FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Applying Semantic Web Concepts to GeoRSS

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Software Engineering/Internet Computing

eingereicht von

Homa Rezaie
Matrikelnummer 0325518

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: o.Univ.-Prof. Dipl.-Ing. Mag. Dr. A Min Tjoa
Mitwirkung: Univ.-Ass. Mag. Dipl.-Ing. Dr. Amin Anjomshoaa

Wien, 03.10.2011

(Unterschrift Verfasserin) (Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Applying Semantic Web Concepts to GeoRSS

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Software Engineering/Internet Computing

by

Homa Rezaie
Registration Number 0325518

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: o.Univ.-Prof. Dipl.-Ing. Mag. Dr. A Min Tjoa
Assistance: Univ.-Ass. Mag. Dipl.-Ing. Dr. Amin Anjomshoaa

Vienna, 03.10.2011

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Homa Rezaie
Am Schöpfwerk 31, 1120 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum) (Unterschrift Verfasserin)

Acknowledgements

I would like to express my sincere gratitude to my direct thesis advisor, Dr. Amin Anjomshoaa, for providing me with a captivating topic as well as for his supervision and unstinting, patient assistance during the course of writing and improving this thesis. His dedication and valuable suggestions have been conducive to its completion.

Moreover, I thank O.Univ.Prof. Dipl.-Ing. Dr.techn. A Min Tjoa for the final evaluation of this work.

Abstract

Today, a tremendous amount of information on the Internet is available to end-users in the form of millions of public web pages.

Some of the user applications on the web contain elements in the form of spatial data or geographical data. Geographical data is information which explains the geographic location in terms of features and boundaries on the Earth's surface. The technologies used in Web 2.0 have increased the importance and widespread use of geospatial context and location-based services. The Web 2.0 offers a new usage of geospatial data for creating a Temporal Browsing Technique, called GeoRSS, which can increase the potential uses of geographical information.

GeoRSS is an emerging standard for encoding location as part of a web feed. These web feeds are used to describe feeds of content such as news articles, text blogs, audio blogs and video blog entries [Wiki].

There is an increasing need to both extend the functionality of GeoRSS services and facilitate the integration of Geo-information. The Semantic Web offers better retrieval methods to facilitate query processes as it improves the information query by better expressing the context and meaning of the queries and information resources.

This thesis introduces the concept of Semantic GeoRSS as an information integration approach that is based on and combines Semantic Web and GeoRSS. The creation of a Semantic GeoRSS requires the presentation of GeoRSS with a formal semantics. For this purpose, appropriate domain ontologies provide a formal vocabulary for describing the GeoRSS information. By means of these GeoRSS ontologies, complex queries can be formally described and run on semantically described resources. In this case, ontologies are used for representing the annotations and the domain of interest.

Meshing the two approaches to create a Semantic GeoRSS increases the potential uses of Geospatial information for more serious use-cases. The thesis will explore this potential by discussing four use cases of Semantic GeoRSS in order to clarify the relation between the Semantic Web and GeoRSS information.

Kurzfassung

Heute haben Benutzer durch das Internet Zugang zu einer überwältigenden Menge an Information in Form von Millionen und Abermillionen öffentlicher Webseiten.

Einige der Benutzeranwendungen im Internet enthalten Elemente in Form von räumlichen oder geographischen Daten. Geographische Daten sind all jene Informationen, die eine geographische Position im Hinblick auf ihre Eigenschaften oder Grenzen auf der Erdoberfläche beschreiben. Die Technologien, die im Web 2.0 zum Einsatz kommen, haben Bedeutung und Verbreitung von Diensten, die auf geographischen oder räumlichen Daten basieren, deutlich zunehmen lassen. Das Web 2.0 bietet neue Nutzungswege um mit geographischen Daten eine Temporal Browsing Technique, auch bekannt als GeoRSS, umzusetzen, welche die potentiellen Einsatzmöglichkeiten von geographischer Information erweitert.

GeoRSS ist ein sich laufend entwickelnder Standard für die Kodierung des Aufenthaltsortes im Rahmen eines Web Feed. Diese Web Feeds werden verwendet, um Inhalte laufend zu aktualisieren, unter anderem bei Nachrichten, Text-Blogs, Audio-Blogs und Video-Blogs [Wiki].

Der Bedarf dafür, einerseits die Funktionalität von GeoRSS-Diensten zu erweitern und andererseits die Integration von geographischer Information zu erreichen, ist im Steigen begriffen. Das Semantische Web bietet verbesserte Möglichkeiten für den Informationsabruf zur Beantwortung von Suchanfragen, da es Kontext und Bedeutung der Anfrage und der Informationsressourcen besser ausdrückt.

Diese Diplomarbeit stellt das Konzept des Semantischen GeoRSS als Methode zur Informationsintegration vor und nutzt dafür eine Kombination des Semantischen Web und GeoRSS. Die Entwicklung eines Semantischen GeoRSS bedarf eines GeoRSS mit einer formalen Semantik. Für diesen Zweck bieten entsprechende Domänenontologien ein formales Vokabular um die GeoRSS-Information zu beschreiben. Mit Hilfe dieser GeoRSS-Ontologien können komplexe Suchanfragen formal beschrieben werden und auf semantisch beschriebene Ressourcen zurückgreifen. In diesem Fall werden Ontologien benutzt, um relevante Anmerkungen und Domänen zu repräsentieren.

Die Verknüpfung der beiden Methoden um ein Semantisches GeoRSS zu erschaffen erweitert die möglichen Anwendungsgebiete von geographischer und räumlicher Information in

praktischen Anwendungsfällen. Die Diplomarbeit beleuchtet das damit erschließbare Potential indem vier Anwendungsfälle für GeoRSS dargestellt und die Verknüpfung des Semantischen Web mit GeoRSS-Information diskutiert wird.

Contents

Contents ix

List of Figures x

Listings xi

1 Introduction 1
- 1.1 Problem Statements . . . 1
- 1.2 Thesis Goal . . . 3
- 1.3 Thesis Method . . . 3
- 1.4 Thesis Structure . . . 4

2 State-of-the-art 7
- 2.1 Available Geo-ontologies . . . 7
- 2.2 Available Geo-Query . . . 12
- 2.3 Available Location Search . . . 19
- 2.4 Available Geospatial Data base Generation . . . 27

3 Concepts Description 31
- 3.1 Semantic Web Technologie . . . 31
- 3.2 Semantic Web & Geospatial Information . . . 37
- 3.3 Geographical Really Simple Syndication . . . 40

4 Use Cases 51
- 4.1 Public-Transport for People with Special Needs (PTPSN) . . . 51
- 4.2 Vehicle Accident and Emergency Services (VA & ES) . . . 55
- 4.3 Tourism Consulter . . . 59
- 4.4 Smart Guide Airplane Landing (SGAL) . . . 62

5 Technical Solutions 67
- 5.1 Functionality Requirements . . . 67
- 5.2 Conceptual Design . . . 69

6 Implementation & Validation 85

6.1 Parser Implementation 85
6.2 Evaluations and Validation of Solution 87
6.3 Validation 100

7 Conclusions **105**
7.1 Conclusions 105

Bibliography **107**

A Appendix A **113**
A.1 VA & ES Main Ontology Model 113
A.2 VA & ES Temporal Ontology 120
A.3 Hospital Surgery GeoRSS feed 122
A.4 Police Station GeoRSS feed 124
A.5 Firebrigade Aviation Unit GeoRSS feeds 126
A.6 Accident GeoRSS feeds 128

List of Figures

2.1 Conceptual design of the geo-ontology [Jones] 11
2.2 Class diagram for the SPAUKGraph and its relation to the IndexProcessors [Kolas] 13
2.3 MashQL Editor [Jarrar] 15
2.4 The architecture of GXQuery [Chen] 17
2.5 DAGIS Query Interface [Ashraful] 19
2.6 ESSE Web client sample screenshot [Zhizhin] 23
2.7 SPIRIT structured query interface [Purves] 25
2.8 A screen shot of the STEWARD [Lieberman] 26

3.1 Graph representation of triple 33
3.2 Result query of required services 35
3.3 Result query for hospital resource 35
3.4 geographical information on the Google Earth 38
3.5 A fragment of the Domain Ontology [Baglioni] 40
3.6 The GeoRSS Information Model [GeoRSS] 44
3.7 Box Mode [GeoRSS] 45
3.8 Polygon Mode [GeoRSS] 45

5.1 Overview of proposed solution 69

5.2 SW & GeoRSS use sequence diagram . . . 71
5.3 Proposed SW & GeoRSS approach . . . 72
5.4 VA & ES use-case architecture . . . 79
5.5 Inference rules in the temporal ontology . . . 80
5.6 Detail of inference rules in the temporal ontology . . . 81
5.7 AV & ES Temporal Ontology Model . . . 84

6.1 Query point and its nearest location point . . . 87
6.2 Query point and its nearest location point . . . 102
6.3 Query point and its nearest location point . . . 102
6.4 Distance the nearest Location . . . 103

Listings

2.1 SPAUK query . . . 13
2.2 SPRQL-ST query . . . 14
3.1 RDF document example . . . 33
3.2 SPARQL query example 1 . . . 34
3.3 SPARQL query example 2 . . . 35
3.4 Rule example . . . 37
3.5 RSS feed example . . . 41
3.6 TU Wien Library RSS feed . . . 42
3.7 GeoRSS Simple encoding . . . 48
3.8 GeoRSS GML example . . . 49
5.1 GoRSS Semantic . . . 72
5.2 GeoRSS Metro Station Karlsplatz . . . 74
5.3 Tourism Consulter temporal ontology (museum information) . . . 75
5.4 Rules of ontology Tourism Consulter . . . 76
5.5 First SPARQL query . . . 77
5.6 Second SPARQL query . . . 78
5.7 VA & ES information in temporal ontology . . . 81
5.8 VA & ES Rules . . . 82
6.1 Ontology Model of the VA & ES . . . 88
6.2 VA & ES Services . . . 89
6.3 VA & ES Hospital Services . . . 90
6.4 VA & ES Police Services . . . 90
6.5 VA & ES Fire Brigade Services . . . 90

6.6 VA & ES Defined-class 91
6.7 Police GeoRSS feed 94
6.8 Hospital GeoRSS feed 95
6.9 Fire brigade GeoRSS feed 96
6.10 Accident Event 97
6.11 Accident Rule 98
6.12 First SPARQL query for the VA & ES 99
6.13 Second SPARQL query for the VA & ES 99
6.14 VA & ES temporal ontology 100
6.15 Applying accident rule into temporal ontology 101
A.1 VA & ES Main Ontology Model 113
A.2 VA & ES Temporal Ontology 120
A.3 GeoRSS AKH Hospital 122
A.4 GeoRSS Lainz Hospital 122
A.5 GeoRSS St.Anna 123
A.6 GeoRSS Police patrol Heldenplatz station 124
A.7 GeoRSS Police patrol station Arndstrße 124
A.8 GeoRSS Brigittenau station 126
A.9 GeoRSS Floridsdorf station 126
A.10 GeoRSS Accident event MariahilferSt _Accident1 128
A.11 GeoRSS Accident event karlsplatz _Accident2 128

CHAPTER 1

Introduction

Setting up the collaboration between GeoRSS and Semantic Web Service is the main step towards developing geographic applications. It is also a good starting point for extending the use of Semantic Web technologies in fields such as GPS or location-based queries. In addition, these extensions of semantic web technology also improve the capabilities/usefulness of location information for the various user groups. In general, the main goal of Semantic GeoRSS is to propose a solution relevant for creating a complete RDF data set, drawing on various web data sources based on the GeoRSS ontology. The main component to be realized is an intelligent search engine for finding information about a particular location from various available sources. This thesis investigates four use cases of the GeoRSS format in order to determine potential applications for combining/meshing the two web technologies. The use cases have been selected from different domains. Each domain in a RDF data set is created according to an individual ontology, which is embedded in the overall/general ontology according to GeoRSS data.

Relations between all use cases for each domain are structured according to Urls, which are based on a related ontology. A use case RDF data set can be used by another use case application, e.g. the Public -Transport ontology can be used for the use case Hotel-Search. Both the process and embedding of location information from GeoRSS in the respective ontology will be discussed in detail in the following sections.

1.1 Problem Statements

Today, a tremendous amount of information on the Internet is available to end-users in the form of millions of public web pages. Some of the user applications on the web contain elements in the form of spatial data or geographical data. Geographical data is information which explains the geographic location in terms of features and boundaries on the Earth's surface. The

technologies used in Web 2.0 have increased the importance and widespread use of geospatial context and location-based services. The Web 2.0 offers a new usage of geospatial data for creating a Temporal Browsing Technique, called GeoRSS, which can increase the potential uses of geographical information.

GeoRSS is an emerging standard for encoding location as part of a web feed. These web feeds are used to describe feeds of content such as news articles, text blogs, audio blogs and video blog entries [Wiki].

While such geographical information is easy to understand for human users, it is difficult for computers to comprehend the location semantics and their context information. In this thesis, the following challenges will be explored:

- How the existing RSS tools and services can be improved for gathering, interpreting, analyzing and visualizing of geo-information?
- How the event's geographical location and event description can be formally defined and interpreted by computers for different use cases?
- How can an automated service find a specific item, according to its proximity to the event's current location (e.g. the hospital nearest to our position for a specific type of injury)?

As a way of answering these questions, we propose the use of Semantic Web and GeoRSS technologies. The aim at the heart of the concept and vision of the Semantic Web is to make web information practically process-able by a computer, thereby making it more effective for its users [Schwitter]. This increase in effectiveness is achieved by locating, collating and relating content, and then drawing conclusions from information found in more separate sources.

Today's on the World Wide Web (WWW), there is different efforts for representation the Geospatial information resources such as navigation systems or Web Maps (e.g. NASA, Google Map, Google Erath). The goal of web-based Map Navigation is to enable people to access location information on the web as easily in the future as they can access documents today. It is a very promising step towards realizing the vision of a Data Web, a reality of information space that allows for the reuse and combination of all published data.

The major problems of location applications based on Semantic web technology, and of web-based Map Navigation in particular, is the lack of services that match and store data from various sources for creating RDF datasets compatible with location information.

In most cases, RDF repositories either store different categories of information in RDF format or store links to other repositories. However, the location information itself is rarely developed with semantic web concepts in mind. Furthermore, they don't have any specific functionality for an intelligent search engine. The state of current RDF repositories is thus insufficient

for many location information applications. This also shows in the lack of sufficient or accurate information that is ultimately available to the user. Location information, which is exchanged through Really Simple Syndication (RSS) web services, can be found at different sources in XML/RDF format. From there, comprehensive structure repositories with complete RDF dataset can be created based on a suitable GeoRSS ontology.

1.2 Thesis Goal

The goal of this thesis is to create a comprehensive structured geographical dataset from an available repository of location information on the Web by using GeoRSS and Semantic Web technology. We propose an approach and components required for creating a complete machine-processable dataset based on geographical location information ontologies from different GeoRSS feeds on the Web. The main proposed component is a temporal ontology for embedding geographical information location about events/services location from various GeoRSS feeds. A further required component is a parser in order to parse the geo-data from different GeoRSS feeds into a machine-processable format.

The geographical location information from each use-case GeoRSS in the final dataset is created according to a main temporal ontology, which is based on a temporal ontology model Relations between all geographical locations for each use-case category are described by URIs , which are based on a related ontology.

The model temporal ontology created in this thesis could be used in applications for our use-cases, such as in the use-case VA&ES to find a hospital that is ready to receive a patient and near the location of an accident.

1.3 Thesis Method

For the analysis and implementation of the goals mentioned above two already existing technologies will be used: the Semantic Web and GeoRSS. The Semantic Web comprises set of technologies (i.e. ontologies, query language, and rule language) that allows machines to understand the meaning of information on the World Wide Web. To create the required basis for use cases, we first must create an ontology model for use case. This ontology introduces a model for embedding geographical information from data sources available on the web. It contains Classes and Subclasses, allowing properties and individuals to be embedded into this model from GeoRSS feeds XML/RDF format.

The ultimate goal of this research is to show that the collaboration between SW & GeoRSS can facilitate the process of events and serving them efficiently using the existing resources

which are also described semantically.

1.4 Thesis Structure

This thesis is structured as follows:

Chapter 1 provides an introduction and the problem statement addressed by this thesis, identifies the essential requirements, and previews the structure of the thesis itself.

Chapter 2 describes the technical background of the thesis. In this chapter we first discuss available Geo-ontologies from different frameworks or systems. In this part we also identify and briefly introduce available Geo-Query approaches to searching spatial data and their graphical interface for the query. Then we present a number of currently available tools for location search and their main features. Finally, we introduce available approaches to geospatial database generation, which integrate spatial data from Semantic Web Services.

Chapter 3 establishes the required functionality and the principle design of the proposed approach (SW & GeoRSS). First, we discuss the concept of the Semantic Web and its standards, including the Resource Description Framework (RDF), the Web Ontology Language (OWL) and Query Languages. Then, we outline the geographical information and its relation with the Semantic Web as well as the advantages of using Geo-ontology to integrate geographical data, which we use as a reference ontology for designing the proposed approach. The final section of this chapter focuses on the GeoRSS and its Encodings, which would be used for our geo-data format.

Chapter 4 introduces the use-cases of this thesis and provides short overview of each use-case. First, we define the use-case ”Public-Transport for People with Special Needs”, which is intended for people who are visually impaired and need to access location information in order to find public transportation stations and learn the stations’ services. Then, we define the use-case ”Vehicle Accident and Emergency Services” and discuss how users gain access to geographical location information with the help of various technologies on the web. The next use-case, ”Tourism consulter”, demonstrates how the Internet and mobile technologies allow users to easily access tourist information systems for detailed information about points of interest such as hotels, public transport, hospitals, etc. by using our proposed approach. With the last use-case, ”Smart Guide Airplane landing”, we illustrate how the user group ”pilots” can benefit from using the two technologies of web services in the Semantic Web and GeoRSS to establish more facilities for searching and accessing geographical information.

Chapter 5 focuses on the technical solution of the proposed approach. First, we describe the required functionality of the use cases and then discuss in detail the conceptual design used to

implement each component.

Chapter 6 presents the implementation of the proposed approach, i.e. using the two technologies SW & GeoRSS on the web, for our use-cases. The process of parsing and embedding the Geo-data from GeoRSS into our temporal ontology is discussed in detail.

In Chapter 7 we conclude the thesis and provide a summary discussion of our approach.

Appendixes give details about the source code that contain our model temporal ontology and some samples of GeoRSS feeds.

CHAPTER 2

State-of-the-art

The present chapter discusses available approaches and existing projects. In Section 2.1 we introduce the available geospatial ontologies. Section 2.2 identifies and briefly discusses available approaches to Geographical information Query for searching spatial data. In section 2.3 we address available solutions for location search of geographical location information or navigating systems. Finally, Section 2.4 provides an overview of available approaches to Geospatial Database Generation.

2.1 Available Geo-ontologies

In recent years the amount of geographical data available to Web users has grown tremendously. As the amount of Geo-data grows, problems of data relevance and information overload become increasingly severe, making it necessary to address these problems with the use of semantic technology [Matthew2007]. As an instance of such technology, the Geo-Ontology provides a sharing semantics for geospatial information. Moreover, the ontology defines classes in a domain, linking them to the description of geographical concepts, thus facilitating the search for geographical information and resources. The following subsections present a number of currently available Geo-Ontologies and discuss their main features.

2.1.1 DAGIS Ontology

DAGIS or Discovering Annotated Geospatial Information Services [Ashraful] is a framework featuring a graphical interface that can be used to query and discover services for geospatial domains. The DAGIS prototype application is useful for finding local businesses in a given geographical region. It uses an Owl-S Service ontology coupled with the geospatial domain-specific

ontology to provide geospatial Semantic Web Services for automatic discovery, dynamic composition and invocation. In the geospatial ontology, the local businesses are categorized according to City, Latitude, Map, State, and Zip code.

The DAGIS comprises three ontologies: a geospatial domain ontology, a QoS (Quality of Service) ontology, and an OWL-S ontology. The QoS ontology is the middle one of the three ontologies in the sense that it defines geospatial domain concepts in the geospatial ontology. The main concepts in the QoS ontology are [Ashraful]:

- Quality: Represents the measurable nonfunctional concept of a service.
- QAttribute: The value of a quality concept is determined by the type of QAttributes that constitute that concept.
- QMeasurement: This describes the measurement of quality, which can be subjective or objective.
- QRelationship: This is used to describe the relationship between two or more quality concepts.

DAGIS improves query mechanisms through automated reasoning using a domain-specific ontology.

2.1.2 SWETO-GS

Semantic Web Technology Evaluation Ontology (SWETO) is able to incorporate instances that are extracted from heterogeneous sources. Extractors are used to create an automatic population. The geospatial extension of SWETO, as defined above, is SWETO-GS [Budak]. It is a spatio-temporal ontology that has three dimensions, i.e. a thematic, spatial, and temporal dimension. The first of these, the thematic dimension, includes either concepts of a general domain, including people, places and organizations, or for a specific domain, e.g. travel and transport. The thematic dimension includes both geographic and non-geographic concepts. The second, i.e. geospatial dimension, more specifically stores the spatial data and relationships. These concepts are described in terms of their coordinates, which are translated from the thematic dimension. The third, temporal dimension is used to store the temporal relations that can exist between concepts. In addition to the data contained by these three dimensions, some metadata can also be associated with the SWETO-GS ontology [Hess].

2.1.3 ONTOAST

ONTOAST (for ONTOlogies in Arom-ST) is a spatio-temporal ontology modeling and semantic query environment, capable of reasoning about spatial, temporal and thematic knowledge. Because of its compatibility with OWL-DL [Miron2007] [Miron2008], it can be used in the Semantic Web context. It increases the spatial-based search capabilities by handling the inference of new qualitative relations. ONTOAST is based on AROM, a generic tool designed to facilitate knowledge modeling and inference. The principal features/advantages of ONTOAST are: first, its capacity to model and reason about spatial features of ontological concepts and, second, its compatibility with the standard ontology language OWL. In general, it increases the flexibility and the expressivity of the language and allows some spatial reasoning even when precise numeric data is unavailable.

ONTOAST contains three categories of qualitative spatial relations - i.e.*topology, orientation and distance* - that can be automatically inferred from existing knowledge or from an explicit user declaration. Thus, when a request is submitted, the answer to the query is built in the following steps:

- using the explicit knowledge, provided that the required relation is already stored in the base
- by carrying out qualitative inferences on the basis of the composition of qualitative relations between objects which are explicitly stated
- by deducing qualitative relations employing some numerical estimation and computation methods
- by applying qualitative reasoning to both explicit and deduced spatial relations.

The ONTOAST ontological knowledge has to be translated into the AROM formalism and stored in a local object-oriented Knowledge Repository. The Knowledge Repository data are filtered using the Spatial and Temporal Contexts Specification in order to reduce the search space. In general, ONTOAST provides a spatial ontology model and manages quantitative data and qualitative spatial relations which can be used to define spatial object, attributes and relations, and to process queries for the modeled knowledge.

2.1.4 SPIRIT Geo-Ontology Component

SPIRIT [Jones] is a search engine for geographical information about places and is focused on tourism queries. The SPIRIT query is associated with a geographical context. In order to automate the search possibilities and make geographical information in this search engine more easily accessible, the Semantic Web is proposed as a solution to enrich geographical data resources with some well-defined meaning. The geo-ontology plays a crucial role in intelligent

spatial search on the Web, serving as a shared vocabulary for the spatial mark-up of Web resources.

The SPIRIT ontology describes the vocabulary and the spatial structure of places for purposes of information retrieval. Overall, the SPIRIT approach facilitates the retrieval of resources and place names that are spatially-related to that in the query; the result accesses resources with an exact locational match to the terms of the query and relates to places that are similar or nearby in location to the query. The SPIRIT query is formulated as a triple expression, i.e. < *term; spatialrelationship; place* >, in which *term* specifies a general non-geographic object, *place* specifies a query term which is geographically referenced, and *spatialrelationship* defines a spatial relationship which relates term and place. For instance, a possible query would be ”universities near Cardiff”.

The geo-ontology components of SPIRIT are shown in Figure 2.1 This figure shows that the ontology is used to generate a polygonal geometric query footprint covering the spatial extent of the query region. The interpretation of the spatial relationship is based on the place. A geographic place is associated with multiple geometric footprints. The query footprint is then used to access the spatial index of web documents. The geometric footprints are supported for each place (Points, Polylines and Polygons). Such footprints can moreover be used for different purposes: for instance, a detailed Polygon may be used to build up the spatial indexing of documents and a central point may be used by the interface for plotting the search results in an interactive map. Consequently, the geo-ontology supports multiple spatial representations of geographic places, including centre points and minimum bounding rectangles, besides more faithful representations of geometries.

For the implementation of the SPIRIT geo-ontology the Oracle Spatial database management system has been used. The following steps describe the access functions developed:

matchPlace: this operation is used by the user interface component to test whether a specific term in a query is a valid geographical place name.

getFeatureID: this operation retrieves the identifiers of places of specified terms and is used by the geo-markup process.

getFootprint: this operation retrieves the geographical footprints for the specified place.

queryDisambiguation: this operation retrieves the broader geographical contexts for the name appearing in a query, using the part of relationship contained in the ontology, and returns the derived geographical hierarchies.

queryExpansion: this operation takes the disambiguated place name and the spatial relationship as arguments and derives the desired geographical search extent for the query.

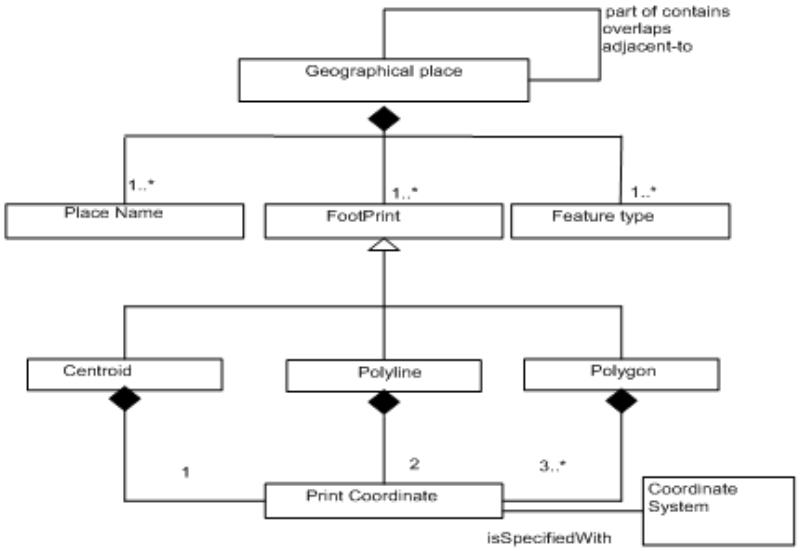

Figure 2.1: Conceptual design of the geo-ontology [Jones]

2.1.5 GKB

Geographic Knowledge Base [Chaves] is a repository based on a domain independent meta-model for integrating geographic knowledge collected from multiple sources. It supports geographic semantics-aware web mining applications that integrate data and knowledge from multiple sources. It contains two tools: converters which load data from various sources and generators which create ontologies in OWL. In addition, GKB includes two instances: one loads with detailed information about the main geographic names of Portugal, another holds information about the main regions around the world in four different languages.

The geographical information to be used is extracted with a tool called GOG-GKB Ontology Generator and is then stored in the GKB repository. GOG exports the information in the OWL format, a vocabulary extension of RDF [1]. The full geographic ontology of Portugal included in the first instance of GKB includes more than 418,000 features and is provided as a public resource. The GKB geographic ontology of Portugal also generates an ontology of geographic names across the world obtained by integrating information from public data sources directly available on the Web.

The GKB is used in three different applications which address problems related to classifying and retrieving web pages according to their geographical scope:

[1] `http://www.w3.org/TR/REC-rdf-syntax/`

- a geographical named entity recognition, classification and grounding tool
- a document classifier for geographical scopes
- an information retrieval interface for geographical queries

In addition to these applications, it is also used in the interface of the geo-information retrieval system, i.e. the Geo-Tumba interface, assisting users in the formulation of queries with a defined geographic scope. Finally, GKB presents a repository containing the semantic relationships between geographic entities extracted from the texts of the Portuguese Web.

2.2 Available Geo-Query

This section examines the convenience of available approaches to or frameworks for Geo-Query of location information.

2.2.1 SPAUK

Spatially AUgmented Knowledgebase (SPAUK) [Kolas] provides high-performance graph query capabilities for searching spatial data with the flexibility and graph search ability of RDF and triplestores. The main goal of this graph query is to provide both effective storage and allow the query of geo-data without sacrificing the flexibility and graph search ability of RDF and triplestores. It provides spatial processing for spatial semantic systems. In addition, it presents the semantic data and the geo-data as a graph.

This graph query divides the query into sub-queries that can then be answered by the various parts of the knowledgebase. The spatial query includes locations and spatial relationships, which are sent to the spatial index and query processor. The part of the non-spatial query will be sent to the underlying triplestore. The final result to the original query is thus a combination form a coherent answer of the two partial queries.

The SPAUK implementation is based on Jena *Semantic Web Framework and Joseki*. SPAUK-Graph belongs to the *com.hp.hpl.jena.graph.Graph* Java interface. Its deals with splitting and combining the information which represents the interface to the data stored.

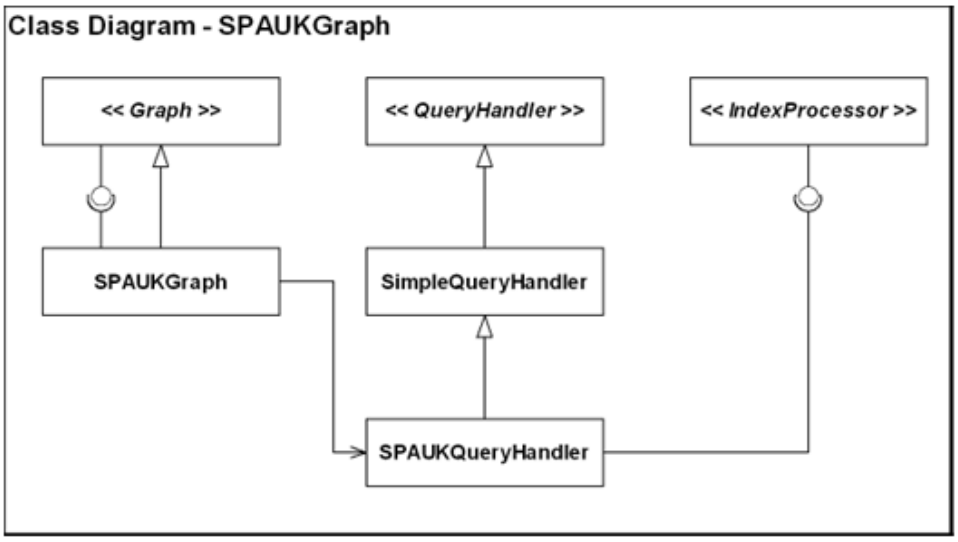

Figure 2.2: Class diagram for the SPAUKGraph and its relation to the IndexProcessors [Kolas]

The SPAUK query process uses SPARQL to support supplementary spatial indices. This query system provides a high-performance graph query for searching data on the web. The following example describes the SPAUK query that would be processed by the spatial index processor:

Listing 2.1: SPAUK query

```
1
2  SELECT ?x  WHERE{
3          ?x a gas:GasStation;
4              georss:where ?y.
5          ?y rcc:part [
6          a gml:Buffer; gml:radius '' 1 ''; gml:bufferGeometry
7         [ a gml:Point;gml:pos  '' 38 -77 ''
8          ].
9           ].
10   }
```

2.2.2 SPARQL-ST

SPARQLST is an extension of SPARQL designed for complex spatiotemporal queries related to politics. This query selects all politicians (and their tenure) that represent a congressional district that is inside a given polygon [Matthew2011]. It supports complex spatial data and temporal RDF queries with GeoRSS geotagging vocabularies. It introduces two types of variable, i.e. a spatial (identified using a % prefix) and a temporal one (identified using a # prefix). The spatial variables define spatial features, rather than a single URI in a set of triples. The temporal

variables relate to a timeframe rather than a URI and are used to retrieve the valid time of each temporal RDF statement.

