MULTISENSOR

Mining and Understanding of multilinguaL contenT for Intelligent Sentiment Enriched coNtext and Social Oriented inteRpretation

FP7-610411

D5.1

Basic semantic infrastructure

Dissemination level:	Public
Contractual date of delivery:	Month 8, 30 June 2014
Actual date of delivery:	Month 8, 30 June 2014
Workpackage:	WP5 < Semantic Reasoning and Decision Support>
Task:	T5.1 Knowledge modelling T5.2 Semantic representation infrastructure
Type:	Report
Approval Status:	Final Draft
Version:	1.0
Number of pages:	44
Filename:	D5.1_BasicSemanticInfrastructure_2014-06-30_v1.0.pdf

Abstract

The objective of this document is to provide the initial selection of ontologies to be used and to produce the first RDF data dump with data stored in OWLIM as a reason-able view. In this context the D5.1 presents the Basic Semantic Infrastructure, which includes the knowledge modelling and the semantic representation. The knowledge modelling is based on two upper ontologies: DOLCE and PROTON. The semantic representation infrastructure is based on the OWLIM RDF repository.

co-funded by the European Union

History

Version	Date	Reason	Revised by
0.1	10-5-2014	Initial input	Maurice Grinberg Vladimir Alexiev
0.2	23.05.2014	First version distributed to partners - basic structure	Kiril Simov Petya Osenowa
0.3	02.06.2014	Additions from partners	Anasatsia Moumtzidou Gerard Casamayor
0.4	23.06.2014	First complete version	Kiril Simov Petya Osenowa
0.5	25.06.2014	Review and comments	Babis Doulaverakis
1.0	29.06.2014	Final version	Kiril Simov Petya Osenova

Author list

Organisation	Name	Contact Information
Ontotext AD	Kiril Simov	kiril.simov@ontotext.com
Ontotext AD	Vladimir Alexiev	vladimir.alexiev@@ontotext.com
Ontotext AD	Petya Osenova	petya.osenova@ontotext.com
Ontotext AD	Maurice Grinberg	maurice.grinberg@ontotext.com
CERTH	Anastasia Moumtzidou	moumtzid@iti.gr
UPF	Gerard Casamayor	gerard.casamayor@upf.edu

Executive Summary

The Basic Semantic Infrastructure (BSI) of the MULTISENSOR project includes two parts: Knowledge Modelling (Task 5.1) and Semantic Representation (Task 5.2). The infrastructure has to meet certain requirements which reflect the business scenarios and the diversity of input sources (texts, images, videos). These requirements are identified to be as follows: Multifaceted architecture; Multimodality; Domain Adaptivity; Language Adaptivity; Metadata coverage; Processing granularity; Proper formalisation; Linked representation; Provenance**.** The Introduction discusses the BSI requirements. In order the supporting architecture to be built, the related state-of-the-art technologies are introduced in Section 2. They refer to Linked Data, SPARQL Query, RDF search, integrated Reason-able views on datasets and FactForge as a Reference Knowledge Stack to the web of data. Section 3 focuses on Knowledge Modelling. It reports on the two selected upper ontologies – DOLCE and PROTON. It also gives more details about FactForge structure and datasets. Next, the modelling of the use case data is outlined within the considered semantic framework. Then, the integration of these data into the Reference Knowledge Stack is described. Section 4 presents the semantic representation infrastructure. The focus in this part is on the OWLIM RDF repository: user interface, data access, data managemest and publishing and linked open publishing. Section 5 outlines the conclusions.

Abbreviations and Acronyms

BSI	Basic Semantic Structure
LOD	Linked Open Data
MS	Milestone
PC	Project Coordinator
QEG	Quality Evaluation Group
R&D	Research and Development
RDF	Rich Description Format
SAP	Self Assessment Plan
SM	Scientific Manager
SPARQL	SPARQL Protocol and RDF Query Language
TM	Technical Manager
WP	Work Package

Table of Contents

1 INTRODUCTION

This deliverable presents the Basic Semantic Infrastructure (BSI) of the project MULTISENSOR. It includes two topics: Knowledge Modelling (Task 5.1) and Semantic Representation (Task 5.2). The first topic is related to the selection of general ontologies and data sets from Linked Open Data in order to unify the data necessary for the tasks within the project, to provide general knowledge about the world facts and to incorporate the domain data from domain ontologies, datasets and information extracted from different media types within MULTISENSOR. The second topic is related to the functionality of the RDF repository for storing, searching and manipulating the project semantic knowledge. Also here we discuss some problems with the existing datasets and how they are processed in order to improve their consistency and usefulness.

1.1 Requirements to the Basic Semantic Infrastructure (BSI)

In this section we present the relations of BSI with the rest of project services. The BSI needs to support the NLP and Multimedia services for extraction of domain knowledge from and searching in text documents, images and videos.
The Basic Semantic Infrastructure has to be conformant to the two business scenarios (pilot use cases), set in MULTISENSOR: *International media monitoring* (Use case 1) and *SME Internationalisation support* (use case 2). The *media monitoring case* reflects two subscenarios: a journalistic media monitoring scenario and a commercial media monitoring (for more details viz. D2.1). In D2.1 also the data requirements have been defined:

- *For Use case 1*: the content of the news articles themselves, the related multimedia content (images, videos, graphs), and the meta information such as language, location, author and source of the news articles
- *For Use case 2*: financial and statistical information about the targeted markets, information about the products and main actors (producers, distributors and potential customers)

The Social media information, relevant for both use cases includes: posts and responses, as well as meta information such as the number of posts, number of favourites, followers, location, visits, language, author etc., depending on what data the social media platform makes available.
The semantic infrastructure has to take into account the following features of the processed data:

- Variety of application zones: market analysis, trend analysis, risk management, competitive management, CV matching, person profiling etc.
- Type of the data sources: print media, video portals, social media, broadcasting (TV, radio, data bases, etc.)
- Various domains: IT, Media, Pharmacy, Finance, etc.
- Relevant metadata depending on the data source type
- Relevant chunks of information: named entities, sentiment, events, etc.
- Multilinguality
- Relations among entities and events
- Ontology mapping
- Linking

- Provenance
- Handling big amounts of data

Having in mind all the above features, we should define the following criteria to the semantic framework:

- **Multifaceted architecture**. This means that it has to support the various processes – analysis, management, profiling.
- **Multimodality**. This means that the diversity of the sources has to be reflected – text, links, video, data bases, images, etc.
- **Domain Adaptivity**. This means that the structure has to be flexible with respect to the conceptual information for various domains (Finance, News, IT, etc.).
- **Language Adaptivity**. This means that the structure has to support the analyses over various languages.
- **Metadata coverage**. This means that the metadata module has to contain all the necessary information to address the various source specifics.
- **Processing granularity**. This means that the relevant chunks of information have to be processed by the NLP tools.
- **Proper formalisation**. This means that the processed data have to be mapped to a domain ontology.
- **Linked representation.** This means that the named entities and events have to be linked via URIs to fact knowledge databases, such as DBPedia, GeoNames, etc.
- **Provenance.** This means that the history of the information flow has to be kept. It also means that a good reasoner is needed to control the data flow.

The Basic Semantic Infrastructure has to be able to support the representation of all the necessary data, as well as their accessibility in order to support the services delivered by the other workpackages of the project. We aim at using resources and technology that are: in-house to the consortium and at the same time suit better to MULTISENSOR tasks.

In the rest of the deliverable we provide an overview of the technologies exploited within the establishment of the Basic Semantic Infrastructure. The core set of ontologies, Linked Open Data sets and some domain ontologies selected from the Basic Semantic Infrastructure data which to be further exploited and extended in order to support the services of the project.

2 TECHNOLOGY OVERVIEW

In this section we briefly describe the main components and technologies used in the implementation of BSI.

2.1 Linked Data

The notion of "linked data" is defined by Tim Berners-Lee (Berners-Lee, 2006), (Bizer, et al. 2009a), as RDF graphs, published on the WWW so that one can explore them across servers by following the links in the graph in a manner similar to the way the HTML web is navigated. It is considered a method for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web, using URIs and RDF.

"Linked data" are constituted by publishing and interlinking open data sources, following the principles of:

1. using URIs as names of things;
2. using HTTP URIs, so that people can look up these names;
3. providing useful information when someone looks up a URI;
4. including links to other URI, so people can discover more things.

In fact, most of the RDF datasets fulfill principles 1 and 2 by design. The new element in these principles concerns the requirement for enabling Semantic Web browsers to load HTTP descriptions of RDF resources, based on their URIs. To this end, data publishers should make sure that:

- the "physical" addresses of the published pieces of data are the same as the "logical" addresses, used as RDF identifiers (URIs);
- upon receiving an HTTP request, the server should return a set of triples that describe the resource;
- re-use identifiers and vocabularies from other datasets, which are also linked data.

Linking Open Data (LOD) (LOD 2014) is a W3C SWEO community project aiming to extend the Web by publishing open datasets as RDF and by creating RDF links between data items from different data sources. Linked Open Data provides sets of referenceable, semantically interlinked resources with defined meaning. The central dataset of the LOD is DBPedia (Bizer, et al. 2009b). Due to the many mappings between other LOD datasets and DBPedia, the latter serves as a sort of a hub in the LOD graph, assuring a certain level of connectivity. LOD is rapidly growing – as of September 2011 it contains more than 326 datasets, and more than 35 billion triples.

The technology behind LOD and the currently available LOD datasets will be used in MULTISENSOR in the following way. First, the semantic data produced via different services within the project will be interconnected with each other and also with datasets from LOD using the technologies of LOD. Also some of the LOD datasets will be exploited as a source of additional background factual knowledge to be used by the services of MULTISENSOR. We will design an URI schema for the entities discovered by the services of the project. Where possible these URIs will be connected to the conceptual information represented within BSI.

These will include classification of the entities with respect to ontologies loaded in BSI; identity resolution between MULTISENSOR entities and BSI entities. For example DBPedia location entities will be used when recognised in text.

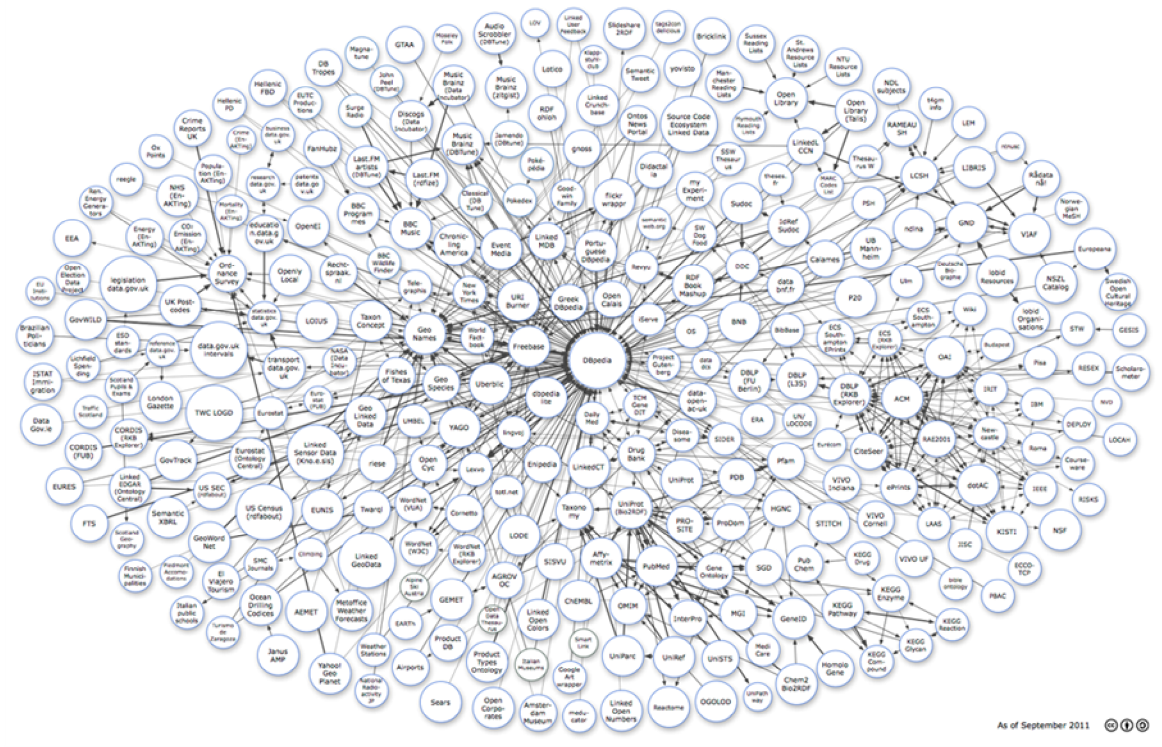


Figure 1: Map of the Datasets in Linking Open Data (LOD) Project (LOD 2014)

At the moment the LOD technology is applied to linguistic information (Linguistic Linked Open Data, LLOD). LLOD is very mature with mapping of lexical databases, while it lacks in corpora mapping. Thus, the challenge will be the mapping done over texts.

