MOLTO

Multilingual Online Translation

Non multa, sed multum

Contract No.:	FP7-ICT-247914
Project full title:	MOLTO – Multilingual Online Translation
Deliverable:	D8.3 W8 Translation and retrieval system for museum object descriptions
Security (distribution level):	Public
Contractual date of delivery:	M36
Actual date of delivery:	March 31, 2013
Modified:	April 22, 2013
Type:	Regular Publication
Status & version:	v2.0
Authors:	Dana Dannélls, Aarne Ranta, Ramona Enache Mariana Damova, Maria Mateva
Task responsible:	UGOT
Other contributors:	Ontotext

ABSTRACT

This is the final report of *Workpackage 8: Case Study: Cultural Heritage*. The major contributions reported are ontology-based multilingual grammar covering 15 languages and cross-language retrieval system for museum object descriptions using Semantic Web technology. Our groundwork for this deliverable was laid in *D8.1: Ontology and corpus study of the cultural heritage domain*,[a] and *D8.2: Multilingual grammar for museum object descriptions*.[b]

[a] http://www.molto-project.eu/biblio/d81-ontology-and-corpus-study-cultural-heritage-domain

[b] http://www.molto-project.eu/biblio/deliverable/multilingual-grammar-museum-object-descriptions

Contents

1 Introduction

The purpose of the work presented here has been to develop a multilingual system that enables interaction with digital museum libraries that are today accessible in semantically-enriched content through the deployment of Semantic Web technologies [14, 15, 16, 23] such as the Resource Description Framework (RDF) [21], RDF Schema (RDFs) [3], and the Web Ontology Language (OWL) [19]. Such a system can become of a value to museum visitors who wish to enrich their cultural knowledge through written or spoken media.

In previous reports [6, 8, 12], we described how to promote multilingual access to museums that usually use their own ontology and terminologies on the World Wide Web (WWW) and presented a prototype that exploits Natural Language Generation (NLG) techniques to render as text the retrieved informational content in six languages [10, 11]. In this deliverable we report on a multilingual grammar methodology that has been developed to enable querying a semantic repository in natural language (NL) and automatically generate well-formed descriptions from a set of RDF statements retrieved as the results. The presented methodology offers flexibility regarding interactivity with generic language-specific and domain-dependent resources – a requirement that has been emphasized by many authors in the past [4, 20, 22, 27, 29].

The proposed method includes several steps of transition from natural language to SPARQL (SPARQL Protocol and RDF Query Language) [18] and from RDF to coherent multilingual descriptions. Querying the Semantic Web in natural language, more specifically, using English to formulate SPARQL queries with the help of controlled natural language (CNL) syntax has been developed before [17]. However, we are not aware of a multilingual Semantic Web approach for querying and generating well-formed descriptions from Semantic Web content for which lexicons are not available. Using the Grammatical Framework (GF) [24, 26], we are able to present a cross-language grammar system covering 15 languages and a cross-language retrieval system that uses this grammar for generating museum object descriptions in the Semantic Web.

2 The GF application grammar

The application grammar that has been developed to enable multilingual interaction with Semantic Web content aims to share as many resources as possible between different, more generic grammars. It contains the following grammars:

- Lexicon: Ontology concepts and properties
- Data: Ontology instances
- Answer: Top, module, provides answers about everything in the domain
- Query: Covering structural and yes/no questions of 11 languages, and SPARQL syntax
- Text: Covering the syntax and discourse patterns of written artwork descriptions

Our application grammar covers 15 languages of those that are available in the Resource Grammar Library (RGL) [25]. More specifically, the supported languages are: Bulgarian, Finnish, Norwegian, Catalan, French, Romanian, Danish, Hebrew, Russian, Dutch, Italian, Spanish, English, German and Swedish. In the following, we describe each of the grammars in more details.

2.1 Lexicon Grammar

The lexicon grammar is intended to cover the ontology terminology (Tbox). It contains linearizations of a subset of the ontology concepts and properties that are covered in the underlying data models [7], more specifically the CIDOC-CRM and the Painting ontologies. Examples of these concepts and the properties are shown in Table 1. Examples of how concepts and properties are presented in the abstract syntax, *LexiconPainting.gf*, are:

```
PTPainting, PTPortrait, PTOilPainting, PTWatercolour,
PTMiniature : PaintingType ;

MCanvas, MWood, MPaper, MLinen : Material ;

PTitle  : Title -> Painting ;
YInt    : Int -> Year ;
SIntInt : Int -> Int -> Size ;
```

The concrete syntaxes, captured in *LexiconPainting{LANG}.gf*,[1] have been created manually by native speaker of the language; functions that are linearized differently depending on the language.

