data model sRDF and the query language sSPARQL proposed recently by our partner NKUA that extends the W3C standards RDF and SPARQL for representing and querying spatial data in the Semantic Web. In the new version of sRDF and sSPARQL, called sRDF++ and sSPARQL++, we opt for a more practical solution to the problem of representing geospatial data and use the OGC standards Well-known Text and GML instead of linear constraints.

Finally, we extended sRDF and sSPARQL to allow the representation and querying of qualitative spatial relations as they have traditionally been studied by the qualitative spatial reasoning community. In the proposed extension of sRDF, called $sRDF^i$, we use a new kind of literals to represent spatial regions about which the known information is incomplete or indefinite.

We expect the data models and languages defined in this deliverable to have impact beyond TELEIOS: in the representation and querying of linked geospatial data sets on the Web, and in the development of a new generation of spatial reasoners for the Web.

In the forthcoming deliverable D2.3 “Theoretical results on query processing for RDF/SPARQL with time and space”, we plan to complete the theoretical developments of Chapter 5 and study the computational complexity of query processing for the languages presented in this deliverable. The results of this analysis will guide the implementation efforts to be undertaken in WP4 of TELEIOS.

Bibliography

[AAB+95] Ilsoo Ahn, Gad Ariav, Don Batory, James Clifford, Curtis E. Dyreson, Ramez Elmasri, Fabio Grandi, Christian S. Jensen, Wolfgang Kaefer, Nick Kline, Krishna Kulkarni, T. Y. Cliff Leung, Nikos Lorentzos, John F. Roddick, Arie Segev, Michael D. Soo, and Suryanarayana M. Sripada. *The TSQL2 Temporal Query Language*. Kluwer Academic Publishers, 1995.

[ADL+09] Sören Auer, Sebastian Dietzold, Jens Lehmann, Sebastian Hellmann, and David Aumueller. Triplify - lightweight linked data publication from relational databases. In *Proceedings of the 17th International Conference on World Wide Web*, page 621âĂŞ630, April 20-24 2009.

[AF01] Alessandro Artale and Enrico Franconi. A survey of temporal extensions of description logics. *Annals of Mathematics and Artificial Intelligence*, 30:171–210, March 2001.

[AHRW76] James R Anderson, Ernest E Hardy, John T Roach, and Richard E Witmer. A land use and land cover classification system for use with remote sensor data. *Development*, 2001, 1976.

[ALH09] Sören Auer, Jens Lehmann, and Sebastian Hellmann. Linkedgeodata: Adding a spatial dimension to the web of data. In Abraham Bernstein, David R. Karger, Tom Heath, Lee Feigenbaum, Diana Maynard, Enrico Motta, and Krishnaprasad Thirunarayan, editors, *International Semantic Web Conference*, volume 5823 of *Lecture Notes in Computer Science*, pages 731–746. Springer, 2009.

[All83] James F Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[ASR+06] Budak I. Arpinar, Amit Sheth, Cartic Ramakrishnan, Usery, Molly Azami, and Mei-Po Kwan. Geospatial Ontology Development and Semantic Analytics. *Transactions in GIS*, 10(4):551–575, July 2006.

[AT08] Alessandro Artale and David Toman. Decidable Reasoning over Timestamped Conceptual Models. In Franz Baader, Carsten Lutz, and Boris Motik, editors, *Proceedings of the 21st International Workshop on Description Logics (DL2008)*, volume 353 of *CEUR Workshop Proceedings*, Dresden, Germany, 2008. CEUR-WS.org.

[ATP+05] Thanos Athanasiadis, Vassilis Tzouvaras, Kosmas Petridis, Frederic Precioso, Yannis Avrithis, and Yiannis Kompatsiaris. Using a multimedia ontology infrastructure for semantic annotation of multimedia content. In *in Proceedgins of 5th International Workshop on Knowledge Markup and Semantic Annotation (SemAnnot '05*, 2005.

[ATSH09] R. Arndt, R. Troncy, S. Staab, and L. Hardman. Comm: A Core Ontology For Multimedia Annotation. In S. Staab and R. Studer, editors, *Handbook on Ontologies*. Springer Verlag, second edition, 2009.

[BCdC99] Philippe Balbiani, Jean-François Condotta, and Luis Fariñas del Cerro. A new tractable subclass of the rectangle algebra. In *IJCAI*, pages 442–447, 1999.

[Ben97] Brandon Bennet. *Logical Representations for Automated Reasoning about Spatial Relationships*. PhD thesis, School of Computer Studies, University of Leeds, 1997.

[Bis08] Yaser A. Bishr. Geospatial semantic web: Applications. In *Encyclopedia of GIS*, pages 391–398. Springer, 2008.