2.2 SPARQL Query and Update Languages

The first version of SPARQL (Prud'hommeaux, et al. 2008), is an SQL-like query language for RDF data, specified by the RDF Data Access Working Group of W3C. It differs from SQL in that it does not contain specific Data Definition Language (DDL) provisions, because the schemata are represented in both RDFS and OWL as standard RDF graphs, so no specific language is required to deal with them.

Version 1.1 of the SPARQL specification extends the query language with new features, (Harris, et al. 2008), but also adds data modification capabilities with the SPARQL Update specification (Schenk, et al. 2010).

The SPARQL Query Language supports four types of queries:

- SELECT queries – return tuples of results just like the SELECT queries in SQL;
- DESCRIBE queries – return an RDF graph. The resulting graph describes the resources, which match the query constraints. Usually, a description of a resource is considered an RDF-molecule (Li, et al. 2009), forming the immediate neighborhood of a URI;

- ASK queries – provide a positive or a negative answer indicating whether or not the query pattern can be satisfied;
- CONSTRUCT queries – return an RDF graph constructed by substituting the variables in the graph template and combining the triples into a single RDF graph by set union.

The SPARQL Update language allows a range of actions, related to remote management and modification of an RDF repository:

- Updates of data contained in the single RDF graph, including insertion of new data, deletion of data, and loading of new files of data into the repository;
- Creation, deletion and management of named graphs.

SPARQL language will be the primary interface between different services of MULTISENSOR services and semantic knowledge represented within the semantic infrastructure of the project. Services that discover new world facts or new conceptual information will use the updated constructs of the language in order to store the new information in the semantic repository. The services that will use the represented semantic knowledge will search the semantic repository via specially designed SPARQL templates to be filled with service specific information during the actual usage. In this way the SPARQL language will not be visible to the end user of the MultiSensor services.

2.3 RDF Search and RDFRank

A feature of OWLIM (OWLIM, an OWL repository and reasoner, will be presented below) that deserves special attention is the so-called RDF Search which provides a novel method for schema-agnostic retrieval of data from RDF datasets. The main rationale behind RDF Search is to allow searching in an RDF graph by keywords and getting usable results (in many cases standalone literals are not useful). Technically, it involves full-text indexing of the URIs in the RDF graph with respect to their "text molecules" – a text snippet, achieved by the concatenation of text from all nodes in the RDF molecule (the RDF molecules of an RDF graph are the finest components into which the graph can be decomposed without loss of information - see (Ding, et al. 2005) of the corresponding URI. The result is a list of URIs, the text molecules of which match the keywords from the query, ranked with a metric that combines standard full-text search Vector Space Model and RDFRank.

RDFRank is a rank that OWLIM can calculate and make available for each URI in the repository via the system predicate http://www.ontotext.com/owlim/hasRDFRank. OWLIM[1] calculates RDFRank for the nodes in an RDF graph similarly to the way in which Google's PageRank (Page, et al. 1999), calculates it for web pages. RDFRank is a static, contextually neutral, measure about the degree of "importance" of an RDF node in a graph, based on the importance of the nodes that are linked to it through statements containing other nodes as a subject and this one as an object.

[1] A semantic repository developed by Ontotext, http://www.ontotext.com/owlim/

Both RDF Search and RDFRank will be exploited within MultiSensor project to support joint textual based search (the search will be done on the basis of semantic annotation of documents and also on the basis of full-text search). RDFRank will be also used in the recommendation system as part of the decision support system of MULTISENSOR project. RDFRank will be exploited in order to determine relevant facts.

2.4 Reason-able Views

Using linked data for data management is considered to have a great potential. On the other hand, several challenges need to be handled in order to make this possible. Reason-able views (Kiryakov, et al. 2009), represent an approach for reasoning with and management of linked data defined at Ontotext and implemented in two systems, namely, FactForge (http://factforge.net) and LinkedLifeData (http://www.linkedlifedata.com). FactForge is based on general world knowledge LOD datasets like DBPedia dataset. LinkedLifeData is domain oriented and it is used to support biomedical research. In the project we will exploit FactForge as a source of background world knowledge.

Reason-able views is an assembly of independent datasets, which can be used as a single body of knowledge with respect to reasoning and query evaluation. The key principles can be summarized as the following instruction:

- Group selected datasets and ontologies in a compound dataset;
- Clean up, post-process and enrich the datasets if necessary. Do this conservatively, in a clearly documented and automated manner, so that (i) the operation can easily be performed each time a new version of one of the datasets is published and (ii) the users can easily understand the intervention made to the original dataset;
- Load the compound dataset in a single semantic repository and perform inference with respect to tractable OWL dialects;
- Define a set of sample queries against the compound dataset. These determine the "level of service" or the "scope of consistency" contract offered by the reason-able view. Each reason-able view is aiming at lowering the cost and the risks of using specific linked data datasets for specific purposes. The design objectives behind each reason-able view are as follows:
    - Make reasoning and query evaluation feasible;
    - Lower the cost of entry through interactive user interfaces and retrieval methods such as URI auto-completion and RDF search (a search modality where RDF molecules are retrieved and ranked by relevance to a full-text style query, represented as a set of keywords);
    - Guarantee a basic level of consistency – the sample queries guarantee the consistency of the data in the same way regression tests guarantee the quality of the software;
    - Guarantee availability – in the same way web search engines are usually more reliable than most of the web sites; they also do caching;

  - Easier exploration and querying of unseen data – sample queries provide re-usable extraction patterns, which reduce the time for getting to know the datasets and their interconnections.

The reason-able view is important for MULTISENSOR Project as an approach to linked data, because the semantic repository of the project will be created on the basis of a reason-able view over LOD datasets. Our goal will be to support as much as possible consistency of the domain specific data. Cases of contradictory information will be also useful for the use cases of MULTISENSOR because they will be results from different sources and opinions. The implementation of reason-able view will be done within OWLIM workbench using the core repository, nested repositories and inference supported within OWLIM and its extensions to be done in the next phases of the project.

2.5 Reference Knowledge Stack as Architecture for LOD - FactForge

FactForge (http://www.factforge.net/, (Bishop, et al. 2011)) is a reason-able view to the web of linked data that includes several of the most central datasets of the LOD cloud: DBPedia, Freebase, Geonames, UMBEL, Wordnet, CIA World Factbook, Lingvoj, MusicBrainz (RDF from Zitgist). Along with the dataset specific schemata and ontologies, the following ones have been loaded in FactForge: Dublin Core, SKOS, RSS, FOAF. They were referred to or imported in the ontologies of the above-listed datasets, so, they were necessary to allow the proper interpretation of the semantics of the data. OWLIM semantic repository is used to load the data and "materialize" the facts that could be inferred from it. FactForge is probably the largest and most heterogeneous body of general factual knowledge that was ever used for logical inference. The inference was performed with respect to a ruleset, derived from the so-called OWL Horst dialect, (ter Horst 2005). The only dataset that requires modification before loading it in FactForge is DBPedia. The following two modifications take place:

- remove the YAGO module as some incorrect classifications of entities and other faults in it were causing the inference of too many faulty statements in FactForge;
- clean up the category hierarchy.

FactForge has been in development for more than a couple of years. The latest developments were motivated by its expected inclusion in different projects. Besides the ongoing updates of the content reflecting the latest versions of the included datasets, FactForge has been extended with OpenCyc and the New York Time’s datasets which are applicable for Use case 1. Apart from other effects, these datasets were added in order to further minimize the dominance of the DBPedia and Wikipedia in the instance identification vocabulary, i.e. the coverage of topics and entities and their naming.

The Reference Knowledge Stack (RKS) can be seen as an extension of FactForge, serving for the purposes of data collection, integration, access and management within MULTISENSOR. Its general idea is to provide a combination of interlinked vocabularies of different size and nature (e.g. instance- vs. schema- level) and to allow them to be used as an access mechanism for LOD and other data.

The major changes that the RKS introduces to FactForge can be summarized as follows:

- PROTON upper-level ontology (to be introduced below) is added to DBPedia, Geonames and Freebase together with its schema-level mappings;

- UMBEL is updated and extended with mappings of DBPedia's categories to it;
- There is also a mapping between PROTON and UMBEL, allowing the classifications of the DBPedia entities (which are in fact Wikipedia articles) with respect to UMBEL to be "visible" through PROTON's class hierarchy as well.
- DBPedia is loaded without its own ontologies and its categorisation hierarchy.

The reason for the latter is that these parts of DBPedia provide semantics, which (i) can cause inference problems and (ii) is not necessary because similar semantics can be inferred, based on the mappings to PROTON and UMBEL. For instance, the sub-class relationships between DBPedia ontology classes are no longer crucial, as all of these classes are already mapped to PROTON and its subsumption hierarchy provides a cleaner and more inference-friendly semantics.

Reference Knowledge Stack will be used in MULTISENSOR as an approach to incorporate of semantic knowledge from different sources potentially having different or contradictory conceptualisation. Some of the reasons to follow this approach will be made clear below in Section 3.

3 KNOWLEDGE MODELLING

In this section we discuss the upper ontologies to be used in the project, the selection of datasets from LOD, their cleaning and mapping, and the selection of some appropriate conceptual sources for Use Cases data. The full set of domain conceptual knowledge to be used in MULTISENSOR project will be presented in Deliverable 5.4.

3.1 Upper Ontology

There were several possibilities for selecting an upper ontology to be used as a basis for the development of the domain ontology. The initial list of ontologies included: DOLCE Ontology[2], SUMO Ontology[3], OpenCyc Ontology[4], Omega Ontology[5], Basic Formal Ontology[6], PROTON Ontology[7], SmartWeb Integrated Ontology[8]. For the selection of the upper ontology with respect to our purposes we considered the following criteria: (1) to be constructed on rigorous basis which reflects OntoClean (Guarino, et al. 2002) (or similar methodology) and suits our domain; (2) to be easy for representation in some of the ontological languages (OWL preferably); (3) existence of domain ontologies constructed with respect to it (in order to facilitate the links with a domain ontology); (4) Support provided to us by the authors of the upper ontology; (4) To be suitable for tasks within MULTISENSOR. After some initial evaluation of the candidate ontologies and consultations with other evaluations of upper ontologies – see (Semy, et al. 2004) and (Oberle, et al. 2006) – we have selected: PROTON Ontology as our upper ontology to provide the upper part of conceptual hierarchy for modelling general world and domain entities, and DOLCE for modelling semantic roles of the participants in the events. In order to use both ontologies we will integrate the necessary sub-hierarchy of concepts from DOLCE within the PROTON ontology.