Most of the ontology concepts that are defined in this grammar as categories are linearized with noun phrases in the concrete syntaxes using the RGL. As shown below.

```
Swe.
PTPainting = mkCN (mkN "målning") ;
PTPortrait =  mkCN (regGenN "porträtt" neutrum) ;
MCanvas = mkMaterial "kanvas" ;

Fre.
PTPainting = mkCN (mkN "tableau") ;
PTPortrait = mkCN (mkN "portrait") ;
MCanvas = mkMaterial "canvas" ;
```

[1] *LANG* is a string to be replaced with a three letter code of the language, e.g. LexiconPaintingSwe.gf, LexiconPaintingFre.gf, etc

```
Fin.
PTPainting = mkCN (mkN "maalaus") ;
PTPortrait = mkCN (mkN "muoto" (mkN "kuva")) ;

Ger.
PTPainting = mkCN painting_N ;
PTPortrait = mkCN (mkN "Porträt" "Porträts" neuter) ;
MCanvas = mkMaterial "Leinwand" ;
```

Where the function *mkMaterial* is defined as follows for all languages.

```
mkMaterial : Str -> NP = \s -> mkNP (mkPN s) ;
```

Two of the ontology concepts that are not linearized with a noun phrase correspond to the categories *Year* and *Size*. These are linearized with prepositional phrases in which the preposition is language dependent. Below are some examples.

```
Bul. YInt i = SyntaxBul.mkAdv prez_Prep (symb (i.s ++ year_Str)) ;
Fin. YInt i = SyntaxFin.mkAdv (prePrep nominative "vuonna") (symb i) ;
Fre. YInt i = SyntaxFre.mkAdv en_Prep (symb i) ;
Ger. YInt i = SyntaxGer.mkAdv in_Prep (symb i) ;
Swe. YInt i = SyntaxSwe.mkAdv noPrep (symb ("år" ++ i.s)) ;
Rus. YInt i = SyntaxRus.mkAdv in_Prep (symb (i.s ++ godu_Str)) ;
```

The ontology properties are defined with operations in the concrete syntaxes. Because an ontology property is linearized differently depending on how it is realized in the target language, these operations are of type: verbs, adverbs and prepositions, as shown below.

```
Swe.
paint_V2 : V2 = mkV2 "måla" ;
painted_A : A = mkA "målad" ;
at_Prep = mkPrep "på" ;

Fin.
paint_V2 = mkV2 "maalata" ;
painted_A = mkA "maalattu" ;

Ger.
paint_V2 : V2 = mkV2 (mkV "malen") ;
painted_A : A = mkA "gemalt" ;
at_Prep = in_Prep ;
```

This approach permits variations regarding the lexical units the ontology properties should be mapped to. It allows to make principled choices about the different realization of an ontology property.

2.2 Data Grammar

As opposed to the lexicon grammar (Section 2.1), the data grammar is created automatically from the data that is extracted from the datasets, more specifically from GCM and DBpedia. It contains instances of the three core concepts that we cover in the text grammar (Section 2.5), namely: Title, Painter and Museum. In the abstract syntax these concepts are defined as categories, in the concrete, they are linearized with strings or noun phrases.

```
MGothenburg_City_Museum : Museum ;
MMus_e_du_Louvre : Museum ;
TGuernica__28painting_29 : Title ;
PRembrandt_Harmenszoon_van_Rijn : Painter ;
```

There are two concrete syntaxes which are produced automatically: *DataPaintingCnc.gf* and *DataPainting{LANG}.gf*. All the instances from the ontology are linearized in *DataPaintingCnc*. Underscores are a side effect of the automated generation process and are left in the actual linearization because they are used as input to the GF smart paradigms.

DataPaintingCnc

```
lincat
  Title, Museum = Str ;
  Painter = {long,short : Str} ;

MGothenburg_City_Museum = "Gothenburg_City_Museum" ;
MMus_e_du_Louvre = "Musée_du_Louvre" ;
TGuernica__28painting_29 = "Guernica" ;
PRembrandt_Harmenszoon_van_Rijn =
   mkPainter "Rembrandt_Harmenszoon_van_Rijn" "Rijn" ;
```

DataPainting{LANG} contains some of the translated name-entities, see Section 3.2, mainly comprasing museum names.