The query language of SPRQL-ST introduces two separate filters: *SPATIAL FILTER*, which defines the distance function through two variables (spatial, temporal), and *TEMPORAL FILTER*, which defines that all triples in a result are valid. This query language contains two time intervals: first, the *intersect interval* introduces the time period during which all statements in the graph pattern instance are valid; second, the *range interval* introduces the time period during which any statement in the graph pattern instance is valid. The following example represents a set of triple polygon in a SPRQL-ST query:

Listing 2.2: SPRQL-ST query

```
SELECT ?p, %g, intersect(#t1, #t2, #t3, #t4)
        WHERE {
                        ?p usgov:hasRole ?r #t1 .
                        ?r usgov:forOffice ?o #t2 .
                        ?o usgov:represents ?c #t3 .
                        ?c stt:located at %g #t4 .
                    SPATIAL FILTER (inside(%g, GEOM(POLYGON ((
                           -75.14 40.88, -70.77 40.88, -70.77 42.35,
                           -75.14 42.35, -75.14 40.88)) ))}
```

This example query selects all politicians' congressional districts inside a given polygon. The result of this filtered spatial query is a statement of the time those politicians have represented those areas. The implementation of SPARQL-ST uses the framework of Oracle 10g, which supports the storage and querying of RDF data and geospatial data. This project provides a single SQL table function,*sparql st*, which inputs a valid SPARQL-ST query and returns a table of the resulting variable mappings.

2.2.3 MashQL

MashQL [Jarrar] is a query-by-diagram language that allows users to build data mashups diagrammatically and pipeline RDF data intuitively. The MashQL uses SPARQL as a backend query language, meaning that the MashQL automatically translates queries into and executes them as SPARQL queries. It also allows users to query data without understanding the technical details or the structure of this data. Moreover, the end-user does not require knowledge about RDF/SPARQL to begin using the tool. This query language is similar to a tree in the sense that the root of this tree is the *query subject* and each branch is a *query restriction* which is used to restrict a certain property of the query subject.

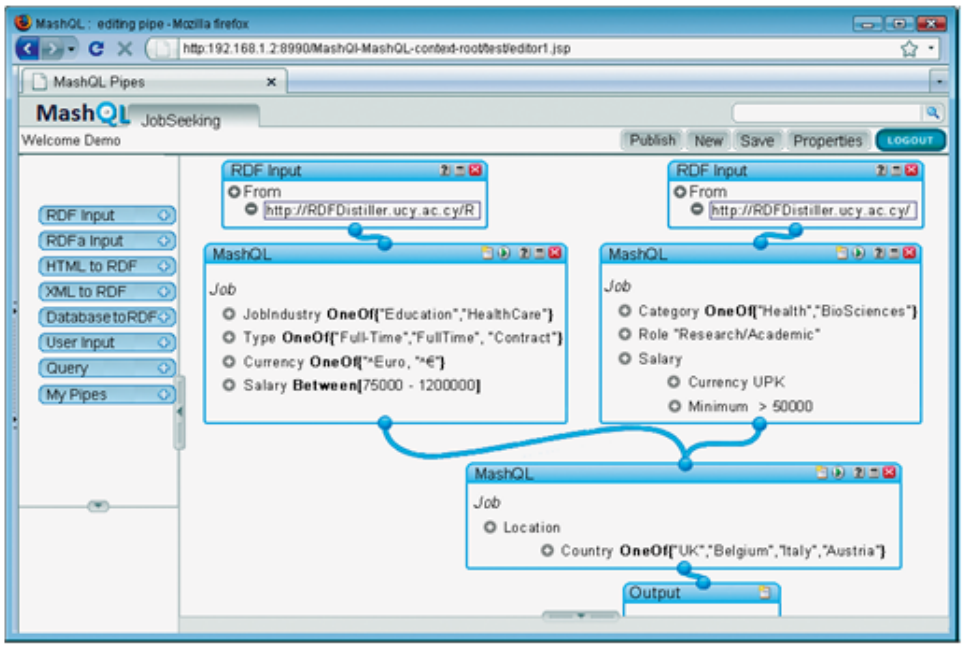

Figure 2.3: MashQL Editor [Jarrar]

The MashQL contains a query editor responsible for understanding data structures instead of the user. Thus, the user simply needs to use drop-down lists to define his/her queries, because the editor performs background queries and dynamically generates these lists. It is implemented on two sides: the server query and mashup editor, and a plug-in for the Firefox browser (Figure 3 . The MashQL uses SPRQL translation for Oracle 11g and Ajax to send queries. The Oracle 11g supports RDF and the database management system (including functionalities such as partitioning, indexing etc.). The MashQL GUI is used the Yahoo Pipes style for visualizing Web feed mashups and JavaScript libraries (see Figure 2.3). The end-user can thus use MashQL to query and mash up the data from the Web as simply as filtering and piping Web feeds.

2.2.4 GXQuery

GXQuery [Chen] is a GML Spatial Data Query Language on XQuery[2] intended to reflect the integration of GML and XQuery. XQuery is designed to data XML query and storage this data in dataset. It supports GML spatial data query by adding GML spatial data types and spatial operators to XQuery. XQuery is used for the evaluation and spread of the GML query language. The principal advantage of XQuery is that it is easy to use and provides powerful functions. It provides strong support for GML spatial data queries through adding spatial extensions to

[2] http://www.w3.org/TR/xquery/

XQuery using self-defined functions.

The Architecture of GXQuery contains the following parts [Chen]:

- XQuery engine: the engine is based on XQuery, performing XML queries through parsing, optimizing, executing and providing a result for the XML query. As GML spatial extensions have been added to XQuery, the XQuery engine can perform parsing, optimizing, and executing of the XML query, and provide a result for the GML query.
- GML parser: The XQuery engine's parsing and constructing the result for the query uses only XML and GML non-spatial data. The GML parser is created after adding a GML parsing module to the XQuery engine, so that GML spatial and non-spatial data can be queried integrally.
- Spatial extension: after adding the GML spatial extension to the XQuery engine and combining it with the GML parser, the XQuery engine performs the query and processes GML spatial data. The spatial extension module contains types of spatial data and spatial operation functions, which conform to XQuery standards.
- GML index accessing interface: the GXQuery engine is accessed by the GML index management module after it has been added to XQuery engine. It obtains the intended GML spatial data through GML index data information and constructs the GML query result.
- GML storage source: the GML spatial data storages use different sources (e.g. native GML spatial database, GML-enabled relational database, or even GML documents). Thus, the GXQuery engine can be used as access interface for different GML storage sources.

Figure 2.4 shows the GXQuery engine that is composed of the XQuery engine, GML parser, spatial extension module and GML index accessing interface module. To implementation of the GXQuery engine uses the JTS (Java Topology Suite) API component as an open source to operate and process 2D linear geometry spatial objects.

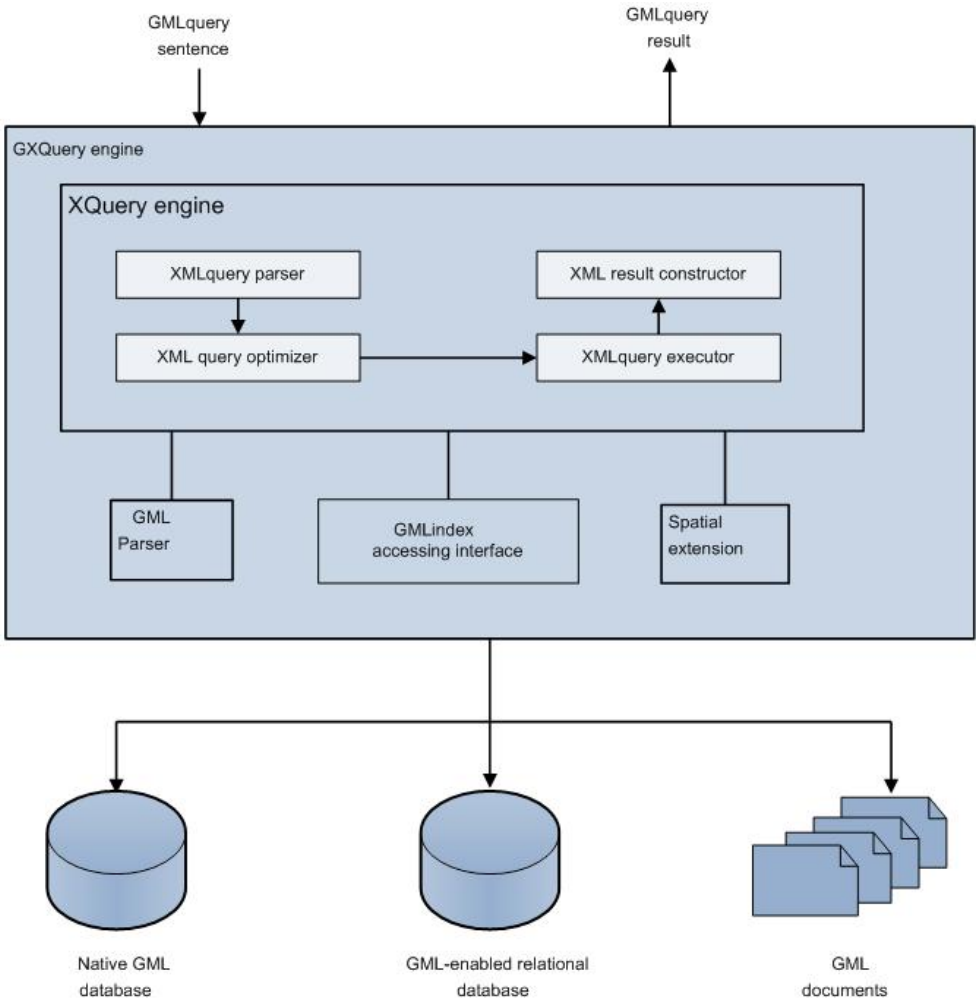

Figure 2.4: The architecture of GXQuery [Chen]

2.2.5 DAGIS

Discovering Annotated Geospatial Information Services (DAGIS) [Ashraful] is a Semantic Web services based framework for the geospatial domain that provides a graphical interface to query and discover services. This framework uses the OWL-S Service ontology coupled for building geospatial Semantic Web Services, providing an advantage over other web 2.0 services and GIS. DAGIS uses algorithms to select the best service via QoS (Quality of Service) based semantic matching. The purpose of DAGIS is to transform data sources into intelligent knowledge repositories on the web.

The DAGIS architecture focuses on devising improved query mechanisms through automated reasoning using a domain specific ontology. It contains two layers: semantic middleware that enables interchangeable components to provide meta-service related functionality (e.g. ser-

vice search and reasoning) and the ontology data that includes major components at each layer.

- *DAGIS Query Browser Portlet*: it gets the user query. It incorporates a Java™ portlet that provides the required interface for the query.
- *DAGIS Agent*: it is located at the layer of the semantic middleware and fetches the query parameters from the user. It uses an OWL-S service ontology for communication between agent and DAGIS Matchmaker.
- *DAGIS Matchmaker*: it performs semantic matching between the submitted queries.
- *DAGIS Composer*: it builds a service chain to solve the user query so that the composed service URI is returned to the Matchmaker.
- *OWL-S Registry*: it stores the Semantic Web Services.
- *WSDL Registry*: it is a public web services registry.
- *WSDL2OWLS Converter*: it converts the WSDL service file into an OWL-S file.

The following screenshot shows the DAGIS interface query. After the user submits his/her query, for the DAGIS Agent to solve this query, the DAGIS agent receives the query and forwards it to the Matchmaker. The Matchmaker then accesses the OWL-S registry to determine a match.

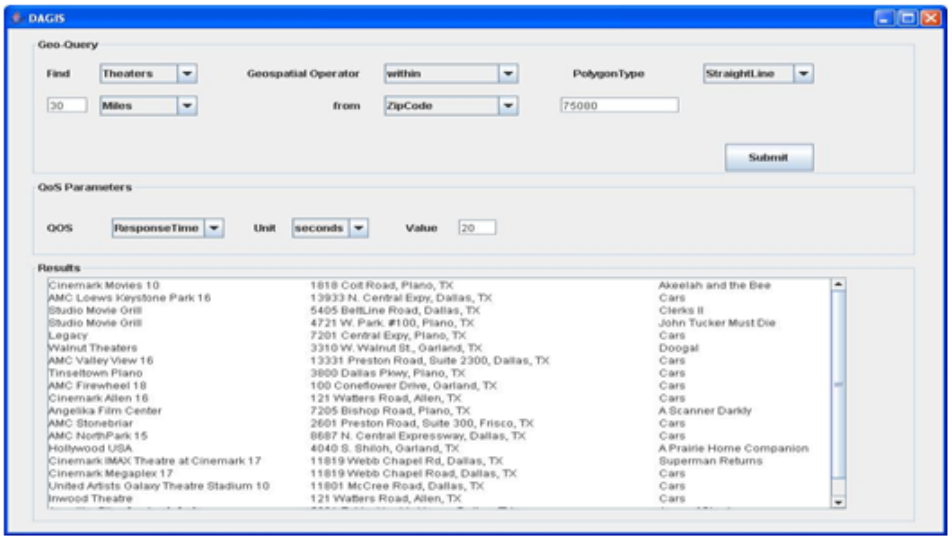

Figure 2.5: DAGIS Query Interface [Ashraful]

2.3 Available Location Search

This section examines the suitability of software tools for searching geographical location information or navigating systems, e.g. GPS (Global Positioning System). There are different navigating systems that can be categorized according to their output methods, interface or several other characteristics.

The following subsections present a number of currently available tools for location search and their main features. They moreover describe and briefly explain some of their distinctive attributes and their operation.

2.3.1 GPS (Global Positioning System) navigation

GPS[3] is a global navigation satellite system that provides reliable location and time information for an unlimited number of people in all weather, at all times and anywhere on the Earth. The system was originally put in place for military applications, but was later made available for civilian use. It is the basis of travel guides (for outdoor locations) such as street finding, using 24 satellites orbiting the earth to indicate permanent positions through signals for 24 hours a day.

[3] http://en.wikipedia.org/wiki/Global_Positioning_System#Basic_concept_of_GPS, http://en.wikipedia.org/wiki/GPS_for_the_visually_impaired

The GPS navigation system can be used for visually impaired people and has been used in the following projects:

Drishti [GPS] integrated several technologies including wearable computers, voice recognition and synthesis, wireless networks, Geographic information system (GIS) and GPS. It augments contextual information for the visually impaired and computes optimized routes based on user preference, temporal constraints (e.g. traffic congestion), and dynamic obstacles (e.g. ongoing ground work, road blocks for special events).

MoBIC [GPS] (Mobility of Blind and Elderly people Interacting with Computers) was developed as a route planning system designed to allow a blind person access to information from many sources such as bus and train timetables as well as electronic maps of the locality.

NOPPA [GPS] is a navigation and guidance system designed to offer public transport passenger and route information by using GPS technology for the visually impaired. It provides an unbroken route for a pedestrian using buses, commuter trains and trams in three neighbor cities' area.

Brunel [GPS] is a navigation system for blind and visually impaired people. It includes high-accuracy GPS positioning, GIS, an electronic compass and wireless digital video transmission. It provides an automated guidance using the information from daily updated digital map datasets for, e.g., roadworks.

Trinetra [GPS] is a system that addresses accessibility concerns of blind people using public transportation systems. Using GPS receivers and staggered Infrared sensors, information is relayed to a centralized fleet management server via a cellular modem. Blind people, using common text-to-speech enabled cell phones can query estimated time of arrival, locality, and current bus capacity using a web browser.

These mobile applications run in Windows Mobile Professional, Standard (Smartphone), and Classic (stand-alone PDA) editions of the operating system.

2.3.2 Web Mapping

On Web 2.0, geographical information play an important role for better deleivery of location information to end-users. The technologies of the web 2.0 such as XML (Ajax) and JavaScript provide maps for users to search and browse geographic information. There are several web maps available on the web like Google Map, Yahoo Map, NASA World Wind, etc. All of these provide their own application programming interface (API) for the integration of web maps in web sites. The main goal of web mapping is to present and allow users to search geographic information without any prior experience or instruction, giving, for instance, driving directions or identifying available hospitals, hotels, public transport stations, and restaurants in a city. There-

fore, the usability of such web maps is an important aspect.

The following subsections present a number of currently available Web Maps and their main features. Distinctive and unusual attributes are briefly highlighted and explained.

2.3.2.1 Google Maps

Google Maps[4] is a Web mapping service application. It provides map data for street maps, a route planner for traveling, a location search for locations such as public transport stations, hospitals, etc. Google Maps uses satellites that support most urban areas in the contiguous Africa, Europe, North America, South America, Southeast Asia, and non-contiguous countries (Australia, China, Comoros, etc.).

The implementation of Google Maps uses JavaScript and XML which provides multiple functions, including support for queries, manipulating maps and editing geo-data. The class user interface map is a JSP page which contains two JavaScript class-references. One class ranks for the Google Map object and the other class rank for the Web Map Service map image and bindings to the Google Map object. A classic Google mapping application uses the AJAX web application module and XMLHttpRequest protocol. Google handles creation of the map by using XMLHttpRequest and the given remote JavaScript file of the browser [Sayar].

2.3.2.2 Google Earth

Google Earth [Travis] is a Web mapping service application with satellite imagery and map data. It provides a geographical search facility for its users, representing a searchable, high-resolution image of the entire planet. It allows the user to easily navigate from one location to another, zooming in to the level of individual trees and buildings.

Google Earth is a GIS program that users can use to overlay datasets of their own choosing. To add new content in the form of XML files and images, the data can be locally loaded into GE as files or published on a web server.

2.3.2.3 NASA World Wind

National Aeronautics and Space Administration World Wind [Shelton] is an excellent visualization tool in a 3-dimensional Earth model that comprises individual satellite and aerial images. NASA receives satellite imagery and elevation data and distributes that imagery via the internet

[4] `http://en.wikipedia.org/wiki/Google_Maps`

to allow users to experience Earth terrain via World Wind. World Wind is open source software that uses to display satellite imagery.

Its goal is to provide the maximum opportunity for geospatial information to be experienced, be it for purposes of education, science, research, business, or government [Hogan]. The user experiences NASA World Wind a searchable, high-resolution image of the planet that brings to life earth features such as landforms, cities and forests.

2.3.2.4 ESSE

Environmental Scenario Search Engine [Zhizhin] is an MS Windows application based on the NASA World Wind 3D globe and is available as a web application built upon OGC WMS and Microsoft Virtual Earth control. It was developed for parallel data mining of a set of conditions inside distributed, very large databases from multiple environmental domains. The goal of ESSE is to allow a user to query the environmental data archives in human linguistic terms. It allows mapping between human language and the computer system.

The slim client accesses the environmental data via ESSE sources on road maps, satellite images or mixed images of the Earth available from Microsoft Virtual Earth. ESSE uses Microsoft Virtual Earth SDK to display vector graphics containing segments and polygons over combined road maps and surface images of the Earth. It is available for Microsoft Internet Explorer and Firefox browsers. The implementation of ESSE uses JavaScript and XML, the Microsoft Virtual Earth API and WMS server. The WMS server returns environmental images as requested by the user.

Its software tools are open-source under BSD license, including data source virtualization API, fuzzy search engine, and VissualESSE clients. Platform-independent Java libraries are hosted by SourceForge (esse.sf.net). Both the MS Windows version of the ESSE engine and the VisualESSE are hosted by CodePlex [Zhizhin].

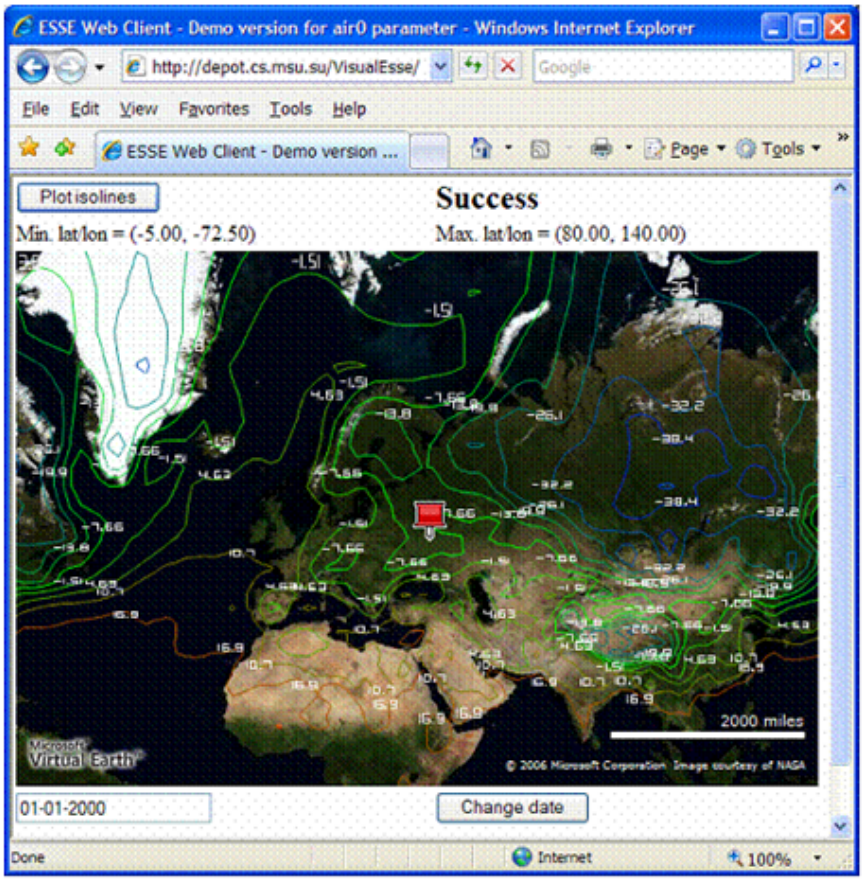

Figure 2.6: ESSE Web client sample screenshot [Zhizhin]

2.3.3 SPIRIT (Spatial Search Engine)

SPIRIT [Jones] is a search engine for geographical information. The components of SPIRIT include: user interface, web document collection, metadata extraction, core search engine, indexes, the geographical ontology, and relevance ranking.

User interface: users can submit a query by specifying a subject of interest and a geographical location. The user query comprises a structured text interface, a free text interface and a map. The user specifies the subject of the query, a place name (e.g. hotels, cities) and a spatial relationship to the place name through the structured text interface. The available spatial relationships are inside, outside and near. The results of such a user query are presented as a text list and as symbols on a map.

Geographical ontology component: This component is explained in section 2.1.4.

Web document collection: in this component SPIRIT employs a terabyte test collection of web documents which are structured to facilitate indexing. The documents contain place names that are associated with one or more document footprints derived from the geographical ontology entries for the respective names.

Indexes: these components support text indexing and spatio-textual indexing for the search functionality of SPIRIT. Spatio-textual indexing combines text indexing and spatial indexing of documents with respect to their document footprint. Spatial indexing also facilitates distance-based relevance ranking, which depends upon analysis of the geometric relationships between query and document footprints [Jones].

Core search Engine: this component provides access to the web document collection and its text and spatio-textual indexes. It receives a query from the user interface and processes it against the collection using the textual and spatio-textual indexes as required.

Relevance ranking: this component takes results retrieved from the SPIRIT database and ranks their relevance with respect to the non-spatial and spatial elements of the query.

Metadata extraction: this component is responsible for the extraction of geographical context from web documents. In addition to developing techniques, work is also being pursued on the detection of features within geo datasets that may then be used to enrich the geographical ontology.

As a result, the SPIRIT geo-ontology plays an important role in providing support for query disambiguation, query expansion through the generation of geometric query footprints, relevance ranking to compare a query footprint to a document footprint, and the extraction of metadata to obtain the geographical context of web documents and geo-datasets [Jones].

The following figure shows the user interface which requires the user to have knowledge of appropriate place names in order to formulate their query.

Figure 2.7: SPIRIT structured query interface [Purves]

2.3.4 STEWARD

Spatio-Textual Extraction on the Web Aiding Retrieval of Documents [Lieberman] is a system for extracting, querying, and visualizing textual references to geographic locations in unstructured text documents. The query string of this system contains a geographical entity, and STEWARD finds documents related to the query by spatial proximity. It focuses on finding geographical scopes of web sites containing multiple documents.

The query can have a purely geographical component, a keyword component, or a combination of both. Therefore, in case of a purely geographical query, STEWARD finds documents that are related to it by spatial proximity. It then returns the documents which are ranked by the extent to which STEWARD determines that the geographic entity in the query corresponds to the geographic focus of the document. In case of a non-geographical query, STEWARD finds documents according to the frequency and distribution of the keywords used. If the query contains both forms, relevant documents are ranked in increasing order of the distance of their geographic focus to the geographic location component of the query string.

The STEWARD system utilizes a user interface written in HTML and AJAX that runs on the Mozilla Firefox browser. The following interface[5] shows STEWARD's response to a textual query.

[5] User STEWARD interface, available at `http://mithra.cs.umd.edu/steward`

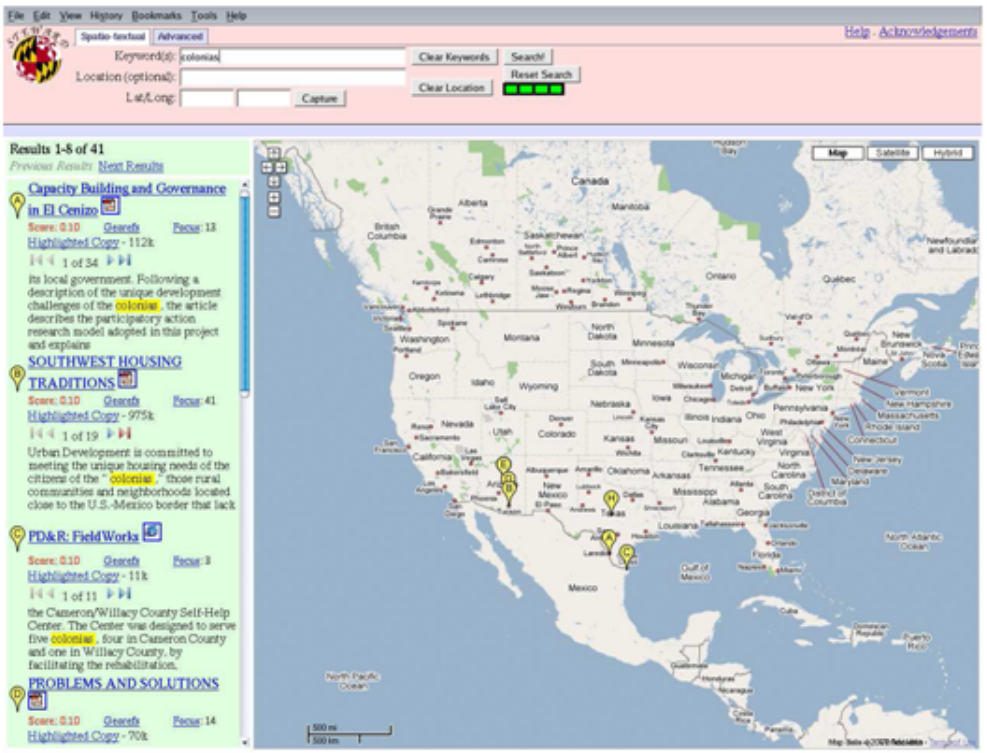

Figure 2.8: A screen shot of the STEWARD [Lieberman]

2.3.5 Kokono

Kokono [Yokoji] is a location-based search system for web documents on the Internet. It finds web documents based on the distance between locations that are described in web documents and a location specified by the user. In other words, this search method is based on the distance between a user-specified location and locations that are described in web documents (e.g. telephone numbers, rail station name etc.). The Kokono system converts these to latitude-longitude pairs or polygons that consist of latitude-longitude pairs. Thus a user-specified location is represented as a latitude-longitude pair. Kokono finds documents related to locations closer to a specified location through using these latitude-longitude pairs to retrieve documents. The system is based on three components:

- A robot component to gather web documents from the Internet.
- A parser component to parse extracts of location information from web documents and convert them into latitude-longitude pairs or polygons.
- A retrieval module component to convert location information specified by the user into a latitude-longitude pair and creates a search circle centered on this pair.

Because of its design, Kokono manages to retrieve location-related web documents overlooked by conventional keyword-based search engines.

2.4 Available Geospatial Data base Generation

This section examines the convenience of available geospatial database generation. Geospatial data is a large resource and an increasing number of applications on the web contain some elements of spatial data. In order for that geospatial data to be useful, it is necessary to integrate geospatial data and Semantic Web technologies. In the following subsections we describe some approaches to this issue and Web Services.

2.4.1 GeoBrain

GeoBrain [Liping] is a Web Services based Geospatial Knowledge Building System designed to help education and research activities in Earth system and geospatial information sciences. It is built on the concepts of geo-object and geo-tree. It makes petabytes of NASA Earth Observing System (EOS) data and information easily accessible to users via any Internet connected computers and offers interoperable, personalized, on-demand data access to and visualization of those huge volumes of NASA EOS data. Users can access, analyze, and model with the enormous amount of NASA EOS data as easily as if they possessed those vast resources locally at their desktops. In addition, users can access the data through Open Geospatial Consortium (OGC) standard interfaces.

The GeoBrain architecture consists of three tiers:

- *The Interoperable Data Server Tier*: this part comprises data servers that provide data to the geospatial service middleware, the application and data analysis systems, application clients, and human users through a common data environment.
- *The Middleware Geospatial Service and Knowledge Management Tier*: this part includes operations necessary to perform geospatial data processing, information extraction, and knowledge management.
- *The Integrated Multiple-protocol Geoinformation Client Tier*: in this part GeoBrain provides an integrated, multiple-protocol geo-information client to users for free.

The developers of GeoBrain have also developed a workflow execution manager (called BPELPower) which is based on Web services, including BPEL, WSDL, WSIF, Xalan, Xerces, UDDI, AXIS, SOAP, JNDI, J2EE (servlets/EJBs/JSPs), Jetspeed (Portlets) and JMX. It runs on top of popular application servers, such as Tomcat, J2EE, JBoss, Weblogic and WebSphere.

2.4.2 e-Merges

In the field of Geospatial Data Integration with Semantic Web Services, the e-Merges [Tanasescu] Approach is an available Spatial Integration of Semantic Web Services. It illustrates a way in which spatially related data (SRD) delivered through SWS can simplify the management of a specific use case by aggregating data originating from different sources and presenting it in a way which is consistent and task relevant. The main aspects of e-Merges are: data integration and context-based navigation of data.

The e-Merges prototype was designed for the *Essex County Council (ECC) Emergency Planning Department*. ECC is a local authority in South-East England. This prototype focuses the scenario on the ECC Emergency Planning department. It is used to implement an Emergency Officer (EO) system assistant in handling the dynamics of an emergency situation (e.g.*a snow hazard or a snow storm*, each presenting different goals) and gathering information related to a certain type of event, at a faster pace and with greater precision.