3.1.1 DOLCE Ontology

Here we present in short the ontological choices behind DOLCE – for more details and explanations see (Masolo, et al. 2002):

Descriptive vs. Revisionary ontology

"A *descriptive ontology* aims at capturing the ontological stands that shape natural language and human cognition. It is based on the assumption that the surface structure of natural language and the so-called commonsense have ontological relevance. As a consequence, the categories refer to cognitive artifacts more or less depending on human perception, cultural imprints and social conventions. Under

[2] http://www.loa-cnr.it/DOLCE.html

[3] http://www.ontologyportal.org/

[4] http://www.cyc.com/cyc/opencyc/overview

[5] http://omega.isi.edu/

[6] http://ontology.buffalo.edu/bfo/BFO.htm

[7] http://proton.semanticweb.org/

[8] http://smartweb.dfki.de/ontology_en.html

this approach, there are no major restrictions on the postulation of ontological categories because overall philosophical or scientific paradigms are neglected. This attitude stands in contrast to the *revisionary approach*. The revisionist considers linguistic and cognitive issues at the level of secondary sources (if considered at all), and does not hesitate to paraphrase linguistic expressions (or to re-interpret cognitive phenomena) when their ontological assumptions are not defensible on scientific grounds." ([13, page 7])

Multiplicative vs. Reductionist ontology

"In designing ontologies, one has to model a considerable amount of concepts. These concepts form a wide taxonomy and are often intertwined in several ways. Since the complexity of the resulting system is quite high, there are considerable advantages in limiting the actual primitives to a small subset of the concepts. If this is possible, then many notions can be reconstructed in terms of the chosen primitives. A reductionist ontologist takes this view as a major guideline; he aims at describing a great number of ontological concepts with the smallest number of primitives. On the other hand, a multiplicative ontologist points at reaching a very expressive system without bothering about the complexity of the ontology. Indeed, the aim is to provide a reliable account of reality despite of the large number of basic concepts needed." ([13, page 8])

Possibilism vs. Actualism

"Actualism claims that only what is real exists, while possibilism admits possibilia (situations or worlds) as well." ([13, page 8])

Endurants and Perdurants distinction

"Classically, endurants (also called continuants) are characterized as entities that are 'in time', they are 'wholly' present (all their proper parts are present) at any time of their existence. On the other hand, perdurants (also called occurrents) are entities that 'happen in time', they extend in time by accumulating different 'temporal parts', so that, at any time t at which they exist, only their temporal parts at moment t are present." ([13, page 11])

Based on this distinction one can define two different approaches to ontology modelling: perdurantism and endurantism – see [14, page 7]. Perdurantism assumes that entities extend in time and in space. That means entities have both spatial and temporal parts (and, therefore, four dimensions). Endurantism treats entities as 3D objects (sometimes called endurants or continuants) that pass through time and are wholly present at each point in time.

DOLCE ontology is *descriptive*, *multiplicative* ontology which adopts *possibilism* as an approach to existence and *perdurantism* as an approach to change. DOLCE is constructed with respect to OntoClean methodology. Descriptivity reflects the primarily usage of the domain ontology – representing the end user view on the domain. The same reason is true for multiplicative nature of the ontology: the end user views over the domain are rarely non-redundant and uniform. The possibilism is important with respect to support interior designs with materials which do not actually exist at the moment, but could exist in principle. Also possibilism will be important when the trends are defined within the trend analyser – the trends in principle will describe non-existing situations, but possible ones. The perdurantism ensures the support of changes of the represented entities in time. The creation of PROTON ontology (described below) follows the same methodology like DOLCE. The main difference is that PROTON is light-weight. But it is already mapped to the ontologies of some of the most important LOD dataset. Thus, our choice is to use PROTON for the common access to LOD datasets and to extend it with necessary parts of DOLCE.

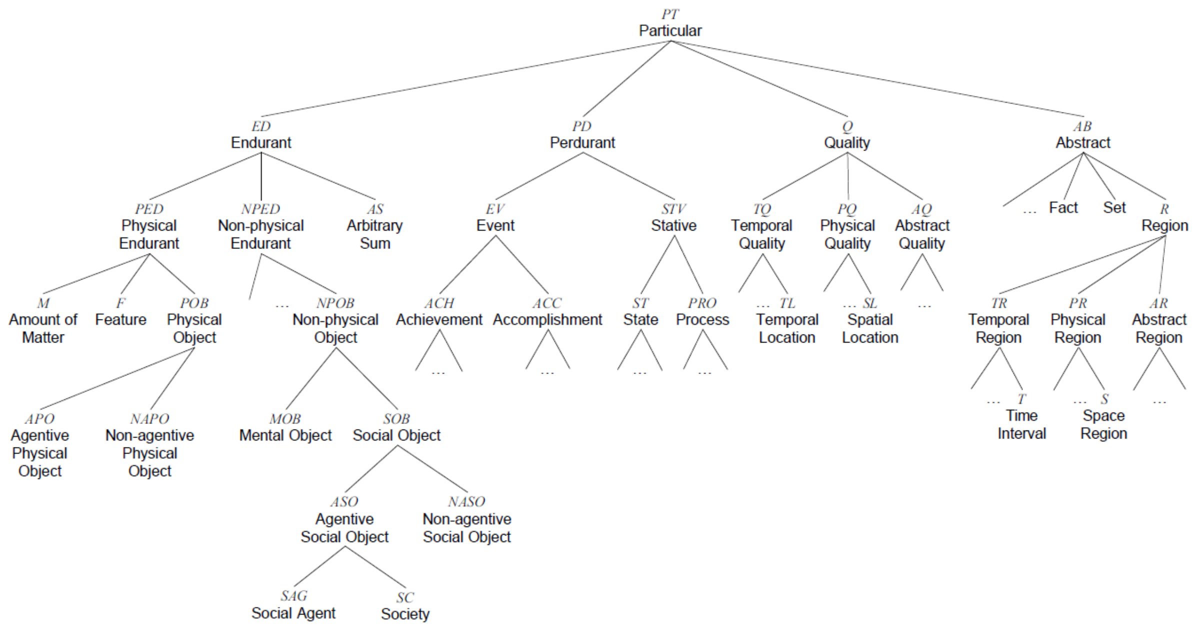


Figure 2: Taxonomy of DOLCE basic categories

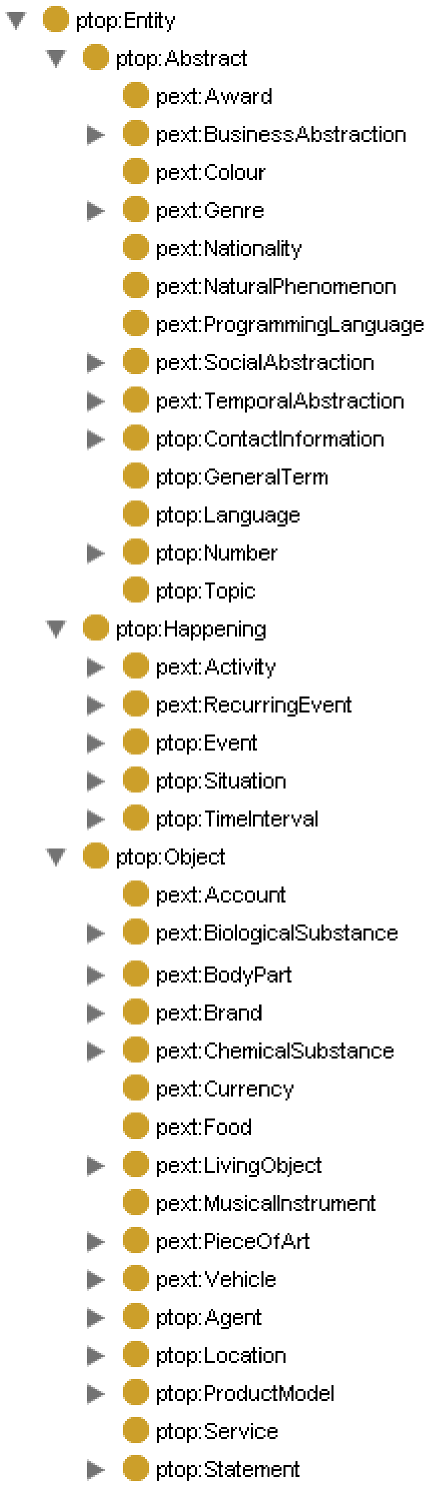
DOLCE ontology contains a sub-hierarchy of Semantic Roles. The classes from this sub-hierarchy will be used in order to generate natural language texts in several languages within the project. One critique of representation of semantic roles within the ontology is that semantic roles could be language dependent. In order to escape from potential problems in this respect we will map the semantic roles to participants in events necessary for MULTISENSOR use cases after we check that the mapping is appropriate for all languages of the project. In this way we will estimate the degree of language independency of the roles encoded in DOLCE.

3.1.2 PROTON Ontology

The PROTON (PROTo ONtology) ontology has been developed in the SEKT project[9] as a light-weight upper-level ontology, serving as a modelling basis across different tasks and domains. PROTON serves better than other popular ontologies for tasks related to automatic entity recognition and more generally Information Extraction (IE) from text, for the sake of semantic annotation (metadata generation). It also provides solid basis of data integration and RDF-based extraction, transformation and loading (ETL) of data. PROTON is constructed according the requirements of OntoClean methodology and in this respect it is compatible to

[9] http://www.ontotext.com/research/sekt

DOLCE ontology. This fact will facilitate the usage of sub-hierarchies from DOLCE as an extensions of PROTON ontology.



```
▼ ptop:Entity
  ▼ ptop:Abstract
      pext:Award
    ▶ pext:BusinessAbstraction
      pext:Colour
    ▶ pext:Genre
      pext:Nationality
      pext:NaturalPhenomenon
      pext:ProgrammingLanguage
    ▶ pext:SocialAbstraction
    ▶ pext:TemporalAbstraction
    ▶ ptop:ContactInformation
      ptop:GeneralTerm
      ptop:Language
    ▶ ptop:Number
      ptop:Topic
  ▼ ptop:Happening
    ▶ pext:Activity
    ▶ pext:RecurringEvent
    ▶ ptop:Event
    ▶ ptop:Situation
    ▶ ptop:TimeInterval
  ▼ ptop:Object
      pext:Account
    ▶ pext:BiologicalSubstance
    ▶ pext:BodyPart
    ▶ pext:Brand
    ▶ pext:ChemicalSubstance
      pext:Currency
      pext:Food
    ▶ pext:LivingObject
      pext:MusicalInstrument
    ▶ pext:PieceOfArt
    ▶ pext:Vehicle
    ▶ ptop:Agent
    ▶ ptop:Location
    ▶ ptop:ProductModel
      ptop:Service
    ▶ ptop:Statement
```

Figure 3: A view of the top part of the PROTON class hierarchy

This Section describes the third version of PROTON ontology. This new version extends the version from 2005 with new classes and properties in order to cover the conceptual knowledge encoded within the most popular Linking Open Data[10] (LOD) datasets. The modularisation of PROTON is also changed – the number of modules is reduced from four to two for convenience reasons. PROTON v3.0 defines in OWL 2 about 500 classes and 200 properties. Mappings are available between PROTON and the schemata of DBPedia, FreeBase and Geonames. PROTON ontology is developed by Ontotext and available for usage under Creative Commons license[11].

PROTON is designed as a light-weight upper-level ontology for use in Knowledge Management and Semantic Web applications, which implies the following specifics:

- PROTON is relatively un-restrictive. It specifies only a hierarchy of classes and domain and range of properties defined within it, but it does not impose any other restrictions on the meaning of the classes and properties.
- PROTON intentionally does not go deep in modeling some general notions, for instance, the conceptualisation of space and time. This is partly because proper models for these notions would require using a logical apparatus, which is beyond the limits acceptable for many of the tasks that PROTON is used for, e.g. querying and management of huge datasets and knowledge bases. Another reason to "keep it simple" is because it is very hard to craft strict and precise conceptualisations for these concepts, which are adequate for a wide range of domains and applications.

From the very beginning we set two additional requirements to PROTON, namely to allow for low cost of adoption and maintenance and for scalable reasoning. The first point requires an easy understanding of a set of classes and properties. The second requires a tractable algorithm for reasoning with the represented conceptual knowledge. Thus, the goal is to make the usage of ontologies and the related reasoning infrastructure as a replacement for the use of DBMSs feasible.

Being lightweight, PROTON matches the intuition behind the arguments coming from the Information Science community, (Sparck Jones 2004)[12] and (Shirky 2005)[13], that the Semantic Web is more likely to yield solutions to real world information management problems if it is based on partial and relatively simple models of the world, used for semantic tagging.

The PROTON ontology contains about 500 classes and 200 properties, providing coverage of the general concepts necessary for a wide range of tasks, including semantic annotation, indexing, and retrieval. The design principles can be summarized as follows:

- domain-independence;
- light-weight logical definitions;

[10] http://linkeddata.org/

[11] http://creativecommons.org/licenses/by-sa/3.0/

[12] Spark Jones, K. 2004. *What's new about the Semantic Web? Some questions.* SIGIR Forum December 2004, Volume 38 Number 2. http://www.sigir.org/forum/2004D-TOC.html

[13] Shirky, Clay. 2005. Ontology is Overrated: Categories, Links, and Tags. Clay Shirky's Writings About the Internet. Economics & Culture, Media & Community. http://www.shirky.com/writings/ontology_overrated.html

- alignment with popular metadata standards;
- good coverage of named entity types and concrete domains (i.e. modelling of concepts such as people, organisations, locations, numbers, dates, addresses); and
- good coverage of instance data in Linked Open Data Reason-able view Fact Forge[14].