DataPainting{LANG}

```
lincat Museum = NP ;

Swe.
MGothenburg_City_Museum = mkMuseum "Göteborgs stadsmuseum" ;
MMus_e_du_Louvre = mkMuseum "Louvren" ;

Ita.
MGothenburg_City_Museum = mkMuseum "museo municipale di Goteburgo" ;
MMus_e_du_Louvre = mkMuseum "Museo del Louvre" ;
```

```
Fre.
MGothenburg_City_Museum = mkMuseum "musée municipal de Göteborg" ;
MMus_e_du_Louvre = mkMuseum "Musée du Louvre" ;

Cat.
MGothenburg_City_Museum = mkMuseum "Gothenburg_City_Museum" ;
MMus_e_du_Louvre = mkMuseum "Museu del Louvre" ;

Ger.
MGothenburg_City_Museum = mkMuseum "Gothenburg_City_Museum" ;
MMus_e_du_Louvre = mkMuseum "Der Louvre " ;
```

The majority of languages linearize the museum name entities with an NP without a gender. They are constructed by the function *mkMuseum*.

```
  mkMuseum : Str -> NP = \s -> mkNP (mkPN s ) ;
```

A special case of *mkMuseum* appears in four languages: Italian, Catalan, Spanish and French, where a masculine gender is assigned to the museum string to get the correct form of the string.

```
  mkMuseum : Str -> NP = \s -> mkNP the_Det (mkN s masculine ) ;
```

In Finnish, a smart paradigm is used to constructed the correct form of the museum string.

```
 mkMuseum : Str -> NP = \s -> case last s of {
    "e" => mkNP (mkPN (mkN s (s + "n") (s + "ja"))) ;
    "n" | "s" => mkNP (mkPN (mkN s (s + "in") (s + "eja"))) ;
    _   => mkNP (mkPN (mkN s))
  } ;
```

2.3 Answer Grammar

The answer grammar has been developed to generate yes/no answers or a coherent text as a response to a query. It is the top module, with both questions and answers, including texts. It contains the following functions.

```
fun
 AAnswer : Answer -> Anything ;
 AMove : Move -> Anything ;

 AYes, ANo : Answer ;
 ADescription : Description -> Answer ;
```

Linearizations of these functions look similar for all languages.

```
lin
 AYes = mkText yes\_Utt ;
 ANo = mkText no\_Utt ;
 ADescription d = d ;
```

2.4 Query Grammar

The approach presented here relies on the assumption that an ontology restricts the number of semantic queries that can be run against it, as it represents a closed world bound by the concepts and properties that are included in it. An ontology has a logically organized structure that semantically characterizes the domain. This allows formulating a controlled language that will exhaustively cover all possible conceptual semantic queries.

The approach to queries is that the abstract syntax is driven by the ontology and the concrete syntax by the resource grammars. There is a query form for everything the ontology can provide answers to. Part of the abstract syntax is generic (such as wh-questions and quantifiers), the other part, the predicates are domain-dependent. In the same way, part of the concrete syntax is language dependent and language independent.

The query grammar builds on Yet Anothe Query Language (YAQL), a grammar that has been developed in workpackage 4. It supports wh-questions, quantifiers, questions, assertions, one/two place predicates, and question adverb/noun phrases.

```
QPainter : Painting -> Query ;    -- who painted x
PPainter : Painter -> Property ; -- x is by Picasso
MQuery   : Query -> Move ;        -- what is the material of x
```

Its lexical entries, e.g wood, canvas, oil painting, etc, come from the module *Lexicon-Painting*. Title, painter and museum names come from the automatically generated module *DataPainting*. Some examples of the queries that can be formulated with this grammar are:

- All About X, Show everything about X
- How many X
- Who is X, What is X
- Some X
- All X painted by Y
- Some X painted on Y
- What is the material of X
- Show everything about all X that are painted on Y
- X is by Y
- X is made of Y

2.4.1 YAQL Grammar

The YAQL grammar is an implementation result that grow out of experiences gained from designing query grammars for previous workpackages. Its main features are:

- a common architecture with base module + domain
- straightforward abstract syntax generation from ontology, with just the minimum of lexical types
  - Kind – usually CN
  - Entity – usually NP
  - Property – can be VP, AP, ClSlash
  - Relation – VPSlash built from V2, AP, comparatives
- all kinds of queries, from which applications can select a subset

The generic grammar structure allows it to be reused by any modules for any domain. *Kind* is loosely coupled with the OWL Entities.