The e-Merges interface uses Google Maps to create a spatial representation of the application that uses the Google Web Toolkit, JavaScript to handle user interaction, and AJAX techniques. The interface supports spatial objects that can have an area-based location (polygon) or a point-based one (symbol). The following steps describe how a new data source process is added [Tanasescu] :

- *Ontological description of service*: the service, which comprises the data types involved and its interface, has to be described in a low level ontology, generally at a level that is low enough to remain close to the data. In many cases this step can be automated.
- *Lifting definition*: the lifting operation facilitates the movement of data type instances from a syntactic level (XML), which is defined in the data schema, to an ontological level (OCML) as specified in the ontology definition. Every time the previous step (see above) can be automated, this process can be as well.
- *Mapping to integration ontologies*: this process can be completely automated by default and customized as required.
- *Goal description*: in order to represent the newly integrated web service, a new goal has to be defined.
- *Mediator description*: the goal then has to be linked to the WS by means of a mediator, often a trivial operation.
- *Lowering definition*: the lowering operation transforms instances of aggregation ontologies into syntactic documents to be used by the server and client applications.

This approach presents advantages for end-users who handle tasks in an environment that contains large amounts of data. It avoids being overwhelmed by either the quantity of information or the complexity of the queries. Even for experts, it is an easier approach to data integration.

2.4.3 SWING

SWING [Andrei] is an Integrated Environment for Geospatial Semantic Web Services. It supports the entire life-cycle of Geospatial Semantic Web Services and demonstrates how the diverse components involved in such services can be integrated successfully. The principal purpose of SWING is to provide a geospatial Semantic Web framework that facilitates the use of geospatial services to solve a specific task in geospatial decision making. The main and technological objectives aspects of SWING are [Roman]:

- To provide an easy-to-use Semantic Web Service framework that contains suitable ontologies and inference tools for the tasks of annotation, discovery, composition, and invocation of geospatial web services.
- To assess the appropriateness of this framework by means of developing a geospatial decision-making application that can dynamically find and provide Semantic Web Services that are interoperable.

The SWING framework consists of the following six components:

- The Application prototype, which is an environment that allows the geospatial decision maker to use resources on the web.
- The Semantic Discovery & Execution, which provides a Semantic Web infrastructure that has access to a repository of semantically described web services and a set of inference tools.
- The Geospatial Ontology, which contains ontologies that provide formal definitions for concepts in a domain and the terms to denote them. This component provides a repository of those ontologies that are used by the Semantic Annotation and the Semantic Discovery & Execution components.
- The Semantic Annotation, the component in which the semi-structured data descriptions of existing geospatial web services are analyzed in order to generate semantic annotations.
- The Gatalog, this component provides a standard web service registry interface which stores entries of established geospatial and non-spatial services.

- The Development Environment, based on IBM's Open Source Eclipse Development Environment and consisting of a number of plug-ins, it integrates and hides the complexity of the other components.

The SWING framework also features several user groups, e.g. geospatial decision makers, data and service providers, application developers and the research community. The main purpose of SWING is the discovery, retrieval and integration of geographic information.

CHAPTER 3

Concepts Description

This chapter describes the Concepts and the required functionality to creation a GeoRSS-data repository of the mentioned Use Cases of Collaborating SW&GeoRSS. The principle of these concepts on the design and proposed collaboration between two technologies on the web (SW& GeoRSS are introduced to address the requirements of our Use Cases.

In Section 3.1 we discuss about Semantic Web and the Semantic Web technologies. These technologies include the Resource Description Framework (RDF), the Web Ontology Language (OWL) and Query Languages. In Section 3.2 we describe about geographical information and its relation with Semantic Web and the advantages of using ontology to integration geographical data. In Section 3.3 we discuss about RSS (Really Simple syndication), Collaboration geographical Data & RSS and GeoRSS encodings.

3.1 Semantic Web Technologie

The Semantic Web [Semantic] as an innovation to the availability of machine-readable metadata would enable automated agents and other software to access the Web more intelligently. The Semantic Web offers a new way for sharing and exploitation of data on the web to better enabling people and computers to work in cooperation. Moreover, the Semantic Web composes an environment search engines, which contains a set integration of data extracted from heterogeneous sources. It will be provided accurate answers to user's queries by using the explicit semantics of information. To represent [Infomesh] data the Semantic Web is usually in triples based structures, which use URIs. Backbone of Semantic Web is the Resource Description Framework (RDF), Web Ontology Language (OWL) and Rules Language that data can be defined by using RDF and OWL.

3.1.1 RDF-Resource Description Framework

RDF [1] (Resource Description Framework) is a framework for representing information in the Web that has been recommended by the World Wide Web Consortium (W3C). It provides a formal way that enables information about any resource to be encoded, machine- readable in the web. Furthermore, it allows interoperability between applications exchanging machine-understandable information on the Web [Nilsson].

RDF represents information to graph-based, which are serialized as XML to describe data about resources on the web. These resources consist of a set of RDF statements that are called triples. Each RDF triple has a uniform structure consisting of the form: subject, predicate, and object, that can be defined in terms of resources, properties, and value as:

- Subject: the statement describes resources by the ensuing predicate and object. The RDF resource is uniquely identified by a URI (called Uniform Resource Identifiers or URIs). A URI can be a URL (Universal Resource Locator).
- Predicate: the property is an entity that describes relations among resources and is uniquely identified by URIs.
- Object: the object is a value for a specified property, which describes the subject.

An RDF statement can be described value properties of resources. The property value is a resource or literal (literal is atomic value string or number). Therefore, an RDF statement is a combination of the preceding three elements (Subject, Predicate, and Object) as a triple. The following format represents an RDF statement:

Resource (subject), has property (predicate), whose value is (object)

A set of triples is called RDF graph (Subject-Predicate-Object) and each triple represents a statement. The following structure is a collection of RDF triples. In this RDF graph we describe, that *"AKH"* as a resource of type hospital has services *"surgeryRoomService and surgeryTeamService"*.

[1] `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`

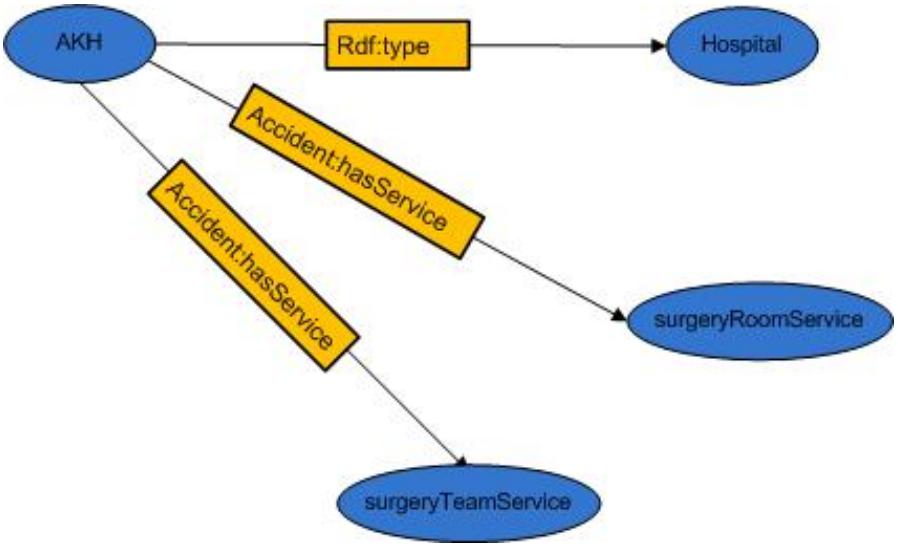

Figure 3.1: Graph representation of triple

In the following example you can see an RDF document that describes some facts about a specific hospital. The combination of RDF and XML provides an object portable across platforms and interoperable between applications.

Listing 3.1: RDF document example

```
1
2
3
4  <?xml version='1.0' ?>
5  <rdf:RDF  >
6  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
7  xmlns:hospital="http://www.hospital.at/hs#">
8       .
9       .
10 <!-- http://www.accident.org/ontologies/Temporal_final.owl#AKH -->
11    <rdf:type rdf:resource="&Accident;Hospital"/>
12    <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
13    <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
14         .
15         .
16         .
17    </rdf:RDF>
```

An RDF/XML document has two types of nodes: resource and property. Resource nodes contain object and subject that usually is showed in this format *rdf:about*. In this example, statement contains: *subject<http://www.hospital.at/hs/hospital>* and for example *predicate <hospital:hasService>* and *object* like *SurgeryTeam*. Property nodes contain literal values, like "AKH".

3.1.2 RDF Query Languages

To accessing information of the RDF information model, we can use an RDF query languages that allows retrieving and manipulating data stored in RDF format. Examples for RDF query languages are: RQL[2] , Versa[3] , N3QL[4] ,RDQL[5] ,R-DEVICE[6],RDFQ[7] ,SeRQL[8] . SPARQL Query language is the most common RDF query language that we also used for realization of our use-cases.

The W3C recommendation SPARQL for an RDF query language, supports querying of multiple RDF graphs. It offers capabilities for querying RDF data by graph patterns and retrieval of solutions is based on graph pattern matching [Corby]. SPARQL syntax is based on the database query language SQL. The following simple SPARQL query example shows a single triple to find out the value of property *"all requested services"* from hospital, police and fire brigade resources:

Listing 3.2: SPARQL query example 1

```
1  prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  prefix Accident:<http://www.accident.org/ontologies/Accident.owl#>
3  prefix Temporal_final:<http://www.accident.org/ontologies/Temporal_final.owl#>
4
5   SELECT  ?service
6
7  WHERE {
8
9          ?MariahilferSt_Accident1   Accident:requestService ?service
10  }
```

In SELECT clause, the query selects resources that are of type *"service"* whose values should appear in the result. In WHERE clause, variables are identified with a"?" prefix.

The following result shows the result of required services.

[2] http://139.91.183.30:9090/RDF/RQL/
[3] http://en.wikipedia.org/wiki/Versa_(query_language)
[4] http://www.w3.org/DesignIssues/N3QL.html
[5] http://www.w3.org/Submission/RDQL/
[6] http://lpis.csd.auth.gr/systems/r-device.html
[7] http://www.w3.org/2001/11/13-RDF-Query-Rules/#RDFQ
[8] http://www.openrdf.org/doc/sesame/users/ch06.html

Services
SurgeryReadyCenter *PoliceReadyCenter* *FirebrigadeReadyCenter*

Figure 3.2: Result query of required services

The following example shows another SPARQL query that consists of multiple triple:

Listing 3.3: SPARQL query example 2

```
1
2  prefix :<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
3  prefix Accident:<http://www.accident.org/ontologies/Accident.owl#>
4 prefix Temporal_final:<http://www.accident.org/ontologies/Temporal_final.owl#>
5 prefix owl:<http://www.w3.org/2002/07/owl#>
6
7  SELECT  ?service ?position
8
9  WHERE {
10         ?service   rdf:type Accident:SurgeryReadyCenter.
11        ?service   Accident:hasPoint  ?position
12                   };
```

The following result shows the result of this query for hospital resources that contain value properties SurgeryReadyCenter and hospital position:

Hospital	Position
AKH Lainz	18.42 25.30 -35.12 12.01

Figure 3.3: Result query for hospital resource

3.1.3 Web Ontology Language-OWL

OWL (OWL Web Ontology Language) is the web ontology language for the definition of Ontologies in the Semantic Web OWL provides, developed by the World Wide Web Consortium (W3C) Web Ontology Working Group [Horrocks2003]. It extended of RDF schema by using RDF meaning for creating classes and properties and instances in ontology (rdfs:classes, etc). The Owl developed from DAML (Darpa Agent Markup Language) and OIL (Ontology Inference Layer) and RDF/S for expressive a web ontology language. three different sub-languages, which are OWL Lite, OWL DL and OWL Full.

OWL Lite: is the easiest Web Ontology Language with simple class hierarchy and simple restriction on properties. According to that the OWL DL can be built using complex combinations of OWL Lite than it is not widely used.

OWL DL: based on Description Logics for the formulation of ontologies. OWL DL is a sublanguage of OWL Full that has restrictions to use the constructs from OWL and RDF. It provides a resource that only contains a class, an object property, a data type, a data type property, an instance, or a data value. This means that, for example a property cannot have at the same time values from a data type and values from a class. Furthermore, properties are sub-classes of *rdfs: property* that can relation resource to resource or resource to value and any classes have several sub-class, which a class has sub-class of class *rdf:resource*. The super- and sub-classes have relationship by *sub-classOf-relationships*.

OWL Full: is the complete language, this means that it contains other OWL constructs (Classes and properties, etc). It allows to combining these constructs with RDF and RDF Schema and mixing of OWL. OWL Full is same language constructs as OWL DL that their difference is on the use of RDF features.

3.1.4 Semantic Web Rules Language (SWRL)

The Semantic Web rules [Horrocks2004] is based on a combination of the OWL DL, OWL Lite sublanguages of the OWL and properties. It is formally expresses an if-then by using the OWL syntax (RDF/XML). The SWRL contains two parts *"if clause (body)"* and *"then clause (head)"*, both of which consist of one or more atoms.

- A rule body is a set of triple patterns (subject predicate object), which must match statements in the input.
- A rule head is a set of triple patterns that is evaluated if clause, when the body matches.

The following example represents a rule with body and head clause of RDF description:

(?f ex:father ?e) (?u ex:brother ?f) ->(?u ex:uncle ?e)

If the **f** is father **e** and u is brother **f** then u is uncle **a**. If the body *(?f ex:father ?a) (?u ex:brother ?f)* is true then assert triple *(?u ex:uncle ?a)* and this property *(ex:uncle)* and will be added to ontology.

As another example shows a rule with multiple bodies and head clause:

Listing 3.4: Rule example

```
1
2
3  prefix Accident: http://www.accident.org/ontologies/Accident.owl#
4
5   [ rege1:  (?a Accident:hasInjury Accident:bleeding)
6                 (?a Accident:hasInjury Accident:canNotMove)
7                 (?a Accident:hasInjury Accident:skullfracture)
8                 ->(?a  Accident:requestService Accident:_SurgeryReadyCenter )]
```

The principle strategies of rule execution are, as follows:

- Forward chaining rules: it starts with the available data, when *"if clause"* is true and thus *"then clause"* is added to our data.

If e has father and father has brother **then** e has uncle

(?f ex:father ?e) (?u ex:brother ?f) ->(?u ex:uncle ?e)

- Backward chaining rules: it starts from the result to the confirm foreword, in fact it makes a query *"has e uncle"* the answer is if e has father and if father has brother, thus.

e has uncle **if** e has father and father has brother

(?u ex:uncle ?e) <- (?f ex:father ?e) (?u ex:brother ?f)

3.2 Semantic Web & Geospatial Information

By far the greatest part of the resources available on the web application refers to information that may be regarded as geographically located. To find geographical information, existing web that relates to a particular location, the Semantic Web search engine proposed a better method for geographical information retrieval by incorporating the data's semantics and exploiting the semantics during the search process [Egenhofer]. The Semantic Geospatial Web is created by development multiple geospatial data and terminological ontologies with a formal Semantic Web.

The Semantic Web provides ontologies that model geographical terminology to the processing of geospatial queries. These ontologies improve communication among humans or computers and search facilities to finding information that relates to a particular location. When user type a name of place in the current search engine, all web pages will be retrieved, that has the same name in their text. Thus the specified place may not be found or may be places that in near to the user query.

This section will discuss how the geographical information is retrieved with the help of the Semantic Web and how the Semantic query processes geographical data.

3.2.1 Geospatial Information

Many information resources on the web contain geographical data. This includes, among others, web pages with geographical information about hospitals, restaurants, airplanes etc. With the increasing spread of technologies on the internet, different efforts are being undertaken in order to represent Geospatial information resources such as navigation systems or Web Maps (e.g. NASA, Google Map, and Google Erath). Most importantly, access to such geographical information on the web offers a huge potential for using the internet and the World Wide Web. In addition, the internet also makes it possible to use maps in a much more advanced way than is possible with a traditional map printed on paper. Ultimately, the geographical information is presented on the Internet in order to provide better services and meet the users' needs.

The following screenshot given in Figure 3.4 shows Google Earth representing several elements: positions on Earth which are linked to (Web) documents, photo collections and other external resources.

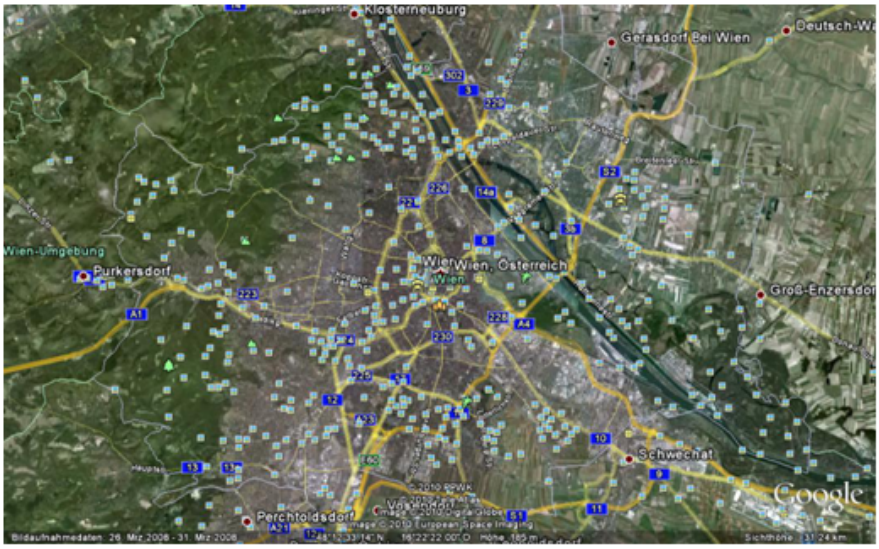

Figure 3.4: geographical information on the Google Earth

The Geospatial web is an integrated, discoverable collection of geographically related *Web Services* and data that spans multiple jurisdiction and geographic regions [Lake]. Main goals of Geospatial information on the web are:

- The web is more useful when is combined with geospatial information.
- Distribute geospatial data and maintenance of digitized and presentation a new services on the web (navigating, search).
- User can access to the geospatial information from different resources on the web

In addition, the web services offer a geographical search engine that is aware of the geographical location of resources. The location search results displays location and short summary information about location.

3.2.2 Geospatial Ontology

A large percentage of documents is stored on the web include geographic information, therefore the web is a place for storage and exchange of geographic data. Geographical information on the Internet provides a high level of capability and flexibility the web services. Users are enabling to access new kinds of services. For instance, user can find a facility location (e.g. doctor's office), public vehicle station that he needs to arrive this location.

Geographic data are used in Web pages of World Wide Web (WWW) for finding locations and routing (e.g. to find a hotel or restaurant and route information include distance, street name and etc). The Geographical information is easy to understand for human but hard for computer (to defining and representing geographic knowledge). To solving this problem the Semantic Web offers a way to share or exchange of information to access information.

In recent years, Semantic Web enables to the development of a Semantic geospatial data by using ontologies. Ontologies improve communication between humans or computers through understanding of essential concepts in some area of interest, such as geographic information. For instance, to create a tourism guide by geographic resources on the web, for finding a special location, route guide and etc. . . . Geospatial ontologies define classes and individuals for representing e.g. geographic regions, their properties, and mutual relationships. By sharing ontological resources in different collections and application domains, interoperability in terms of geographical locations can be obtained and intelligent end-user services such as Semantic search, browsing, and visualization be facilitated [Kauppinen].

Ontology is a formal, explicit, shared conceptualization of a domain. Domain ontology represents the concepts and vocabulary used within a community of interest [Ressler]. As example of domain ontology the following figure describes a fragment of the Urban Ontology. A GeographicObject can have one or more spatial Object to define its location in space. The UrbanObject represents a city with different entities such as location (e.g. squares, roads) or buildings (e.g. museum, hotel, hospital).

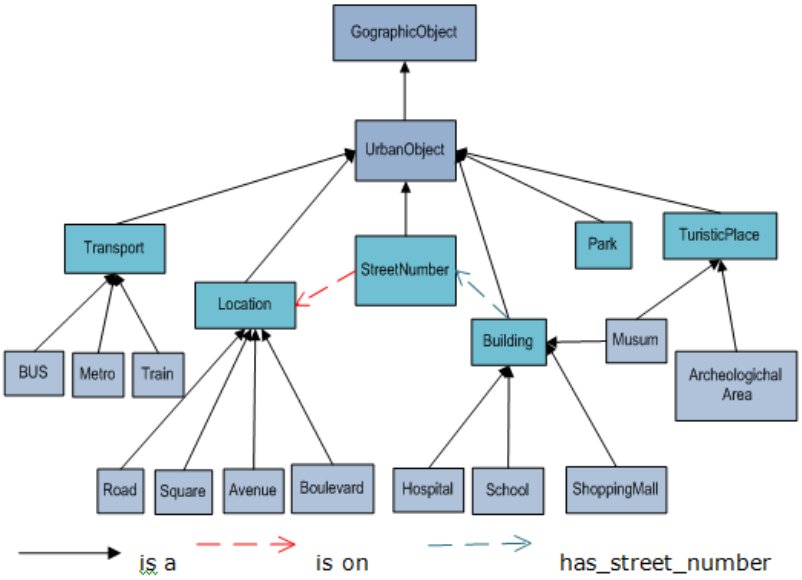

Figure 3.5: A fragment of the Domain Ontology [Baglioni]

In addition, ontology has an important role to development of the Geo-Semantic Web to be facilitating search of the geographical information. The Geo-Semantic is presented by using the web ontology language (OWL).

3.3 Geographical Really Simple Syndication

In this section we will introduce the RSS feed and collaboration with geographical information location that called GeoRSS. GeoRSS is proposed for encoding RSS feeds with location information and its point latitude/longitude. As an importance goal of collaboration geographical information and RSS is to maintenance and integration geo-information codes on the Web. In the next step, we discuss in detail the GeoRSS Encoding.

3.3.1 RSS (Really Simple Syndication)

Today, many people in their daily lives acquire information about news headlines of interest. The web services exhibit a way for internet users to receive information updates from different websites without having to visit them individually by means of RSS technology. RSS (Really Simple Syndication, Rich Site Summary or RDF Site Summary) is a kind of web technology that provides frequently updated information from the Internet. RSS was formally used for weblogs

or news website such as Reuters, CNN, and the BBC, which provide access to rapidly changing news. More recently, RSS is an important technology for researchers, specialists, educators and Student that can access to update information, database searches, events and use of geographical information.

Generally, RSS is a format for aggregating web content in one place, which allows users to aggregate information from many different Internet sources [Cold] [Richardson]. Therefore, user does not need to visit consistently of web sites that there are new contents. RSS allows the publishers easily syndicate their information, which can other websites publish this information. Syndicates or feeds are a form of advertising for the publishers, which can easily spread throughout the Internet [Gill].

RSS uses an XML format to content distribution and to frequently update the content and facilitates the content syndication(feed) of web sites that these updates are automatically delivered to users through an RSS feed. RSS feed consists of a main element <RSS >and feed element <channel >. The RSS feed can consist one or many <channel >that very channel consists of items that each items usually consist a <title >, <link >, <description >. Moreover, there are many other elements that can be added to the channel such as <image >, <language >and <category >. The following XML document is an example for RSS feed:

Listing 3.5: RSS feed example

```
1
2     <?xml version="1.0" encoding="UTF-8"?>
3     <rss version="2.0">
4     <channel>
5     <title>Channel Title</title>
6     <link>Channel URL</link>
7     <description>Channel Description</description>
8     <language>Channel Language </language>
9     <item>
10    <title> Item(1) Title</title>
11    <link> Item(1) Link</link>
12    <description>Item(1) Description</description>
13    </item>
14    ...
15    ...
16    <item>
17    <title> Item(n)Titel</title>
18    <link> Item(n) Link</link>
19    <description>Item(n) Description</description>
20
21    </item>
22    </channel>
23    ...
24    </rss>
```

The first line defines the XML version and the character encoding and the next line determine the RSS identifies that this is in this case version 2.0. The next line contains the <channel>

element, which used to describe the RSS feed and it has three required child elements:

<title >:Defines the title of the channel e.g. TU Wien Home Page <link >:Defines the hyperlink to the channel e.g.`http://www.tuwien.ac.at` <description >:Describes the channel e.g. Vienna University of Technology

Each channel element consist one or more item elements, which defines information that the feed wants to present. The items have three required child elements:

<title >: Defines the title of the item e.g. Library <link >:Defines the hyperlink to the item e.g.`http://www.ub.tuwien.ac.at/eng/index.html` <description >:Describes the item e.g. Vienna University of Technology Library.

Listing 3.6: TU Wien Library RSS feed

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <rss version="2.0">
3   <channel>
4   <title>Tu Wien Home Page</title>
5   <link>http://www.tuwien.ac.at</link>
6   <description>This is an example of tu wien RSS feed</description>
7   <item>
8   <title> Library</title>
9   <link> http://www.ub.tuwien.ac.at/eng/index.html</link>
10  <description>Vienna University of Technology Library</description>
11  </item>
12 </channel>
```

RSS feed can be created manually using text editor such as Notepad but the XML file must be updated automatically too. Thus, there are several software, web sites (RSS Creator [9], Tools [10], software like FeedForAll [11]) to create feeds automatically, which have fewer errors and less time consuming.

RSS feeds can be read using a feed reader, RSS reader or an aggregator, which allow users to organize the information they receive by category. There are three different Types of RSS reader [Jie]:

- Web-based aggregators: There are many Web-based aggregators such as Bloglines[12] or Newsgators[13] . Web portals such as My Yahoo[14] , Google Reader[15].

[9] (www.create-rss.com)
[10] (www.free-rss.page2go2.com)
[11] (www.feedforall.com)
[12] http://bloglines.com
[13] http://www.newsgator.com
[14] http://myyahoo.com
[15] http://reader.google.com

- Desktop-based aggregators such as FeedDemon[16].
- Web browsers with plug-in or built-in feed readers include Opera[17] Web Browser, Mozilla Firefox[18] , and Internet Explorer (IE) with add-in software Onfolio[19] installed.

RSS aggregators are applications that support people in reading and managing large collections of RSS feeds. The aggregator is maintained RSS feeds in the form of a list of URLs and automatically checks to update each of the RSS feeds in this list. Therefore, the aggregator would be added the new contents to the last update.

Finally, RSS is a convenient tool to save or retain updated information that internet users frequently visit of websites that are their favorite. Thus, RSS give users the latest updates information (e.g. about weather, local news, new music), also enables users to see at a quick look whether is a website or blogs has been updated without visiting again.

3.3.2 Collaboration Geographical Data and RSS

Geographical information is as a large resource data available on the web that is one of the most popular web sites usages (e.g. location search, route planning navigation and emergency response). Recent Web technologies provide more facility of geographical information such as storage, organization, searching and representation, which can be represented as Web maps such as *Google Map, Yahoo Map, NASA World Wind or Google Earth* that constitute a area to demonstrate the geographical data to the public. For instance, web sites respective to road constriction, locations (e.g. hospital, restaurant) etc., can publish a geo-information to the World Wide Web, in a magnitude and breadth that has not existed before.

The geo-location information is important for internet users insofar as it specifies location information and the position of places. Assuming that a tourist is planning to visit a museum and would like to know: What is the nearest museum to his position? How can he go there? What are the opening hours of that museum? In order to provide answers to such questions, the query response would use Geographically Encoded Objects for RSS (Really Simple Syndication) feeds or, in other words, GeoRSS technology.

The GeoRSS proposes a method for description of information location as an RSS feeds of internet content. The goal of collaboration geographical information and RSS is to maintenance and integration geo-information codes on the Web from fracturing into various encodings that it will be executed through RSS. In addition, GeoRSS facilitates the immediate dissemination

[16] `http://feedDemon.com`
[17] `http://www.opera.com/`
[18] `http://firefox.com`
[19] `http://www.onfolio.com`

of RSS feeds to specific users based on event type and location, view traffic event on Mariahilfer Avenue at eleven o'clock that user can access to via his computer or mobile. Therefore, GeoRSS provides the ability for availability to geographical information as a coordinates (latitude and longitude) and location description on the web.

Use of GeoRSS feeds may benefit:

- **Sharing:** GeoRSS is best use of a single RSS item such as a blog post, sensor reading, story, photograph, or the tracking of a moving object [Turner]
- **Searching:** GeoRSS will make it possible to search for location information/ events [Lesage]. For instance, it can be used to search a hospital and information or event respective to it.
- **Aggregation:** GeoRSS aggregates RSS and geospatial information, which is supported by Google Maps API [Zhang] [Lesage].

Figure 3.6 represents GeoRSS UML model:

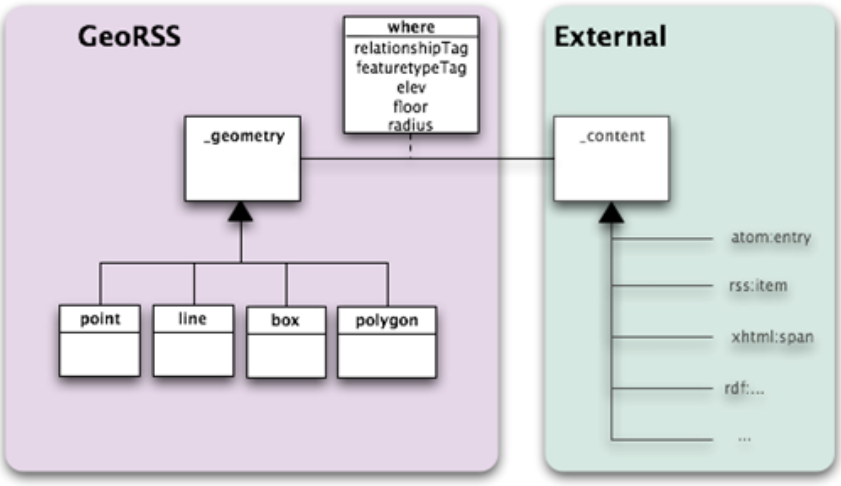

Figure 3.6: The GeoRSS Information Model [GeoRSS]

The GeoRSS information model is consisted geometry objects *point, line, box* and *polygon*, their attributes *relationship, feature type, elevation, floor* and *radius* and their external information entities *XML, XHTML, RDF* etc.., which describe the content of GeoRSS. The four geometric shapes are in the following way [GeoRSS]:

Point: it contains a single coordinate pair a latitude value and a longitude value.

Line: it contains two or more coordinate pairs that each pair contains a latitude value and a longitude value.

Box: it contains exactly two coordinate pairs that each pair contains a latitude value and a longitude value.

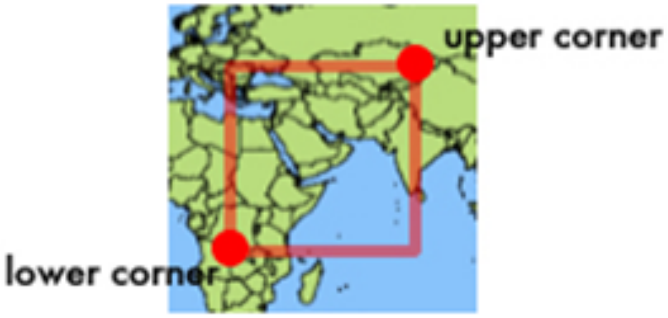

Figure 3.7: Box Mode [GeoRSS]

Polygon: it contains at least four coordinate pairs that each pair contains a latitude value and a longitude value.

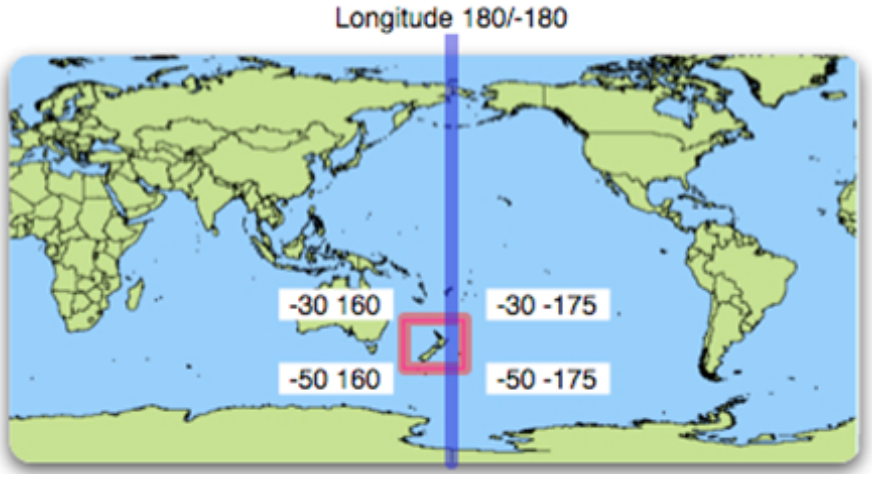

Figure 3.8: Polygon Mode [GeoRSS]

The attributes of GeoRSS model are in the following way [GeoRSS]:

Relationship Tag: GeoRSS is a way of relating Web content to Earth features. The GeoRSS model allows for a single string containing a relationshiptag. No constraints are placed on this string. The intent is to allow a relationship folksonomy[20] to emerge. The default relationship, "is-located-at" simply indicates that the subject of the content is located at the GeoRSS feature [GeoRSS].

Elevation: In order to provide a means of expressing an elevation, the Simple form of GeoRSS has two special tags. These tags are not meant to be used in the GML version since elevation values would be properly expressed based in more precise terms. The tags are elev and floor [GeoRSS].

- **elev** is meant to contain "common" GPS elevation readings, i.e. height in meters from the WGS84 ellipsoid, which is a reading that should be easy to get from any GPS device [GeoRSS].
- **floor** is meant to contain the floor number of a building. In some countries the numbering is different than in other countries, but since we'll know the location of the building, it should be fairly unambiguous [GeoRSS].

Feature Type Tag: GeoRSS geometry is meant to represent a real feature of the Earth's surface. The GeoRSS model allows for a single string containing a featuretypetag. No constraints are placed on this string. The intent is to allow a Feature Type folksonomy to emerge. The default is llocation- [GeoRSS].

3.3.3 GeoRSS Encoding

Geo-tagging allows users to add the geographic coordinates latitude and longitude (e.g. 45°18'N, 85°54'W) in Web content such as document, photograph, audio sample, or some other type of data. Geo-tagging help users to find location information through improvement in data sources integrity as a form geographic coordinates in the location data. Geo-tagging can also be used in information services for example to find emergency information about a location disaster. GeoRSS was proposed for Geo-tagging Really Simple Syndication (RSS) feeds with location information of geographic coordinates <geo:lat >and <geo:long >that is specified the geographical location of each <item >element. GeoRSS is encoded of three types: *Simple, GML* and *W3C Geo*.

The Simple encoding was developed to support point, line, polygon geometry. The GML (Geographic Markup Language) encoding describes complex geographic geometry. The W3C

[20]The folksonomy is combination of folks "people" and onomy "management", which has been suggested by Thomas Wonder Wall. Folksonomy is a system for organizing information, classification and retrieval of information resources on the Web.

Geo encoding describes a single point location. The format of the GeoRSS geometry tags are:

Point tag:
<georss:point>23.256 -51.62</georss:point>

Line tag:
<georss:line>23.256 -51.62 46.46 -109.48 63.44 -79.36 </georss:line>

Polygon:
<georss:polygon>23.256 -51.62 46.46 -109.48 63.44 -79.36 23.256 -51.62 </georss:polygon>

Box :
<georss:box>23.256 -51.62 46.46 -109.48</georss:box>

The attributes are another tags specified as GeoRSS elements and the format of the attributes tags are:

Feature:
<georss:point>23.256 -51.62</georss:point>
<georss:featureTypeTag>university</georss:featureTypeTag>
<georss:relationshipTag>is-technical-at</georss:relationshipTag>
<georss:featureName>Vienna</georss:featureName>

Elevation:
<georss:point>23.256 -51.62</georss:point >
<georss:elev>156 </georss:elev >
<georss:point>23.256 -51.62 </georss:point >
<georss:floor>8</georss:floor>

Elevation, specified in GeoRSS elements, can be expressed as ”elev” or ”floor” that means height in meters and floor number of a building.

Radius:
<georss:point>23.256 -51.62</georss:point>
<georss:radius>100</georss:radius>

To added GeoRSS tags to RSS, let us definition the GeoRSS namespace that shuld be added to RSS feed namespace:

xmlns:georss=”http://www.georss.org/georss”

xmlns:gml=”http://www.opengis.net/gml”

Then we can add the GeoRSS tag e.g.<georss:point>latitude longitude</georss:point>in to RSS feed. The following example defines GeoRSS Simple encoding:

Listing 3.7: GeoRSS Simple encoding