The ontology is encoded in OWL 2 RL[15] sub-set (that is also eligible OWL 2DL) and split into two modules: Top and Extension. A snapshot of the top of PROTON class hierarchy is given on Figure 3.

The design principles behind PROTON are presented in greater detail in (Terziev et al. 2005)[16]. In the following sections we provide an overview of its modules and its structure.

3.2 FactForge

FactForge is a reason-able view to the web of linked data. FactForge implements the idea of Reference Knowledge Stack using PROTON Ontology (extended with elements from DOLCE Ontology) as a common upper ontology to cover all the LOD dataset specific ontologies. In this section we present some of the main steps of the development of FactForge. These steps (with some modification if necessary) will be exploited in order to incorporate the domain knowledge and facts in BSI as extensions of FactForge. The development of FactForge could be divided into six main steps:

1. Selecting the LOD datasets
2. Checking each dataset for consistency
3. Mapping the PROTON concepts to the respective LOD datasets concepts
4. Cleaning the datasets from any discrepancies between the concepts in the different datasets and PROTON
5. Loading all datasets in a joint repository
6. Loading owl:sameAs statements and checking for consistency

Here, we also present solutions for resolving discrepancies when mapping concepts from the central datasets in LOD (also loaded in FactForge) and PROTON, as well as the way of cleaning the datasets.

In some of the cases, we have to add new instances, which are introduced via inference rules.

Ultimately, FactForge provides a deeper understanding of: the Linked Open Data available on the web, some peculiarities of the datasets conceptualisation and the problems of integrating the different LOD datasets. In MULTISENSOR project FactForge will be the main source of background instance data. This instance data will be extended with domain factual

[14] http://www.ontotext.com/factforge

[15] http://www.w3.org/TR/owl2-profiles/

[16] Ivan Terziev, Atanas Kiryakov, Dimitar Manov. 2005. Base Upper-level Ontology (BULO) Guidance. Deliverable 1.8.1, SEKT project, July 2005, (http://proton.semanticweb.org/D1_8_1.pdf)

knowledge from domain terminological lexicons, ontologies and datasets. Also it will be extended with factual knowledge extracted by the services of MULTISENSOR project. In the rest of the section we present the mechanisms for mapping conceptual information, cleaning data sets. Our goal is twofold: first, to present the approach on the basis of concrete datasets because it will be exploited in future during the incorporation of domain data; second, to ensure some consistency of the datasets that are already loaded in BSi of MULTISENSOR.

3.2.1 **Mapping rules**

This section describes the methodology for creating a correspondence between two dataset conceptualisations of the real world. When constructing such correspondence, several manipulations of the datasets facts are conducted: (1) introducing new individuals; (2) deleting some individuals; (3) modifying some individuals; (4) inserting/deleting/updating relations between individuals; (5) inserting/deleting/updating characteristics of the individuals. The idea behind LOD is that such transformations are minimal. Ideally, there should be no transformations at all. We respect this recommendation, as much as possible, when constructing FactForge, except in such cases where the resulting reason-able view contradicts the conceptualisation of the PROTON ontology.

So in developing FactForge, the first aim of Ontotext was to support a full querying of the resulting repository via PROTON classes and properties. This means that classes and properties from PROTON were allowed to be part of the formulation of SPARQL queries. Only rdfs:subClassOf or rdfs:subPropertyOf statements are used in order to ensure a complete mapping coverage of the PROTON ontology to the other schemas in FactForge. Generally, the mapping statements can be arbitrary couples but in most cases they are simply rdfs:subClassOf or rdfs:subPropertyOf statements between classes or properties explicitly defined in the PROTON ontology and the ontology or the schema of a given dataset. For example[17]:

dbp:SportsTeam rdfs:subClassOf pext:Team .

foaf:homepage rdfs:subPropertyOf pext:hasWebPage .

However, due to the different conceptualisations, in some cases a more complex mapping is needed. For example, in the Geonames dataset geographical objects are classified by codes

[17] Here are the namespace declarations used in the document:

@prefix ptop:	<http://www.ontotext.com/proton/protontop#> .
@prefix pext:	<http://www.ontotext.com/proton/protonext#> .
@prefix owl:	<http://www.w3.org/2002/07/owl#> .
@prefix rdf:	<http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:	<http://www.w3.org/2000/01/rdf-schema#> .
@prefix dbp:	<http://dbpedia.org/ontology/> .
@prefix dbp-prop:	<http://dbpedia.org/property/> .
@prefix fb:	<http://rdf.freebase.com/ns/>
@prefix foaf:	<http://xmlns.com/foaf/0.1/> .

and not by an ontology hierarchy. In such cases the mapping is done by more complex statements such as:

```
[ rdf:type owl:Restriction ;
 owl:onProperty <http://www.geonames.org/ontology#featureCode> ;
 owl:hasValue <http://www.geonames.org/ontology#A.PCL> ]
        rdfs:subClassOf        pext:Country .
```

Some of these compound statements require adding new individuals. In such cases, the OWLIM inference rules are used to create the necessary additions. Here is an example:

```
//dbp-ont:PrimeMinister rdfs:subPropertyOf [ptop:hasPosition [pupp:hasTitle]].
       Id:PM
        p <rdf:type> <dbp-ont:PrimeMinister>
        ---------------------------------------
        p <ptop:hasPosition> j
        j <pext:hasTitle> <pext:PrimeMinister>
```

Here, the inference rule is necessary because the conceptualisations in the DBPedia ontology and in the PROTON ontology are different. In DBPedia, Prime Minister belongs to a class of politicians, which is a class of person, while in PROTON, Prime Minister is a title of a job position. Therefore, in DBPedia, the respective Prime Minister is an individual whereas in PROTON he is an individual who has a position with the title PrimeMinister. As the instance data about the position itself (j in the rule above) is missing in the DBPedia dataset, it has to be created so that the mapping between the two ontologies is consistent.

3.2.2 DBPedia Ontology and Dataset

The DBPedia dataset is created by extracting structured information from Wikipedia and presenting it in an RDF form (http://DBPedia.org/About). The conceptualisation of the DBPedia dataset is based on the categories that are designed and implemented in Wikipedia, i.e. the data in the info-box section of the articles. This conceptualisation is presented as an ontology. For our purposes, we have used version 3.8. It contains 359 classes, 800 object properties and 975 data types[18]. In addition, some of the classes and properties of these other ontologies are used in the definition of the DBPedia ontology. The instances in the DBPedia dataset are classified according to the conceptual information in its ontology, but also to some other ontologies - http://schema.org; http://xmlns.com/foaf/spec/. Reuse of classes and properties from one ontology in the definition of another ontology is considered to be desirable behavior, having in mind the cost of the creation of ontologies. On the other hand, the provided mapping between ontologies can cause problems as we will see below. Also using classes and properties from ontologies that are not explicitly used in the dataset is

[18] Keep in mind that the actual DBPedia ontology will not be loaded within the BSI in order not to infer facts that might contradict the conceptualisation of PROTON ontology. But the DBPedia instance data contains class and property names from the ontology. Their usage in FactForge is ensured by the mapping from PROTON to DBPedia ontology.

often useless. Thus, we have to clean the dataset from unwanted statements. In the majority of cases, the conceptualisations of the DBPedia and PROTON ontologies are compatible and the mapping between them is straightforward as discussed earlier. However, there are still some differences as illustrated in the following two examples:

Architect as a Person

In the DBPedia ontology, many roles in society, mainly performed by persons, are formalized as subclasses of the class dbp-ont:Person.

```
dbp-ont:Architect
        rdf:type owl:Class;
        rdfs:subClassOf dbp-ont:Person .
```

The definition in PROTON is:

```
pext:Architect
        rdf:type pext:Profession ;
        rdfs:comment "A profession of planning, design and oversight of the
        construction of buildings and some other  artefacts. (Wikipedia)"@en .
```

and

```
pext:Profession
        rdf:type owl:Class ;
        rdfs:subClassOf pext:SocialFunction .
```

The main difference is that in PROTON the class pext:Architect is defined as a profession and a social function, in order for someone (or something) to have this profession. This means that not only persons can perform it. While in DBPedia the definition follows the logic that all architects described in Wikipedia are, in fact, persons.

It is relatively easy to overcome such conceptual differences by an appropriate mapping between the two ontologies:

```
dbp-ont:Architect rdfs:subClassOf [ rdf:type owl:Restriction ;
        owl:onProperty pext:hasProfession ;
        owl:hasValue pext:Architect ] .
```

This statement determines that all instances of dbp-ont:Architect correspond to the instances of the PROTON ontology with the profession pext:Architect.

Sport as an Activity

Another example is the definition of Sport. In the DBPedia ontology, it is defined as:

```
dbp-ont:Sport
```

```
        rdf:type owl:Class;
        rdfs:comment "A sport is commonly defined as an organized, competitive,
        and skillful physical activity."@en;
        rdfs:subClassOf dbp-ont:Activity .
```

and in PROTON as:

```
    pext:Sport
        rdf:type owl:Class ;
        rdfs:comment "A specific type of sport game"@en ;
        rdfs:subClassOf pext:SocialAbstraction .
```

The difference is that in DBPedia, Sport is a specific activity and its characteristics such as game rules, number of participants, etc. are not defined in the class dbp-ont:Sport. In PROTON the characteristics of the sport game are defined in the class pext:Sport as a social abstraction. The actual realisation of the definition as a sport event is an instance of Activity.

Unfortunately, any mapping between the two ontologies cannot solve this conceptual difference. The following mappings:

```
    dbp-ont:Activity
        rdfs:subClassOf pext:Activity .
```

and

```
    dbp-ont:Sport
        rdfs:subClassOf pext:Sport .
```

automatically make all instances of the class dbp-ont:Sport in PROTON to be simultaneously instances of the classes ptop:Happening and ptop:Abstract, which are mutually disjoint.

In FactForge such conceptualisation differences between the two ontologies are solved by not loading the DBPedia ontology into the FactForge repository. In this way, we make use of the richness of the DBPedia instance data but impose the conceptualisation of the PROTON ontology over it.

Another reason for not loading the DBPedia ontology is that the definitions in it also contain mappings to other ontologies. We believe that including ontology statements referring to classes (properties, etc) of other ontologies is not a good practice. First, presenting the necessary conceptualisation requires importing of the other ontology. And second, this can introduce some contradictions in the ontology that uses these statements.

For example, the DBPedia ontology contains some statements from the Schema ontology (**http://schema.org**), which take care of the mapping between this ontology and the DBPedia ontology. However, because DBPedia is not an extension of the Schema ontology, it is better to store these statements separately. If they are included in the definitions of the DBPedia classes, this can lead to some contradictions as illustrated in the examples below:

```
dbp-ont:University
        rdf:type owl:Class;
        rdfs:subClassOf dbp-ont:EducationalInstitution ;
        owl:equivalentClass schema:CollegeOrUniversity .
```

and

```
dbp-ont:College a owl:Class;
        rdf:type owl:Class;
        rdfs:subClassOf dbp-ont:EducationalInstitution ;
        owl:equivalentClass schema:CollegeOrUniversity .
```

Using owl:equivalentClass makes these two classes - dbp-ont:University and dbp-ont:College - the same. Such equivalent statements are difficult to be noticed in the DBPedia ontology as it is full of, them but, it is also not very easy to use DBPedia without such statements.

The instance data also contains statements that result from inferences from the DBPedia ontology. In order to avoid all conceptualisations that follow from the DBPedia ontology we have to clean the DBPedia instance data from such inferences. Here are some examples:

Subclass - Superclass inference

In the DBPedia instance data, each instance of sport is classified as Sport but also as an Activity. Therefore, even if we do not load the DBPedia ontology in the FactForge repository, this inference is present in the instance data. Thus, the classification of the DBPedia sport instances will also be wrong in PROTON when mapping PROTON to DBPedia.