2.4.2 SPARQL Grammar

In the context of the Semantic Web, semantic data is accessible via the SPARQL endpoint,[2] as in our Museum Reason-able View of Linked Open Data (LOD) [7]. One of the bottlenecks of SPARQL is that formulating a query requires knowledge of the query language and of the schemata underlying the datasets in the knowledge representation infrastructure. To avoid this, natural language/controlled natural language mechanism could be used to help the user formulate queries by suggesting the valid words. These words are in fact the lexicalizations of the concepts and properties that are available in the knowledge representation infrastructure.

The SPARQL grammar that consists of several modules: *YAQL1SPARQL, QueryPaintingSPARQL, LexiconPaintingSPARQL* and *DataPaintingSPARQL*, was developed to provide exactly this functionality. The grammar builds on existing categories and functions to allow formulating NL queries that are in turn being translated into a single SPARQL query. The grammar has been developed on top of YAQL, hence, all of the natural language sentences that are supported by the query grammar, and that are listed in the beginning of this section (Section 2.4) are defined in the SPARQL grammar. It uses the same lexicon and data sets that have been mapped from the ontology and linearized with corresponding language strings.

To allow translation to SPARQL some extensions were required. For example, the category *Query* has defined with three parameters to support queries of type *MQuery*:

```
Query = {wh1 : Str ; wh2 : Str ; prop : Str} ;
```

Each category is linearized with a different string depending on the category in question.

[2] http://museum.ontotext.com/sparql

```
QMaterial p =
  {wh1= "?material"; prop = p; wh2=" painting:hasMaterial ?material ." };
QMuseum p =
  {wh1= "?museum"; prop = p; wh2=" painting:hasCurrentLocation ?museum ."};
QPainter p =
  {wh1= "?painter"; prop = p; wh2=" painting:createdBy ?painter ."};
```

The *prop* category is generated from the module *DataPaintingSPARQL* and is constructed with the appropriated SPARQL sub-strings.

```
lin PTitle t = "FILTER (regex(?title,"++ t ++",\"i\"))"
 ++ "$n" ++ "FILTER (lang(?title) = 'en') " ;
```

Query parameters are then used to generate structured SPARQL strings using simple concatenating.

```
MQuery q = "PREFIX painting:
<http://spraakbanken.gu.se/rdf/owl/painting.owl#> $n
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> $n
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> $n
SELECT distinct"++ q.wh1 ++ "$n WHERE { $n
?painting rdf:type painting:Painting; $n
rdfs:label ?title; $n " ++ q.wh2 ++ "$n" ++ q.prop++"}" ;
```

Some examples of SPARQL generations are:[3]

QueryPainting: MQuery (QColour (PTitle TMonaLisa_jpg))
QueryPaintingEng: what are the colours of Mona_Lisa
QueryPaintingSPARQL:

```
PREFIX painting: <http://spraakbanken.gu.se/rdf/owl/painting.owl#> $n
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> $n
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> $n
SELECT distinct ?color $n
WHERE { $n ?painting rdf:type painting:Painting . $n
?painting rdfs:label ?title
FILTER (str(?title)= "Mona_Lisa" ) .
$n ?painting painting:hasColor ?color . }
```

QueryPainting: MQuery (QMaterial (PTitle TMonaLisa_jpg))
QueryPaintingEng: what is the material of Mona_Lisa
QueryPaintingSPARQL:

[3]The *$n* stands for new line identifier for the backend to post-process.

```
PREFIX painting: <http://spraakbanken.gu.se/rdf/owl/painting.owl#> $n
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> $n
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#> $n
SELECT distinct ?material $n
WHERE { $n ?painting rdf:type painting:Painting . $n
?painting rdfs:label ?title
FILTER (str(?title)= "Mona_Lisa" ) . $n
?painting painting:hasMaterial ?material . }
```