[BLOR05] François Bry, Bernhard Lorenz, Hans J. Ohlbach, and Mike Rosner. A Geospatial World Model for the Semantic Web. In François Fages and Sylvain Soliman, editors, *Principles and Practice of Semantic Web Reasoning*, volume 3703 of *Lecture Notes in Computer Science*, chapter 14, pages 145–159. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2005.

[BMM96] R. Brunelli, O. Mich, and C. M. Modena. A survey on video indexing. *Journal of Visual Communications and Image Representation*, 10:78–112, 1996.

[BP10] Sotiris Batsakis and Euripides G. M. Petrakis. SOWL: spatio-temporal representation, reasoning and querying over the semantic web. In *Proceedings of the 6th International Conference on Semantic Systems*, I-SEMANTICS '10, New York, NY, USA, 2010. ACM.

[BPS+05] Stephan Bloehdorn, Kosmas Petridis, Carsten Saathoff, Nikos Simou, Yannis Avrithis, H. Siegfried, Yiannis Kompatsiaris, and Michael G. Strintzis. Semantic annotation of images and videos for multimedia analysis. In *In Proceedings of the 2nd European Semantic Web Conference, ESWC 2005*, volume 3532, pages 592–607, 2005.

[CBGG97] Anthony G. Cohn, Brandon Bennett, John Gooday, and Nicholas Mark Gotts. Qualitative Spatial Representation and Reasoning with the Region Connection Calculus. *Geoinformatica*, 1(3):275–316, 1997.

[CCR93] Zhan Cui, Anthony G. Cohn, and David A. Randell. Qualitative and Topological Relationships in Spatial Databases. In *Advances in Spatial Databases*, 1993.

[CG09] M. Cristani and N. Gabrielli. Practical Issues of Description Logics for Spatial Reasoning. *AAAI*, 2009.

[Cho94] Jan Chomicki. Temporal query languages: A survey. In *ICTL '94: Proceedings of the First International Conference on Temporal Logic*, pages 506–534, London, UK, 1994. Springer-Verlag.

[dCLL05] Paulo C. da Costa, Kathryn B. Laskey, and Kenneth J. Laskey. PR-OWL: A Bayesian Ontology Language for the Semantic Web. In *International Semantic Web Conference (ISWC-2005)*, pages 23–33, 2005.

[DD02] Chris Date and Hugn Darwen. *Temporal Data and the Relational Model*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[DDKB07] Mihai Datcu, Sergio D'Elia, Roger King, and Lorenzo Bruzzone. Introduction to the special section on image information mining for earth observation data. *IEEE Transactions on Geoscience and Remote Sensing*, 45(4):795–798, April 2007.

[DK04] S. S. Durbha and R. L. King. Knowledge mining in earth observation data archives: a domain ontology perspective. In *Geoscience and Remote Sensing Symposium IEEE International*, pages 172–173b, 2004.

[DKS09] Stamatia Dasiopoulou, Ioannis Kompatsiaris, and Michael G. Strintzis. Applying fuzzy DLs in the extraction of image semantics. *Journal on Data Semantics XIV*, 14:105–132, 2009.

[dLSV+10] Alexander de León, Victor Saquicela, Luis M. Vilches, Boris Villazón-Terrazas, Freddy Priyatna, and Oscar Corcho. Geographical Linked Data: a Spanish Use Case. In *I-SEMANTICS*, 2010.

[Dor98] Christoph Dornheim. Undecidability of plane polygonal mereotopology. In *KR*, pages 342–355, 1998.

[DPP04] Zhongli Ding, Yun Peng, and Rong Pan. A Bayesian Approach to Uncertainty Modeling in OWL Ontology. In *Proceedings of the International Conference on Advances in Intelligent Systems - Theory and Applications*, 2004.

[DSDM02] M. Datcu, K. Seidel, S. D'Elia, and P. G. Marchetti. Knowledge-driven information mining in remote-sensing image archives. *ESA Bulletin 110*, 110:26–33, May 2002.

[EF91] Max J. Egenhofer and Robert D. Franzosa. Point-Set Topological Spatial Relations. In *Proceedings of International Journal of Geographical Information Systems*, volume 5(2), pages 161–174, 1991.

[Ege02] Max J. Egenhofer. Toward the semantic geospatial web. In *Proceedings of the 10th ACM international symposium on Advances in geographic information systems*, GIS '02, pages 1–4, New York, NY, USA, 2002. ACM.

[End72] H.B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 1972.

[Fra91] A.U. Frank. Qualitative spatial reasoning about cardinal directions. In *AUTOCARTO-CONFERENCE-*, volume 6, pages 148–148. ASPRS AMERICAN SOCIETY FOR PHOTOGRAMMETRY AND, 1991.