To clean this instance data statement a deletion statement of the following type has been created:

```
delete {?s a dbp-ont:SuperClass} where
        {
        ?s a dbp-ont:SubClass .
        ?s a dbp-ont:SuperCLass .
        }
```

Here is an example:

```
delete {?s a dbp-ont:Activity} where
        {
        ?s a dbp-ont:Sport .
        ?s a dbp-ont:Activity .
        }
```

In this way, if there is a statement for a subclass, we delete all the statements for the super classes. After that, we use the inference mechanisms of the repository to make the inferences that follow from the mapping to the PROTON ontology.

rdfs:domain and rdfs:range statements

In the DBPedia instance data, some statements for domain and range have properties connected to instances that do not belong to the appropriate classes. Such unclassified instances in DBPedia could be wrongly classified in PROTON, based on these domain and range statements. In order to clean such cases we use queries of the following type:

```
delete {?s dbp-ont:DBPediaProperty ?y } where
        {?s dbp-ont:DBPediaProperty ?y .
        ?y rdf:type ?c .
        filter(
                ?c = dbp-ont:Class01
                || ?c = dbp-ont:Class02
                || ... ## List of all unappropriate classes
            )
        }
```

A complete example of such a query is given in Appendix A.

Apart from the deleted statements discussed earlier, all instance data described by statements using classes that are not from the DBPedia ontology have been deleted.

In this way, the DBPedia instance data has a clean interpretation in terms of the PROTON conceptualisation. This ensures that FactForge is will provide consistent instance data for the services of the project.

3.2.3 Freebase dataset

Freebase (http://www.freebase.com/) is a community-curated database of well-known people, places, and things. In Freebase, real-world entities are represented as topics. For example, there are topics for movie stars, countries, cities, etc. The information for each topic is structured in three levels as defined in the Freebase schema. The first layer comprises several domains (76 in number). Each domain is defined by type (second layer) and each type has properties (third layer). All topics are described as belonging to one or more types via the property http://rdf.freebase.com/ns/type.object.type

The types are connected via the special relation *inclusion of type*. This relation connects more specific types with more general types. For example, the type https://www.freebase.com/base/litcentral/named_person includes the type: https://www.freebase.com/people/person. However, it is not possible to interpret this relation as superclass-to-subclass relation, because it is not strict in the sense that each instance of the subclass inherits the properties of the instance of the super class. For example, the type https://www.freebase.com/film/actor also includes the type https://www.freebase.com/people/person. But its definition is: "The Film Actor type includes people (and credited animals) who have appeared in any film ...". Therefore, in most cases, the instances of the type https://www.freebase.com/film/actor are people but there are also cases where they are not. Consequently, the interpretation of the type inclusion relation is not strict with respect to inheritance of the properties from the included type. In the example above, if the film actor is a person, then he or she inherits all properties from the type for persons. But if it is not a person, then it does not inherit any of these properties. Instead, it inherits properties from some other type(s).

These peculiarities of the Freebase schema impose some restrictions over the mapping to the PROTON ontology. Mapping so many types and properties requires more extensive work. Therefore, for our purposes, we have mapped only the types with more than 500 instances in the Freebase dataset to the PROTON concepts. Another criterion is that the mapping does not produce any misclassification of some instances. For many types the mapping is straightforward:

```
fb:location.location
            rdf:type owl:Class;
            rdfs:comment "The Location type is used for
                          any topic with a fixed location..."@en ;
            rdfs:label "Location";
      rdfs:subClassOf ptop:Location .
```

For types representing professions and other social roles, the mappings are similar to the mapping used for the DBPedia ontology:

```
fb:military-militarycommander
      rdfs:subClassOf
            [rdf:type owl:Restriction ;
             owl:onProperty pext:hasTitle ;
             owl:allValuesFrom pext:Commander].
```

Some of the types are mediators between a type and a grouping of several other types. This is mainly used to represent event information. For example, the type *Website ownership* describes an event of owning a website by an agent for some period. A website can be owned by different agents in different periods, thus it is important that these 'owning' events are represented as different instances in the dataset.

At present, we have not yet mapped the mediator types to PROTON. For this type of mapping it is necessary to use an appropriate subclass of the class ptop:Happening. For example, the type *Website ownership* can be mapped to a subclass of the class ptop:Situation, where the start and end date of the ownership are stated, the owner and the address of the website are specified, etc. As this requires huge extension of PROTON, it is not featured in the current version. In MULTISENSOR project these classes will be extensively used in supporting Natural Language Generation. The necessary classes will be mapped to new domain dependent events. In these cases also the semantic roles from DOLCE ontology will be mapped to the corresponding participants in the domain events.

In the original dataset, there are also several errors in the instance classification. For example, organisation and location are very often represented by the same instance. More specifically, the types fb:location.location and fb:organization.organization have 42763 instances in common. We believe that such cases result from the linguistic intuition of the users who created the data in question. In many cases, the same word denotes both the meaning of an institution and a location. We do not consider this a good practice for semantic representation in LOD and we think that it should be avoided. The different classes (types in Freebase) have different properties. In spite of this, the Freebase types are not strict in inheriting properties, some types are still not mutually compatible (intuitively). For example, due to this misclassification, the instance of the United State of America (https://www.freebase.com/m/09c7w0) is not only an instance of the types Country, Location but also of Food. Therefore, we believe that such knowledge has to be represented in a completely different way.

It is important to note that correcting such cases of instance classifications to many disjoint types (classes) is outside the scope of usage of FactForge in MULTISENSOR project. We will perform it only when this is required by the use cases implemented within the project. The presented approach to cleaning, loading and mapping of dataset conceptual information to the upper ontology will be used also for the incorporation of the domain specific information in MULTISENSOR.

3.3 Modelling and Integration of Use Case Data in Reference Knowledge Stack

As it was mentioned above, the final selection of domain ontologies and their usage within the project will be presented in Deliverable 5.4. Here we briefly represent the methodology to be followed in the process of modelling of use case data and its integration in the reference knowledge stack as defined above. Initial list of sources of conceptual information and ontologies within Energy domain will be presented below. In D2.1 also the most frequent domains have been identified (p. 17):

- Automotive
- IT / Telecommunication
- Finance / Insurance
- Media / Advertisement
- Pharmacy / Medicine
- Biotechnology
- Food industry

However, the frequency criterion is not the strongest one for the selection of an illustrative domain. There should exist appropriate thesauri, ontologies, etc. One requirement for the selection of the ontologies is that they have aligned lexicons. For the first experiments with domain conceptual data we have selected two ontologies that correspond to the last

requirement. The first one is EuroVoc[19] - a European multilingual thesaurus. It covers many domains of importance for European Union on detailed level of granularity. We consider EuroVoc as a middle level ontology to cover all the domains of MULTISENSOR project. EuroVoc is already available in RDF format which will facilitate its inclusion in the Reference Knowledge Stack. One domain that has been viewed as appropriate for MULTISENSOR is Energy domain. In this domain we selected REEGLE - Clean Energy Info Portal[20]. It contains conceptual and instant data in Energy domain. REEGLE is also available in RDF format.
The incorporation of these ontologies within the reference knowledge stack will complete the following steps:

1. Preparation of an RDF version of the ontology and the instance data. Formalisation in OWL language. This first step will produce a formally consistent dataset;
2. Checking of the consistency of the dataset on conceptual level. The difference from the consistency in the previous point is the definition of consistency axioms which to ensure the consistency of the model. Disjoint axioms are one example of such axioms. For example, locations are different from events. In cases of mismatches the dataset is cleaned;
3. Establishment of a mapping to already represented ontologies. At the beginning there will be two ontologies: PROTON and part of DOLCE. Later on, domain ontologies to support the use cases, like EuroVoc and REEGLE, will be added and will be mapped to existing ontologies in BSI;
4. The mapping might require the addition of inference rules for the completion of the RDF instance data to support the access via SPARQL language;
5. Loading of the dataset in the semantic infrastructure of the MULTISENSOR project.
6. Establishment of identity equivalence on instance level. This equivalence will be stated via owl:sameAs statements. For example, countries represented already in EuroVoc will be mapped to countries represented in FactForge.

The mapping procedure of step 3 will be developed in detail and will be implemented within Task 4.2: *Mapping discovery and validation*. This procedure will differ from the approach taken during the mapping of PROTON to dataset ontologies in FactForge. In FactForge manual examination of ontologies was taken. Task 4.2 will develop and implement an automatic procedure which will match ontologies on the basis of their shape, labels and instance knowledge. Similar steps will be followed for the incorporation of the semantic data extracted from text and multimedia documents. Some relaxations on the consistency will be accepted in order to allow the representation of contradictory information from different documents.

[19] http://eurovoc.europa.eu/

[20] http://www.reegle.info/

Energy Domain

There is a lot of multilingual data in the energy domain. However, the structured and linked data seems to be sparse. Below we consider the data in the following categories: available data sources and ontologies.

Data sources

1. Open Linked Data: There are some concepts and events in addition to names in DBPedia and FreeBase. The existing labels and relations can be used in the project. However, the coverage will not be sufficient. These data will be rather used as example model for incorporating more energy-related terms and concepts.
2. General Thesauri: These thesauri cover also the energy domain to different extent and for various languages.
   - 2.1 REEGLE glossary (http://www.reegle.info/glossary/1660). This glossary is in the energy domain. It displays terms with their definitions, synonyms and relations to other terms. The glossary is multilingual. It covers at least the following languages: English, Spanish, Portuguese, French, German.
   - 2.2 STW Thesaurus for Economics (http://zbw.eu/stw/versions/latest/descriptor/18231-5/about.en.html). This thesaurus includes also terms about energy in German. It provides links to the terms in DBPedia and other thesauri.
   - 2.3 General Multilingual Environmental Thesaurus
   - 2.4 (http://www.eionet.europa.eu/gemet/concept?ns=1&cp=9335). The thesaurus includes also terms about energy in all the EU languages. It provides links to EuroVoc and AgroVoc.
   - 2.5 AGROVOC (Agricultural Vocabulary) (http://aims.fao.org/aos/agrovoc/c_8394). It is available also as LOD and in a multilingual set.
   - 2.6 EuroVoc (EU's multilingual thesaurus) (http://eurovoc.europa.eu/748). It is available also as LOD and in many languages.
   - 2.7 Library of Congress Subject Headings (http://id.loc.gov/authorities/subjects/sh85146874). It is in English, but available in many formats (RDF, SKOS, XML, JSON, etc.)
   - 2.8 Crowd-sourced data
     - Enipedia (http://enipedia.tudelft.nl/wiki/Main_Page) – a specialized portal to various energy-related resources. It covers power generation (power plants, fuel type, emissions, geography, etc) and other energy topics. It is developed with Semantic Media Wiki. It is hand-curated and includes data imported from other sources. Its content can be explored with SPARQL.
     - Open Energy Information (http://en.openei.org/wiki/Main_Page) (OpenEI). It is a knowledge sharing online community dedicated to connecting people with the latest energy information and data. Access is provided to energy-related information via geographic discovery, visualisations and apps, and topic-oriented gateways. There are numerous datasets for exploration.

Ontologies

1. DEHEMS ontology (http://www.dehems.eu/). The Home Appliance Ontology is described in: http://www.dehems.eu/cms/wp-content/uploads/2011/04/D8.1-Paper3.pdf. It is based on SUMO upper ontology.
2. Ontology for Urban energy systems (https://workspace.imperial.ac.uk/urbanenergysystems/Public/2010NGInfra-VanDamKeirstead.pdf). It uses the SynCity ("Synthetic City") modelling system.
3. SPITFIRE ontology (http://spitfire-project.eu/incontextsensing/ontology.php). The ontology has been designed for sensor metadata and activities, but extended also to energy-efficiency requirements.
4. E-SAVE ontology (http://www.e-save.eu/?wpfb_dl=7). It covers the energy efficiency monitoring domain across the supply chain.

To sum up, concerning the data sources, there are many multilingual and linked data, related to energy domain which can be re-used, further structured and enriched. The main sources would be EuroVoc and REEGLE as the most advanced resources with respect to multilinguality and linking. The other ones will be used for enrichment with additional terms and relations. Concerning the available ontologies, many of them are connected to energy saving and also to popular formats like SKOS. Thus, they are also potentially useful to our project model.

4 SEMANTIC REPRESENTATION INFRASTRUCTURE

In this section we discuss two topics: the software infrastructure for the Basic semantic infrastructure and the management of datasets for loading into the RDF repository.