2.5 Text Grammar

From the knowledge representation infrastructure a set of RDF triples are returned as results from a SPARQL query (see examples in Section 4). Our text grammar has been designed to generate a coherent natural language descriptions from a selected set of the returned triples. More specifically, our grammar covers eight concepts that are most commonly used to describe a painting, including: Title, Painter, Painting type, Material, Colour, Year, Museum and Size. The corresponding ontology concepts are listed in Table 1. In the Text Grammar they are defined as categories and are captured in one function *DPainting* which has the following representation in the abstract syntax.[4]

```
 DPainting :
 Painting -> Painter -> PaintingType ->
 OptColours -> OptSize -> OptMaterial ->
 OptYear  -> OptMuseum  -> Description ;
```

In the function *DPainting* five arguments have been implemented as optional, i.e. *OptColour, OptSize, OptMaterial, OptYear* and *OptMuseum.* Each of these categories can be left out in a text. Some examples from the abstract and the concrete syntaxes illustrate how they are implemented:

```
  MkYear : Year -> OptYear ;
  MkMuseum : Museum -> OptMuseum ;
  MkSize : Size -> OptSize ;
  MkMaterial : Material -> OptMaterial ;

Dut.
  MkYear year = mkOptAdv year ;
  MkMuseum museum = mkOptAdv (SyntaxDut.mkAdv at_Prep (museum)) ;
  MkSize size = mkOptAdv (SyntaxDut.mkAdv (mkPrep []) size) ;
  MkMaterial material = mkOptAdv (SyntaxDut.mkAdv on_Prep material) ;
```

[4] The category *Painting* actually represents the title of the painting.

Table 1: Examples of the ontology concepts and properties covered in the grammar.

GF category	Ontology concept	Ontology property statement
PTPainting	Painting	SubClassOf:Artwork
		createdBy:Painter
		hasCreationDate:TimePeriod
		hasMaterial:Material
		hasTitle:Title
		hasDimension:Dimension
		hasCurrentLocation:Museum
		hasColor:Color
Painter	Painter	SubClassOf:Human
Year	TimePeriod	fromTimePeriodValue:Value
		toTimePeriodValue:Value
Material	Material	sameAs:E57_Material
Title	Title	sameAs:E35_Title
Size	Dimension	hasMeasure:Measure
		hasUnitOfMeasure:UnitOfMeasure
Museum	Museum	SubClassOf:Building

```
Eng.
  MkYear year = mkOptAdv year ;
  MkMuseum museum = mkOptAdv (SyntaxEng.mkAdv at_Prep museum) ;
  MkSize size = mkOptAdv (SyntaxEng.mkAdv part_Prep
                       (mkNP (mkCN size_N size))) ;
  MkMaterial material = mkOptAdv (SyntaxEng.mkAdv on_Prep material) ;

Fin.
  MkYear year = mkOptAdv year ;
  MkMuseum museum = mkOptAdv (SyntaxFin.mkAdv in_Prep museum) ;
  MkSize size = mkOptAdv (SyntaxFin.mkAdv part_Prep size) ;
  MkMaterial material = mkOptAdv (SyntaxFin.mkAdv for_Prep material) ;

Fre.
  MkYear year = mkOptAdv year ;
  MkMuseum museum = mkOptAdv (SyntaxFre.mkAdv dative museum) ;
  MkSize size = mkOptAdv (SyntaxFre.mkAdv part_Prep size) ;
  MkMaterial material = mkOptAdv (SyntaxFre.mkAdv on_Prep material) ;

Ita.
  MkYear year = mkOptAdv year ;
```

```
  MkMuseum museum = mkOptAdv (SyntaxIta.mkAdv dative museum) ;
  MkSize size = mkOptAdv (SyntaxIta.mkAdv part_Prep size) ;
  MkMaterial material = mkOptAdv (SyntaxIta.mkAdv on_Prep material) ;

Ger.
  MkYear year = mkOptAdv year ;
  MkMuseum museum = mkOptAdv (SyntaxGer.mkAdv at_Prep museum) ;
  MkSize size = mkOptAdv (SyntaxGer.mkAdv (mkPrep [] nominative) size) ;
  MkMaterial material = mkOptAdv (SyntaxGer.mkAdv on_Prep material) ;
```

With this approach, different texts can be generated, depending on the information that is available in the ontology. A minimal description consists of a title, a painter and a painting type. This approach allows for efficient multilingual linearizations [9], as opposed to the previous one [13], where semantic patterns were defined with different functions and thus required an extensive linguistic effort to linearize. Below follow some examples of texts generated in English to exemplify the different descriptions we are able to generate from one single function call with a varying number of instantiated parameters.