```
1
2
3  <?xml version="1.0" encoding="utf-8"?>
4  <feed xmlns="http://www.w3.org/2005/Atom"
5    xmlns:georss="http://www.georss.org/georss">
6    <title>TU Wien</title>
7    <subtitle>Vienna University of Technology</subtitle>
8    <link href="http://www.tuwien.ac.at/"/>
9    <updated>2010-07-23T08:30:02Z</updated>
10   <author>
11     <name>Homa Rezaie</name>
12     <email>Rezaiehoma@gmail.com</email>
13   </author>
14   <entry>
15     <title>Library</title>
16     <link href="http://www.ub.tuwien.ac.at/eng/index.html"/>
17     <updated>2010-07-22T07:32:38Z</updated>
18     <summary>Vienna University of Technology Library</summary>
19     <georss:point>23.256 -51.62</georss:point>
20   </entry>
21 </feed>
```

We present the GeoRSS GML elements which are used to describe and different geographical information:

Point:

```
<georss:where>
  <gml:Point>
    <gml:pos>
      23.256 -51.62
    </gml:pos>
  </gml:Point>
</georss:where>
```

Polygon:

```
<georss:where>
  <gml:Polygon>
    <gml:exterior>
      <gml:LinearRing>
        <gml:posList>
          23.256 -51.62 46.46 -109.48 63.44 -79.36 23.256 -51.62
```

```
          </gml:posList>
        </gml:LinearRing>
      </gml:exterior>
    </gml:Polygon>
  </georss:where>
```

Line:

```
<georss:where>
  <gml:LineString>
    <gml:posList>
     23.256 -51.62 46.46 -109.48 63.44 -79.36
    </gml:posList>
  </gml:LineString>
</georss:where>
```

Box:

```
<georss:where>
  <gml:Envelope>
    <gml:lowerCorner>23.256 -51.62 </gml:lowerCorner>
     <gml:upperCorner>46.46 -109.48 </gml:upperCorner>
   </gml:Envelope>
 </georss:where>
```

Listing 3.8: GeoRSS GML example

```
1
2  <?xml version="1.0" encoding="utf-8"?>
3  <feed xmlns="http://www.w3.org/2005/Atom"
4    xmlns:georss="http://www.georss.org/georss">
5    <title>TU Wien</title>
6    <subtitle>Vienna University of Technology</subtitle>
7    <link href="http://www.tuwien.ac.at/"/>
8    <updated>2010-07-23T08:30:02Z</updated>
9    <author>
10     <name>Homa Rezaie</name>
11     <email>Rezaiehoma@gmail.com</email>
12   </author>
13   <entry>
14     <title>Library</title>
15     <link href="http://www.ub.tuwien.ac.at/eng/index.html"/>
16     <updated>2010-07-22T07:32:38Z</updated>
17     <summary>Vienna University of Technology Library</summary>
18     <georss:where>
19        <gml:Point>
20           <gml:pos>23.256 -51.62</gml:pos>
21        </gml:Point>
22     </georss:where>
```

```
23    </entry>
24  </feed>
```

CHAPTER 4

Use Cases

In this chapter we will discuss the uses cases proposed and defined in this thesis in order to present the relation between the two web technologies Semantic Web and GeoRSS. The use-case ”PTPSN” investigates how impaired people can access information on public transport (e.g. the change of timetables). Second use-case ”VA& ES” uses the end-user's information when an accident happens and combine them with the appropriate information of required services (e.g. hospital information, police information, and etc.) in order to provide the selected list of services. Third use-case ”Tourism Consulter” presents how the Internet technologies allow users to easily access location-based tourist information systems. Fourth use-case ”SGAL”, using the two technologies of web services *Semantic Web and GeoRSS*, will illustrate how the proposed method can be used for flight information scenarios.

4.1 Public-Transport for People with Special Needs (PTPSN)

The people who are visually impaired do not have access to suitable location information. This lack of access limits their use of public transportation and effectively denies them equal access to many locations, such as public buildings. This indicates a clear need for navigation systems that improve the way-finding abilities of the visually impaired people, especially for finding public transportation stations and the stations' interior layout. Navigation systems that could provide this information would allow the visually impaired to become familiar with the location, allowing them to recognize when they have arrived at the desired location. This would effectively eliminate the need to constantly ask people along the way in order to verify whether or not they have reached the correct place.

Knowledge of the basic spatial relationships between objects in one's environment, including information about position, direction, desired location, route planning etc., is essential to the

concept of *orientation* as a basic component of human way-finding. The visually impaired are at a considerable disadvantage with respect to this crucial aspect, as they often lack accurate information about their current position. This kind of information is essential to finding an integrated solution that would facilitate way-finding for the visually impaired.

Recent years have brought an increase of orientation and way-finding technologies as well as navigation systems for impaired people. With the help of navigation systems, geographical information and web services, visually impaired people can already access a wealth of location information and services such as information about indoor or outdoor environments, travel timetables, etc. However, despite these recent technological advances, there is still considerable need for Human Computer Interaction research. This section developed the concept of a navigation system for the visually impaired that would integrate a Geographical information system and environment information system with the aid of Semantic Web and GeoRSS technologies.

The main goal of public transport for impaired people is to facilitate and improve users' ability to access geographically dispersed data provided by a set of corresponding services, while they are on the move. Thus, users are enabled to utilize information depending on their geographic location and their specific requirements.

4.1.1 Use case "PTPSN" Overview

4.1.1.1 Public-Transport User Groups

Impaired people may have a wide variety of functional impairments, and it should be noted that only some disabled people are blind or depend on wheelchairs. However, any activity limitation that prohibits the free movement of a person means that the person has mobility impairment.

The following sub-divisions give some indication of the wide range of mobility impairments: [Buhalis]

- Blind/partially blind people
- Deaf and people with hearing problems
- People unable to walk, i.e. wheelchair users
- People who have difficulties with walking
- People with cognitive impairments/learning difficulties

For many of these people having knowledge of where they are or where they are going is very important. Thus, they need to have a personalized intelligent navigation system that updates the information, locates the public transport requested and guides the user to the position of the nearest/best accessible public transport services that meet the user requirements such as

Accessibility, time table, etc...

4.1.1.2 Public-Transport User Needs

Obviously, user needs between the distinct subgroups of impaired people differ considerably. In order to illustrate just how much, this section provides/discusses some examples of how geographical information would have to be accessed through Web Services. The user needs of the introduced groups can be summarized as follows [Fink]:

- For blind users, for instance, location information must be presented as audio output. Only then can it help them to access the information immediately and allow them to arrive at the public transport station as quickly as possible. Moreover, blind people need an internal representation or model of their environment. The crucial point is that the system should be able to navigate its users through finding the way from their current position to the target object or location [Weber].
- The user group comprised of users that either depends on a wheelchair or have difficulties with walking needs detailed location information concerning the accessibility of public-transport stations (e.g. the existence and dimensions of stairs and elevators, the type and width of doors). The information about the layout of interior and exterior of a station is very important and should therefore be automatically provided (e.g., this information will be updated every 20 seconds). All this information must be presented both on a display and as audio output [Nordin].
- For the user group including the deaf and people suffering from cognitive impairments, the information must be presented on a display in full pages of text. In addition, these users need to be informed and updated, again on a display in the form of text, about any information that comes from central stations and is broadcasted by loudspeaker (e.g. for changes in the timetable or announcements of the expected time until the arrival of transportation) [Brewer].

In recent years, the access that impaired people with special needs have to geographic information through the web has been studied. The following section will propose how impaired people can be guided to the target object or location with the help of web technologies such as Semantic Web and Geographic Really Simple Syndication. With a collaboration of both technologies, these users could gain easy access to correct and exact information.

4.1.2 Use case ”PTPSN” Scenario

In order to ground our study, we present this use case as an example that will illustrate the most important features of the collaboration between SW & GeoRSS on the web.

The use case scenario is defined as follows: A person who is visually impaired is in a city and would like to organize a day by visiting some places, meeting friends etc. This makes it necessary for him to plan his movement to his intended destinations, possibly using a combination of public transport means (e.g. bus, metro, taxi, and tramway). Furthermore, let us also assume that this exemplary user moves with a wheelchair and does not know the destination of his route in advance. Moreover, he needs to know information about his route and public transport locations on the way, including the following:

- shortest way to his destination;
- nearest public transport station information;
- services at this station (e.g. does it have a lift or a special access for wheelchairs);
- information about events affecting his route, means of public transport and the station (e.g. delays, construction work etc.);

Because this information is both essential and impossible to obtain by normal means, he needs a navigation system that can help him arrive at his destination. He could express some of his requests through a query, which consists of user requests.

Let us assume, for instance, that our exemplary user would like to go and meet a friend at a coffee house and wants to take the shortest way from his current location to the coffeehouse. Since he moves with a wheelchair and wants to use public transportation, he will look for the nearest metro station, which must have either a lift or special access for wheelchairs, but he also needs to know about events affecting the station or the chosen means of transportation (e.g. the change of a time table or the down-time of a lift from 2 pm until 2:30 pm for maintenance etc.).

Current search engines are incapable of processing this user query. In order to deal with such a query, first of all, the query should have access to an appropriate data source. In this case, that entails that the information about public transport stations should be updated and accurate, the events information must exist in a source data etc. Moreover, we need a suitable strategy to constitute data sources which can keep the integrated data in different domains. Thus, to reach the desired result and improve the use of the geographic data source, we propose collaboration between SW & GeoRSS which can understand:

- where the user is and which matching services exist;
- what the user requests and what he needs to know ;

The desired outcome is that the user will be presented a suitable result of this query by the help of SW & GeoRSS technologies. In this use case, that means, the location of the nearest metro station that meets the user requirements according to station's announced services.

4.2 Vehicle Accident and Emergency Services (VA & ES)

The World-Wide Web provides the ability to access geographical location information around the globe. Nowadays, this ability is being used on various web sites which contain information about hospitals, restaurant, hotels etc. However, much of this information is presented to web users as an address and/or a point on a map in the form of an image with a short description. Unfortunately, current search engines do not analyze the geographical location of queries to help people find information that relates to a particular location. Thus, when the user searches for a particular place by entering the name of a place, query results often include web pages that are not geographically relevant to the user query [Gravano] [Jones]. In an emergency case, users can obtain a proper result to their query only if they can completely specify the information they need. In this section we will investigate how users gain access to geographical location information in the use case "VA&ES" with the help of various technologies on the web.

When we attempt to find a geographical information location, it is important that the result provided refers to a location close to the query. Unfortunately, there are numerous different locations we do not need. Therefore, we propose collaboration between two existing technologies on the web in order to improve the use of information locations so as to help users find a specific location and accurate information. For instance, the scenario is an accident that has taken place on the street and somebody has been injured in the form of a broken foot. The exemplary user in this scenario seeks a hospital, which should be the nearest hospital to the user's position and have an orthopedic department. If the user directs his query at a standard search engine, e.g., google.at or my.yahoo.com, the search engine will then use this information to rank web sites which have no relevance to the query and eventually return references to all hospitals and orthopedic departments in the world. Note that this variety of queries is not equivalent to the search engine strategy for *"hospital AND orthopedic department AND user's position"* since such a query would miss references to find a hospital, too. We thus propose the Semantic Web as a way to improve the organization of information and access to geographical location information.

Moreover, we also propose the use of GeoRSS technology to obtain information related to a specific location. This technology offers an advantage by establishing a relationship between geographic coordinates and a set of new information available about that specific location. Thus, the web provides a new resource for location information, which users can use to access locations close to their position. GeoRSS technology plays a very important role in this use case since emergency responses in the event of an accident requires more information about the hospital, police and fire brigade. Such information can be vital if, for instance, a hospital's orthopedic department does not have power at the moment or if the orthopedic specialist is currently not at the hospital that would be closest to the user's position. Through the collaboration between the Semantic Web and GeoRSS, this information will be provided with ontology to standardize different resources used to answer queries.

Accident and Emergency Services is based on the idea to make use of the technologies of Semantic Web and GeoRSS on the web. We believe that both of these technologies can benefit

substantially from the geographic location information for Emergency Services on vehicle accidents for three reasons:

- The location data is still not integrated for responses to emergency queries. Since the information from a hospital is not stored in different domains (domain 1 being surgery capabilities, domain 2 relevant events, and domain 3 the admission of patients), a complex query that relates various to each other is not possible;
- The location data is available only in some forms such as street addresses, place names or short descriptions like the data in car navigation systems or web map services. If, for instance, we try to find information about a hospital via Google or another map service, we would not receive enough information (i.e. free beds, specific units and departments etc.);
- The technology that current search engines use presents a severe limitation on to the search for geographical information. If, for instance, we were to look for a hotel that is close to downtown, near the metro and in the 4-star category, the access to geographical data would be limited

In the following section, we describe how Semantic Web and GeoRSS collaborate through a classification and facilitating retrieval of geographical information in this use case. In addition, we present a specific transportation problem in city, propose a way to reduce this problem, and explicate why using Semantic Web and GeoRSS makes sense in this use case.

4.2.1 Use case ”VA&ES”: Overview

Today, access to emergency services during a vehicle accident on the street is usually provided through a call centre as a police service. After receiving the interpretation of the accident, the call centre will be handling all services required by this accident. If, for instance, a call informs the call centre about casualties, it will request ambulance services of the nearest hospital.

To achieve an improved handling of emergency services, we propose to use two technologies on the web to allow an intelligent search by information integration. The information is sorted in one GeoRSS feed each. The locations information consists of the following data, which will be embedded into domain ontology:

- Position and information of hospital
- Position and information of police
- Position and information of fire brigade

Furthermore, the information of the accident events which are pushed to server, should include the following information:

- Event's position
- Type of vehicle
  - Car
  - Bus
  - Tramway
  - Motorcycle
  - Bicycle
- Type of accident [Mikulik]
  - Crash with pedestrian
  - Crash with another vehicle
  - Crash with solid obstacle
- Damage [Mikulik]
  - Killed
  - bleeding
  - Burn
  - Heavy injured
  - Slightly injured

The service information feed (GeoRSS), which is needed to respond to a user query, would be embedded into domain ontology. The result of the collaboration between the two technologies is a semantic temporal model that is generated and represents the service providers and their corresponding services. An event feed would be made by the user, when an accident has happened, and it would be refined by user requirements. In summary, we propose that emergency services can be improved with the efficient and effective collaboration between web technologies.

4.2.2 Use case "VA&ES" Scenario

A possible scenario for *Vehicle accident and emergency Services* is discussed in the following. For the sake of illustration, we assume that an accident has happened on a street in the city of Vienna. This accident occurred between two vehicles, a car and a motorcycle. The motorcycle's driver was injured and his foot is broken. Therefore, in this situation an exemplary user would

need the police for their report on the accident, the ambulance for the motorcycle's driver and the fire brigade for the damaged vehicle.

If the user does not know in advance at which geographical point these three are located, he could formulate some requests based on the type of accident and his current geographical location. The user might, for instance, need a hospital which has an orthopedic department, is close to the accident's location and is also a public hospital. Or the accident might require the police because the accident (i.e. its type) involves a car and a motorcycle. If, on the other hand, the accident was between two bicycles, it would not need the police. And if, finally, both vehicles were damaged in the accident, the fire brigade would be needed. Depending on the specifics of the actual accident, the user needs information about certain places and some services, e.g. a hospital with orthopedic department, police station or fire brigade, which are closest to the user's current position and the accident?

To answer the user request, numerous distributed and heterogeneous data sources are accessed [Zhub] and the relevant information resources should be consulted in order to find an appropriate response to the user query. The query should be guided to proper data sources and should select some desirable locations. Take, for instance, the search for a hospital: The query has accessed many hospitals, all of which are near the user, but which ones are suitable? At this point, an appropriate approach to finding a hospital should be adopted, i.e. one that will yield better results.

What should the user know to receive an accurate answer?

- *Where is the user*? The user's geographical position should be defined; if the user does not know at which position he is, he cannot receive an accurate answer, because there are many information sources that are in fact irrelevant to the user.
- *What does the user need*? In other words, the user should know whether somebody is injured, how and to what degree, as well as whether any vehicles are damaged. This information will help to receive an apposite answer as the probability for an inaccurate search is more likely, the less information is known.
- *Is the search query incorrect*? If there are two or more different possible locations, e.g. two streets with the same name, the user should retry to select the option and should check his requirements before submitting them.

To summarize the approach we illustrated with this scenario: we proposed a solution based on the collaboration between two technologies on the web. The geographical information locations update (GeoRSS) should be integrated into different domains. The better approach to defining these domains is offered by Semantic Web technology. These technologies will provide effective means to provide responses to user queries. If the user gives exact information on his position and the nature of the event, he will receive an accurate answer.

4.3 Tourism Consulter

Today, there are numerous different navigation systems which allow users to find places and retrieve information on their current location. For instance, a user requests the nearest restaurant or hotel to his position. However, these systems do not contain information that would also be desirable for the user, such as reservations, menus and other services. Moreover, some users would like to be notified about events occurring at any time, such as when the selected hotel cannot accept the passengers this morning. Incidentally, users generally do not like to access the same information many times, unless they are explicated and accessible at any time.

Location information concerns many different locations, e.g. museums, historical buildings, hotels, public transport stations, restaurants, parks etc., which are points of interest for tourists. In the case of many locations, information is already useful and available for developing tourism systems. Through accessing locations databases and web technologies (GeoRSS& Semantic Web), it's safe to say, these technologies allow more flexible access to information for tourist support. Furthermore, users can thus access different services (e.g. through mobile technology) from any location, making traveling itself more flexible.

The use case "Tourist consulter" presented in this section is based on the idea to use more technologies on the web. At present, the Internet and mobile technologies allow users to easily access tourist information systems for detailed information about things like hotels. However, existing tourist guides have severe shortcomings when it comes to finding location information or a particular event's location. To overcome this problem, this section proposes the collaboration between two technologies on the web, i.e. the Semantic Web and GeoRSS, for use in tourist services. The GeoRSS technology would present the geographical location information and events in RSS format, while the Semantic Web provides explicit meaning and organizes the geographical information into different domain ontologies.

We believe that both technologies can benefit substantially from geographical location information to provide access to tourism services because:

- With the use of GeoRSS, the location information will be updated every few minutes, thus always allowing tourists to access the new information about locations or events.
- The Semantic Web technologies have proven to be a valid solution for the integration of information location; and [Noy]
- Only part of the information that is available for a given location is actually needed by a tourist, and this selection will be saved to different domains and kept accessible.

In this use case, we describe the potential of web services using Semantic Web technologies and GeoRSS for geo-spatial information integration. Through the collaboration of both technologies, tourists can use tourist systems with more confidence and their queries will receive response of higher value.

4.3.1 Use case ”Tourism consulter”: Overview

The internet already has the largest source of tourist information, and the information extracted from these sources can help guide users to their destinations. The ”tourism consulter” would provide the tourist with information on such matters as sightseeing, hotel accommodation or museums according to his geographic position. By using Semantic Web for geographic data sources would be integrated into different domain ontologies which, in the processing of a user query, could be related to each other.

With regard to the increasing popularity of Web services, today's tourists no longer need to search for locations with a map in hand. At present, there are numerous guide systems on the web, which present touristic points of interest in cities or other areas. To improve the performance of tourist services, we propose the ”Tourism consulter”, which would use the technologies of Semantic Web & GeoRSS. The following areas offer possibilities for applying the proposed ”Tourism consulter”:

- Public transport; tourists should know which public transport stations are near their position and which services they offer, as well as any events affecting these stations (e.g. they would like to know that the nearest station has a lift or that there has been a change in the timetable).
- Sightseeing; in this area, tourists can select some places in their domain of interest (e.g. palaces, attractive squares, old buildings with attractive architecture & design, central parks, museums & exhibitions etc.). Tourists would also have the opportunity to select, for example, that they would like to visit a palace which is located near the metro and not far from his position, but which also offers guided tours for tourists.
- Music & stage shows; this option helps tourists find concert halls and stages (e.g. venues for opera & operetta, classical music, musicals, dance, theater, pop, rock, jazz etc.). Should tourists decide to visit one of these venues, they have access to all information and events. This includes, for instance, date and time of a performance, time to play, information about a performer or group or changes of the timetable.
- Shopping & restaurants; this option includes all the restaurants, shopping centers, shops and coffeehouses in a city, including events such as opening hours, specialties of a shop, and type of food. If, for instance, a tourist would like to shop, he would search for a shopping center near his hotel where he can buy clothes and which is near both an Italian Restaurant and a cinema that has a reservation service.
- Hotels & Accommodations; finding a hotel is one of the most important things for tourists. They can find suitable accommodations on grounds of their requirements. They can, for instance, request a hotel which is opposite the beach, has 4 stars, features a restaurant that has international food, and whose staff speaks German.

- Emergency services; this option includes some emergency services in a city (i.e. police, emergency medical services, pharmacies and the fire brigade). If, for instance, a tourist has lost his passport, he could search for a police station which is not only near his position but also has an officer that can speak French.

These applications represent some common areas of tourist requests in which the proposed "Tourism consulter" would provide an intelligent search for tourism services

.

4.3.2 Use case "Tourism consulter" Scenario

The scenario for the use case "Tourism consulter" is the following: A tourist is in a city and would like to visit a place of historical interest which lies outside the city limits. He can either travel with public transport or look for a hotel in the downtown area near a hospital which has a coronary care unit. Thus, he would like to plan his visit by using a combination of public transport (e.g. bus or train) or his whereabouts with a combination of emergency services.

In this situation, the tourist needs information on the following:

- What route planner to use in the city.
- Regarding public transport, how can he travel to his destination outside the city? Where is the closest public transport station to his position?
- What is the history of the place he wants to visit?
- Which hotel is closest to a hospital with a coronary unit in the downtown area?
- Which service does this hotel have and what information on events is available right now?

We can thus say that our exemplary tourist needs a service system that allows him to select his requests. For instance, he would like to go and visit a museum or some famous buildings (e.g. a guildhall or church) of the city; or he would like to know the program for concerts, movies or the theater; or else he would like to visit another city of interest and needs information about his distance to this city, historical information about the city, weather conditions at this time and miscellaneous other information. Thus, his destination can be a place of interest (e.g. famous buildings that he can select from the ones present in the city, which are near his position and reachable through public transport) or his goal can simply be to know current events at the location (e.g. a change of the timetable or the distance from his position).

To handle the tourist's query, a huge amount of information is available through various tourist service systems on the web sites through which these heterogeneous information systems are distributed. In many cases, the information location does not contain details, does not specify events or provide sufficient explanation about a location. Frequently, users therefore get information that they did not explicitly ask for or don't need. Therefore, the information

location should be subdivided into different kinds and different levels of detail, which presents a challenge [Fonseca]. Beyond this, the query has to be directed to a relevant database (i.e. data sources for historical data on the location, public transport, points of interest etc.) so that the result will present the relevant information for the location the tourist is interested in.

But we still need to discuss the conditions for obtaining good results:

First, the tourist should determine his position because "Tourist consulter" will guide the user from his position to his destination. Sometimes the user may need more than one service concurrently, which must be near his position (e.g. the user would like to visit a museum, but to arrive there, he should use a vehicle and, first of all, needs to be certain of his current position).

Second, the tourist should have decided what his point of interest is: where does he want to go and how does he want to travel? He can determine a number of options for his query (e.g. he wants to go to a restaurant which has French food, is near his hotel and has live music at 6pm). The "Tourist consulter" will guide the user to his target and inform him of events at the desired location, if the user makes his query in the appropriate way.

To provide a technical design for the "Tourist consulter", we propose to navigate the tourist with the help of two existing technologies (GeoRSS & Semantic Web) on the web. In collaboration, both technologies offer an intelligent and flexible search engine for this use case which can help by providing accurate search results and a high degree of user satisfaction. On the one hand, the GeoRSS feed provides the ability to access the location information with events location; on the other hand, the Semantic Web offers a better solution to managing GeoRSS feeds in an intelligent way. Thus, with using these technologies, tourists can always confidently trust the query results.

4.4 Smart Guide Airplane Landing (SGAL)

At present, a large proportion of information and/or resources available on the web consists of and is presented as geographical information (GI), i.e. information on a city, including address, hotels, metro, schools, businesses, hospitals, parks etc. Thus most activities on the internet are a result of the proliferation of geographical services (e.g. route planning for a city, finding the location of, for example, and tourist services). Today, such services are used by everybody in different user groups. For instance, the "tourist" user group can use them to obtain geographical information about sightseeing, music, shopping etc. in a city, or the "driver" user group can access urban information through GPS. With our increased ability to use geographic information and new technologies on the web, we would like to turn out attention to a special user group which may use this information in the future of web services.

We propose pilots as a new user group that could use the geographic information during flight and landing. As a group, these users need more information and knowledge about locations. We believe that geographical information needs to be introduced and used, and will be a major driving force behind future web services. However, geographic information is used for different applications, but must be described so as to encompass domains as semantic. Web services offer technologies to establish more facilities for searching, accessing, extracting, interpreting and processing information. The use case "Smart Guide Airplane landing", using the two technologies of web services Semantic Web and GeoRSS, will illustrate how the user group "pilots" can benefit from this potential. The proposed solution thus uses advanced technologies to access and present geographical information, supports specific location data and shows integration of geographic data.

In this use case, we assume that a pilot would like to know which airport is nearest to his position, which runway he can land on, and – in the case of an emergency landing – which emergency services are available at this airport? In other words, the pilot needs enough information in order to correctly predict the aircraft's behavior, but unnecessary information would present a data overload for the pilot. In the following, we shall illustrate how the pilot can access enough information with the help of both technologies. The following data resources should be provided for the SGAL since the pilot needs them:

- All information on geographical location and events that are relevant to airplane landing should be collected;
- The information should be updated every 20 seconds
- The updated data should be stored in different domain ontologies, which are related to each other within the search engine.

Consequently, intelligent data that is useful for a pilot query about landing an airplane would be provided through a collaboration of two existing technologies of web services (SW & GeoRSS).

4.4.1 Use case "SGAL" Overview

Today there is a variety of technological means for navigation which help pilots achieve a smooth and safe landing. Pilots generally do not have to worry about finding a free runway and can easily land on an open field that has been provided. As airline traffic grew, the need to provide systems for navigation increased likewise. With the use of GPS receivers, pilots have access to information on the nearest airport, including identifier, bearing and distance to the airport – all of which is provided for the 10 to 20 airports closest to the airplane's current position [Williams].

For the improved and effective use of current navigation systems and geographic information, we propose the collaboration of Semantic Web and GeoRSS to allow an intelligent search. Such a search would give the pilot access to the information he needs and guide him to the nearest airport. The exemplary pilot in our use case needs the following information:

- Airport position (pilot requests the nearest airport to his position);
- Airport services for passengers and airplanes (e.g. emergency services for passengers, since a passenger has a heart problem and needs medical attention; or the airplane itself has a technical problem and requires technical support for the airplane's specification);
- Airport's free runways;
- Airport's events (e.g. a runway will be free after 5 minutes or another airplane has just made a emergency landing and all airplanes must wait before landing);
- Airport's weather report;
- Airport's traffic;

This information is provided directly by the respective airports to the airplane, meaning that it does not exist on the web, and can be accessed by the pilot through making a query. This use case presents an advantage with the use of current technologies and collaboration between them and web services. In our view, this will provide the most effective way of presenting information on the nearest airport to the pilot.

4.4.2 Use Case "SGAL" Scenario

In the following, we construct a simple scenario to illustrate the role and benefits of both technologies (SW & GeoRSS). The scenario for the use case SGAL application is the following:

Let us assume a Boeing 767 airplane is flying over a major metropolitan area when it encounters a technical problem and should be landed as soon as possible. The pilot will search for the following [Kolas]:

- Where are the nearest airports to his position?
- Are these airports capable of supporting a 767?
- Would the weather conditions at the airports allow an emergency landing?
- At which airport will a runway be most easily cleared?

- What other activities are currently underway at each potential location?
- Which airport has access to the best medical facilities and mechanical facilities?
- Which airport has emergency services for passengers?

In order to receive an adequate response to the above query, the pilot needs a service system that cans accesses to the information in the space of a few second. The query search we propose has access to the different geographical data repositories, containing some geospatial (location information) and some nongeospatial data (events). Therefore, this query has to be considered a compound of several different queries, which have a relationship to each other that can be expressed with the word AND. Moreover, the answer should in fact contain several answers which, at the end, combine to identify an exact location or several options for the pilot.

For instance, the pilot makes the following query: Which airport is near the pilot's position and is there a free runway and are there emergency services for passengers? If the answer to this query is positive, the pilot will have received an exact airport with events information (e.g. there is no free runway at present, but there will be after 3 minutes). Otherwise, the search engine would propose other airports (e.g. an airport which isn't near the current position, but has a free runway and emergency services for passengers etc.).

We would like to investigate how pilots can find an airport which conforms to their requirements and how to generate events data, because most of the available data sources don't have search functionalities based on the current location of the pilot. We are, however, convinced that with the use of the two web technologies, SGAL has access to the adequate information. We can obtain geographic airport information from the GPS satellite and modify it in two ways:

- The data is modified with geo-data, with adds a few or more geo-tags to the RSS that hold geographic data and will be combined into a GeoRSS.
- For more effective identification, automation and integration of the GeoRSS data, the Semantic Web integrates the geographic data into different domain ontologies.

The collaboration between the Semantic Web and GeoRSS suggested above enhances the potential and capability of using geo-data. Consequently, instead of searching disparate sources of information, the pilot has access to the appropriate data sources with one query only - which would break down the composite query into its components and return an appropriate answer.

CHAPTER 5

Technical Solutions

This chapter describes the technical solution and the fundamental design of the proposed approach my means of the Semantic Web technology.

Section 5.1 concentrates on the required functionality of this approach, while Section 5.2 describes details of the design of each component for the proposed use-cases. The section on component-design illustrates the four steps of designing this approach and presents the application of selected use-case for Vehicle Accident and Emergency Services (VA & ES).

5.1 Functionality Requirements

To illustrate the required functionality, let us use one of the use cases that will be implemented at a later stage. The selected use-case is Vehicle Accident and Emergency Services (VA & ES). In the context of hospital services and as an example, let us focus surgery department service whose specifications and assume that properties of surgery service are listed in the resources of different hospitals, such as AKH, St.Anna, Lainz and others. Furthermore, we also assume that the surgery department of the hospital AKH has special properties, e.g. *"surgeryTeamservice", "surgeryRoomService"*. This means that the surgery department has a surgery team, a surgery room and a free bed to retain a patient after surgery. In contrast, the hospital Linz lists other properties for this item, for example *"surgeryRoomService"and "surgeryTeamService"*. In addition, the AKH may also provide any additional pieces of information about the surgery department to receive the patient. This information will be required to find a hospital in which the surgery department is ready to receive a patient and its location is near the accident's location.

While we have information about the respective surgery departments of hospitals, there remain three main problems which should be solved to use this information. First, we lack comprehensive access to all information to find a hospital that is not only ready to receive a patient

now, but is also near an accident's location. Second, we need information which details the current situation of the surgery department. For instance, the surgery information must be updated every 20 sec. Third, the existing information is available without any clear geographic background, i.e. we don't have the geographical point of the hospitals.

Thus the main problems in using the information available on the web are:

- There is a lack of web services to store information from various resources in a unique structure which facilitates the search for and finding of a specific location near the user.
- Much of the information on the web does not include geographical information.
- Much of the geographical information on the web is not integrated (with Semantic Web technolohies).
- While geographical information is easy to understand for human users, it is difficult for computers to define and represent geographic knowledge.

As the first step, geographical information and service information for the use case VA & ES will be received and parsed from different sources that are accessible via corresponding GeoRSS feeds. Since RSS is becoming more and more prevalent as a way to publish and share information, it becomes increasingly important that locations are described in an interoperable manner [Zhang]. By using the RSS format, the event information from an accident relating to the police, hospital, and fire brigade will be published every 20 seconds.Furthermore, the new geographical information is automatically inserted into the GeoRSS feed by the information providers. Therefore the location information must be parsed from GeoRSS feeds.

As the second step, the geo-data from GeoRSS feeds will be embedded into uniform ontology which will be explained letter and refer to temporal ontology. In this step the GeoRSS data are extracted and added to temporal ontology, which we will need to answer the required queries.

The third step uses semantic inference and applies semantic rules to find matching triples for introduced use-cases. In this example, if surgery department has some specific services that match the user requirements, then it will be ready to accept and handle that patient. The inference (entailment) rules define how new propositions are derived from previously established ones [Bar].

The following figure represents a simple example of the functionality requirements for the collaboration between Semantic Web & GeoRSS.

To summarize, the required steps of proposed solution are:

- An ontology model to create reference model for management of geographical information from data sources available on the web.

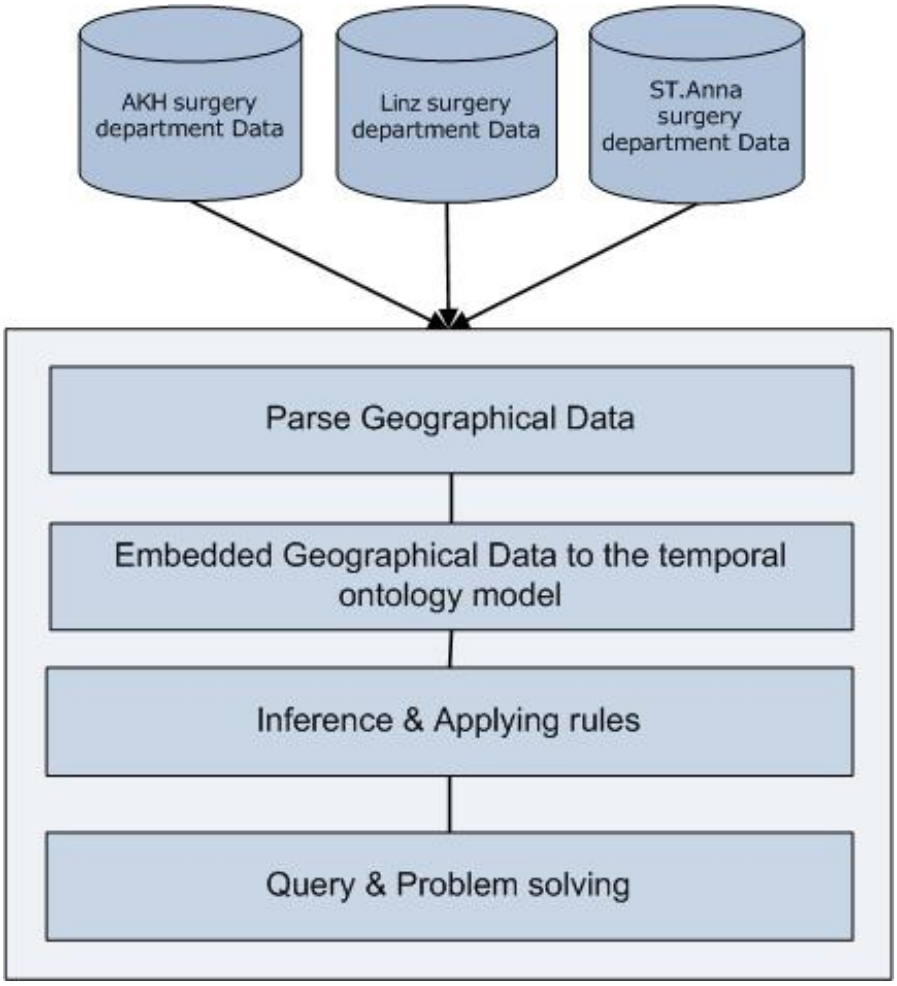

Figure 5.1: Overview of proposed solution

- Parsing geographical information about the location of events from different sources.
- Embedding event information into the temporal ontology from different data sources.
- Inference and applying some rules to find appropriate services.
- The appropriate Semantic queries on the temporal ontology will result the required services which are prioritized based on the distance to user's location.

5.2 Conceptual Design

In this section we will define the workflow and essential components to create a comprehensive temporal model on GeoRSS information. Figure 5.3 shows the overview our approach for the

Applying Semantic Web Concept to GeoRSS.

Geo-Data parsing: In the first phase, information about geographical location is parsed from various GeoRSS feeds, which we need to create an intelligent data base. The GeoRSS feeds include the semantic-encoded information of available services that comply with the proposed Main ontology. This feeds data is used to define the location of events and their corresponding services every few seconds, i.e. the geo-data will be parsed every few seconds.

Geo-Data embedding: In the second phase, available geo-data from different GeoRSS feeds is embedded into the temporal ontology in RDF format. In the proposed approach an ontology model for each use case is created as described in Section 4. After parsing each use-case event. Therefore, an RDF dataset with corresponding domain information will be created for each use case.

Inference and applying rules: In the third phase, by defining some rules the problem constraints will be applied to the information for finding the best solution (for instance, finding a hospital that meets the requirements of a GeoRSS event of the type accident).

Query and problem solving: After adding the triples, in this step the Semantic Web query language will be used to find all required services from temporal repository and list them to find the nearest location to user's position.

The SW & GeoRSS approach comprises the following activities shown in Figure 5.2.

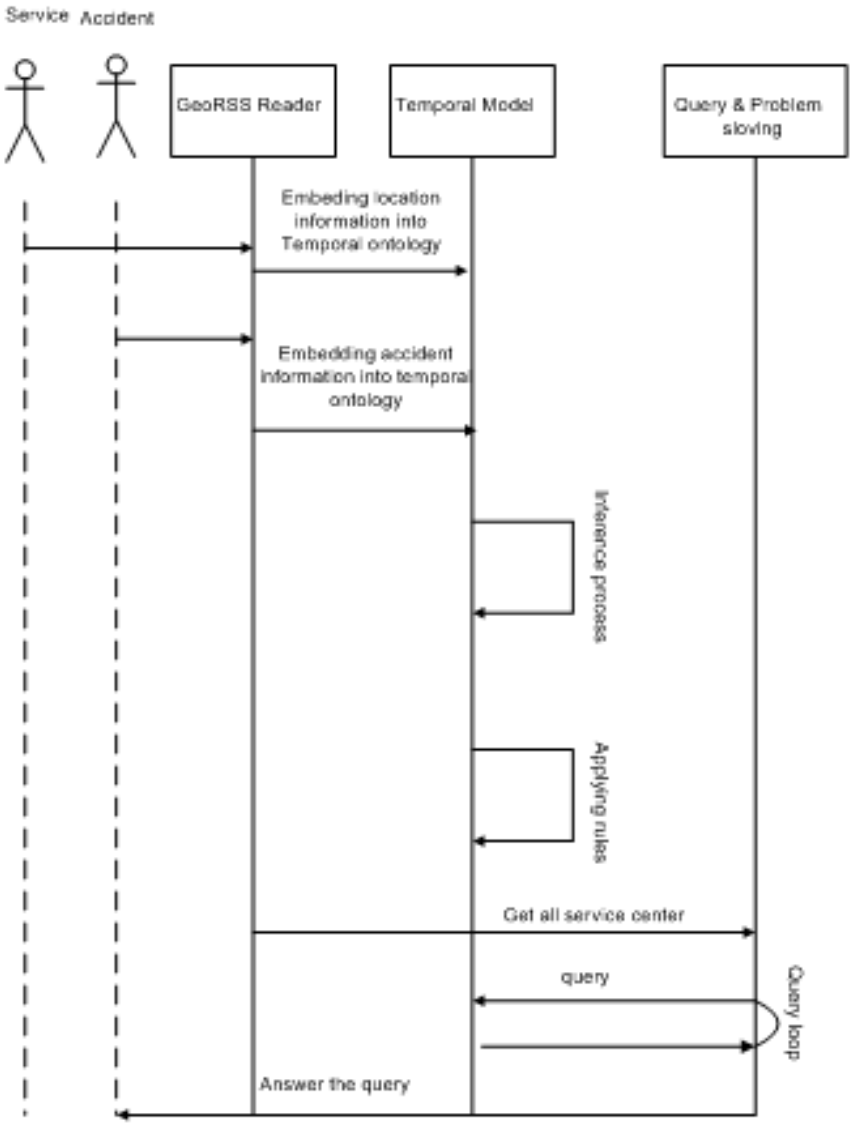

Figure 5.2: SW & GeoRSS use sequence diagram

An overview of the proposed approach is shown in Figure 5.3:

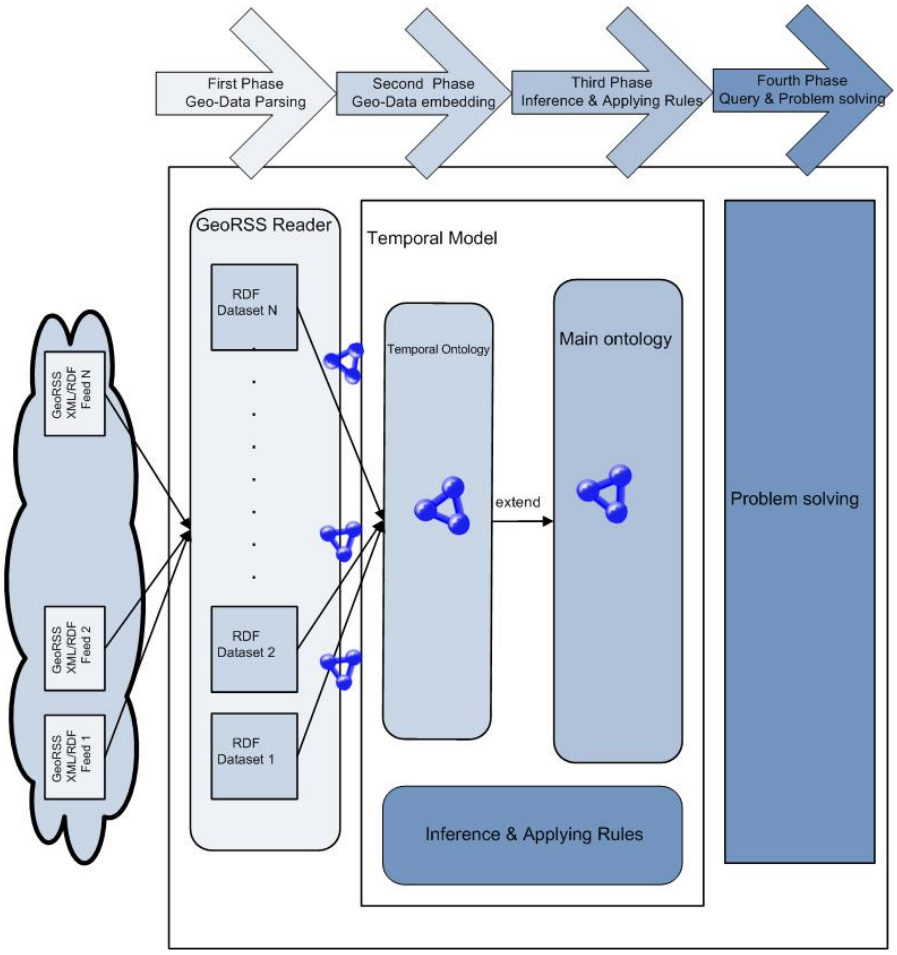

Figure 5.3: Proposed SW & GeoRSS approach

5.2.1 Parsing ComponentDesign

For the purposes of this thesis, XML/RDF GeoRSS feeds contain geographical location information that is updated every few seconds. This data will be parsed from different feeds and stored in a single OWL/RDF ontology. In this section, we briefly discuss the requirements for parsing XML/RDF geo-data to OWL ontology.

The following snippet of GeoRSS shows how semantic information can be embedded into a GeoRSS feed:

Listing 5.1: GoRSS Semantic