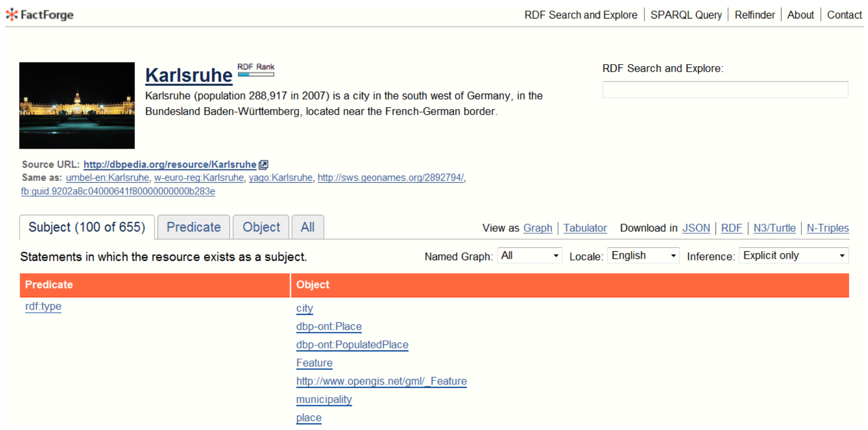
4.1 OWLIM Architecture and Functionality

The standard data model for representing data in the project is RDF. The semantics of the data is expressed in RDF Schema, (Brickley, et al. 2004), and OWL, (Bechofer, et al. 2004). The integration of data and their access is further facilitated by the fact that the datasets are interlinked to one or more of the elements of the Reference Knowledge Stack (see above). While for specific tasks within MULTISENSOR some non-RDF-based representations are also considered, the facilities for data publication and access are entirely based on the above standards.

Complying to the Semantic Web-related specifications of W3C means that there should be facilities for querying datasets using SPARQL, as well as for searching in and exploring RDF graphs. Both the user-interfaces[21] and the APIs will be based on the Forest linked-data front-end framework, which is developed in Ontotext as an open-source library. It allows the exploration of data, hosted in a semantic repository compliant with Sesame. The Forest framework allows for customizing the services provided on its basis.

For instance, the user interfaces of FactForge are based on Forest, but they still look quite different and provide different look and feel.

While the Forest framework allows the exploration of data in any Sesame-compliant repository, in FactForge it uses a repository, hosted by OWLIM, which includes advanced features such as full-text indexing and querying, optimized handling of equivalence search, full text search (FTS), geo-spatial owl:sameAs, etc.


FactForge
RDF Search and Explore | SPARQL Query | Relfinder | About | Contact
RDF Rank
RDF Search and Explore:
Subject (100 of 655) | Predicate | Object | All
View as Graph | Tabulator
Download in JSON | RDF | N3/Turtle | N-Triples
Named Graph: All
Locale: English
Inference: Explicit only

Karlsruhe

Karlsruhe (population 288,917 in 2007) is a city in the south west of Germany, in the Bundesland Baden-Württemberg, located near the French-German border.

Source URL: http://dbpedia.org/resource/Karlsruhe
Same as: umbel-en:Karlsruhe, w-euro-reg:Karlsruhe, yago:Karlsruhe, http://sws.geonames.org/2892794/, fb:guid.9202a8c04000641f80000000000b283e

Statements in which the resource exists as a subject.

Predicate	Object
rdf:type	city
	dbp-ont:Place
	dbp-ont:PopulatedPlace
	Feature
	http://www.opengis.net/gml/_Feature
	municipality
	place

Figure 4: The Resource Exploration View of FactForge

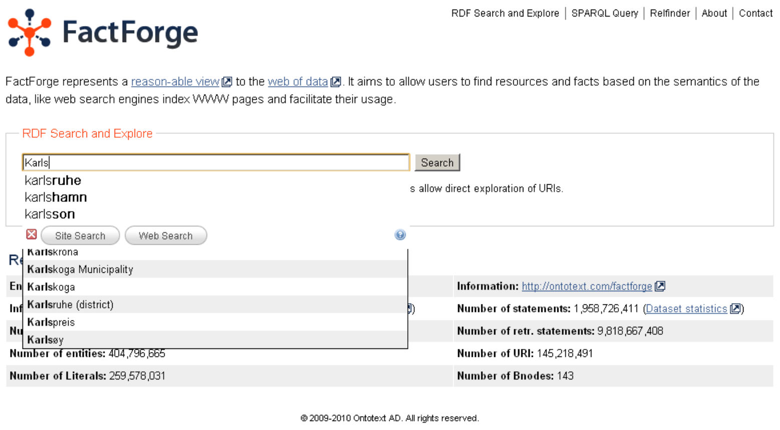
[21] Note that these user interfaces are appropriate for the developers of the client services within MULTISENSOR project. For the end users of MULTISENSOR services there will be more appropriate user interfaces in which the access to semantic repository will not be visible in terms of SPARQL queries.

4.1.1 User Interface for Exploration and Querying

In some user interface components there will be a need for access to conceptual knowledge stored in the OWLIM repository. The main access methods to be exposed are as follows:

Incremental URI auto-suggest mechanism

It allows users to easily find the entity identifiers that they are looking for. On the one hand, it allows finding URIs that they do not know or cannot fully type in. On the other, it improves the speed of exploring known URIs, because users do not need to type their full length, but can type just few characteristic symbols and select them;


FactForge
RDF Search and Explore | SPARQL Query | Relfinder | About | Contact
RDF Search and Explore
Karls
Search
Site Search
Web Search

FactForge represents a reason-able view to the web of data. It aims to allow users to find resources and facts based on the semantics of the data, like web search engines index WWW pages and facilitate their usage.

karlsruhe
karlshamn
karlsson

s allow direct exploration of URIs.

Karlskrona
Karlskoga Municipality
Karlskoga
Karlsruhe (district)
Karlspreis
Karlsøy

Information: http://ontotext.com/factforge
Number of statements: 1,958,726,411 (Dataset statistics)
Number of retr. statements: 9,818,667,408
Number of entities: 404,796,665
Number of URI: 145,218,491
Number of Literals: 259,578,031
Number of Bnodes: 143

© 2009-2010 Ontotext AD. All rights reserved.

Figure 5: Incremental URI auto-suggest mechanism

One-node-at-a-time exploration

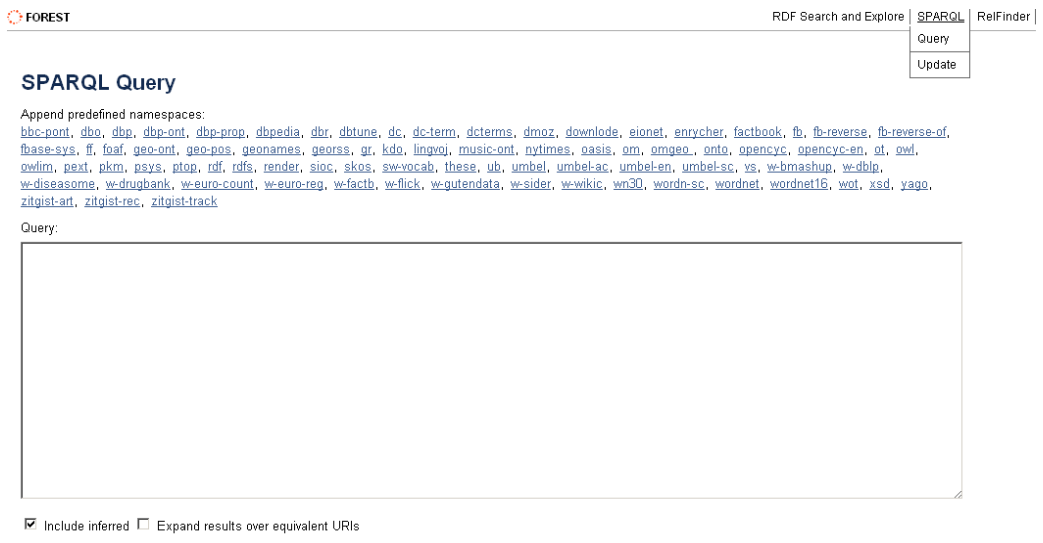
This is a standard “linked data browser” metaphor where each URI is represented by a table of two columns – predicates and objects of statements, where this URI appears as a subject. The user can navigate to other URIs by following the hyperlinks of related resources presented in this table. Forest has several features, which allow more ergonomic exploration of repositories containing large volumes of data from multiple datasets: presenting resources in the UI by their “preferred labels”, instead of their URIs (when available); filtering data by language locale and named graph; filtering data by their explicit/implicit status; presenting all URIs that are owl:sameAs - equivalents of the one currently being explored; presenting a text snippet and an image on top of the page (when available);

RDF Search, retrieving a ranked list of URIs by keywords

From a user interface point of view, Forest renders the results of the RDF Search as a list of URIs, represented by their preferred labels and text snippets, when such are available (see Figure 5);

SPARQL querying interface

To facilitate query writing, the form provides: shortcuts for the namespace prefixes used in the repository; references to sample queries and check-boxes, which allow one to determine whether inferred facts should be considered during query evaluation and whether query results should be “expanded” with respect to the standard owl:sameAs semantics (see Figure 6);



SPARQL Query

Append predefined namespaces:
bbc-pont, dbo, dbp, dbp-ont, dbp-prop, dbpedia, dbr, dbtune, dc, dc-term, dcterms, dmoz, downlode, eionet, enrycher, factbook, fb, fb-reverse, fb-reverse-of, fbase-sys, ff, foaf, geo-ont, geo-pos, geonames, georss, gr, kdo, lingvoj, music-ont, nytimes, oasis, om, omgeo , onto, opencyc, opencyc-en, ot, owl, owlim, pext, pkm, psys, ptop, rdf, rdfs, render, sioc, skos, sw-vocab, these, ub, umbel, umbel-ac, umbel-en, umbel-sc, vs, w-bmashup, w-dblp, w-diseasome, w-drugbank, w-euro-count, w-euro-reg, w-factb, w-flick, w-gutendata, w-sider, w-wikic, wn30, wordn-sc, wordnet, wordnet16, wot, xsd, yago, zitgist-art, zitgist-rec, zitgist-track

Query:

[✓] Include inferred ☐ Expand results over equivalent URIs

Figure 6: SPARQL querying interface

RelFinder

The graphical search facility called ‘RelFinder’ (Haim, et al. 2009) allows finding paths between selected nodes. This is a computationally intensive activity and the results are displayed and updated dynamically. The resulting graph can be reshaped by the user with simple click and drag operations. Entities within the emerging graph can be selected and a properties box provides links to the sources of information about the entity (see Figure 7).

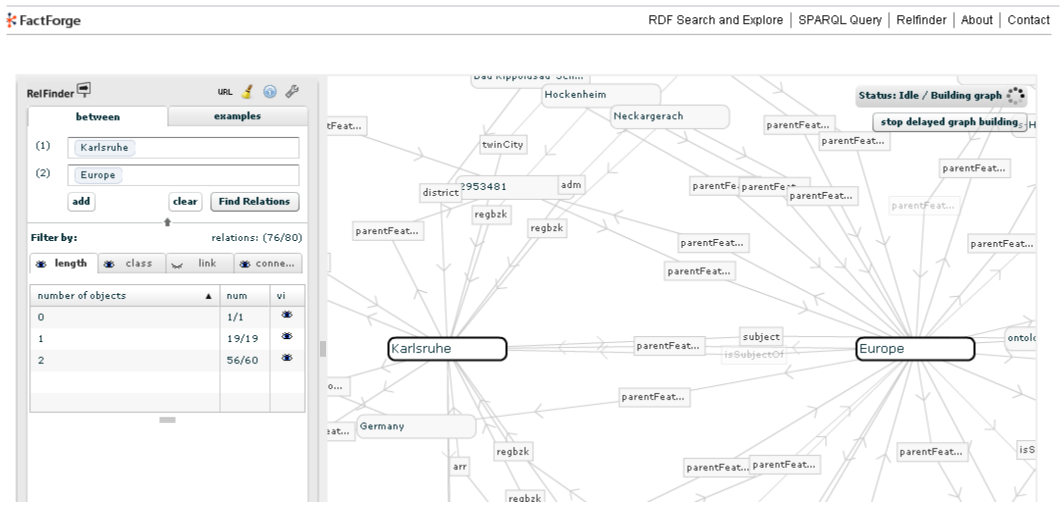


Figure 7: RelFinder

4.1.2 APIs for Data Access

Programs will be able to access the data published within the repository via SPARQL end-point, which at a technical level represents a web service conformant with the SPARQL RDF Protocol (Clark, et al. 2008). In essence, it allows programs to submit SPARQL queries for evaluation to a remote query processor (a semantic repository) and to retrieve results in a standard format.