- Interior was painted on canvas by Edgar Degas in 1868. It measures 81 by 114 cm and it is painted in red and white. This painting is displayed at the Philadelphia Museum of Art.

- Interior was painted by Edgar Degas in 1868. It measures 81 by 114 cm. This painting is displayed at the Philadelphia Museum of Art.

- Interior was painted on canvas by Edgar Degas in 1868. It is painted in red and white. This painting is displayed at the Philadelphia Museum of Art.

- Interior was painted by Edgar Degas. It measures 81 by 114 cm and it is painted in red and white. This painting is displayed at the Philadelphia Museum of Art.

- Interior was painted on canvas by Edgar Degas. It measures 81 by 114 cm and it is painted in red and white.

- Interior was painted by Edgar Degas in 1868. This painting is displayed at the Philadelphia Museum of Art.

- Interior was painted by Edgar Degas.

The design of the text grammar is similar to the query grammar, i.e. the abstract syntax is driven by the ontology and is language independent. The concrete syntax is both language dependent and language independent. The language dependent part is the choice of the main verbs, tenses and phrases. The language independent part relates to how we package semantic concepts. In the current implementation we limited the length of a description to three sentences. The examples from the concrete syntaxes below show how the first sentence of a text decription (*s1*) is linearized in the different languages.

```
Bul. s1 : Text = mkText (mkS (mkCl painting (mkVP (mkVP (mkVP
      (passiveVP paint_V2) material.s) (SyntaxBul.mkAdv from_Prep
      (title painter.long))) year.s))) ;

Eng. s1 : Text = mkText (mkS pastTense (mkCl painting (mkVP (mkVP (mkVP
      (passiveVP paint_V2) material.s)(SyntaxEng.mkAdv by8agent_Prep
      (title painter.long))) year.s))) ;

Fin. s1 : Text = mkText (S_OVS noPart (mkTemp presentTense anteriorAnt)
      positivePol (PredClPlus (title painter.long) (mkVP (mkVP (mkVP
      paint_V2 (mkNP (mkCN paintingtype painting))) material.s) year.s))) ;

Swe. s1 : Text = mkText (mkS pastTense mkCl painting (mkVP (mkVP (mkVP
      (PassV2 paint_V2) material.s) (SyntaxSwe.mkAdv by8agent_Prep
      (title painter.long))) year.s))) ;

Ita. s1 : Text = mkText (mkS (mkCl painting (mkVP (mkVP (mkVP
      (mkVP dipinto_A) material.s) (SyntaxIta.mkAdv by8agent_Prep
      (title painter.long))) year.s))) ;

Fre. s1 : Text = mkText (mkS anteriorAnt (mkCl painting (mkVP (mkVP (mkVP
      (passiveVP paint_V2) material.s) (SyntaxFre.mkAdv by8agent_Prep
      (title painter.long))) year.s))) ;

Ger. s1 : Text = mkText (mkS pastTense (mkCl painting (mkVP (mkVP
      (mkVP (passiveVP paint_V2) year.s) (SyntaxGer.mkAdv von_Prep
      (title painter.long))) material.s))) ;

Rus. s1 : Text = mkText (mkS pastTense (mkCl painting (mkVP (mkVP (mkVP
      (passiveVP paint_V2) (SyntaxRus.mkAdv part_Prep
      (title painter.long masculine animate))) material.s) year.s))) ;
```

The above extracts from the concrete syntaxes show how we exploit RGL to linearize the first sentence of a description. This sentence comprises four semantic categories. Some of the distinguishing differences between the languages are: in Finnish the use of an active voice, in Italian, present tense, in French, past participle. The order of the categories is also different. In German the material string appears at the end of the sentence as opposed to the other languages where year is often the last string. Examples of the multilingual descriptions generated with these grammars are shown in Figure 1.