[GBE+00] Ralf Hartmut Güting, Michael H. Böhlen, Martin Erwig, Christian S. Jensen, Nikos A. Lorentzos, Markus Schneider, and Michalis Vazirgiannis. A Foundation for Representing and Quering Moving Objects. *ACM Transantions on Database Systems*, 25(1):1–42, 2000.

[GBM07a] Rolf Grütter and Bettina Bauer-Messmer. Combining owl with rcc for spatiotерminological reasoning on environmental data. In *OWLED*, 2007.

[GBM07b] Rolf Grütter and Bettina Bauer-Messmer. Towards spatial reasoning in the semantic web: A hybrid knowledge representation system architecture. In Sara Irina Fabrikant and Monica Wachowicz, editors, *The European Information Society*, Lecture Notes in Geoinformation and Cartography, pages 349–364. Springer Berlin Heidelberg, 2007.

[GC05] Roberto García and Òscar Celma. Semantic Integration and Retrieval of Multimedia Metadata. In Siegfried Handschuh, Thierry Declerck, and Marja-Riitta Koivunen, editors, *5th International Workshop on Knowledge Markup and Semantic Annotation (SemAnnot 2005)*, Galway, Ireland, November 2005. CEUR Workshop Proceedings.

[GDH08] J. Goodwin, C. Dolbear, and G Hart. Geographical Linked Data: The Administrative Geography of Great Britain on the Semantic Web. *Transactions in GIS*, 12:19–30, 2008.

[GHV05] Claudio Gutierrez, Carlos Hurtado, and Ro Vaisman. Temporal RDF. In *European Conference on the Semantic Web (ECSWï£¡05)*, pages 93–107, 2005.

[GHV07] Claudio Gutierrez, Carlos Hurtado, and Alejandro Vaisman. Introducing Time into RDF. *IEEE Transactions on Knowledge and Data Engineering*, 19(2):207–218, 2007.

[GKR+03] Stéphane Grumbach, Manolis Koubarakis, Philippe Rigaux, Michel Scholl, and Spiros Skiadopoulos. Spatio-temporal Models and Languages: An Approach Based on Constraints. In *Spatio-Temporal Databases: The CHOROCHRONOS Approach*, pages 177–201. Springer, 2003.

[GMH08] Rolf Grütter, Bettina B. Messmer, and Martin Hägeli. Extending an ontology-based search with a formalism for spatial reasoning. In *Proceedings of the 2008 ACM symposium on Applied computing*, SAC '08, pages 2266–2270, New York, NY, USA, 2008. ACM.

[GPP95] Michelangelo Grigni, Dimitris Papadias, and Christos H. Papadimitriou. Topological inference. In *IJCAI (1)*, pages 901–907, 1995.

[Gra91] G. Grahne. *The Problem of Incomplete Information in Relational Databases*, volume 554 of *Lecture Notes in Computer Science*. Springer Verlag, 1991.

[GRS98] Stéphane Grumbach, Philippe Rigaux, and Luc Segoufin. The DEDALE System for Complex Spatial Queries. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data (SIGMOD '98)*, pages 213–224, New York, NY, USA, 1998. ACM.

[Grz51] A. Grzegorczyk. Undecidability of some topological theories. *Fundamenta Mathematicae*, 38:137–152, 1951.

[GS03] Dengfeng Gao and Richard T. Snodgrass. Temporal slicing in the evaluation of xml queries. In *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB 2003)*, pages 632–643, 2003.

[GS05] Ralf Hartmut Guting and Markus Schneider. *Moving Objects Databases*. Morgan Kaufmann Publishers Inc., 2005.

[Gut94] Ralf Hartmut Guting. An introduction to spatial database systems. *VLDB Journal*, 3:357–399, 1994.

[HAB08] C. Hudelot, J. Atif, and I. Bloch. Fuzzy spatial relation ontology for image interpretation. *Fuzzy Sets and Systems*, 159(15):1929–1951, August 2008.

[HAmPS06] Farshad Hakimpour, Boanerges Aleman-meza, Matthew Perry, and Amit Sheth. Data Processing in Space, Time and Semantics Dimensions. In *Terra Cognita 2006 - Directions to the Geospatial Semantic Web, in conjunction with the Fifth International Semantic Web Conference (ISWC '06)*, 2006.

[HFK10] Frederik Hogenboom, Flavius Frasincar, and Uzay Kaymak. A review of approaches for representing RCC8 in OWL. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 1444–1445, New York, NY, USA, 2010. ACM.

[HHR+09] Michael Hausenblas, Wolfgang Halb, Yves Raimond, Lee Feigenbaum, and Danny Ayers. Scovo: Using statistics on the web of data. In *ESWC*, volume 5554 of *Lecture Notes in Computer Science*, pages 708–722. Springer, 2009.