Several comments can be made about the SPARQL end-point, providing the project data. As a start, it should be mentioned that SPARQL, as a query language, is agnostic with respect to any form of inference.

In other words, the SPARQL specification does not allow one to specify whether statements that are inferred, or can be inferred in the process of query evaluation, should be considered by the query engine during evaluation. More generally the question is whether reasoning with respect to the semantics of the data and the ontologies loaded in the repository should affect the results of the query. OWLIM allows the clients to determine their preference how to use the inference in the query evaluation with the help of a special-purpose system predicate in the FROM NAMED clause of a SPARQL query. These system predicates are documented in section 8.6 of the OWLIM's User Guide (Ontotext, 2014).

As the data being published by project, or through its publishing facilities, are meant to be entirely open and public, there is no need of any access control. Still, to ensure the smooth operation of the SPARQL end-point, limitations will be enforced on the maximum size of query results that can be loaded through the public interface and the maximum time for query evaluation. The most important rationale behind such a constraint is to ensure that the SPARQL queries that happen to require massive computational resources are not going to hamper the performance of the entire service.

4.1.3 Data Management and Publishing

The API to be used for Data Management and Publication will be SPARQL 1.1 Update (Kiryakov, et al. 2009). It was selected because:

- it is the most comprehensive standardized mechanism for management of RDF data;
- it covers the requirements for all low-level tasks that need to be handled in relation to publication and management of data in the context of MULTISENSOR.

Essentially, there are two levels of granularity at which data can be updated using the SPARQL update:

- modifications to the content of a single (named) RDF graph;
- creation and removal of entire named graphs in a repository.

As long as the metadata about the named graphs is RDF, it can be managed with the standard mechanisms for modifying RDF graph content, applied to the default graph of the dataset or to any other graph that can be used for storing such metadata. The following is a short summary of the basic operations for modifying the content of an RDF graph, provided by the SPARQL Update:

- INSERT DATA / DELETE DATA allow adding or removing specific statements that are included inline in the query;
- INSERT / DELETE allow inserting and/or deleting statements, based on query patterns; it is much more powerful and generic than the DATA variants of the operators;
- LOAD inserts all the data from a remote graph into a named graph or into the default graph of the repository;

- CLEAR clears the content of a named graph or of the default graph; the graph itself is not removed.

Ontotext had dedicated considerable efforts in enabling OWLIM to support the SPARQL Update specification and, more generally, SPARQL version 1.1 (Harris, et al. 2008). Initially the efforts were to achieve this by integrating OWLIM with Jena, which already supported SPARQL 1.1. Version 3.4 of OWLIM was the first one to include Jena integration and SPARQL 1.1 support. With version 3.5, released at the end of March 2011, all functionality, even the advanced features of OWLIM is available through the Jena APIs and through SPARQL. An intermediate version was recently provided for independent evaluation and demonstrated outstanding results in the Berlin SPARQL Benchmark (Bizer, et al. 2014). There is an ongoing work to extend the open-source Sesame framework with SPARQL 1.1 support – this will enable OWLIM to deliver even better performance with respect to SPARQL 1.1. At the current phase of the project however, the SPARQL Update functionality will not be available through the public end-points and interfaces, because it is still necessary to properly define the requirements for access control and implement the corresponding policies. Without access control, free public access to interfaces that allow data modification is likely to result in an unpredictable and unusable service. This reveals the need to carefully investigate the requirements in order to deliver the best balance between freedom of modification and stability.

4.1.4 **Linked Data Publishing**

The publishing RDF data as “linked data” requires the following:

The “physical” addresses of the published pieces of data should be the same as the “logical” addresses, used as RDF identifiers (URIs);

Upon receiving an HTTP request, the server should return a set of triples that describe the resource;

1. Re-use identifiers and vocabulary from other datasets that are also linked data.

Meeting the first requirement is only a matter of properly selecting the namespaces, used for constructing URL the URIs.

The HTTP requests to this sub-domain will be directed to a Forest-based service, which will be used to provide access to MULTISENSOR data; Forest provides support for linked data publishing and can answer HTTP GET requests in accordance with the “linked data etiquette”. The separate bodies of data will be loaded in the repository in one or several named graphs, dedicated to them, so that they can be managed separately, but also queried together with the rest of the data.

To meet the third requirement, if the datasets subject to publishing are not already properly linked to other linked data datasets, the standard procedure will be to get them linked to one or more of the dataset in the Reference Knowledge Stack. This approach is appropriate for the following reasons:

- RKS already includes several of the most popular datasets of linked data, e.g. DBPedia, Freebase, Geonames and MusicBrainz, and many of the most popular ontologies and schemata (e.g. FOAF and SKOS);

- RKS does not limit the diversity because it includes multiple alternative references for the most popular concepts. For instance, most of the locations can be found in more than three of the datasets in RKS (DBPedia, Freebase, Geonames). Most of the popular classes and relationships are also available through several ontologies and schemata (PROTON, DBPedia).

We will use these recommendations in the project in order to store the domain related information from other sources on the web or from the MULTISENSOR services.

OWLIM is the software basis of the BSI. In the next phases of MULTISENSOR BSI will be extended with additional reasoning services to be more suitable to support the specific use case services. The decision support system will be heavily integrated with OWLIM in order to process the necessary facts effectively.

5 CONCLUSIONS

The Basic Semantic Infrastructure that would reflect the business cases of the Multisensor project will use two upper ontologies – PROTON and DOLCE – for knowledge modelling of the relevant information. The most important requirements for the selection of these specific ontologies were: the possibility for cleaning the unnecessary statements, connectivity to linked open datasets, such as DBPedia, Freebase, etc., the possibility of applying a good inference mechanism over the data modelling flow. The ontologies will be mapped to the respective domain lexicons for all the surveyed languages. The infrastructure will also rely on the OWLIM repository for the semantic representation of the data. For RDF-ized and linked data, the FactForge reason-able view will be used as a Reference Knowledge Stack.

6 REFERENCES

Bechofer, S, van Harmelen, F., Hendler, J., Horrocks, I., McGuinness, D. L., Patel-Schneider, P. F., and Stein, L. A. 2004. "OWL Web Ontology Language Reference", In: Dean, M., Schreiber, G. (Eds.), W3C Recommendation, February 10, 2004. http://www.w3.org/TR/owl-ref/.

Berners-Lee, T. 2006. "Design Issues: Linked Data", http://www.w3.org/DesignIssues/LinkedData.html

Bizer, C., Heath, T., and Berners-Lee, T. 2009a. "Linked Data – The Story so Far", In: Heath, T., Hepp, M. and Bizer, C. (Eds.) Special Issue on Linked Data, International Journal on Semantic Web and Information Systems (IJSWIS), http://linkeddata.org/docs/ijswis-special-issue

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S. 2009b. "DBpedia – A Crystallization Point for the Web of Data", Journal of Web Semantics: Science, Services and Agents on the World Wide Web, Issue 7, Pages 154–165.

Bizer, Ch., Schultz, A. BSBM V3 Results. Retrieved May 2014. http://www4.wiwiss.fu-berlin.de/bizer/BerlinSPARQLBenchmark/results/V6/index.html#comparison

Brickley, D., Guha, R.V, (eds.) 2004. "Resource Description Framework (RDF) Schemas", W3C Recommendation, 10 February 2004. http://www.w3.org/TR/rdf-schema/

Clark, K.G., Feigenbaum, L., Torres, E. (eds.). SPARQL Protocol for RDF. W3C Recommendation 15 January 2008. http://www.w3.org/TR/2008/REC-rdf-sparql-protocol-20080115/

Ding, L. et al., 2005. "Tracking RDF Graph Provenance using RDF Molecules", Proceedings of the 4th International Semantic Web Conference

Harris, S., Seaborne, A. (eds.), 2008. SPARQL 1.1 Query Language. http://www.w3.org/TR/2010/WD-sparql11-query-20101014/

Heim, P., Hellmann, S., Lehmann, J., Lohmann, S., Stegemann, T. 2009. "RelFinder: Revealing Relationships in RDF Knowledge Bases", In Proc. of the 4th International Conference on Semantic and Digital Media Technologies SAMT.

Kiryakov, A., and Momtchev, V. 2009. "Two Reason-able Views to the Web of Linked Data", Presentation at the Semantic Technology Conference 2009, San Jose. http://www.slideshare.net/ontotext/two-reasonable-views-to-the-web-of-linked-data.

Li, Y., Cunningham, H., Roberts, A., Kiryakov, A., Momtchev, V., Greenwood, M., Aswani, N., Damljanovic, D. 2009. "Selection Components (report accompanying two software deliverables)". LarKC project deliverable D2.2.1, 2.5.1.

Masolo,C., Borgo, S., Gangemi, A., Guarino, N., Oltramari, A., Luc Schneider, L. 2002. "The WonderWeb Library of Foundational Ontologies", WonderWeb Deliverable D17, August 2002. http://www.loa-cnr.it/Publications.html.

Oberle, D., Ankolekar, A., Hitzler, P., Cimiano, P., Sintek, M., Kiesel, M., Mougouie, B., Vembu, S., Baumann, S., Romanelli, M., Buitelaar, P., Engel, R., Sonntag, D., Reithinger, N., Loos, B., Porzel, R., Zorn, H.-P., Micelli, V., Schmidt, C ., Weiten, M., Burkhardt, F., Zhou, J. 2006. "DOLCE ergo SUMO: On Foundational and Domain Models in SWIntO (SmartWeb Integrated Ontology)", Submission to Journal of Web Semantics.

Semy S., Pulvermacher, M., Obrst, L. 2004. "Toward the Use of an Upper Ontology for U.S. Government and Military Domains: An Evaluation", MITRE Technical Report 04B0000063.

Ontotext. OWLIM-Enterprise (Replication Cluster) Version 5.4. Retrieved on May 2014. http://owlim.ontotext.com/display/OWLIMv54/OWLIM-Enterprise

Page, L., Brin, S., Motwani, R., Winograd, T. "The PageRank citation ranking: Bringing order to the Web", http://dbpubs.stanford.edu:8090/pub/showDoc.Fulltext?lang=en&doc=1999-66&format=pdf&compression. (1999)

Prud'hommeaux, E., Seaborne, A. 2008. "SPARQL Query Language for RDF", W3C Recommendation 15(2008), http://www.w3.org/TR/rdf-sparql-query/

Schenk, S., Gearon, P. "SPARQL 1.1 Update", W3C Working Draft 26 January 2010. http://www.w3.org/TR/2010/WD-sparql11-update-20100126/

LOD. World Wide Web Consortium (W3C): Linking Open Data. W3C SWEO community project home page, as of June 2014. http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData (2014)

Bishop, B., Kiryakov, A., Ognyanoff, D., Peikov, I., Tashev, Z., Velkov, R. 2011. "FactForge: A fast track to the web of data", Submission for Semantic Web Journal, Special Issue: Real-World Applications of OWL. http://www.semantic-web-journal.net/content/new-submission-factforge-fast-track-web-data.

ter Horst, H. J. 2005. "Combining RDF and Part of OWL with Rules: Semantics, Decidability, Complexity", In Proc. of ISWC 2005, Galway, Ireland, LNCS 3729, pp. 668-684

Guarino, N., Welty, C. 2002. "Evaluating Ontological Decisions with OntoClean", Communications of the ACM. 45(2):61-65. New York:ACM Press