```
1
2  <?xml version="1.0"?>
3   <rss version="2.0"
4   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
5   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
6   xmlns:owl="http://www.w3.org/2002/07/owl#"
7   xmlns:georss="http://www.georss.org/georss/"
8   xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
9   xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
10  xmlns:gml="http://www.opengis.net/gml#"
11  xmlns:dc="http://purl.org/dc/elements/1.1/">
12   <channel rdf:about="http://www.police.at/rss">
13     <title>Police Station GeoRSS feed</title>
14     <link>http://www.police.at/</link>
15     <description>My example channel</description>
16     <item>
17      <title>News about Arndtstr Police station</title>
18      <link>http://www.police.at/Station/Arndtstr/</link>
19      <description>other things happened today</description>
20
21      <!-- http://www.accident.org/ontologies/Temporal_final.owl#Arndtstr -->
22     <owl:NamedIndividual rdf:about="&Temporal_final;Arndtstr">
23         <rdf:type rdf:resource="&Accident;TraficPolice"/>
24         <Accident:hasService rdf:resource="&Accident;hasFreePolicepatrol"/>
25         <Accident:hasService rdf:resource="&Accident;hasTrafficPolice"/>
26         <Accident:hasService rdf:resource="&Accident;hashelicopterPolic"/>
27     </owl:NamedIndividual>
28                 <georss:where>
29               <georss:point>
30                    <gml:pos>
31                       <haspoint>98.46-256.258</haspoint>
32               </gml:pos>
33             </georss:point>
34          </georss:where>
35
36          </item>
37            </channel>
38  </rss>
```

This component is responsible for searching for the required geo-data in a feed, e.g. finding all available data for a specific location's information (e.g. information about a hotel, airport, bus station etc.), and for loading this data to a defined destination. For instance, an airport provides information for the landing of airplanes in XML/RDF format. For the extraction of the geo-data, we should define the necessary geographical information, which is needed to land at the airport. It is formulated as a query for information on landing from the GeoRSS landing feed (e.g. free runways, airport position, airport services, airport traffic, and the airport's weather report) so that the query result will contain our needed information.

5.2.2 Embedding Component Design

As described in the previous section, we parse the geodata from different GeoRSS feeds into some RDF snippets. The next step is to add these individual RDF snippets datasets to the respective domain ontology and creating the temporal ontology. This temporal model contains an ontology model that describes all classes and sub-classes for each use cases. Finally the semantic inference & rules will be applied to this model (see figure 5.3).

Each use case offers specific services e.g. in the use case "PTPSN", a metro station offers services such as lift or a special access route for wheelchairs, etc. However, some properties may change after a few seconds. If, for instance, a metro station has a black out for 5 minutes, these properties will be changed. With the support of our ontology model, all properties are described in homogeneous levels, which mean that we could parse all properties that are needed for service category (e.g. properties related to Metro Station *Karlsplatz*) in our ontology model. As an example, consider the term *"hasService"* as a property of Metro Station Karlsplatz from the use case *"PTPSN"* in our ontology. Furthermore, in the case of a specific Metro Station Karlsplatz, we describe all other properties for this individual, like , *"escalator"*, which is part of the GeoRSS Metro Station feed. The following code shows the description of one individual *"Metro Station Karlsplatz"* in the GeoRSS XML/RDF feed.

Listing 5.2: GeoRSS Metro Station Karlsplatz

```
1
2  <?xml version="1.0"?>
3
4   <rss version="2.0"
5   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
6   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
7   xmlns:owl="http://www.w3.org/2002/07/owl#"
8   xmlns:georss="http://www.georss.org/georss/"
9   xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
10  xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
11  xmlns:gml="http://www.opengis.net/gml#"
12  xmlns:dc="http://purl.org/dc/elements/1.1/">
13   <channel rdf:about="http://firebrigade.at/news.rss">
14     <title>Firebrigade AviationUnit GeoRSS feed</title>
15     <link>http://www.firebrigade.at/</link>
16     <description>My example channel</description>
17
18  <item>
19      <title>News about Karlsplatz metro station</title>
20      <link>http://www.Station.at#MStation/Karlsplatz</link>
21      <description>New Event Station</description>
22
23          <!-- http://www.metro.org/ontologies/Temporal_final.owl#Karlsplatz -->
24    <owl:NamedIndividual rdf:about="&Temporal_final:Karlsplatz ">
25         <rdf:type rdf:resource="&Metro;MetroStation"/>
26         <Metrot:hasService rdf:resource="&Metro;lift"/>
27         <Metro:hasService rdf:resource="&Metro;stair"/>
28         <Metro:hasService rdf:resource="&Metro;escalator"/>
29         <Metrot:hasService rdf:resource="&Metro;wayAccessWheelchairs"/>
30         <Metro:hasEventInfo rdf:resource="&Metro;noPower"/>
```

```
        <Metro:hasEventInfo rdf:resource="&Metro;trainDelay"/>
    </owl:NamedIndividual>
              <georss:where>
             <georss:point>
                  <gml:pos>
                   <haspoint>18.30 -14.321</haspoint>
                </gml:pos>
           </georss:point>
          </georss:where>

</item>

  </channel>
</rss>
```

After defining all services/events in XML/RDF GeoRSS feeds and the semantic relations of each property the geographical data will be embedded into ontology model.

5.2.3 Inference & Applying Rule Component Design

As described in the two last sections, we parsed geo-data from GeoRSS XML/RDF feeds and embedded this geo-data into an Owl/RDF temporal ontology model, giving us an RDF geo-dataset compiled from different feeds. The next step is to add triples with the use of Semantic Web constraints and inference process. We defined in our ontology model some define classes that automatically classify domain objects and classes in the temporal ontology.

Listing 5.3: Tourism Consulter temporal ontology (museum information)

```


    <!-- http://www.accident.org/ontologies/Temporal_final.owl#Leopold -->

    <owl:NamedIndividual rdf:about="&Temporal_final;Leopold">
        <rdf:type rdf:resource="&Museum;MuseumInfo"/>
        <Museum:hasServices rdf:resource="&Museum;onlineTicketBuy"/>
        <Museum:hasProgram rdf:resource="&Museum;nightExhibitions"/>
        <Museum:hasProgram rdf:resource="&Museum;nightProgram"/>
        <Museum:hasProgram rdf:resource="&Museum;exhibitions"/>
        <Museum:hasServices rdf:resource="&Museum;touristGuideGerman"/>
        <Museum:hasServices rdf:resource="&Museum;touristGuideSpain"/>
        <Museum:hasServices rdf:resource="&Museum;touristGuideEnglish"/>
        <Museum:hasServices rdf:resource="&Museum;onlineShop"/>
        <Museum:hasServices rdf:resource="&Museum;onlineOrder"/>
        <Museum:hasProgram rdf:resource="&Museum;childExhibition"/>
        <Museum:hasProgram rdf:resource="&Museum;childfilm"/>
        <Museum:hasProgram rdf:resource="&Museum;childTheater"/>
        <Museum:hasProgram rdf:resource="&Museum;childCulture"/>
        <Museum:hasProgram rdf:resource="&Museum;childMusicr"/>
        <Museum:hasServices rdf:resource="&Museum;onlinePay"/>
        <Museum:hasServices rdf:resource="&Museum;onlinereservation"/>
    </owl:NamedIndividual>
```

```
25
26      <!-- http://www.accident.org/ontologies/Temporal_final.owl#Albertina -->
27
28     <owl:NamedIndividual rdf:about="&Temporal_final;Albertina">
29          <rdf:type rdf:resource="&Museum;MuseumInfo"/>
30          <Museum:hasServices rdf:resource="&Museum;onlineTicketBuy"/>
31          <Museum:hasProgram rdf:resource="&Museum;nightExhibitions"/>
32          <Museum:hasProgram rdf:resource="&Museum;exhibitions"/>
33          <Museum:hasServices rdf:resource="&Museum;touristGuideGerman"/>
34          <Museum:hasServices rdf:resource="&Museum;touristGuideSpain"/>
35          <Museum:hasServices rdf:resource="&Museum;touristGuideEnglish"/>
36          <Museum:hasServices rdf:resource="&Museum;onlineShop"/>
37          <Museum:hasServices rdf:resource="&Museum;onlineOrder"/>
38          <Museum:hasServices rdf:resource="&Museum;onlinePay"/>
39          <Museum:hasServices rdf:resource="&Museum;onlinereservation"/>
40          <Museum:hasProgram rdf:resource="&Museum;childExhibition"/>
41          <Museum:hasProgram rdf:resource="&Museum;childfilm"/>
42          <Museum:hasProgram rdf:resource="&Museum;childCulture"/>
43          <Museum:hasProgram rdf:resource="&Museum;childMusic"/>
44          <Museum:hasServices rdf:resource="&Museum;groupvisits"/>
45         <Museum:hasServices rdf:resource="&Museum;onlinebooking"/>
46    </owl:NamedIndividual>
```

The rules of this ontology are as follows:

Listing 5.4: Rules of ontology Tourism Consulter

```
Rule1: (?Museum  Museum:hasServices  Museum:onlinePay)
            (?Museum  Museum:hasServices  Museum:onlineTicketBuy)
            (?Museum  Museum:hasServices  Museum:onlinereservation)->
                  (?Museum  Museum:hasServices  Museum:onlineTicketService )

Rule2: (?Museum  Museum:hasProgram  Museum:nightExhibitions)
            (?Museum  Museum:hasProgram  Museum:nightProgram) ->
                  (?Museum  Museum:hassServices  Museum:nightVisit)

Rule3: (?Museum  Museum:hasServices  Museum:onlineShop)
            (?Museum  Museum:hasServices  Museum:onlineOrder)
            (?Museum  Museum:hasServices  Museum:onlinePay)  ->
                  (?Museum  Museum:hasServices  Museum:onlineShopService)

Rule4: (?Museum  Museum:hasServices  Museum: touristGuideGerman)
            (?Museum  Museum:hasServices Museum: touristGuideSpain)
            (?Museum  Museum:hasServices Museum:touristGuideEnglish) ->
                 (?Museum  Museum:hasServices Museum:multiLanguagetouristGuide)

Rule5: (?Museum  Museum:hasProgram  Museum:childExhibition)
           (?Museum  Museum:hasProgram  Museum:childFilm)
           (?Museum  Museum:hasProgram  Museum:childTheater)
           (?Museum  Museum:hasProgram  Museum:childCulture)
           (?Museum  Museum:hasProgram  Museum:childMusic) ->
                (?Museum  Museum:hasServices  Museum:childProgramPackage)
```

The first rule provides the information about online ticket service, i.e. if the user can online ticket buy/reservation/online pay/online tickets book this museum has online service for ticket. The second rule states that if the museum has some program in night like music, theater, dance,

and etc and there are exhibitions in the night in this museum then there are services for visiting at night. The third rule determines that if there is online shopping and online order and online pay, there is an online shop service. The fourth rule states that if guided tours are offered in different languages (e.g. German, Italian, Spain, English), there is a multiple language guide. The fifth rule determines that if there is some program for children like culture, theater and etc., the museum services offer a package program for children. As a result, these rules help to organize information on museum services so that the user would be given access to the right information location.

5.2.4 Query & Problem Solving Component Design

According to the three last steps described above, we have RDF temporal repository from different GeoRSS geo-data feeds and RDF instances, which are the outputs of the embedding component and the inference & applying rules component. We now have these standardized geographical information is in a temporal RDF repository.

The first query list all requested services that user needs via the following query:

The ontology model also includes a class which specifically defines our defined-classes. This class contains all services/ events that our user group needs for accessing geographical information about locations. To access the location information, the first query will list all services / events and the second query provides access to the location that has these services/events . For instance, the use-case Smart Guide Airplane landing (SGAL) needs the following services for landing an airplane at an airport near its position:

- Free runway for landing
- Supporting services for airplane of type X after landing
- Supporting of emergency landing
- Emergency services for passengers

The first query lists all services requested by the user

Listing 5.5: First SPARQL query

```
SELECT  ?service

WHERE {

?Airplan_A01   Airport:requestService ?service

   }
```

The second query lists all airports which has the requested services with their points through the following query:

Listing 5.6: Second SPARQL query