[HKS02] Steffen Hölldobler, Tran D. Khang, and Hans-peter Störr. A Fuzzy Description Logic with Hedges as Concept Modifiers. In *In Proceedings InTech/VJFuzzy'2002*, pages 25–34, 2002.

[HLM99] Volker Haarslev, Carsten Lutz, and Ralf Möller. A Description Logic with Concrete Domains and a Role-forming Predicate Operator. In *Journal of Logic and Computation*, volume 9, 1999.

[HM02] Volker Haarslev and Ralf Möller. Practical Reasoning in Racer with a Concrete Domain for Linear Inequations. In *Proceedings of the International Workshop on Description Logics (DL-2002*, pages 91–98, 2002.

[HMW01] Volker Haarslev, Ralf Möller, and Michael Wessel. The description logic alcnhr+ extended with concrete domains: A practically motivated approach. In *In Proceedings of International Joint Conference on Automated Reasoning, IJCARâĂŹ2001*, pages 29–44. Springer, 2001.

[HMW04] Volker Haarslev, Ralf MÃűller, and Michael Wessel. Querying the semantic web with racer + nrql. In *In Proceedings of the KI-2004 International Workshop on Applications of Description Logics (ADLâĂŹ04*, 2004.

[HP04] Jerry R. Hobbs and Feng Pan. An ontology of time for the semantic web. *ACM Transactions on Asian Language Information Processing (TALIP)*, 3(1):66–85, March 2004.

[HPSB+04] I. Horrocks, P.F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. SWRL: A semantic web rule language combining OWL and RuleML. *W3C Member submission*, 21, 2004.

[HRSS03] A. Hoogs, J. Rittscher, G. Stein, and J. Schmiederer. Video content annotation using visual analysis and a large semantic knowledgebase. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 327–334, 2003.

[HSBW10] Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. YAGO2: a spatially and temporally enhanced knowledge base from Wikipedia. Research Report MPI-I-2010-5-007, Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany, November 2010.

[Hun01] J. Hunter. Adding Multimedia to the Semantic Web - Building an MPEG-7 Ontology. In *International Semantic Web Working Symposium (SWWS)*, 2001.

[HV06] Carlos A. Hurtado and Alejandro A. Vaisman. Reasoning with Temporal Constraints in RDF. In *Principles and Practice of Semantic Web Reasoning*, pages 164–178. Springer, 2006.

[HW05] L. Hollink and M. Worring. Building a visual ontology for video retrieval. In *Proceedings of the 13th annual ACM international conference on Multimedia*, MULTIMEDIA '05, pages 479–482, New York, NY, USA, 2005. ACM.

[hyd09] *An approach to comparing different ontologies in the context of hydrographical information*, May 2009.

[IL84] T. Imielinski and W. Lipski. Incomplete Information in Relational Databases. *Journal of ACM*, 31(4):761–791, 1984.

[ISO03] ISO 19115:2003 Geographic information – Metadata. Technical report, International Organization for Standardization (TC 211), 2003.

[JS99] Christian S. Jensen and Richard T. Snodgrass. Temporal data management. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):36–44, 1999.

[KDH05] D. Kolas, M. Dean, and J. Hebeler. Geospatial Semantic Web: Architecture of Ontologies. In *GeoSpatial Semantics*, pages 1–10, 2005.

[KG05] Yarden Katz and Bernardo C. Grau. B.C.: Representing Qualitative Spatial Information in OWL-DL. In *In: Proceedings of OWL: Experiences and Directions*, 2005.

[KK10] Manolis Koubarakis and Kostis Kyzirakos. Modeling and Querying Metadata in the Semantic Sensor Web: The Model stRDF and the Query Language stSPARQL. In Lora Aroyo, Grigoris Antoniou, Eero Hyvönen, Annette ten Teije, Heiner Stuckenschmidt, Liliana Cabral, and Tania Tudorache, editors, *ESWC (1)*, volume 6088 of *Lecture Notes in Computer Science*, pages 425–439. Springer, 2010.

[KKK09] Kostis Kyzirakos, Manolis Koubarakis, and Zoi Kaoudi. Data models and languages for registries in SemsorGrid4Env. Deliverable D3.1, SemSorGrid4Env, 2009.

[KKK10] Kostis Kyzirakos, Manos Karpathiotakis, and Manolis Koubarakis. Developing Registries for the Semantic Sensor Web using stRDF and stSPARQL (short paper). In *Proceedings of the 3rd International Workshop on Semantic Sensor Networks (SSN10), number* , volume 668, 11 2010.