A Example query for dataset cleaning

Here is a complete example for cleaning of wrong statements for the property dbp-ont:birthPlace. Using the knowledge from the DBPedia ontology we construct an SPARQL query which deleted all the statements in which the birth place is not a place.

```
delete {?s dbp-ont:birthPlace ?y } where
        {?s dbp-ont:birthPlace ?y .
         ?y rdf:type ?c .
        filter(?c = dbp-ont:AcademicJournal || ?c = dbp-ont:Activity
         || ?c = dbp-ont:AdministrativeRegion || ?c = dbp-ont:Aircraft
         || ?c = dbp-ont:Airline || ?c = dbp-ont:Airport
         || ?c = dbp-ont:Album || ?c = dbp-ont:AmericanFootballLeague
         || ?c = dbp-ont:AmericanFootballTeam || ?c = dbp-ont:Amphibian
         || ?c = dbp-ont:AnatomicalStructure || ?c = dbp-ont:Animal
         || ?c = dbp-ont:Arachnid || ?c = dbp-ont:Archaea
         || ?c = dbp-ont:ArchitecturalStructure || ?c = dbp-ont:Arena
         || ?c = dbp-ont:Artery || ?c = dbp-ont:Artwork
         || ?c = dbp-ont:Asteroid || ?c = dbp-ont:Atoll
         || ?c = dbp-ont:AustralianFootballLeague || ?c = dbp-ont:AutoRacingLeague
         || ?c = dbp-ont:Automobile || ?c = dbp-ont:AutomobileEngine
         || ?c = dbp-ont:Award || ?c = dbp-ont:Bacteria
         || ?c = dbp-ont:Band || ?c = dbp-ont:BaseballLeague
         || ?c = dbp-ont:BaseballTeam || ?c = dbp-ont:BasketballLeague
         || ?c = dbp-ont:BasketballTeam || ?c = dbp-ont:Beverage
         || ?c = dbp-ont:BiologicalDatabase || ?c = dbp-ont:Biomolecule
         || ?c = dbp-ont:Bird || ?c = dbp-ont:BodyOfWater
         || ?c = dbp-ont:Bone || ?c = dbp-ont:Book
         || ?c = dbp-ont:BowlingLeague || ?c = dbp-ont:BoxingLeague
         || ?c = dbp-ont:Brain || ?c = dbp-ont:Bridge
         || ?c = dbp-ont:BroadcastNetwork || ?c = dbp-ont:Broadcaster
         || ?c = dbp-ont:Building || ?c = dbp-ont:CanadianFootballLeague
         || ?c = dbp-ont:CanadianFootballTeam || ?c = dbp-ont:Canal
         || ?c = dbp-ont:Cave || ?c = dbp-ont:CelestialBody
         || ?c = dbp-ont:ChemicalCompound || ?c = dbp-ont:ChemicalElement
         || ?c = dbp-ont:ChemicalSubstance || ?c = dbp-ont:Church
         || ?c = dbp-ont:City || ?c = dbp-ont:ClubMoss
         || ?c = dbp-ont:College || ?c = dbp-ont:Colour
|| ?c = dbp-ont:ComicBook || ?c = dbp-ont:Company
|| ?c = dbp-ont:Conifer || ?c = dbp-ont:Constellation
|| ?c = dbp-ont:Continent || ?c = dbp-ont:Convention
|| ?c = dbp-ont:Country || ?c = dbp-ont:CricketLeague
|| ?c = dbp-ont:Crustacean || ?c = dbp-ont:CurlingLeague
|| ?c = dbp-ont:Currency || ?c = dbp-ont:Cycad
|| ?c = dbp-ont:CyclingLeague || ?c = dbp-ont:Database
|| ?c = dbp-ont:Decoration || ?c = dbp-ont:Device
|| ?c = dbp-ont:Disease || ?c = dbp-ont:Drug
|| ?c = dbp-ont:EducationalInstitution || ?c = dbp-ont:Election
|| ?c = dbp-ont:Embryology || ?c = dbp-ont:EthnicGroup
|| ?c = dbp-ont:Eukaryote || ?c = dbp-ont:EurovisionSongContestEntry
|| ?c = dbp-ont:Event || ?c = dbp-ont:Fern
|| ?c = dbp-ont:FieldHockeyLeague || ?c = dbp-ont:Film
|| ?c = dbp-ont:FilmFestival || ?c = dbp-ont:Fish
|| ?c = dbp-ont:Flag || ?c = dbp-ont:FloweringPlant
|| ?c = dbp-ont:Food || ?c = dbp-ont:FootballMatch
|| ?c = dbp-ont:FormulaOneRacing || ?c = dbp-ont:Fungus
|| ?c = dbp-ont:Galaxy || ?c = dbp-ont:Game
|| ?c = dbp-ont:Gene || ?c = dbp-ont:GeneLocation
|| ?c = dbp-ont:GeopoliticalOrganisation || ?c = dbp-ont:Ginkgo
|| ?c = dbp-ont:GivenName || ?c = dbp-ont:Gnetophytes
|| ?c = dbp-ont:GolfLeague || ?c = dbp-ont:GovernmentAgency
|| ?c = dbp-ont:GovernmentType || ?c = dbp-ont:GrandPrix
|| ?c = dbp-ont:Grape || ?c = dbp-ont:GreenAlga
|| ?c = dbp-ont:HandballLeague || ?c = dbp-ont:HistoricBuilding
|| ?c = dbp-ont:HistoricPlace || ?c = dbp-ont:HockeyTeam
|| ?c = dbp-ont:Holiday || ?c = dbp-ont:Hospital
|| ?c = dbp-ont:Hotel || ?c = dbp-ont:HumanGene
|| ?c = dbp-ont:HumanGeneLocation || ?c = dbp-ont:IceHockeyLeague
|| ?c = dbp-ont:Ideology || ?c = dbp-ont:Infrastructure
|| ?c = dbp-ont:InlineHockeyLeague || ?c = dbp-ont:Insect
|| ?c = dbp-ont:Instrument || ?c = dbp-ont:Island
|| ?c = dbp-ont:LacrosseLeague || ?c = dbp-ont:Lake
|| ?c = dbp-ont:Language || ?c = dbp-ont:LaunchPad
|| ?c = dbp-ont:LawFirm || ?c = dbp-ont:LegalCase
|| ?c = dbp-ont:Legislature || ?c = dbp-ont:Letter
|| ?c = dbp-ont:Library || ?c = dbp-ont:Lighthouse
|| ?c = dbp-ont:Locomotive || ?c = dbp-ont:LunarCrater
|| ?c = dbp-ont:Lymph || ?c = dbp-ont:Magazine
|| ?c = dbp-ont:Mammal || ?c = dbp-ont:MeanOfTransportation
|| ?c = dbp-ont:MilitaryConflict || ?c = dbp-ont:MilitaryUnit
|| ?c = dbp-ont:Mineral || ?c = dbp-ont:MixedMartialArtsEvent
|| ?c = dbp-ont:MixedMartialArtsLeague || ?c = dbp-ont:Mollusca
|| ?c = dbp-ont:Monument || ?c = dbp-ont:Moss
|| ?c = dbp-ont:MotorcycleRacingLeague || ?c = dbp-ont:Mountain
|| ?c = dbp-ont:MountainPass || ?c = dbp-ont:MountainRange
|| ?c = dbp-ont:MouseGene || ?c = dbp-ont:MouseGeneLocation
|| ?c = dbp-ont:Muscle || ?c = dbp-ont:Museum
|| ?c = dbp-ont:MusicFestival || ?c = dbp-ont:MusicGenre
|| ?c = dbp-ont:Musical || ?c = dbp-ont:MusicalWork
|| ?c = dbp-ont:Name || ?c = dbp-ont:NationalSoccerClub
|| ?c = dbp-ont:NaturalPlace || ?c = dbp-ont:Nerve
|| ?c = dbp-ont:Newspaper || ?c = dbp-ont:Non-ProfitOrganisation
|| ?c = dbp-ont:OlympicResult || ?c = dbp-ont:Olympics
|| ?c = dbp-ont:Organisation || ?c = dbp-ont:PaintballLeague
|| ?c = dbp-ont:Painting || ?c = dbp-ont:Park
|| ?c = dbp-ont:PeriodicalLiterature || ?c = dbp-ont:Place
|| ?c = dbp-ont:Planet || ?c = dbp-ont:Plant
|| ?c = dbp-ont:Play || ?c = dbp-ont:PoliticalParty
|| ?c = dbp-ont:PoloLeague || ?c = dbp-ont:PopulatedPlace
|| ?c = dbp-ont:PowerStation || ?c = dbp-ont:ProgrammingLanguage
|| ?c = dbp-ont:Project || ?c = dbp-ont:ProtectedArea
|| ?c = dbp-ont:Protein || ?c = dbp-ont:PublicTransitSystem
|| ?c = dbp-ont:Race || ?c = dbp-ont:RadioControlledRacingLeague
|| ?c = dbp-ont:RadioStation || ?c = dbp-ont:RailwayLine
|| ?c = dbp-ont:RailwayTunnel || ?c = dbp-ont:RecordLabel
|| ?c = dbp-ont:ReligiousBuilding || ?c = dbp-ont:Reptile
|| ?c = dbp-ont:ResearchProject || ?c = dbp-ont:Restaurant
|| ?c = dbp-ont:River || ?c = dbp-ont:Road
|| ?c = dbp-ont:RoadJunction || ?c = dbp-ont:RoadTunnel
|| ?c = dbp-ont:Rocket || ?c = dbp-ont:RouteOfTransportation
|| ?c = dbp-ont:RugbyClub || ?c = dbp-ont:RugbyLeague
|| ?c = dbp-ont:Sales || ?c = dbp-ont:SambaSchool
|| ?c = dbp-ont:School || ?c = dbp-ont:Sculpture
|| ?c = dbp-ont:Settlement || ?c = dbp-ont:Ship
|| ?c = dbp-ont:ShoppingMall || ?c = dbp-ont:Single
|| ?c = dbp-ont:SiteOfSpecialScientificInterest || ?c = dbp-ont:SkiArea
|| ?c = dbp-ont:Skyscraper || ?c = dbp-ont:SnookerWorldRanking
|| ?c = dbp-ont:SoccerClub || ?c = dbp-ont:SoccerClubSeason
|| ?c = dbp-ont:SoccerLeague || ?c = dbp-ont:SoccerLeagueSeason
|| ?c = dbp-ont:SoccerTournament || ?c = dbp-ont:SoftballLeague
|| ?c = dbp-ont:Software || ?c = dbp-ont:Song
|| ?c = dbp-ont:SpaceMission || ?c = dbp-ont:SpaceShuttle
|| ?c = dbp-ont:SpaceStation || ?c = dbp-ont:Spacecraft
|| ?c = dbp-ont:Species || ?c = dbp-ont:SpeedwayLeague
|| ?c = dbp-ont:SpeedwayTeam || ?c = dbp-ont:Sport
|| ?c = dbp-ont:SportsEvent || ?c = dbp-ont:SportsLeague
|| ?c = dbp-ont:SportsTeam || ?c = dbp-ont:SportsTeamSeason
|| ?c = dbp-ont:Stadium || ?c = dbp-ont:Station
|| ?c = dbp-ont:Stream || ?c = dbp-ont:SupremeCourtOfTheUnitedStatesCase
|| ?c = dbp-ont:Surname || ?c = dbp-ont:Tax
|| ?c = dbp-ont:TelevisionEpisode || ?c = dbp-ont:TelevisionSeason
|| ?c = dbp-ont:TelevisionShow || ?c = dbp-ont:TelevisionStation
|| ?c = dbp-ont:TennisLeague || ?c = dbp-ont:TennisTournament
|| ?c = dbp-ont:Theatre || ?c = dbp-ont:TopicalConcept
|| ?c = dbp-ont:Town || ?c = dbp-ont:TradeUnion
|| ?c = dbp-ont:Tunnel || ?c = dbp-ont:University
|| ?c = dbp-ont:Unknown || ?c = dbp-ont:Valley
|| ?c = dbp-ont:Vein || ?c = dbp-ont:VideoGame
|| ?c = dbp-ont:VideogamesLeague || ?c = dbp-ont:Village
|| ?c = dbp-ont:Volcano || ?c = dbp-ont:VolleyballLeague
|| ?c = dbp-ont:WaterwayTunnel || ?c = dbp-ont:Weapon
|| ?c = dbp-ont:Website || ?c = dbp-ont:WineRegion
|| ?c = dbp-ont:WomensTennisAssociationTournament || ?c = dbp-ont:Work
|| ?c = dbp-ont:WorldHeritageSite || ?c = dbp-ont:WrestlingEvent
|| ?c = dbp-ont:WrittenWork || ?c = dbp-ont:Year || ?c = dbp-ont:YearInSpaceflight )
}
```