```
SELECT  ?location  ?Position

WHERE {

        ?location  rdf:type ?LandingServiceReady.
        ?location  rdf:type ?PassengerServiceReady.
        ?location  Airport:hasPoint ?Position
        }
```

5.2.5 Use-case VA & ES Component Design

In this section we will design the use case Vehicle Accident and Emergency Services (VA & ES) as an example. Figure 5.4 shows the VA & ES use-case architecture. The class "Domain Concept" defines all domains in this ontology. For the first phase, the needed geographical information is parsed from different feeds. The GeoRSS feeds (events/services) are in XML/RDF format for this use case and are defined for:

- Police
- Medical center
- Fire brigade
- Accident

In the second phase, geographical information is embedded into the ontology model of the use case VA & ES, which contains classes and subclasses for each category in this use case. For instance, the category Medical center contains the class Hospital, Clinic, and several subclasses that the information from different GeoRSS medical center feeds which will be embedded into temporal ontology domain *"MedicalCenter"*. In this use case we defined some services for each category, therefore a class with name *"Service"* is defined which contains subclasses that they defines all services for each category. For instance, the subclass *"hospitalServices"* contains all services which are related to the medical center.

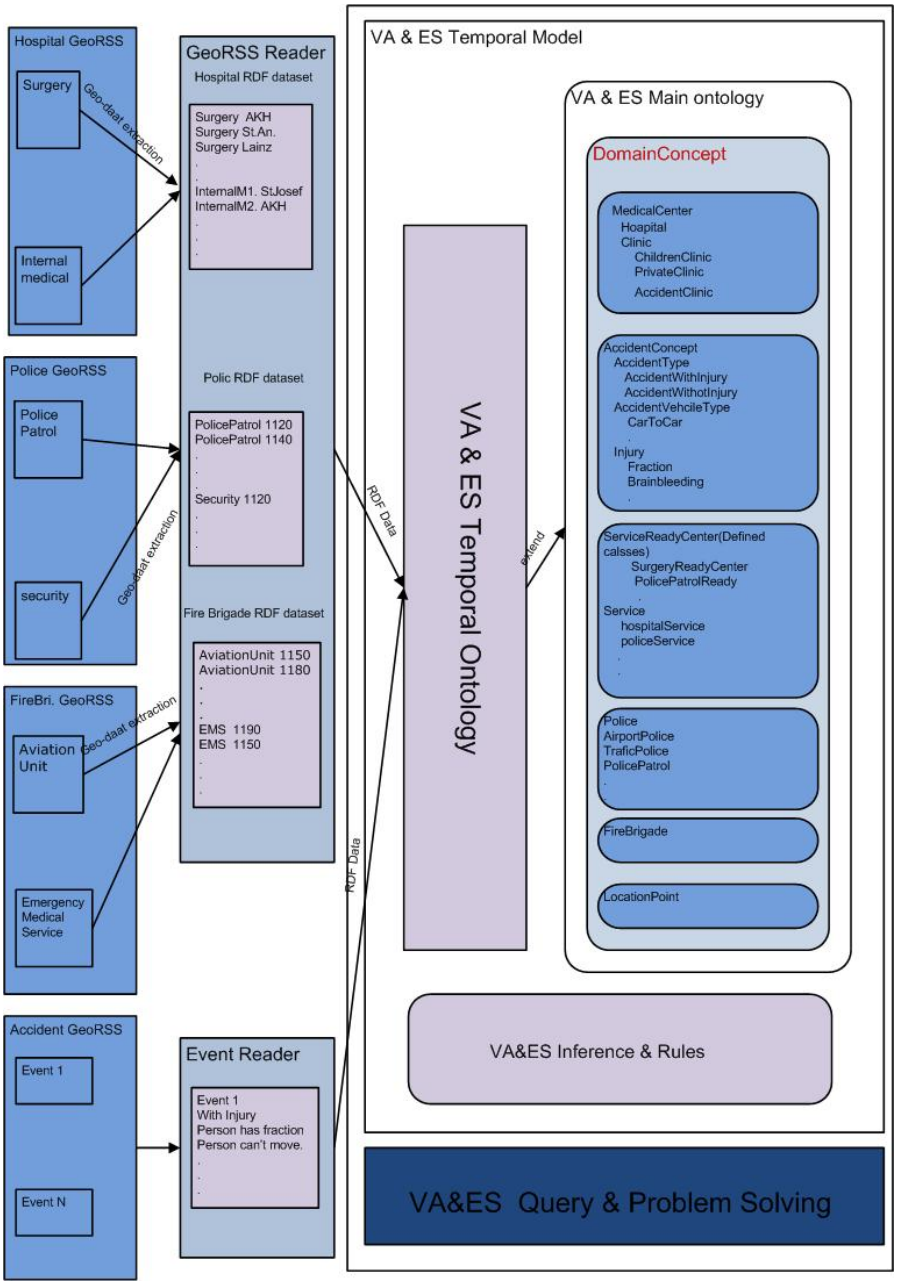

Figure 5.4: VA & ES use-case architecture

In our example of *"hospitalServices"*we have services e.g. InternalMedicine, LabDiagnostice, SurgeryRoomService, SurgeryTeamService, and etc... Also there is a class in VA & ES ontology model which defines the define classes with name *"ServiceReadyCenter"*. This class represents our services that are required for accident events. For instance we assume the defined _calss *SurgeryReadyCenter* as a service center that has a surgery room service and also

a surgery team service. We also assume, if a Medical center has this two requirements then it is ready to reception patients who need surgery. During the inference phase, the surgery ready medical centers can be extracted from this defined class. After embedding all request into VA & ES temporal ontology, the reasoner will be started and we will get the centers (for instance hospitals) classified under required classes (for instance *SurgeryReadyCenter*) which is shown in the following screenshots.

Figure 5.5 defines inference rule in the temporal ontology model for class ”SurgeryReady-Center”. The service center is ready if it has a ready surgery room and surgery team. All centers that contain the both services after inference will be classified under ”SurgeryReadyCenter”.

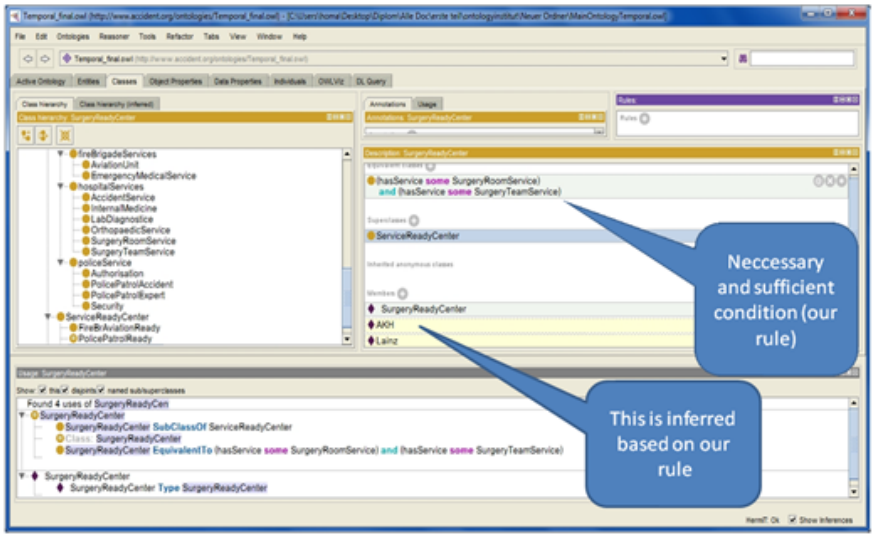

Figure 5.5: Inference rules in the temporal ontology

Figure 5.6 defines detail of our inference rules for the specific case of ”AKH” hospital. This hospital has surgery room service ready and surgery team service ready then it is a surgery center which is ready to accept patient that needs surgery.

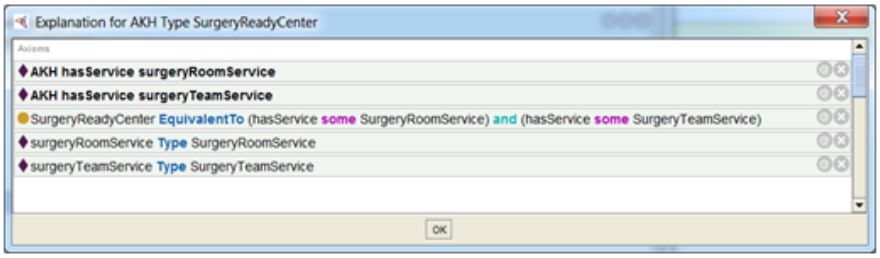

Figure 5.6: Detail of inference rules in the temporal ontology

When an accident happens, the GeoRSS of accident is sent to our system. The information event from accident GeoRSS will refer to the concepts in main ontology. We defined a set of injuries, accident types and accident vehicle types in the main ontology under domain concepts, which determine the needed services for different accidents.

The following snippet of temporal ontology shows the sample semantic information which is extracted from different GeoRSS feeds:

Listing 5.7: VA & ES information in temporal ontology

```
1
2
3    <!-- http://www.accident.org/ontologies/Temporal_final.owl#AKH -->
4      <owl:NamedIndividual rdf:about="Temporal_final;AKH">
5          <rdf:type rdf:resource="&Accident;Hospital"/>
6          <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
7          <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
8          <Accident:hasPoint>344.13 -251.224</Accident:hasPoint>
9      </owl:NamedIndividual>
10
11     <!-- http://www.accident.org/ontologies/Temporal_final.owl#Lainz -->
12     <owl:NamedIndividual rdf:about="Temporal_final;Lainz">
13         <rdf:type rdf:resource="&Accident;Hospital"/>
14         <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
15         <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
16         <Accident:hasPoint>256.23 -851.204</Accident:hasPoint>
17     </owl:NamedIndividual>
18
19            <!-- http://www.accident.org/ontologies/Temporal_final.owl#St.Anna -->
20         <owl:NamedIndividual rdf:about="&Temporal_final;St.Anna">
21         <rdf:type rdf:resource="&Accident;Childrenclinic"/>
22         <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
23            <Accident:haspoint>32.401 -551.101</Accident:haspoint>
24         </owl:NamedIndividual>
25
26 <!-- http://www.accident.org/ontologies/   Temporal_final.owl#MariahilferSt_Accident1 -->
27           <owl:NamedIndividual rdf:about="&Temporal_final;MariahilferSt_Accident1">
28           <Accident:hasPoint> 46.183 -123.816 </Accident:hasPoint>
29           <rdf:type rdf:resource="&Accident;AccidentWithInjury"/>
30           <Accident:hasInjury rdf:resource="&Accident;canNotMove"/>
31           <Accident:hasInjury rdf:resource="&Accident;bleeding"/>
32           <Accident:hasInjury rdf:resource="&Accident;skullfracture"/>
```

```
33          <Accident:hasAccidentVehicelType rdf:resource="&Accident;carToCar"/>
34      </owl:NamedIndividual>
```

In the third phase, several rules will be defined for finding the best solution from our temporal ontology based on accident properties and requirements (for instance, finding a hospital that meets the requirements of a GeoRSS event of the type accident). The first step of AV & ES rule is implemented as define classes in our proposed ontology model (see the screenshot 1). The second step of VA & ES are some rules. The following listings show some sample rules for accident use-case:

Listing 5.8: VA & ES Rules

```
1
2
3   [ rege1:  (?a Accident:hasInjury Accident:bleeding)
4             (?a Accident:hasInjury Accident:canNotMove)
5                     (?a Accident:hasInjury Accident:skullfracture)->
6                               (?a  Accident:requestService Accident:_SurgeryReadyCenter )]
7
8   [ rege2: (?a Accident:hasInjury Accident:burn)
9             (?a Accident:hasInjury Accident:hasInjury Accident:canNotMove)
10                  ->(?a Accident:requestService Accident:_DempartmentBurnReadyCenter)]
11
12   [ rege3:
13            (?a Accident:hasInjury Accident:hemorrahge)
14            (?a Accident:hasInjury Accident:footFracture)
15                  ->(?a Accident:requestService Accident:_OrthopaedicsReadyCenter)]
16
17   [ rege4:
18            (?a Accident:hasVehicel Accident:carToCar)
19            (?a Accident:hasCarDamage Accident:carBodyDamage)
20                  (?p Accident:personhascomplaint Accident:indict)
21                  ->(?a Accident:requestService Accident:_PolicePatrolReady)]
22
23   [ rege5:
24            (?a Accident:hasVehicel Accident:carToCar)
25            (?a Accident:hasCarDamage Accident:carBodyDamage)
26                  ->(?a Accident:requestService Accident:_FireBrigadeReady)]
27
28     [ rege6:
29            (?a Accident:hasVehicel Accident:carToCar)
30            (?a Accident:hasCarDamage Accident:carBodyDamage)
31                   (?a Accident:hasCarDamage Accident:carfire)
32                  ->(?a Accident:requestService Accident:_FireBrigadeReady)]
```

The first three rules define the needed hospital services. For instance if the person has bleeding and cannot move and has skull fracture then he needs surgery. The forth rules defines the needed request for police service and the fifth to sixth rules define the needed services of fire brigade service. After create an inferred model using Jena, these rules will be applied to temporal ontology and will result some triples about required services.

the Semantic query language *"SPARQL"* will be used to created some query for the accident information to find the services which are nearest to the current user position. In the temporal ontology model is defined the needed services for accident that we discussed in the last section (Inference & Applying Rules). To access the needed services, first we make a query to find all services that are ready, for instance the needed hospital services which are ready or the needed police services and other services. After listing all services ready we make the second query to find service centers which are ready. For instance user needs surgery and police patrol, the first query list *"_SurgeryReadyCenter" "_PolicePatrolReady"* and the second query list all centers which are ready e.g. for *SurgeryReadyCenter* the hospitals "AKH" and "Lainz" are ready.

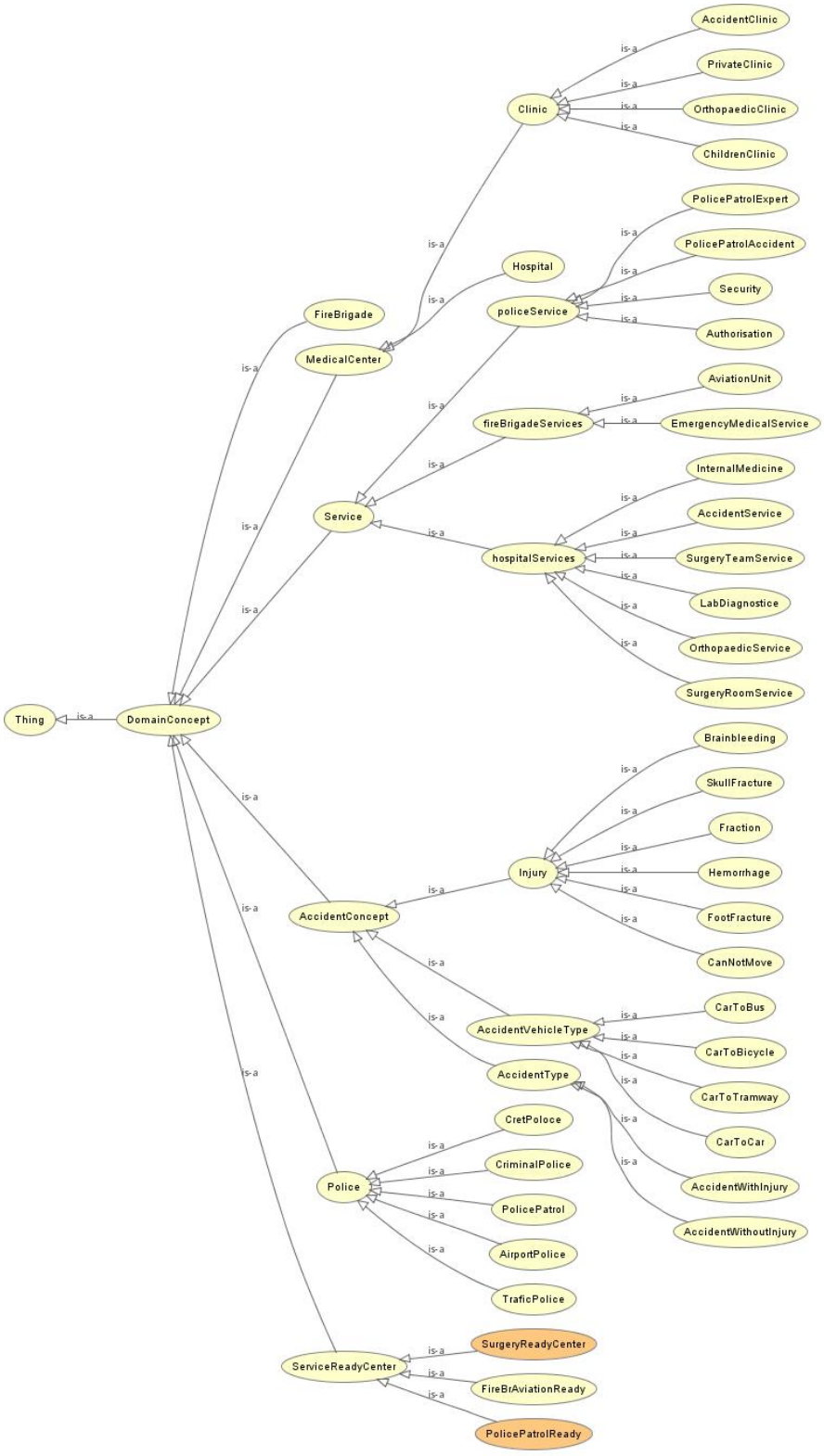

Figure 5.7: AV & ES Temporal Ontology Model

CHAPTER 6

Implementation & Validation

This section describes the implementation of the approach proposed in this thesis, using the Java programming language (JSE-1.6) and Jena API. We discuss in detail the technologies used to implement each component. As proof of concept the Vehicle Accident and Emergency Services use-case (VA& ES) will be explored.

6.1 Parser Implementation

To implement the parsing component, we assume that a Java class (parsingData) would parse the geo-data from different GeoRSS feeds. The requested data is in GeoRSS format with an embedded RDF snippet and will be embedded into an ontology model to create a temporal repository.

The class parsingData is responsible to parse the geo-data from different GeoRSS feeds and provide the RDF dataset. The input parameter of this class is *feed URL* (defines the name of GeoRSS feed e.g. police.rss) which contains location services/events and location.

6.1.1 Embedding Implementation

To implement the second component, i.e. the embedding of geo-data obtained from parsed GeoRSS feeds, we assume that the Java class *"embeddingStatement"*' is used to embed the geo-data into a temporal ontology.

The class *embeddingStatement* is responsible for embedding the parsed RDF data mentioned in the previous section into the temporal ontology model. The input is the extracted RDF snippet plus location information in RDF form.

6.1.2 Applying Rule implementation

As indicated in the last section, several rules will be provided for the event/service. To implement the rule component we use rule language to create these rules as well as the Jena library *com.hp.hpl.jena.reasoner.rulesys.Rule*. This library provides a selection of simple rule engines for Jena inference models.

The rule component looks first in the temporal ontology model (data repository) and then uses the Jena library *com.hp.hp1.jena.reasoner.Reasoner* to create an OWL reasoner which uses rules to create an inferred model.

6.1.3 Query & Problem Solving Implementation

The Jena API is used in a more advanced way to implement the query component. Jena is an open source RDF framework designed to support RDF schema inference and query. For each process of this component, a compatible query has to be written (e.g., choose requested services from the RDF temporal dataset).

The implementation of the query component uses the Semantic Web query language SPARQL, which is described in section 3 (RDF query language). To perform the query process via Jena we use the *com.hp.hp1.jena.query.** library. The Java class *geoQuery* is responsible for listing the requested services/events and the locations at which these services/events are situated. The input parameter of this class is *ontologyName*, which defines the temporal ontology file, and its output is the result of the query.

6.1.4 Distance Implementation

While this thesis focuses on the geo-ontology model, another highly important application of geographical data is to allow users to search for the geographical point nearest to their current position. Therefore, this section will discuss the matter of determining a location's proximity to a user's position, for example which hotel is nearest to a tourist's current position. The location query point is between two consecutive sampled positions, the user's own position point and the position point of the destination location. There are several possible approaches to finding the geographical point that is closest to the query point, including, for instance, *Linear Search, Space partitioning, Locality hashing* [Distance] etc.

Although the nearest point to the query point is defined in terms of the distance between the query point and any other points in the data set as the shortest distance, this cannot be used to determine any route. To calculate the distance between two points, we use Pythagoras' Theorem. This algorithm is a very simple means of finding the distance points that are closest to the user's position. Figure 6.1 illustrates object **A** with a number of location points **B1,B2,B3,B4**

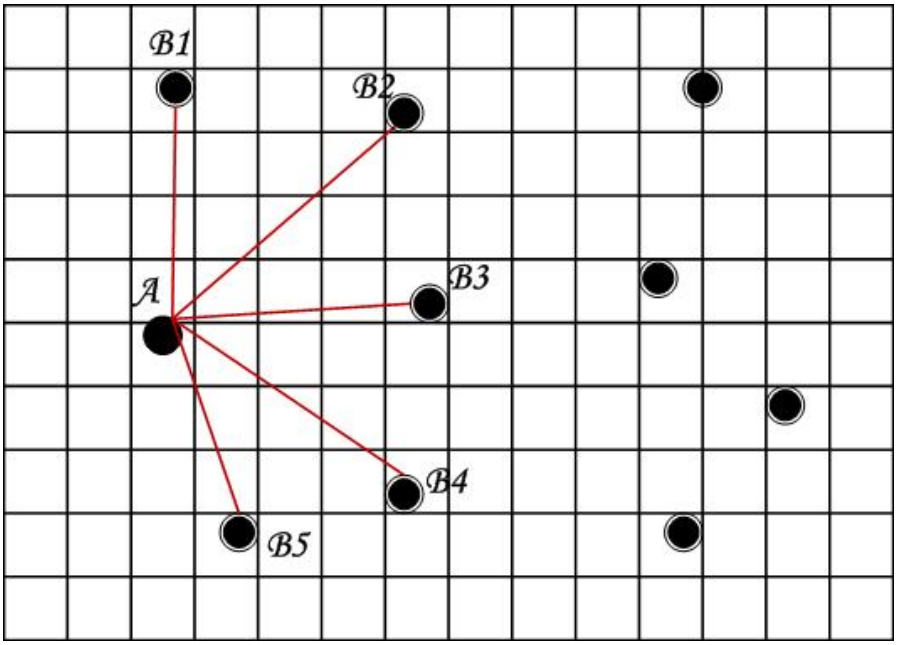

Figure 6.1: Query point and its nearest location point

(i,j) and **B5 (k,l)** – the goal is to determine which location point is closest to the user **A**.

To implement this part we use the java class *PointCalculator* which is responsible for calculating the distance between geographical points. The input parameters of this class are: user point (x1, y1) and the points of the geographical locations obtained as the result to the class *geoQuery*, e.g. location1 (x2, y2), location2 (x3, y3) etc. The nearest location is the output of the class *PointCalculator* . This rather direct and simple method could be replaced with more complicated methods of calculating path based on street maps.

6.2 Evaluations and Validation of Solution

In order to demonstrate the approach proposed in this section, we will now discuss the ontology model structure of the use-case Vehicle Accident and Emergency Services (VA & ES) as designed in section 4.2.4. This section will also define the parsed and embedded information as well as the implementation of the rule and query components for the VA & ES use-case.

6.2.1 VA & ES Geo-Data structure

As mentioned in chapter 4, the use-case VA & ES offers sets of information about an accident event and the location of needed services (e.g. hospital information). The VA & ES ontology model is created by using the Protege Semantic web ontology development platform.

The following code shows the ontology model of the VA & ES use-case that defines classes, sub-classes, properties and some individuals. MedicalCenter, police and firebrigade, Accident-Concept are classes/subclasses in which some individuals must later be embedded. The classes also contain subclasses which define different services for those classes (e.g. sub-class Service contains classes *"hospitalServices", "policeServices", and "firebrigadeServices"*).

Listing 6.1: Ontology Model of the VA & ES

```
1
2  <!DOCTYPE rdf:RDF
3
4  .
5  .
6  .
7
8           // Object Properties
9  <owl:ObjectProperty rdf:about="http://www.accident.org/ontologies/Accident.owl#hasAccidentVehicelType">
10 <owl:ObjectProperty rdf:about="http://www.accident.org/ontologies/Accident.owl#hasInjury">
11 <owl:ObjectProperty rdf:about="http://www.accident.org/ontologies/Accident.owl#hasService">
12            // Data properties
13
14   <owl:DatatypeProperty rdf:about="http://www.accident.org/ontologies/Accident.owl#hasPoint">
15         // Classes
16  <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentClinic">
17 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Clinic"/>
18 <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentConcept">
19 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>
20 <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#TraficPolice">
21 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Police"/>
22 <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#fireBrigadeServices">
23 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
24 <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#hospitalServices">
25 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
26  <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#policeServices">
27 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
28 .
29 .
30 .
31
32 // Individuals
33 <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#canNotMove">
34 <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#carToCar">
35     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#hemorrhage">
36     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#skullfracture">
37 .
38 .
39 .
40
41 </rdf:RDF>
```

The RDF temporal datasets of services/events and related requested services entities will be created in the next section. Each service/event category in the RDF temporal repository is created according to an individual geo-ontology. Moreover, each category of service/event is de-

fined by a particular namespace and each instance of a category is identified by a particular URI which includes the ”Accident” namespace.

The following section describes the internal structure of this temporal repository as well as the relation between service instances and the related requested services within this temporal repository according to the relevant geo-ontology classes.

6.2.1.1 Service

This class defines a set of services for hospital services, police services, and fire brigade services in the VA & ES ontology model. It uses the following format:

- The first part of URI is identical for each service category `http://www.accident.org/ontologies/Accident.owl`
- The second part is the name of that particular class of services (e.g. hospitalServices)

Listing 6.2: VA & ES Services

```
1
2
3
4   <rdf:RDF
5  .
6  .
7      <!-- http://www.accident.org/ontologies/Accident.owl#hospitalServices -->
8
9  <owl:Class rdf:about="http://www.accident.org/ontologies Accident.owl#hospitalServices">
10 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
11    </owl:Class>
12 .
13 .
14
15     <!-- http://www.accident.org/ontologies/Accident.owl#policeService -->
16
17   <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#policeService">
18 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
19     </owl:Class>
20 .
21 .
22
23     <!-- http://www.accident.org/ontologies/Accident.owl#fireBrigadeServices -->
24  <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#fireBrigadeServices">
25   <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
26     </owl:Class>
27 .
28 .
29
30 </rdf>
```

Each service lists the following services:

- *hospitalServices*: this service contains all services of a medical center, e.g. *InternalMedicine, LabDiagnostic OrthopaedicServices* etc. The example below includes the following services in the AV & ES ontology model:

Listing 6.3: VA & ES Hospital Services

```
1
2
3  <!-- http://www.accident.org/ontologies/Accident.owl#SurgeryRoomService -->
4  <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#SurgeryRoomService">
5  <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
6   </owl:Class>
7  <!-- http://www.accident.org/ontologies/Accident.owl#SurgeryTeamService -->
8   <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#SurgeryTeamService">
9   <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
10   </owl:Class>
11   <!-- http://www.accident.org/ontologies/Accident.owl#OrthopaedicService -->
12   <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#OrthopaedicService">
13   <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
14      </owl:Class>
15 .
16 .
17 .
```

- *policeService*: this service contains all services of a police station, e.g. *AirportService, TrafficService, AccidentService* etc. As an example, the following services are included in the AV & ES ontology model:

Listing 6.4: VA & ES Police Services

```
1
2   <!-- http://www.accident.org/ontologies/Accident.owl#PolicePatrolExpert -->
3  <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#PolicePatrolExpert">
4   <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#policeService"/>
5      </owl:Class>
6   <!-- http://www.accident.org/ontologies/Accident.owl#Security -->
7   <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Security">
8   <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#policeService"/>
9      </owl:Class>
10 <!-- http://www.accident.org/ontologies/Accident.owl#PolicePatrolExpert -->
11 <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#PolicePatrolExpert">
12 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#policeService"/>
13     </owl:Class>
14 .
15 .
16 .
```

- fireBrigadeServices: this service contains all services of a fire brigade station, e.g. *AviationUnit, EmergencyMedicalService* etc.

As an example, the following services are included in the AV & ES ontology model:

Listing 6.5: VA & ES Fire Brigade Services

```
1
2
3     <!-- http://www.accident.org/ontologies/Accident.owl#_FireBrAviationReady -->
```

```
<owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#_FireBrAviationReady">
 <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#FireBrAviationReady"/>
    </owl:NamedIndividual>

 <!-- http://www.accident.org/ontologies/Accident.owl#AviationUnit -->
 <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AviationUnit">
 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#fireBrigadeServices"/>
    </owl:Class>
.
.
.
```

6.2.1.2 ServiceReadyCenter

This class contains our defined-classes which are described in the VA & ES inference rules in order to relate accident location to service centers (hospital, police, and fire brigade). The following code shows the VA & ES defined-class:

Listing 6.6: VA & ES Defined-class

```


<rdf:RDF
.
.
    <!-- http://www.accident.org/ontologies/Accident.owl#PolicePatrolReady -->

<owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#PolicePatrolReady">
   <owl:equivalentClass>
     <owl:Class>
       <owl:intersectionOf rdf:parseType="Collection">
         <owl:Restriction>
          <owl:onProperty rdf:resource="http://www.accident.org/ontologies/Accident.owl#hasService"/>
            <owl:someValuesFrom rdf:resource="http://www.accident.org/ontologies/Accident.owl#PolicePatrolAccident"/>
              </owl:Restriction>
              <owl:Restriction>
            <owl:onProperty rdf:resource="http://www.accident.org/ontologies/Accident.owl#hasService"/>
          <owl:someValuesFrom rdf:resource="http://www.accident.org/ontologies/Accident.owl#PolicePatrolExpert"/>
       </owl:Restriction>
     </owl:intersectionOf>
   </owl:Class>
  </owl:equivalentClass>
 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#ServiceReadyCenter"/>
    </owl:Class>
.
.
    <!-- http://www.accident.org/ontologies/Accident.owl#SurgeryReadyCenter -->

<owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#SurgeryReadyCenter">
 <owl:equivalentClass>
  <owl:Class>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Restriction>
        <owl:onProperty rdf:resource="http://www.accident.org/ontologies/Accident.owl#hasService"/>
         <owl:someValuesFrom rdf:resource="http://www.accident.org/ontologies/Accident.owl#SurgeryRoomService"/>
          </owl:Restriction>
          <owl:Restriction>
           <owl:onProperty rdf:resource="http://www.accident.org/ontologies/Accident.owl#hasService"/>
           <owl:someValuesFrom rdf:resource="http://www.accident.org/ontologies/Accident.owl#SurgeryTeamService"/>
          </owl:Restriction>
         </owl:intersectionOf>
        </owl:Class>
       </owl:equivalentClass>
     <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#ServiceReadyCenter"/>
    </owl:Class>
.
.
```

```
48
49  </rdf>
```

6.2.2 Parsing Geo-data from VA & ES GeoRSS feed

As mentioned in chapter 4, the use-case VA & ES provides complete sets of information about hospitals, police, fire brigades and their accident-related services in XML/RDF format. As a basis for good relations between the ontology classes, the following information needs to be extracted from the GeoRSS feeds:

- Police information: VA & ES provides a separate XML/RDF file for each category of the geographical information on police services (e.g. police patrol, criminal investigation department, and traffic police) from police-related GeoRSS feeds. These geo-information files contain the information about the event/service of police and the geographical point of locations. For example, these are some instances for the service *"PoliceService"* :

  police patrol position (assigned to a task or not)

  police patrol expert

  security

  police patrol accident

  location point of police patrol

This information defines the position and number of police patrols in a given police station (how many police patrols are ready in the station for this mission) and its geographical position/location. This information will help to find a police patrol that is both ready to assume a new mission and is near the accident's location.

- Hospital information: VA& ES provides a set XML/RDF files for each category of the geographical information on hospitals (e.g. hospital, infirmary, children's hospital, and clinic). This information contains event/services of each type of hospital in addition to geo-location points. The services contain different departments of each type (e.g. Surgery services, internal medicine services, and orthopedics services). The following is an example of the department Surgery in the instance *"hospitalService"*:

  Surgery team Service

  Surgery room Service

This information defines the position of the accident surgery which can contain events related to this department, e.g. a surgery team will be ready in 5 min or a surgery room does not have power and will be ready to use in 20 min.

- Fire brigade information: VA & ES provides similar XML/RDF files of fire brigade geo-data, containing information about the type of mission that can be conducted (e.g. Aviation Unit, Urban Search and Rescue, Fire Investigation, Rural Fire Service, Rescue Services and rendering ambulance aid to the sick or injured and convey them to hospital) as well as the location point of different fire service stations. For example, these are some instances for the service *"firebrigadeService"*:

  Aviation Unit is ready for new mission

  Aviation Unit has water tank

  Aviation Unit contains emergency services

The Aviation Unit's information describes the nature and position of this service.

- Accident information: VA & ES also provides the same XML/RDF files of event information (e.g. the event's position, type of vehicle, type of accident, and damage) as described in chapter 4. The following instances illustrate this information:

  Person has bleeding

  Person cannot move

  Person has skull fracture

  Car has body damage

  Type of accident (e.g. including injuries to a person or not)

Using this information about the accident, we add the services needed for a certain type of accident, for example: The important component of the extraction engine consists of java classes to extract the geo-information from relevant VA & ES GeoRSS feeds. The inputs of these classes are the names of the categories for services (*policeService, hospitalService, firebrigadeService, accidentEvent*) and parse all files relevant to these categories of services. By use of these classes each event/service of emergency services and accident event would be extracted.

6.2.3 Embedding Geo-Data into VA & ES Temporal Ontology Model

In this section we assume that in order to embed geo-data into the temporal ontology we need parsed data as described in the previous section, provided in the following format:

- Police GeoRSS feeds: the following police GeoRSS XML/RDF format will be embedded into the VA & ES temporal ontology model after running the java program embedding class:

Listing 6.7: Police GeoRSS feed