[KKKK09] Kostis Kyzirakos, Zoi Kaoudi, Manolis Karpathiotakis, and Manolis Koubarakis. Distributed data structures and algorithms for a Semantic Sensor Grid registry. Deliverable D3.2 Version 1.0, SemSorGrid4Env, August 2009.

[KKR90] P.C. Kanellakis, G.M. Kuper, and P.Z. Revesz. Constraint Query Languages. In *Proceedings of the 9th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 299–313, 1990.

[Kol08] Dave Kolas. Supporting Spatial Semantics with SPARQL. In *Terra Cognita Workshop (Terra Cognita 2008)*, Karlsruhe, Germany, 2008.

[Kou94a] M. Koubarakis. Complexity Results for First-Order Theories of Temporal Constraints. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR'94)*, pages 379–390. Morgan Kaufmann, San Francisco, CA, May 1994.

[Kou94b] M. Koubarakis. Database Models for Infinite and Indefinite Temporal Information. *Information Systems*, 19(2):141–173, March 1994.

[Kou94c] M. Koubarakis. Foundations of Indefinite Constraint Databases. In A. Borning, editor, *Proceedings of the 2nd International Workshop on the Principles and Practice of Constraint Programming (PPCP'94)*, volume 874 of *Lecture Notes in Computer Science*, pages 266–280. Springer Verlag, 1994.

[Kou94d] Manolis Koubarakis. Database models for infinite and indefinite temporal information. *Information Systems*, 19:141–173, 1994.

[Kou97] M. Koubarakis. The Complexity of Query Evaluation in Indefinite Temporal Constraint Databases. *Theoretical Computer Science*, 171:25–60, January 1997. Special Issue on Uncertainty in Databases and Deductive Systems, Editor: L.V.S. Lakshmanan.

[Kou06] M. Koubarakis. Temporal CSPs. In *Handbook of Constraint Programming*. Elsevier, 2006.

[KPHZ10] Roman Kontchakov, Ian Pratt-Hartmann, and Michael Zakharyaschev. Interpreting topological logics over euclidean spaces. In *KR*, 2010.

[Kri63] S. Kripke. Semantical considerations on modal logic. *Acta philosophica fennica*, 16(1963):83–94, 1963.

[KS00] Manolis Koubarakis and Spiros Skiadopoulos. Querying temporal and spatial constraint networks in ptime. *Artificial Intelligence*, 123(1-2):223–263, 2000.

[KS07] Dave Kolas and Troy Self. Spatially Augmented Knowledgebase. In *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007)*, volume 4825 of *LNCS*, pages 785–794, Berlin, Heidelberg, November 2007. Springer Verlag.

[KSF+03] Manolis Koubarakis, Timos K. Sellis, Andrew U. Frank, Stéphane Grumbach, Ralf Hartmut Güting, Christian S. Jensen, Nikos A. Lorentzos, Yannis Manolopoulos, Enrico Nardelli, Barbara Pernici, Hans-Jörg Schek, Michel Scholl, Babis Theodoulidis, and Nectaria Tryfona, editors. *Spatio-Temporal Databases: The CHOROCHRONOS Approach*, volume 2520 of *Lecture Notes in Computer Science*. Springer, 2003.

[LL10] Sanjiang Li and Weiming Liu. Topological relations between convex regions. In *AAAI*, 2010.

[LM07] Carsten Lutz and Maja Miličić. A tableau algorithm for description logics with concrete domains and general tboxes. *J. Autom. Reason.*, 38:227–259, April 2007.

[Lop10] Xavier Lopez. Quering Spatial Data with SPARQL. ORACLE presentation sent to NKUA group by email on June 18, 2010.

[Lut04] C. Lutz. Combining Interval-based Temporal Reasoning with General TBoxes. *Artificial Intelligence*, 152(2):235–274, February 2004.

[LXL+05] Yanhui Li, Baowen Xu, Jianjiang Lu, Dazhou Kang, and Peng Wang. A Family of Extended Fuzzy Description Logics. In *Proceedings of the 29th Annual International Computer Software and Applications Conference - Volume 01*, COMPSAC '05, pages 221–226, Washington, DC, USA, 2005. IEEE Computer Society.

[Mah93] M. Maher. A Logic Programming View of CLP. In *Proceedings of the 10th International Conference on Logic Programming*, pages 737–753, 1993.

[MFKN07] V. Milea, F. Frasincar, U. Kaymak, and T. Noia. An OWL-based approach towards representing time in web information systems. In *4th International Workshop of Web Information Systems Modeling (WISM 2007)*, 2007.

[MGVO08] Alina Dia Miron, Jerome Gensel, and Marlene Villanova-Oliver. Semantic Analysis for the Geospatial Web - Application to OWL-DL Ontologies. In *Proceedings of the 8th Symposium on Web and Wireless Geographical Information System (W2GIS)*, volume 5373, pages 37–49. Springer, 2008.