```
1
2
3
4  <rss version="2.0"
5  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
6  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
7  xmlns:owl="http://www.w3.org/2002/07/owl#"
8  xmlns:georss="http://www.georss.org/georss/"
9  xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
10 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
11 xmlns:gml="http://www.opengis.net/gml#"
12 xmlns:dc="http://purl.org/dc/elements/1.1/">
13  <channel rdf:about="http://www.police.at/rss">
14    <title>Police Station GeoRSS feed</title>
15    <link>http://www.police.at/</link>
16    <description>My example channel</description>
17    <item>
18     <title>News about Arndtstr Police station</title>
19     <link>http://www.police.at/Station/Arndtstr/</link>
20     <description>other things happened today</description>
21
22
23    <!-- http://www.accident.org/ontologies/Temporal_final.owl#Arndtstr -->
24
25    <owl:NamedIndividual rdf:about="&Temporal_final;Arndtstr">
26        <rdf:type rdf:resource="&Accident;TraficPolice"/>
27        <Accident:hasService rdf:resource="&Accident;hasFreePolicepatrol"/>
28        <Accident:hasService rdf:resource="&Accident;hasTrafficPolice"/>
29                <Accident:hasService rdf:resource="&Accident;hashelicopterPolic"/>
30    </owl:NamedIndividual>
31
32          <georss:where>
33             <georss:point>
34                   <gml:pos>
35                <haspoint>98.46-256.258</haspoint>
36                </gml:pos>
37           </georss:point>
38         </georss:where>
39
40         </item>
41
42   </channel>
43 </rss>
```

- Hospital GeoRSS feeds: the following hospital GeoRSS XML/RDF format will be embedded into the VA & ES temporal ontology model after running the java program embedding class.

Listing 6.8: Hospital GeoRSS feed

```
<rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://hospital.at/news.rss">
    <title>Lainz Surgery ENT</title>
    <link>http://hospital.at/</link>
    <description>My example channel</description>
    <item>
     <title>News about Lainz Surgery</title>
     <link>http://www.hospital.at/SurgeryENT/Lainz/</link>
     <description>other things happened today</description>

   <!-- http://www.accident.org/ontologies/Temporal_final.owl#Lainz -->
    <owl:NamedIndividual rdf:about="Temporal_final;Lainz">
        <rdf:type rdf:resource="&Accident;Hospital"/>
        <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
        <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
    </owl:NamedIndividual>
              <georss:where>
              <georss:point>
                   <gml:pos>
                 <Accident:hasPoint>256.23 -851.204</Accident:hasPoint>
             </gml:pos>
            </georss:point>
           </georss:where>

         </item>
   </channel>
 </rss>
```

- Fire brigade GeoRSS feeds: the following fire brigade GeoRSS XML/RDF format will be embedded into the VA & ES temporal ontology model after running the java program embedding class.

Listing 6.9: Fire brigade GeoRSS feed

```
<rss version="2.0"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:georss="http://www.georss.org/georss/"
xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
xmlns:gml="http://www.opengis.net/gml#"
xmlns:dc="http://purl.org/dc/elements/1.1/">
 <channel rdf:about="http://firebrigade.at/news.rss">
   <title>Firebrigade AviationUnit GeoRSS feed</title>
   <link>http://www.firebrigade.at/</link>
   <description>My example channel</description>
   <item>
    <title>News about fire brigade Brigittenau station </title>
    <link>http://www.firebrigade.at/Brigittenau/</link>
    <description>other things happened today</description>


   <!-- http://www.accident.org/ontologies/Temporal_final.owl#Brigittenau -->

   <owl:NamedIndividual rdf:about="&Temporal_final;Brigittenau">
       <rdf:type rdf:resource="&Accident;FireBrigade"/>
       <Accident:hasService rdf:resource="&Accident;hasAviationUnit"/>
       <Accident:hasService rdf:resource="&Accident;hasAviationUnitsalvation"/>
      <Accident:hasService rdf:resource="&Accident;hasAviationUnitwatertank"/>
   </owl:NamedIndividual>
                      <georss:where>
                     <georss:point>
                            <gml:pos>
                           <haspoint>256.223 -531.204</haspoint>
                    </gml:pos>
                 </georss:point>
                </georss:where>

        </item>

  </channel>
</rss>
```

- Accident GeoRSS feeds: the following Accident GeoRSS XML/RDF format will be embedded into the VA & ES temporal ontology model after running the java program embedding class.

Listing 6.10: Accident Event

```
1
2  <rss version="2.0"
3  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
4  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
5  xmlns:owl="http://www.w3.org/2002/07/owl#"
6  xmlns:georss="http://www.georss.org/georss/"
7  xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
8  xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
9  xmlns:gml="http://www.opengis.net/gml#"
10 xmlns:dc="http://purl.org/dc/elements/1.1/">
11  <channel rdf:about="http://www.cityaccident.at/news.rss">
12    <title>City Accident GeoRSS feed</title>
13    <link>http://www.cityaccident.at/</link>
14    <description>My example channel</description>
15    <item>
16     <title>News about accident on the Mariahilfer street</title>
17     <link>http://www.cityaccident.at/accident/</link>
18     <description>other things happened today</description>
19
20 <!-- http://www.accident.org/ontologies/Temporal_final.owl#Karlsplatz_Accident2 -->
21
22    <owl:NamedIndividual rdf:about="&Temporal_final;Karlsplatz_Accident2">
23        <rdf:type rdf:resource="&Accident;AccidentWitouthInjury"/>
24                <Accident:hasCarDamage rdf:resource="&Accident;carBodyDamage"/>
25        <Accident:hasCarDamage rdf:resource="&Accident;carfire"/>
26                <Accident:personhascomplaint rdf:resource="&Accident;indict"/>
27                <Accident:hasVehicel rdf:resource="&Accident;carToTramway"/>
28
29    </owl:NamedIndividual>
30                <georss:where>
31              <georss:point>
32                 <gml:pos>
33                       <Accident:hasPoint> 47.25  -43.16 </Accident:hasPoint>
34                 </gml:pos>
35                </georss:point>
36             </georss:where>
37
38         </item>
39
40   </channel>
41 </rss>
```

Complete examples of these GeoRSS feeds are available in Appendix A3, A4, A5, and A6.

6.2.4 VA & ES Inference and Applying Rule Implementation

After embedding the geo-data into the ontology model as detailed in section 5.3.3, the rule component provides several RDF triples which will be added to the use-case temporal ontology (i.e. the use-case temporal RDF repository). In case of our example, the accident event ”Accident”, there exists for this event several RDF triple in the VA & ES temporal RDF repository that has the following properties:

- hasInjury person cannot move
- hasInjury person has bleeding
- Person has skull fracture
- Car has body damage
- Accident type is with injury
- Accident vehicle type is car to car

Using this information about the accident, we can add some required services as defined in our proposed ontology model to the accident information in the temporal repository. The java class ”CreateRules” provides RDF triples as an output. For the sake of illustration, the following services will be added to the accident information as an example:

- If person has bleeding and cannot move and has skull fracture then he needs surgery ready center
- If the type of vehicle accident is car to car and person has indict and car has damage then the needed service ready is police patrol ready
- If the type of vehicle accident is car to car and car has damage with fire then the needed service ready is fire brigade

As an example, the following cod shows the Jena rule for the VA & ES use-case. In regard to the accident information, the requested service ”_SurgeryReadyCenter” will be added to the accident information that defines which services are needed for the user query.

Listing 6.11: Accident Rule

```
1
2
3
4  prefix owl: http://www.w3.org/2002/07/owl#
5  prefix rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#
6  prefix rdfs: http://www.w3.org/2000/01/rdf-schema#
7  prefix georss: http://www.georss.org/georss/#
8  prefix Accident: http://www.accident.org/ontologies/Accident.owl#
9  prefix Temporal_final: http://www.accident.org/ontologies/Temporal_final.owl#
10
11  [ rege1:  (?a Accident:hasInjury Accident:bleeding)
12            (?a Accident:hasInjury Accident:canNotMove)
13                 (?a Accident:hasInjury Accident:skullfracture)
14                 ->(?a  Accident:requestService Accident:_SurgeryReadyCenter )]
```

6.2.5 VA & ES Query and Problem Solving

When the process of embedding the services (police service, hospital service and fire brigade) is completed, we have the VA & ES RDF temporal repository as well as individual RDF instances, which are the outputs of the embedding and rule components. Now, by calling the Jena class Query, we can query all requested services from the temporal repository and access the accident query result which contains the requested services and the nearest location point to the current user position.

The Semantic query of the VA & ES consists of two parts:

- The first query lists all requested services using the following SPARQL query:

Listing 6.12: First SPARQL query for the VA & ES

```
1
2  prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
3  prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>
4  prefix xsd:<http://www.w3.org/2001/XMLSchema#>
5  prefix Accident:<http://www.accident.org/ontologies/Accident.owl#>
6  prefix Temporal_final:<http://www.accident.org/ontologies/Temporal_final.owl#>
7  prefix owl:<http://www.w3.org/2002/07/owl#>
8
9   SELECT  ?service
10
11  WHERE {
12
13  Temporal_final:MariahilferSt_Accident1   Accident:requestService ?service
14
15                             }
```

All services will be listed with name, e.g. *"_SurgeryReadyCenter", "_PoliceReadyCenter", "_FirebrigadeReadyCenter"* etc., for the requested services as specified in the accident information. This list defines which services the user needs that will remove the first character from the name of the service to get the class name, for example *"_SurgeryReadyCenter"* becomes *"SurgeryReadyCenter"*. We use this class name in the second query to list all service centers.

- The second query lists all centers that are ready to provide the requested services, using the following SPARQL query:

Listing 6.13: Second SPARQL query for the VA & ES

```
1
2
3  prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
4  prefix rdfs:<http://www.w3.org/2000/01/rdf-schema#>
5  prefix xsd:<http://www.w3.org/2001/XMLSchema#>
6  prefix Accident:<http://www.accident.org/ontologies/Accident.owl#>
7  prefix Temporal_final:<http://www.accident.org/ontologies/Temporal_final.owl#>
```

```
prefix owl:<http://www.w3.org/2002/07/owl#>

 SELECT  ?hospital ?position

 WHERE {
       ?hospital   rdf:type Accident:SurgeryReadyCenter.
       ?hospital   Accident:hasPoint  ?position

                                                  }
```

6.3 Validation

In this section, in order to prove the validity of our solution we will compare the result of the use-case VA & ES services after and before performing the described processes. The RDF description provided in appendix A.1 illustrates the RDF definition of the VA & ES ontology model. In this model some classes and subclass are defined as simple. The following code shows the VA & ES temporal ontology model:

Listing 6.14: VA & ES temporal ontology

```

<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY Accident "http://www.accident.org/ontologies/Accident.owl#" >
 <!ENTITY Temporal_final "http://www.accident.org/ontologies/Temporal_final.owl#" >
]>

<rdf:RDF xmlns="http://www.accident.org/ontologies/Temporal_final.owl#"
     xml:base="http://www.accident.org/ontologies/Temporal_final.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <owl:Ontology rdf:about="http://www.accident.org/ontologies/Temporal_final.owl">
    <owl:imports rdf:resource="http://www.accident.com/ontologies/Accident.owl"/>
    </owl:Ontology>
    <!-- http://www.accident.org/ontologies/Temporal_final.owl#AKH -->

    <owl:NamedIndividual rdf:about="&Temporal_final;AKH">
       <Accident:hasPoint> 344.13 -251.224</Accident:hasPoint>
       <rdf:type rdf:resource="&Accident;Hospital"/>
       <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
       <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
    </owl:NamedIndividual>



    <!-- http://www.accident.org/ontologies/Temporal_final.owl#Lainz -->

    <owl:NamedIndividual rdf:about="&Temporal_final;Lainz">
        <rdf:type rdf:resource="&Accident;Hospital"/>
        <Accident:hasPoint>256.23 -851.204</Accident:hasPoint>
        <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
        <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
    </owl:NamedIndividual>

   <!-- http://www.accident.org/ontologies/Temporal_final.owl#St.Anna -->

    <owl:NamedIndividual rdf:about="&Temporal_final;St.Anna">
        <Accident:hasPoint> 76.103 -180.311 </Accident:hasPoint>
```

```
        <rdf:type rdf:resource="&Accident;ChildrenClinic"/>
        <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
    </owl:NamedIndividual>

 <!-- http://www.accident.org/ontologies/Temporal_final.owl#MariahilferSt_Accident1 -->
    <owl:NamedIndividual rdf:about="&Temporal_final;MariahilferSt_Accident1">
        <Accident:hasPoint> 46.183 -123.816 </Accident:hasPoint>
        <rdf:type rdf:resource="&Accident;AccidentWithInjury"/>
        <Accident:hasInjury rdf:resource="&Accident;canNotMove"/>
        <Accident:hasInjury rdf:resource="&Accident;bleeding"/>
        <Accident:hasInjury rdf:resource="&Accident;skullfracture"/>
         <Accident:hasAccidentVehicelType rdf:resource="&Accident;carToCar"/>
      </owl:NamedIndividual>

<!-- http://www.accident.org/ontologies/Temporal_final.owl#Karlsplatz_Accident2 -->
        <owl:NamedIndividual rdf:about="&Temporal_final;Karlsplatz_Accident2">
        <Accident:hasPoint> 47.25  -43.16 </Accident:hasPoint>
        <rdf:type rdf:resource="&Accident;AccidentWitouthInjury"/>
        <Accident:hasCarDamage rdf:resource="&Accident;carBodyDamage"/>
        <Accident:hasCarDamage rdf:resource="&Accident;carfire"/>
        <Accident:personhascomplaint rdf:resource="&Accident;indict"/>
        <Accident:hasVehicel rdf:resource="&Accident;carToTramway"/>

    </owl:NamedIndividual>
.
.
.
</rdf:RDF>
```

6.3.1 Rule Result

After running the java class using Jena API, several rules will be added to our temporal ontology model. As an example of the accident event that we defined above, each accident event requires emergency services (e.g. police services). In the case of ”MariahiferSt_Accident1” the following rule applies: If the person has a skull fracture, is bleeding and cannot move, then he needs surgery. This rule provided an RDF triple which describes the needed service for the injured person. The following code shows the added RDF triple ”Accident:requestService red:resource= ”http://www.accident.org/ontologyies”Accident:_SurgeryReadyCenter” of the VA & ES temporal ontology for the ”MariahiferSt_Accident1” event.

Listing 6.15: Applying accident rule into temporal ontology

```

<rdf:RDF
.
.
<rdf:Description rdf:about="http://www.accident.org/ontologies/Temporal_final.owl#MariahilferSt_Accident1">
    <Accident:requestService rdf:resource="http://www.accident.org/ontologies/Accident.owl#_SurgeryReadyCenter"/>
    <Accident:hasAccidentVehicelType rdf:resource="http://www.accident.org/ontologies/Accident.owl#carToCar"/>
    <Accident:hasInjury rdf:resource="http://www.accident.org/ontologies/Accident.owl#skullfracture"/>
    <Accident:hasInjury rdf:resource="http://www.accident.org/ontologies/Accident.owl#bleeding"/>
    <Accident:hasInjury rdf:resource="http://www.accident.org/ontologies/Accident.owl#canNotMove"/>
    <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentWithInjury"/>
    <Accident:hasPoint> 46.183 -123.816 </Accident:hasPoint>
.
.
.
</rdf:Description>
.
.
```

```
19 </rdf:RDF>
```

The result of first query defines which services the accident event needs. We defined those services with use Jena rules in the last part.

6.3.2 Query

First Query Result

The result of the first query defines which services are required by the accident event. We have already defined those services with the use of rules above. To return to our example of the use-case AV & ES, with use the Semantic rule provided a triple for ”MariahilferStAccident1” determining that this accident needs surgery service. The first accident query ”Temporal_final: MariahilferSt_Accident1 Accident:requestServices ?service” of ”MariahilferStAccident1” defines the service ”_SurgeryReadyCenter”, which is defined by our Jena rule.

```
-------------------------------------------------------------------------
| service                                                               |
=========================================================================
| <http://www.accident.org/ontologies/Accident.owl#_SurgeryReadyCenter> |
-------------------------------------------------------------------------
```

Figure 6.2: Query point and its nearest location point

Second Query Result

The second query defines service centers, including their geographical position, which are ready to provide the required services. We defined in our ontology model which hospital has surgery ready center (e.g. if hospital has surgery team service and room services then hospital is ready). The following result defines the hospitals AKH and Lainz as ready.

```
-------------------------------------------------------------------------------
| hospital                                                    | position      |
===============================================================================
| <http://www.accident.org/ontologies/Temporal_final.owl#AKH>   | " 344.13 -251.224" |
| <http://www.accident.org/ontologies/Temporal_final.owl#Lainz> | "256.23 -851.204"  |
-------------------------------------------------------------------------------
```

Figure 6.3: Query point and its nearest location point

6.3.3 Distance the nearest location

To find the hospital nearest to user position, we use the java class ”PointCalculator” to calculate the distance between two points. In the section above, the second query defined a list of hospitals which are ready. The result to the query of the use-case VA & ES are hospitals ”AKH” with position: 344.13, -251.224 and ”Lainz” with position: 256.23, -851.204. To calculate which hospital is nearest to the accident’s position: 46.183, -123, we use Pythagoras’ Theorem. This calculates the distance between accident position and Lainz at: 757.1083441311421 and the distance between accident and AKH is at 324.0450790754274, showing that the hospital nearest to accident position is AKH.

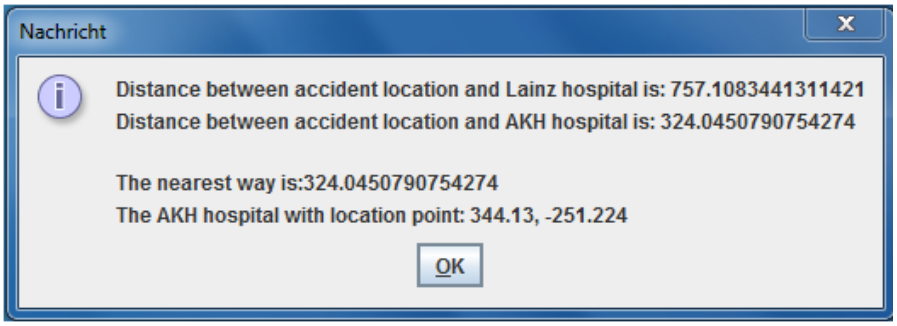

Figure 6.4: Distance the nearest Location

CHAPTER 7

Conclusions

7.1 Conclusions

In this thesis, we have presented an approach to establishing a relation between the Semantic Web and GeoRSS, based on three main requirements for using geographical location information from various GeoRSS feeds:

- It provides all geographical information from different GeoRSS feeds in machine-processable GeoRSS reader
- It embeds geographical information from various sources in a unique processable structure
- It provides this information for the intelligent search of geographical location information (e.g., related information about finding a hospital and its services), making it easier to use for geographical applications.

In order to propose a comprehensive solution to these requirements in this thesis, we analyzed relevant state-of-the-art solutions. Here we shortly identified and discussed several popular approaches and solutions, which can be used in different phases of our approach, using SW & GeoRSS, to create a comprehensive geographical data RDF dataset.

To provide geographical information from various GeoRSS feeds in machine- processable formats, the location information would be parsed from different GeoRSS feeds to a machine-processable format (i.e. converting the dataset from XML format to XML/RDF format) In this part the information for the use-cases would be parsed from different feeds on the web in order to convert their format form XML into RDF.

In order to embed the geographical data from various feeds in a unique processable structure, we have created a temporal ontology model. As a basis for our work, we briefly described

and analyzed several currently available geo-ontologies and their main features. Furthermore, we developed a temporal ontology for the purpose of embedding geo-data from feeds in order to provide an intelligent search algorithm for finding information about geographical locations from various GeoRSS feeds. This component embeds the services/events connected to specific locations, parsed from different feeds, into a temporal ontology to create comprehensive RDF instances.

To provide a search engine for this information with the use of Semantic Web technologies, we first outlined the advantages of available approaches to location search. We then proposed a temporal ontology as a geographical ontology and the individual components of our approach to create a search engine for geographical location information.

In order to demonstrate the approach using SW & GeoRSS for searching geographical information in the use-cases discussed in this thesis, we used the example of the use-case VA & ES. We demonstrated in detail the process of parsing and embedding data from different GeoRSS feeds (e.g., various hospitals, police stations, and fire brigade stations) and completely implemented the VA & ES use-case.

Bibliography

[Andrei] Andrei M., Berre ., A.J., Costa L., Duchesne P., Fitzner D., Grcar M., Hoffmann J., Klien E., Langlois J., Limyr A., Maué P., Schade S., Steinmetz N., Tertre F., Vasiliu L., Zaharia R.& Zastavni N.:. Swing: An integrated environment for geospatial semantic web services, (2008). Published in: Proceedings of 5th European Semantic Web Conference (ESWC 2008),.

[Ashraful] Ashraful A., Ganesh S., Latifur K. & Bhavani T.:. Dagis: A geospatial semantic web services discovery and selection framework, (2007). Published in:Proceeding GeoS'07(2007) Proceedings of the 2nd international conference on GeoSpatial semantics.

[Baglioni] Baglioni M., Masserotti M. V., Renso C. & Spinsanti L.: . Building geospatial ontologies from geographical databases, (2010). Published in: GeoS'07 Proceedings of the 2nd international conference on GeoSpatial semantics.

[Bar] Bar-Haim R., Dagan I., Greental I. & Shnarch E.:. Semantic inference at the lexical-syntactic level, (2007). Published in: Proceeding AAAI'07 Proceedings of the 22nd national conference on Artificial intelligence.

[Brewer] Brewer J.:. How people with disabilities use the web, (2005). www.w3.org/WAI/intro/people-use-web/Overview. Last visited: 09.2011.

[Budak] Budak A., Amit S., Cartic R., E Lynn U., Molly A. & MeiPo K.:. Geospatial ontology development and semantic analytics and sweto: Large-scale semantic web test-bed, (2006). Transactions in GIS (2006).

[Buhalis] Buhalis D. , Eichhorn V., Michopoulou E. & Miller G.: . Accessibility market and stakeholder analysis., (2005). Publisher: OSSATE.

[Chaves] Chaves M. S. , Silva M. J. & Martins B.:. A geographic knowledge base for semanticweb applications, (2005). In Proceedings of SBBD-05, the 20th Brazilian Symposium on Databases.

[Chen] Chen J., He B. & Wang W.:. Gxquery: A gml spatial data query language, (2010). Publication type: IEEE International Conference on 10.1109/ICIME.2010.5478020 Page(s): 271 – 275.

[Cold] Cold S. J.:. Using really simple syndication (rss) to enhance student research, (2006). Published in: Newsletter ACM SIGITE Newsletter Homepage archive Volume 3 Issue 1, January 2006.

[Corby] Corby O. & FaronZucker C.:. Implementation of sparql query language based on graph homomorphism, (2007). Published in: Proceeding ICCS '07 Proceedings of the 15th international conference on Conceptual Structures: Knowledge Architectures for Smart Applications.

[Distance] Available: http://en.wikipedia.org/wiki/Distance Last visited: 09.2011.

[Egenhofer] Egenhofer M. J.:. Toward the semantic geospatial web, (2002). Published in: bProceeding GIS '02 Proceedings of the 10th ACM international symposium on Advances in geographic information system.

[Fink] Adaptable and adaptive information provision for all users, including disabled and elderly people, (1998). In: The New Review of Hypermedia and Multimedia, Vol. 4 (1998) , p. 163-188.

[Fonseca] Fonseca F. T. & Egenhofer M.J.: . Ontology-driven geographic information system, (1999). Published in: Proceeding GIS '99 Proceedings of the 7th ACM international symposium on Advances in geographic information systems.

[GPS] http://en.wikipedia.org/wiki/GPS_for_the_visually_impaired Last visited:09.2011.

[GeoRSS] http://www.georss.org/Encodings Last visited:09.2011.

[Gill] Gill K. E.: . Blogging, rss and the information landscape: A look at online news, (2005). Workshop on the Weblogging Ecosystem (2005).

[Gravano] Gravano L., Vasileios H. & Richard L.:. Categorizing web queries according to geographical locality, (2003). Published in:ProceedingnCIKM '03 Proceedings of the twelfth international conference on Information and knowledge management New York, NY, USA ©200.

[Hess] Hess N. G., Cirano I. & Silvana C.:. Towards a geographic ontology reference model for matching purposes, (2007). Publication type: Conference paper. Brazil: November 2528, 2007.

[Hogan] Hogan P. & Kuehnel F.:. Nasa world wind, open source 4d geospatial visualization platform: .net& java, (2006). EOS Transactions of the American Geophysical Union, Fall Meeting, San Francisco, Abstract ED23C02.

[Horrocks2003] Horrocks I., Patel-Schneider P. F. & Harmelen F. V.: . The making of a web ontology language, (2003). Journal of Web Semantics.

[Horrocks2004] Horrocks I., Patel-Schneider P. F. Boley H., Tabet, S. Grosof B. & Dean M.:. Swrl: A semantic web rule language combining owl and ruleml, (2004). W3C Member Submission, 21 May 2004.

[Infomesh] http://infomesh.net/2001/swintro/ Last visited: 09.2011.

[Jarrar] Jarrar M. & Dikaiakos M. D.: . Mashql: A query-by-diagram topping sparql towards semantic data mashups. - university of cyprus, (2008). Proceeding ONISW '08 Proceeding of the 2nd international workshop on Ontologies and information systems for the semantic web.

[Jie] Jie L. & Wendy G. W.:. Rss made easy: A basic guide for librarians, (2007). Journal title: MEDICAL REFERENCE SERVICES QUARTERLY, Publisher : HAWORTH PRESS INC.

[Jones] Jones C. B, . Purves R., Ruas A., Sanderson M., Sester M., Van Kreveld M. & Weibel R.:. Spatial information retrieval and geogaphical ontologies an overview of the sprit project, (2002). In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.

[Kauppinen] Kauppinen T., Väätäinen J. & Hyvönen E. :. Creating and using geospatial ontology time series in a semantic cultural heritage portal, (2008). Published in: Proceeding ESWC'08 Proceedings of the 5th European semantic web conference on The semantic web: research and applications.

[Kolas] Kolas D. & Self T.:. Spatially-augmented knowledgebase, (2007). Proceeding ISWC'07/ASWC'07 Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference.

[Lake] Lake R. & Farley J.:. Infrastructure for the geospatial web, (2007). In: The Geospatial Web - How Geobrowsers, Social Software and the Web 2.0 are Shaping the Network Society London: Springer (2007) , p. 15-26.

[Lesage] Lesage N.:. Open geospatial consortium inc, (2007). Category: OGC™ Discussion Paper.

[Lieberman] Lieberman M. D., Samet H. & Sankaranarayanan J.:. Steward: Architecture of a spatio-textual search engine, (2007). Published in: Proceeding GIS '07 Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems.

[Liping] Liping D.:. Geobrain-a web services based geospatial knowledge building system,the implementation of geospatial web services at geobrain, (2004). Proceedings of NASA Earth Science Technology Conference 2004. June 22-24, 2004.

[Matthew2007] Matthew P., Amit S., Ismailcem A. & Farshad H.: . Geospatial and temporal semantic analytics, (2007). Submitted as book chapter to: Encyclopedia of Geoinformatics.

[Matthew2011] Matthew P., Prateek J. & Amit S.:. Sparqlst: Extending sparql to support spatiotemporal queries, (2011). Submitted as book chapter to: Geospatial Semantics and the Semantic Web. Publisher Springer Publishers Inc.

[Mikulik] Mikulik J.:. Identification of accident location by use of gps and possibilities of its application, (2009). Publication type: Conference pape, Available: http://www.internationaltransportforum.org/irtad/pdf/seoul/3Mikulik.pdf, Last visited: 09.2011.

[Miron2007] Miron A.D., Gensel J. & VillanovaOliver M. & Martin H.:. Semantic analysis for the geospatial web – application to owl-dl ontologies, towards the geo-spatial querying of the semantic web with ontoast, (2007). Publication in: Proceeding W2GIS'07 Proceedings of the 7th international conference on Web and wireless geographical information systems.

[Miron2008] Miron A.D., Gensel J. & VillanovaOliver M.:. Semantic analysis for the geospatial web – application to owl-dl ontologies, towards the geo-spatial querying of the semantic web with ontoast, (2008). Publication type: Conference paper.

[Nilsson] Nilsson M.:. The semantic web: How rdf will change learning technology standards, (2001). Center for User-Oriented IT-design, Royal Institute of Technology, Stockholm September 27, 2001.

[Nordin] Nordin M. J. & Ali A. M.:. Navigation and localization for visually impaired people using weighted topological map, (2009). Journal of Computer Science.

[Noy] Noy N. F.:. Semantic integration: A survey of ontologybased approaches, (2004). Published in: Newsletter ACM SIGMOD Record Homepage archive Volume 33 Issue 4, December 2004.

[Purves] Purves R., Syed A. K., Yang B. & Weibel R.:. A cartographic visualisation interface for spatial information retrieva, (2005). conference presentation - International Cartographic Conference, ICC 2005, July 2005.

[Ressler] Ressler J., Dean M. & Kolas D.: . Geospatial ontology trade study, (2010). Published in: Proceeding of the 2010 conference on Ontologies and Semantic Technologies for Intelligence.

[Richardson] Richardson W.:. The abcs of rss, (2005). Source: Technology & Learning;May2005, Vol. 25 Issue 10, p20. Source type: Trade Publication.

[Roman] Roman D. , Klien E. & Skogan D.:. Swing– a semantic web services framework for the geospatial domain (position paper), (2006). In proceedings of the workshop TTerra Cognita 2006 - Directions to the Geospatial Semantic Webïn conjunction with ISWC 2006, Athens, Georgia, USA.

[Sayar] Sayar A. , Aydin G. , Pierce M. & Fox G.:. Integrating ajax approach into gis visualization web services, (2006). In Proceedings of IEEE International Conference on Internet and Web Applications and Services ICIW'06 February 23-25, 2006.

[Schwitter] Schwitter R. & Tilbrook M.: . Controlled natural language meets the semantic web,, (2004). Proceedings of the Australasian Language Technology Workshop 2004, Macquarie University, 8th December 2004, pp. 55-62, 2004.

[Semantic] http://en.wikipedia.org/wiki/Semantic_Web Last visited:09.2011.

[Shelton] Shelton C. A.:. Nasa world wind open source project. Publication type: Dockument.

[Tanasescu] Tanasescu V., Gugliotta A., Domingue J., Villarías L., Davies R., Rowlatt M., Richardson M. & Stincic S.: . Geospatial data integration with semantic web services: the emerges approach, (2007). Submitted as book chapter to: The Geospatial Web.

[Travis] Travis M. S. & Lakshmanan V.:. Utilizing google earth as a gis platform for weather applications, (2006). In Proceedings of the 22nd International Conference on Interactive Information Processing Systems for Meteorology, Oceanography, and Hydrology.

[Turner] Turner A.:. O'reilly short cuts, introduction to neogeography, (2006). publisher: Addison-Wesley.

[Weber] Weber G., Völkel T.& Kühn R. : . Mobility impaired pedestrians are not cars: Requirements for the annotation of geographical data, (2008). Publication type: Conference paper.

[Wiki] http://en.wikipedia.org/wiki/GeoRSS Last visited:09.2011.

[Williams] Williams K. W.:. Gps design considerations: Displaying nearest airport, (1998). Publisher: Washington, D.C. : U.S. Federal Aviation Administration, Office of Aviation Medicine, [1998].

[Yokoji] Yokoji S., Takahashi K. & Miura N.:. Kokono search: A location based search engine, (2001). Publication type: Conference paper.

[Zhang] Zhang J. & Shi H.:. Geospatial visualization using google maps: A case study on conference presenters, (2007). Conference: International Multi-Symposium of Computer and Computational Sciences IMSCCS , pp. 472476, 2007.

[Zhizhin] Zhizhin M., Kihn E., Lyutsarev V., Berezin A., Poyda A., Mishin D., Medvedev D. & Voitsekhovsky D.:.

[Zhub] Zhub F. Turner M., Kotsiopoulos I., Bennett K., Russell M., Budgen D., Brereton P., Keane J., Layzell P., Keane J., Layzell P., Rigby M. & Xu J.:. Dynamic data integration using web services, (2004). Published in: Proceeding ICWS '04 Proceedings of the IEEE International Conference on Web Services.

APPENDIX A

Appendix A

A.1 VA & ES Main Ontology Model

Listing A.1: VA & ES Main Ontology Model

```
<?xml version="1.0"?>


<!DOCTYPE rdf:RDF [
    <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
    <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
    <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
    <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
    <!ENTITY Accident "http://www.accident.com/ontologies/Accident.owl#" >
]>


<rdf:RDF xmlns="http://www.accident.org/ontologies/Accident.owl#"
     xml:base="http://www.accident.org/ontologies/Accident.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:Accident="http://www.accident.com/ontologies/Accident.owl#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <owl:Ontology rdf:about="http://www.accident.com/ontologies/Accident.owl"/>



    <!--
    ///////////////////////////////////////////////////////////////////////////////////////
    //
    // Object Properties
    //
    ///////////////////////////////////////////////////////////////////////////////////////
     -->


    <!-- http://www.accident.org/ontologies/Accident.owl#hasAccidentVehicelType -->

    <owl:ObjectProperty rdf:about="http://www.accident.org/ontologies/Accident.owl#hasAccidentVehicelType">
        <rdfs:subPropertyOf rdf:resource="&owl;topObjectProperty"/>
    </owl:ObjectProperty>



    <!-- http://www.accident.org/ontologies/Accident.owl#hasInjury -->
```

```
42
43      <owl:ObjectProperty rdf:about="http://www.accident.org/ontologies/Accident.owl#hasInjury">
44          <rdfs:subPropertyOf rdf:resource="&owl;topObjectProperty"/>
45      </owl:ObjectProperty>
46
47      <!-- http://www.accident.org/ontologies/Accident.owl#hasService -->
48
49      <owl:ObjectProperty rdf:about="http://www.accident.org/ontologies/Accident.owl#hasService">
50          <rdfs:subPropertyOf rdf:resource="&owl;topObjectProperty"/>
51      </owl:ObjectProperty>
52
53      <!-- http://www.w3.org/2002/07/owl#topObjectProperty -->
54
55      <owl:ObjectProperty rdf:about="&owl;topObjectProperty"/>
56
57      <!--
58      ///////////////////////////////////////////////////////////////////////////////////
59      //
60      // Data properties
61      //
62      ///////////////////////////////////////////////////////////////////////////////////
63       -->
64
65      <!-- http://www.accident.org/ontologies/Accident.owl#hasPoint -->
66
67      <owl:DatatypeProperty rdf:about="http://www.accident.org/ontologies/Accident.owl#hasPoint">
68          <rdfs:subPropertyOf rdf:resource="&owl;topDataProperty"/>
69      </owl:DatatypeProperty>
70
71
72      <!-- http://www.w3.org/2002/07/owl#topDataProperty -->
73
74      <owl:DatatypeProperty rdf:about="&owl;topDataProperty"/>
75
76      <!--
77      ///////////////////////////////////////////////////////////////////////////////////
78      //
79      // Classes
80      //
81      ///////////////////////////////////////////////////////////////////////////////////
82       -->
83
84      <!-- http://www.accident.com/ontologies/Accident.owl#FootFracture -->
85
86      <owl:Class rdf:about="&Accident;FootFracture">
87          <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Injury"/>
88      </owl:Class>
89
90      <!-- http://www.accident.org/ontologies/Accident.owl#AccidentClinic -->
91
92      <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentClinic">
93          <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Clinic"/>
94      </owl:Class>
95
96      <!-- http://www.accident.org/ontologies/Accident.owl#AccidentConcept -->
97
98      <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentConcept">
99          <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>
100 </owl:Class>
101
102     <!-- http://www.accident.org/ontologies/Accident.owl#AccidentService -->
103
104     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentService">
105     <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
106     </owl:Class>
107
108     <!-- http://www.accident.org/ontologies/Accident.owl#AccidentType -->
109
110     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentType">
111    <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentConcept"/>
112     </owl:Class>
113
114     <!-- http://www.accident.org/ontologies/Accident.owl#AccidentVehicleType -->
```

```
115
116     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentVehicleType">
117      <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentConcept"/>
118     </owl:Class>
119
120     <!-- http://www.accident.org/ontologies/Accident.owl#AccidentWithInjury -->
121
122     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentWithInjury">
123       <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentType"/>
124     </owl:Class>
125
126     <!-- http://www.accident.org/ontologies/Accident.owl#AccidentWithoutInjury -->
127
128     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AccidentWithoutInjury">
129         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentType"/>
130     </owl:Class>
131
132     <!-- http://www.accident.org/ontologies/Accident.owl#AirportPolice -->
133
134     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AirportPolice">
135         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Police"/>
136     </owl:Class>
137     <!-- http://www.accident.org/ontologies/Accident.owl#Authorisation -->
138
139     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Authorisation">
140         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#policeService"/>
141     </owl:Class>
142
143     <!-- http://www.accident.org/ontologies/Accident.owl#AviationUnit -->
144
145     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#AviationUnit">
146  <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#fireBrigadeServices"/>
147     </owl:Class>
148
149     <!-- http://www.accident.org/ontologies/Accident.owl#Brainbleeding -->
150
151     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Brainbleeding">
152         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Injury"/>
153     </owl:Class>
154
155     <!-- http://www.accident.org/ontologies/Accident.owl#CanNotMove -->
156
157     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#CanNotMove">
158         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Injury"/>
159     </owl:Class>
160
161     <!-- http://www.accident.org/ontologies/Accident.owl#CarToBicycle -->
162
163     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#CarToBicycle">
164     <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentVehicleType"/>
165     </owl:Class>
166
167     <!-- http://www.accident.org/ontologies/Accident.owl#CarToBus -->
168
169     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#CarToBus">
170         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentVehicleType"/>
171     </owl:Class>
172
173     <!-- http://www.accident.org/ontologies/Accident.owl#CarToCar -->
174
175     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#CarToCar">
176         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentVehicleType"/>
177     </owl:Class>
178
179     <!-- http://www.accident.org/ontologies/Accident.owl#CarToTramway -->
180
181     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#CarToTramway">
182         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentVehicleType"/>
183     </owl:Class>
184
185     <!-- http://www.accident.org/ontologies/Accident.owl#ChildrenClinic -->
186
187     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#ChildrenClinic">
```

```
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Clinic"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#Clinic -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Clinic">
       <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#MedicalCenter"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#CretPoloce -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#CretPoloce">
       <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Police"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#CriminalPolice -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#CriminalPolice">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Police"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#DomainConcept -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>



    <!-- http://www.accident.org/ontologies/Accident.owl#EmergencyMedicalService -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#EmergencyMedicalService">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#fireBrigadeServices"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#FireBrAviationReady -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#FireBrAviationReady">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#ServiceReadyCenter"/>
    </owl:Class>


    <!-- http://www.accident.org/ontologies/Accident.owl#FireBrigade -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#FireBrigade">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>
        <owl:disjointWith rdf:resource="http://www.accident.org/ontologies/Accident.owl#MedicalCenter"/>
        <owl:disjointWith rdf:resource="http://www.accident.org/ontologies/Accident.owl#Police"/>
        <owl:disjointWith rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
        <owl:disjointWith rdf:resource="http://www.accident.org/ontologies/Accident.owl#locationPoint"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#bleeding -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#bleeding">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Injury"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#Hemorrhage -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Hemorrhage">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Injury"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#Hospital -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Hospital">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#MedicalCenter"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#Injury -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Injury">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#AccidentConcept"/>
    </owl:Class>
```

```
261
262     <!-- http://www.accident.org/ontologies/Accident.owl#InternalMedicine -->
263
264     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#InternalMedicine">
265         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
266     </owl:Class>
267
268
269
270     <!-- http://www.accident.org/ontologies/Accident.owl#LabDiagnostice -->
271
272     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#LabDiagnostice">
273         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
274     </owl:Class>
275
276     <!-- http://www.accident.org/ontologies/Accident.owl#MedicalCenter -->
277
278     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#MedicalCenter">
279         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>
280         <owl:disjointWith rdf:resource="http://www.accident.org/ontologies/Accident.owl#Police"/>
281         <owl:disjointWith rdf:resource="http://www.accident.org/ontologies/Accident.owl#locationPoint"/>
282     </owl:Class>
283
284     <!-- http://www.accident.org/ontologies/Accident.owl#OrthopaedicClinic -->
285
286     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#OrthopaedicClinic">
287         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Clinic"/>
288     </owl:Class>
289
290     <!-- http://www.accident.org/ontologies/Accident.owl#OrthopaedicService -->
291
292     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#OrthopaedicService">
293         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
294     </owl:Class>
295
296     <!-- http://www.accident.org/ontologies/Accident.owl#Police -->
297
298     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Police">
299         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>
300         <owl:disjointWith rdf:resource="http://www.accident.org/ontologies/Accident.owl#locationPoint"/>
301     </owl:Class>
302
303
304
305     <!-- http://www.accident.org/ontologies/Accident.owl#PolicePatrol -->
306
307     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#PolicePatrol">
308         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Police"/>
309     </owl:Class>
310
311     <!-- http://www.accident.org/ontologies/Accident.owl#PolicePatrolAccident -->
312
313     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#PolicePatrolAccident">
314         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#policeService"/>
315     </owl:Class>
316
317
318     <!-- http://www.accident.org/ontologies/Accident.owl#PolicePatrolExpert -->
319
320     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#PolicePatrolExpert">
321         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#policeService"/>
322     </owl:Class>
323
324     <!-- http://www.accident.org/ontologies/Accident.owl#PolicePatrolReady -->
325
326 <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#PolicePatrolReady">
327   <owl:equivalentClass>
328     <owl:Class>
329       <owl:intersectionOf rdf:parseType="Collection">
330         <owl:Restriction>
331          <owl:onProperty rdf:resource="http://www.accident.org/ontologies/Accident.owl#hasService"/>
332           <owl:someValuesFrom rdf:resource="http://www.accident.org/ontologies/Accident.owl#PolicePatrolAccident"/>
333             </owl:Restriction>
```

```
            <owl:Restriction>
           <owl:onProperty rdf:resource="http://www.accident.org/ontologies/Accident.owl#hasService"/>
          <owl:someValuesFrom rdf:resource="http://www.accident.org/ontologies/Accident.owl#PolicePatrolExpert"/>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
 <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#ServiceReadyCenter"/>
    </owl:Class>



    <!-- http://www.accident.org/ontologies/Accident.owl#PrivateClinic -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#PrivateClinic">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Clinic"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#Security -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Security">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#policeService"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#Service -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#Service">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>
        <owl:disjointWith rdf:resource="http://www.accident.org/ontologies/Accident.owl#locationPoint"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#ServiceReadyCenter -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#ServiceReadyCenter">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>
    </owl:Class>



    <!-- http://www.accident.org/ontologies/Accident.owl#SkullFracture -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#SkullFracture">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Injury"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#SurgeryReadyCenter -->

<owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#SurgeryReadyCenter">
 <owl:equivalentClass>
  <owl:Class>
    <owl:intersectionOf rdf:parseType="Collection">
      <owl:Restriction>
        <owl:onProperty rdf:resource="http://www.accident.org/ontologies/Accident.owl#hasService"/>
         <owl:someValuesFrom rdf:resource="http://www.accident.org/ontologies/Accident.owl#SurgeryRoomService"/>
          </owl:Restriction>
          <owl:Restriction>
           <owl:onProperty rdf:resource="http://www.accident.org/ontologies/Accident.owl#hasService"/>
           <owl:someValuesFrom rdf:resource="http://www.accident.org/ontologies/Accident.owl#SurgeryTeamService"/>
          </owl:Restriction>
         </owl:intersectionOf>
        </owl:Class>
       </owl:equivalentClass>
     <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#ServiceReadyCenter"/>
    </owl:Class>

    <!-- http://www.accident.org/ontologies/Accident.owl#SurgeryRoomService -->

    <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#SurgeryRoomService">
        <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
    </owl:Class>


    <!-- http://www.accident.org/ontologies/Accident.owl#SurgeryTeamService -->
```

```
407
408     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#SurgeryTeamService">
409         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#hospitalServices"/>
410     </owl:Class>
411
412     <!-- http://www.accident.org/ontologies/Accident.owl#TraficPolice -->
413
414     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#TraficPolice">
415         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Police"/>
416     </owl:Class>
417
418   <!-- http://www.accident.org/ontologies/Accident.owl#fireBrigadeServices -->
419
420     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#fireBrigadeServices">
421         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
422     </owl:Class>
423
424
425
426     <!-- http://www.accident.org/ontologies/Accident.owl#hospitalServices -->
427
428     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#hospitalServices">
429         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
430     </owl:Class>
431
432     <!-- http://www.accident.org/ontologies/Accident.owl#locationPoint -->
433
434     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#locationPoint">
435         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#DomainConcept"/>
436     </owl:Class>
437
438 <!-- http://www.accident.org/ontologies/Accident.owl#policeService -->
439
440     <owl:Class rdf:about="http://www.accident.org/ontologies/Accident.owl#policeService">
441         <rdfs:subClassOf rdf:resource="http://www.accident.org/ontologies/Accident.owl#Service"/>
442     </owl:Class>
443
444
445     <!--
446     ///////////////////////////////////////////////////////////////////////////////////////
447     //
448     // Individuals
449     //
450     ///////////////////////////////////////////////////////////////////////////////////////
451      -->
452
453
454
455
456     <!-- http://www.accident.com/ontologies/Accident.owl#footFracture -->
457
458     <owl:NamedIndividual rdf:about="&Accident;footFracture">
459         <rdf:type rdf:resource="&Accident;FootFracture"/>
460     </owl:NamedIndividual>
461
462     <!-- http://www.accident.org/ontologies/Accident.owl#_FireBrAviationReady -->
463
464     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#_FireBrAviationReady">
465         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#FireBrAviationReady"/>
466     </owl:NamedIndividual>
467
468 <!-- http://www.accident.org/ontologies/Accident.owl#_PolicePatrolReady -->
469
470     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#_PolicePatrolReady">
471         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#PolicePatrolReady"/>
472     </owl:NamedIndividual>
473
474  <!-- http://www.accident.org/ontologies/Accident.owl#_SurgeryReadyCenter -->
475
476     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#_SurgeryReadyCenter">
477         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#SurgeryReadyCenter"/>
478     </owl:NamedIndividual>
479
```

```
480
481
482     <!-- http://www.accident.org/ontologies/Accident.owl#canNotMove -->
483
484     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#canNotMove">
485         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#CanNotMove"/>
486     </owl:NamedIndividual>
487
488     <!-- http://www.accident.org/ontologies/Accident.owl#carToCar -->
489
490     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#carToCar">
491         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#CarToCar"/>
492     </owl:NamedIndividual>
493
494     <!-- http://www.accident.org/ontologies/Accident.owl#bleeding -->
495
496     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#bleeding">
497         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#bleeding"/>
498     </owl:NamedIndividual>
499
500     <!-- http://www.accident.org/ontologies/Accident.owl#hemorrhage -->
501
502     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#hemorrhage">
503         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#Hemorrhage"/>
504     </owl:NamedIndividual>
505
506     <!-- http://www.accident.org/ontologies/Accident.owl#skullfracture -->
507
508     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#skullfracture">
509         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#SkullFracture"/>
510     </owl:NamedIndividual>
511
512     <!-- http://www.accident.org/ontologies/Accident.owl#surgeryRoomService -->
513
514     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#surgeryRoomService">
515         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#SurgeryRoomService"/>
516     </owl:NamedIndividual>
517     <!-- http://www.accident.org/ontologies/Accident.owl#surgeryTeamService -->
518
519     <owl:NamedIndividual rdf:about="http://www.accident.org/ontologies/Accident.owl#surgeryTeamService">
520         <rdf:type rdf:resource="http://www.accident.org/ontologies/Accident.owl#SurgeryTeamService"/>
521     </owl:NamedIndividual>
522 </rdf:RDF>
523
524 <!-- Generated by the OWL API (version 3.0.0.1469) http://owlapi.sourceforge.net -->
```

A.2 VA & ES Temporal Ontology

Listing A.2: VA & ES Temporal Ontology

```
1  <?xml version="1.0"?>
2
3
4  <!DOCTYPE rdf:RDF [
5      <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
6      <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
7      <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
8      <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
9      <!ENTITY Accident "http://www.accident.org/ontologies/Accident.owl#" >
10         <!ENTITY Temporal_final "http://www.accident.org/ontologies/Temporal_final.owl#" >
11 ]>
12
13
14 <rdf:RDF xmlns="http://www.accident.org/ontologies/Temporal_final.owl#"
15      xml:base="http://www.accident.org/ontologies/Temporal_final.owl"
16      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
17      xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
18      xmlns:owl="http://www.w3.org/2002/07/owl#"
```

```
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
    <owl:Ontology rdf:about="http://www.accident.org/ontologies/Temporal_final.owl">
        <owl:imports rdf:resource="http://www.accident.com/ontologies/Accident.owl"/>
    </owl:Ontology>









    <!-- http://www.accident.org/ontologies/Temporal_final.owl#AKH -->

    <owl:NamedIndividual rdf:about="&Temporal_final;AKH">
        <Accident:hasPoint> 344.13 -251.224</Accident:hasPoint>
        <rdf:type rdf:resource="&Accident;Hospital"/>
        <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
        <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
    </owl:NamedIndividual>



    <!-- http://www.accident.org/ontologies/Temporal_final.owl#Lainz -->

    <owl:NamedIndividual rdf:about="&Temporal_final;Lainz">
        <rdf:type rdf:resource="&Accident;Hospital"/>
        <Accident:hasPoint>256.23 -851.204</Accident:hasPoint>
        <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
        <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
    </owl:NamedIndividual>


    <!-- http://www.accident.org/ontologies/Temporal_final.owl#St.Anna -->

    <owl:NamedIndividual rdf:about="&Temporal_final;St.Anna">
            <Accident:hasPoint> 76.103 -180.311 </Accident:hasPoint>
        <rdf:type rdf:resource="&Accident;ChildrenClinic"/>
        <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
    </owl:NamedIndividual>

          <!-- http://www.accident.org/ontologies/Temporal_final.owl#MariahilferSt_Accident1 -->
    <owl:NamedIndividual rdf:about="&Temporal_final;MariahilferSt_Accident1">
        <Accident:hasPoint> 46.183 -123.816 </Accident:hasPoint>
        <rdf:type rdf:resource="&Accident;AccidentWithInjury"/>
        <Accident:hasInjury rdf:resource="&Accident;canNotMove"/>
                <Accident:hasInjury rdf:resource="&Accident;bleeding"/>
        <Accident:hasInjury rdf:resource="&Accident;skullfracture"/>
               <Accident:hasAccidentVehicelType rdf:resource="&Accident;carToCar"/>

    </owl:NamedIndividual>

</rdf:RDF>
```

A.3 Hospital Surgery GeoRSS feed

A.3.1 AKH Hospital

Listing A.3: GeoRSS AKH Hospital

```
<?xml version="1.0"?>

 <rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://hospital.at/news.rss">
    <title>AKH Surgery ENT</title>
    <link>http://hospital.at/</link>
    <description>My example channel</description>
    <item>
     <title>News about AKH Surgery</title>
     <link>http://www.hospital.at/SurgeryENT/AKH/</link>
     <description>other things happened today</description>

   <!-- http://www.accident.org/ontologies/Temporal_final.owl#AKH -->
    <owl:NamedIndividual rdf:about="Temporal_final;AKH">
        <rdf:type rdf:resource="&Accident;Hospital"/>
        <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
        <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
    </owl:NamedIndividual>

              <georss:where>
             <georss:point>
                   <gml:pos>
                <Accident:hasPoint>344.13 -251.224</Accident:hasPoint>
              </gml:pos>
             </georss:point>
           </georss:where>

         </item>

   </channel>
 </rss>
```

A.3.2 Lainz Hospital

Listing A.4: GeoRSS Lainz Hospital

```
<?xml version="1.0"?>

 <rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://hospital.at/news.rss">
    <title>Lainz Surgery ENT</title>
    <link>http://hospital.at/</link>
    <description>My example channel</description>
    <item>
     <title>News about Lainz Surgery</title>
```

```
    <link>http://www.hospital.at/SurgeryENT/Lainz/</link>
    <description>other things happened today</description>

  <!-- http://www.accident.org/ontologies/Temporal_final.owl#Lainz -->
   <owl:NamedIndividual rdf:about="Temporal_final;Lainz">
       <rdf:type rdf:resource="&Accident;Hospital"/>
       <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
       <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
   </owl:NamedIndividual>

              <georss:where>
             <georss:point>
                  <gml:pos>
                 <Accident:hasPoint>256.23 -851.204</Accident:hasPoint>
             </gml:pos>
            </georss:point>
          </georss:where>

        </item>

  </channel>
</rss>
```

A.3.3 St.Anna Hospital

Listing A.5: GeoRSS St.Anna

```
<?xml version="1.0"?>

 <rss version="2.0"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://hospital.at/news.rss">
    <title> St.Anna Surgery ENT</title>
    <link>http://hospital.at/</link>
    <description>My example channel</description>
    <item>
     <title>News about St.Anna Surgery</title>
     <link>http://www.hospital.at/SurgeryENT/ St.Anna/</link>
     <description>other things happened today</description>

          <!-- http://www.accident.org/ontologies/Temporal_final.owl#St.Anna -->
        <owl:NamedIndividual rdf:about="&Temporal_final;St.Anna">
        <rdf:type rdf:resource="&Accident;Childrenclinic"/>
        <Accident:hasService rdf:resource="&Accident;surgeryRoomService"/>
        <Accident:hasService rdf:resource="&Accident;surgeryTeamService"/>
    </owl:NamedIndividual>

               <georss:where>
             <georss:point>
                  <gml:pos>
               <Accident:haspoint>32.401 -551.101</Accident:haspoint>
             </gml:pos>
            </georss:point>
          </georss:where>

        </item>

  </channel>
</rss>
```

A.4 Police Station GeoRSS feed

A.4.1 Police patrol Heldenplatz station

Listing A.6: GeoRSS Police patrol Heldenplatz station

```
<?xml version="1.0"?>

 <rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://www.police.at/rss">
    <title>Police Station GeoRSS feed</title>
    <link>http://www.police.at/</link>
    <description>My example channel</description>
    <item>
     <title>News about Heldenplatz  Police station</title>
     <link>http://www.police.at/Station/ Heldenplatz /</link>
     <description>other things happened today</description>


    <!-- http://www.accident.org/ontologies/Temporal_final.owl# Heldenplatz-->

    <owl:NamedIndividual rdf:about="&Temporal_final; Heldenplatz">
        <rdf:type rdf:resource="&Accident;TraficPolice"/>
        <Accident:hasService rdf:resource="&Accident;hasFreePolicepatrol"/>
        <Accident:hasService rdf:resource="&Accident;hasTrafficPolice"/>
                <Accident:hasService rdf:resource="&Accident;hashelicopterPolic"/>
    </owl:NamedIndividual>

          <georss:where>
             <georss:point>
                  <gml:pos>
                <haspoint>98.46-256.258</haspoint>
                  </gml:pos>
             </georss:point>
           </georss:where>

         </item>

   </channel>
 </rss>
```

A.4.2 Police patrol station Arndstrße

Listing A.7: GeoRSS Police patrol station Arndstrße

```
<?xml version="1.0"?>

 <rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://www.police.at/rss">
    <title>Police Station GeoRSS feed</title>
    <link>http://www.police.at/</link>
```

```
    <description>My example channel</description>
    <item>
     <title>News about Arndtstr Police station</title>
     <link>http://www.police.at/Station/Arndtstr/</link>
     <description>other things happened today</description>


    <!-- http://www.accident.org/ontologies/Temporal_final.owl#Arndtstr -->

    <owl:NamedIndividual rdf:about="&Temporal_final;Arndtstr">
        <rdf:type rdf:resource="&Accident;TraficPolice"/>
        <Accident:hasService rdf:resource="&Accident;hasFreePolicepatrol"/>
        <Accident:hasService rdf:resource="&Accident;hasTrafficPolice"/>
                <Accident:hasService rdf:resource="&Accident;hashelicopterPolic"/>
    </owl:NamedIndividual>

              <georss:where>
             <georss:point>
                   <gml:pos>
                  <haspoint>98.46-256.258</haspoint>
              </gml:pos>
             </georss:point>
           </georss:where>

         </item>
   </channel>
 </rss>
```

A.5 Firebrigade Aviation Unit GeoRSS feeds

A.5.1 Brigittenau station

Listing A.8: GeoRSS Brigittenau station

```
<?xml version="1.0"?>

 <rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://firebrigade.at/news.rss">
    <title>Firebrigade AviationUnit GeoRSS feed</title>
    <link>http://www.firebrigade.at/</link>
    <description>My example channel</description>
    <item>
     <title>News about fire brigade Brigittenau station </title>
     <link>http://www.firebrigade.at/Brigittenau/</link>
     <description>other things happened today</description>


    <!-- http://www.accident.org/ontologies/Temporal_final.owl#Brigittenau -->

    <owl:NamedIndividual rdf:about="&Temporal_final;Brigittenau">
        <rdf:type rdf:resource="&Accident;FireBrigade"/>
        <Accident:hasService rdf:resource="&Accident;hasAviationUnit"/>
        <Accident:hasService rdf:resource="&Accident;hasAviationUnitsalvation"/>
                <Accident:hasService rdf:resource="&Accident;hasAviationUnitwatertank"/>
    </owl:NamedIndividual>

          <georss:where>
             <georss:point>
                  <gml:pos>
               <haspoint>256.223 -531.204</haspoint>
              </gml:pos>
             </georss:point>
           </georss:where>

         </item>

   </channel>
 </rss>
```

A.5.2 Floridsdorf station

Listing A.9: GeoRSS Floridsdorf station

```
<?xml version="1.0"?>

 <rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://firebrigade.at/news.rss">
    <title>Firebrigade AviationUnit GeoRSS feed</title>
    <link>http://www.firebrigade.at/</link>
```

```
    <description>My example channel</description>
    <item>
     <title>News about fire brigade Floridsdorf station </title>
     <link>http://www.firebrigade.at/Floridsdorf/</link>
     <description>other things happened today</description>


    <!-- http://www.accident.org/ontologies/Temporal_final.owl#Floridsdorf -->

    <owl:NamedIndividual rdf:about="&Temporal_final;Floridsdorf">
        <rdf:type rdf:resource="&Accident;FireBrigade"/>
        <Accident:hasService rdf:resource="&Accident;hasAviationUnit"/>
        <Accident:hasService rdf:resource="&Accident;hasAviationUnitsalvation"/>
                 <Accident:hasService rdf:resource="&Accident;hasAviationUnitwatertank"/>
    </owl:NamedIndividual>

          <georss:where>
             <georss:point>
                  <gml:pos>
              <haspoint>18.30 -14.321</haspoint>
              </gml:pos>
             </georss:point>
           </georss:where>

         </item>

   </channel>
 </rss>
```

A.6 Accident GeoRSS feeds

A.6.1 Accident event MariahilferSt _Accident1

Listing A.10: GeoRSS Accident event MariahilferSt _Accident1

```
<?xml version="1.0"?>

 <rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
 xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://www.cityaccident.at/news.rss">
    <title>City Accident GeoRSS feed</title>
    <link>http://www.cityaccident.at/</link>
    <description>My example channel</description>
    <item>
     <title>News about accident on the Mariahilfer street</title>
     <link>http://www.cityaccident.at/accident/</link>
     <description>other things happened today</description>

            <!-- http://www.accident.org/ontologies/Temporal_final.owl#MariahilferSt_Accident1-->

    <owl:NamedIndividual rdf:about="&Temporal_final;MariahilferSt_Accident1">
        <rdf:type rdf:resource="&Accident;AccidentWithInjury"/>
        <Accident:hasInjury rdf:resource="&Accident;canNotMove"/>
                <Accident:hasInjury rdf:resource="&Accident;bleeding"/>
        <Accident:hasInjury rdf:resource="&Accident;skullfracture"/>
                <Accident:hasCarDamage rdf:resource="&Accident;carBodyDamage"/>
        <Accident:hasCarDamage rdf:resource="&Accident;carfire"/>
                <Accident:personhascomplaint rdf:resource="&Accident;indict"/>
                <Accident:hasVehicel rdf:resource="&Accident;carToCar"/>

    </owl:NamedIndividual>

            <georss:where>
             <georss:point>
                <gml:pos>
                     <Accident:hasPoint> 46.183 -123.816 </Accident:hasPoint>
                </gml:pos>
               </georss:point>
            </georss:where>

         </item>
   </channel>
 </rss>
```

A.6.2 Accident event karlsplatz _Accident2

Listing A.11: GeoRSS Accident event karlsplatz _Accident2

```
<?xml version="1.0"?>

 <rss version="2.0"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:georss="http://www.georss.org/georss/"
 xmlns:Temporal_final="http://www.accident.org/ontologies/Temporal_final.owl#"
 xmlns:Accident="http://www.accident.org/ontologies/Accident.owl#"
 xmlns:gml="http://www.opengis.net/gml#"
```

```
xmlns:dc="http://purl.org/dc/elements/1.1/">
 <channel rdf:about="http://www.cityaccident.at/news.rss">
   <title>City Accident GeoRSS feed</title>
   <link>http://www.cityaccident.at/</link>
   <description>My example channel</description>
   <item>
    <title>News about accident on the Mariahilfer street</title>
    <link>http://www.cityaccident.at/accident/</link>
    <description>other things happened today</description>

           <!-- http://www.accident.org/ontologies/Temporal_final.owl#Karlsplatz_Accident2 -->

   <owl:NamedIndividual rdf:about="&Temporal_final;Karlsplatz_Accident2">
       <rdf:type rdf:resource="&Accident;AccidentWitouthInjury"/>
              <Accident:hasCarDamage rdf:resource="&Accident;carBodyDamage"/>
       <Accident:hasCarDamage rdf:resource="&Accident;carfire"/>
              <Accident:personhascomplaint rdf:resource="&Accident;indict"/>
              <Accident:hasVehicel rdf:resource="&Accident;carToTramway"/>

   </owl:NamedIndividual>
            <georss:where>
             <georss:point>
                <gml:pos>
                      <Accident:hasPoint> 47.25  -43.16 </Accident:hasPoint>
                </gml:pos>
             </georss:point>
            </georss:where>

        </item>

  </channel>
</rss>
```