[MHCW08] Phivos Mylonas, Hermann Hellwagner, Pablo Castells, and Manolis Wallace. Signal, image and video processing (SIVP) special issue on âĂİ multimedia semantics, adaptation and personalizationâĂİ Editorial. *Signal, Image and Video Processing*, 2(4):287–288, December 2008.

[MPE01] MPEG-7. Multimedia Content Description Interface. Standard No. ISO/IEC 15938, 2001.

[Neb04] Douglas D. Nebert. *Developing Spatial Data Infrastructures: The SDI Cookbook*. GSDI, 2004.

[NVOH05] Frank Nack, Jacco Van Ossenbruggen, and Lynda Hardman. That Obscure Object of Desire: Multimedia Metadata on the Web (Part II). *IEEE Multimedia*, 12(1):54–63, 2005.

[OGC07] Open Geospatial Consortium Inc OGC. OpenGIS Geography Markup Language(GML) Encoding Standard. OpenGIS Implementation Standard, 08 2007. http://portal.opengeospatial.org/files/?artifact_id=20509.

[OGC10a] Open Geospatial Consortium Inc OGC. Geography Markup Language (GML) simple features profile. OpenGIS Implementation Standard Profile, 07 2010. http://portal.opengeospatial.org/files/?artifact_id=39853.

[OGC10b] Open Geospatial Consortium Inc OGC. GeoSPARQL - A geographic query language for RDF data. Proposal for an OGC Draft Candidate Standard, 04 2010.

[OGC10c] Open Geospatial Consortium Inc OGC. OpenGIS Geography Markup Language (GML) Application Schema for Earth Observation Products. OpenGIS Implementation Standard, 02 2010.

[OGC10d] Open Geospatial Consortium Inc OGC. OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option. OpenGIS Implementation Standard, 08 2010. http://portal.opengeospatial.org/files/?artifact_id=25354.

[OGC10e] Open Geospatial Consortium Inc OGC. OpenGIS Implementation Standard for Geographic information - Simple feature access - Part 1: Common Architecture. OpenGIS Implementation Standard, 08 2010. http://portal.opengeospatial.org/files/?artifact_id=25355.

[ORS10] Ermelinda Oro, Massimo Ruffolo, and Steffen Staab. Sxpath: extending xpath towards spatial querying on web documents. *Proceedings VLDB Endowment*, 4:129–140, November 2010.

[PAG06] Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. Semantics and Complexity of SPARQL. In *Proceedings of the 5th International Semantic Web Conference (ISWC 2006)*, pages 30–43, 2006.

[PAG09] Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. Semantics and complexity of sparql. *ACM Trans. Database Syst.*, 34(3), 2009.

[PAS+06] Kosmas Petridis, Dionysios Anastasopoulos, Carsten Saathoff, Yiannis Kompatsiaris, and Steffen Staab. MOntoMat-Annotizer: Image annotation. linking ontologies and multimedia low-level features. In *In KES 2006 - 10th Intnl. Conf. on Knowledge Based, Intelligent Information and Engineering Systems*, 2006.

[PCJ+07] Ross S. Purves, Paul Clough, Christopher B. Jones, Avi Arampatzis, Benedicte Bucher, David Finch, Gaihua Fu, Hideo Joho, Awase Khirni Syed, Subodh Vaid, and Bisheng Yang. The design and implementation of spirit: a spatially aware search engine for information retrieval on the internet. *International Journal of Geographical Information Science*, 21(7):717–745, 2007.

[Per02] George Percivall. ISO 19119 and OGC Service Architecture. In *Proceedings of FIG XXII International Congress*, Washington D.C., USA, April 2002.

[Per08] Matthew Perry. *A Framework to Support Spatial, Temporal and Thematic Analytics over Semantic Web Data*. PhD thesis, Wright State University, 2008.

[PHS06] Matthew Perry, Farshad Hakimpour, and Amit P. Sheth. Analyzing Theme, Space, and Time: an Ontology-based Approach. In *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems (GIS'06)*, pages 147–154, 2006.

[PR01] Agnes Voisard Philippe Rigaux, Michel Scholl. *Spatial Databases*. Morgan Kaufmann Publishers, 2001.

[PS04] Bijan Parsia and Evren Sirin. Pellet: An OWL DL Reasoner. In *3rd International Semantic Web Conference (ISWC2004)*, 2004.

[PSHJ07] Matthew Perry, Amit P. Sheth, Farshad Hakimpour, and Prateek Jain. Supporting Complex Thematic, Spatial and Temporal Queries over Semantic Web Data. In *Proceedings of the Second International Conference on GeoSpatial Semantics (GeoS)*, pages 228–246, 2007.

[PT97] Dimitris Papadias and Yannis Theodoridis. Spatial relations, minimum bounding rectangles, and spatial data structures. *International Journal of Geographical Information Science*, 11(2):111–138, 1997.

[PUS08] Andrea Pugliese, Octavian Udrea, and V. S. Subrahmanian. Scaling RDF with Time. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 605–614, 2008.

[RCC92a] David A. Randell, Zhan Cui, and Anthony G. Cohn. A Spatial Logic based on Regions and Connection. In *PROCEEDINGS 3RD INTERNATIONAL CONFERENCE ON KNOWLEDGE REPRESENTATION AND REASONING*, 1992.

[RCC92b] David A. Randell, Zhan Cui, and Anthony G. Cohn. A spatial logic based on regions and connection. In *KR*, pages 165–176, 1992.

[Ren02] Jochen Renz. *Qualitative spatial reasoning with topological information*. Springer, Berlin, Heidelberg, 2002.

[Rev02] Peter Z. Revesz. *Introduction to Constraint Databases*. Springer, 2002.

[RN99] J. Renz and B. Nebel. On the Complexity of Qualitative Spatial Reasoning: A Maximal Tractable Fragment of the Region Connection Calculus. *Artificial Intelligence*, 1-2:95–149, 1999.

[RN01] Jochen Renz and Bernhard Nebel. Efficient methods for qualitative spatial reasoning. *J. Artif. Int. Res.*, 15:289–318, October 2001.

[RN07] Jochen Renz and Bernhard Nebel. Qualitative spatial reasoning using constraint calculi. In *Handbook of Spatial Logics*, pages 161–215. Springer, 2007.

[RSSG03] Philippe Rigaux, Michel Scholl, Luc Segoufin, and Stephane Grumbach. Building a constraint-based spatial database system: model, languages, and implementation. *Information Systems*, 28(6):563–595, 2003.

[RSV00] P. Rigaux, M. Scholl, and A. Voisard. *Introduction to Spatial Databases: Applications to GIS*. Morgan Kaufmann, Reading, MA, 2000.

[RTSS98] C. S. Jensen R. T. Snodgrass, M. H. Bohlen and A. Steiner. Transitioning temporal support in tsql2 to sql3. In O. Etzion, S. Jajodia, and S. Sripada, editors, *Temporal Databases: Research and Practice*, pages 150–194. Springer, 1998.

[RV08] Flavio Rizzolo and Alejandro A. Vaisman. Temporal xml: Modeling, indexing, and query processing. *VLDB J.*, 17(5):1179–1212, 2008.

[SA85] R. Snodgrass and I. Ahn. A Taxonomy of Time in Databases. In *Proceedings of ACM SIGMOD International Conference on Management of Data*, pages 236–246, 1985.

[SAEJ07] Philip D. Smart, Alia I. Abdelmoty, Baher A. El Geresy, and Christopher B. Jones. A framework for combining rules and geo-ontologies. In *Proceedings of the 1st international conference on Web reasoning and rule systems*, RR'07, pages 133–147, Berlin, Heidelberg, 2007. Springer-Verlag.

[SASK08] N. Simou, Th Athanasiadis, G. Stoilos, and S. Kollias. Image indexing and retrieval using expressive fuzzy description logics. *Signal, Image and Video Processing*, 2(4):321–335, December 2008.

[SC10] Sven Schade and Simon Cox. Linked Data in SDI or How GML is not about Trees. In *13^{th} AGILE International Conference on Geographic Information Science*, 2010.

[Sch86] A. Schrijver, editor. *Theory of Integer and Linear Programming*. Wiley, 1986.

[SFGP09] Mari C. Suarez-Figueroa and Asuncion Gomez-Perez. NeOn Methodology for Building Ontology Networks: a Scenario-based Methodology. In SERVICES & SEMANTIC TECHNOLOGIES S3T 2009 International Conference on SOFTWARE, editor, *Proceedings of the International Conference on SOFTWARE, SERVICES & SEMANTIC TECHNOLOGIES (S3T 2009)*, 2009.

[SGD10] Sven Schade, Carlos Granell, and Laura Díaz. Augmenting SDI with Linked Data. In *Proceedings of the Workshop on Linked Spatiotemporal Data*, 2010.

[SHH+07] C. G. M. Snoek, B. Huurnink, L. Hollink, M. de Rijke, G. Schreiber, and M. Worring. Adding Semantics to Detectors for Video Retrieval. *IEEE Transactions on Multimedia*, 9(5):975–986, August 2007.

[SKS+07] Chi-Ren Shyu, M. Klaric, G. J. Scott, A. S. Barb, C. H. Davis, and K. Palaniappan. GeoIRIS: Geospatial information retrieval and indexing System—Content mining, semantics modeling, and complex queries. *Geoscience and Remote Sensing, IEEE Transactions on*, 45(4):839–852, 2007.

[Sno87] R. Snodgrass. The Temporal Query Language TQuel. *ACM Transcactions on Database Systems*, 12(2):247–298, June 1987.

[SP08] A. Sheth and M. Perry. Traveling the Semantic Web through Space, Time, and Theme. *IEEE Internet Computing*, 12(2):81–86, 2008.

[SPG+07] E. Sirin, B. Parsia, B. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):51–53, June 2007.

[SS09] Markus Stocker and Evren Sirin. Pelletspatial: A hybrid rcc-8 and rdf/owl reasoning and query engine, 2009.

[SSB+09] Simon Schenk, Carsten Saathoff, Anton Baumesberger, Frederik Jochum, Er Kleinen, Steffen Staab, and Ansgar Scherp. SemaPlorerâĂŤInteractive Semantic Exploration of Data and Media based on a Federated Cloud. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7:298–304, 2009.

[SSP+07] Giorgos Stoilos, Giorgos B. Stamou, Jeff Z. Pan, Vassilis Tzouvaras, and Ian Horrocks. Reasoning with very expressive fuzzy description logics. *J. Artif. Intell. Res. (JAIR)*, 30:273–320, 2007.

[SSSK06] Giorgos Stoilos, Nikos Simou, Giorgos B. Stamou, and Stefanos D. Kollias. Uncertainty and the semantic web. *IEEE Intelligent Systems*, 21(5):84–87, 2006.

[ST07] Arno Scharl and Klaus Tochtermann, editors. *The Geospatial Web: How Geobrowsers, Social Software and the Web 2.0 are Shaping the Network Society*. Springer, London, 2007.

[STMD08] R. Singh, A. Turner, M. Maron, and A. Doyle. GeoRSS: Geographically Encoded Objects for RSS Feeds. Retrieved May 18, 2008, from `http://georss.org/gml`, 2008.

[Str05] Umberto Straccia. Description logics with fuzzy concrete domains. In Fahiem Bachus and Tommi Jaakkola, editors, *21st Conference on Uncertainty in Artificial Intelligence (UAI-05)*, pages 559–567, Edinburgh, Scotland, 2005. AUAI Press.

[SWS+00] Arnold W. M. Smeulders, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. Content-Based image retrieval at the end of the early years. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(12):1349–1380, December 2000.

[TPC04] Chrisa Tsinaraki, Panagiotis Polydoros, and Stavros Christodoulakis. Integration of OWL Ontologies in MPEG-7 and TV-Anytime Compliant Semantic Indexing. In *Advanced Information Systems Engineering*, pages 143–161. Springer, 2004.

[TWHC04] Goce Trajcevski, Ouri Wolfson, Klaus Hinrichs, and Sam Chamberlain. Managing Uncertainty in Moving Objects Databases. *ACM Transactions of Database Systems*, 29(3):463–507, 2004.

[VBVTS+10] Luis M. Vilches-Blázquez, Boris Villazón-Terrazas, Victor Saquicela, Alexander de León, Oscar Corcho, and Asunción Gómez-Pérez. GeoLinked Data and INSPIRE through an Application Case. In *GIS*, pages 446–449, 2010.

[VKvB89] Marc Vilain, Henry Kautz, and Peter van Beek. Constraint Propagation Algorithms for Temporal Reasoning: A Revised Report. In D.S. Weld and J. de Kleer, editors, *Readings in Qualitative Reasoning about Physical Systems*, pages 373–381. Morgan Kaufmann, 1989.

[VONH04] Jacco Van Ossenbruggen, Frank Nack, and Lynda Hardman. That Obscure Object of Desire: Multimedia Metadata on the Web (Part I). *IEEE Multimedia*, 11(4):54–63, 2004.

[VSS+08] M. Vacura, V. Svatek, C. Saathoff, T. Franz, and R. Troncy. Describing Low-Level Image Features Using The COMM Ontology. In *ICIP Workshop on Multimedia Information Retrieval: New Trends and Challenges*. IEEE, October 2008.

[Wes02] Michael Wessel. On Spatial Reasoning with Description Logics. In *Proceedings of the International Workshop on Description Logics 2002 (DL2002), number 53 in CEUR-WS*, volume 31, pages 156–163, 2002.

[WM09] M. Wessel and R. Moller. Flexible software architectures for ontology-based information systems. *Journal of Applied Logic*, 7(1):75–99, March 2009.

[ZG02] Rong Zhao and William I. Grosky. Narrowing the semantic gap - improved Text-Based web document retrieval using visual features. *IEEE Transactions on Multimedia*, 4:189–200, 2002.