TextPaintingBul: The Potato Eaters е нарисувана от Vincent van Gogh през 1885 година. Тя е с размер 82 см на 115 см. Този експонат е изложен в Музей ван Гог.
TextPaintingCat: The Potato Eaters està pintat per Vincent van Gogh en 1885. Ell és de 82 sobre 115 cm. Aquesta pintura està exposada al Museu Van Gogh.
TextPaintingDan: The Potato Eaters blev malet af Vincent van Gogh i 1885. Det er 82 ganger 115 cm. Dette maleri er udstillet på Van Gogh-museet.
TextPaintingDut: The Potato Eaters werd in 1885 door Vincent van Gogh geschilderd. Het werk is 82 bij 115 cm. Dit schilderij wordt in Van Gogh Museum getoond.
TextPaintingEng: The Potato Eaters was painted by Vincent van Gogh in 1885. It measures 82 by 115 cm. This painting is displayed at the Van Gogh Museum.
TextPaintingFin: Maalauksen The Potato Eaters on maalannut Vincent van Gogh vuonna 1885. Se on kokoa 82 kertaa 115 cm. Tämä maalaus on esillä Van Gogh Museumissa.
TextPaintingFre: The Potato Eaters a été peint par Vincent van Gogh en 1885. Il est de 82 sur 115 cm. Ce tableau est exposé au Musée van Gogh.
TextPaintingGer: The Potato Eaters wurde in 1885 von Vincent van Gogh gemalt. Das Werk ist 82 mal 115 cm. Dieses Bild ist ausgestellt im Van Gogh Museum.
TextPaintingHeb: אוכלי הבולבוסים היא תמונה של וינסנט ואן גוך משנת 1885. היא בגודל 82 על 115 סנטימטר. יצירה זו מוצגת במוזיאון ואן גוך.
TextPaintingIta: The Potato Eaters è dipinto da Vincent van Gogh in 1885. Misura di 82 su 115 cm. Questo dipinto è esposto al Van Gogh Museum.
TextPaintingNor: The Potato Eaters ble malt av Vincent van Gogh i 1885. Det er 82 ganger 115 cm. Denne malerien er utstillt på Van Gogh Museum.
TextPaintingRon: The Potato Eaters este pictat de către Vincent van Gogh în 1885. Este din 82 pe 115 cm. Acest tablou este expus în Van Gogh Museum.
TextPaintingRus: The Potato Eaters нарисовалося Vincent van Gogh в 1885 году. Она с размером 82 см на 115 см. Эта картина видится в Музей Винсента ван Гога.
TextPaintingSpa: The Potato Eaters está pintado por Vincent van Gogh en 1885. Mide 82 por 115 cm. Esta pintura está expuesta en el Museo van Gogh.
TextPaintingSwe: The Potato Eaters målades av Vincent van Gogh år 1885. Den är 82 gånger 115 cm. Den här målningen är utställd på Van Gogh-museet.

TextPaintingBul: Portrait of Alof de Wignacourt е нарисувана от Caravaggio през 1607 година. Тя е с размер 195 см на 134 см. Този експонат е изложен в Лувър.
TextPaintingCat: Portrait of Alof de Wignacourt està pintat per Caravaggio en 1607. Ell és de 195 sobre 134 cm. Aquesta pintura està exposada al Museu del Louvre.
TextPaintingDan: Portrait of Alof de Wignacourt blev malet af Caravaggio i 1607. Det er 195 ganger 134 cm. Dette maleri er udstillet på Louvre.
TextPaintingDut: Portrait of Alof de Wignacourt werd in 1607 door Caravaggio geschilderd. Het werk is 195 bij 134 cm. Dit schilderij wordt in Musée du Louvre getoond.
TextPaintingEng: Portrait of Alof de Wignacourt was painted by Caravaggio in 1607. It measures 195 by 134 cm. This painting is displayed at the Musée du Louvre.
TextPaintingFin: Maalauksen Portrait of Alof de Wignacourt on maalannut Caravaggio vuonna 1607. Se on kokoa 195 kertaa 134 cm. Tämä maalaus on esillä Louvressa.
TextPaintingFre: Portrait of Alof de Wignacourt a été peint par Caravaggio en 1607. Il est de 195 sur 134 cm. Ce tableau est exposé au Musée du Louvre.
TextPaintingGer: Portrait of Alof de Wignacourt wurde in 1607 von Caravaggio gemalt. Das Werk ist 195 mal 134 cm. Dieses Bild ist ausgestellt in der Der Louvre.
TextPaintingHeb: דיוקן של אלוף דה ווינגקורט היא תמונה של קאראווג'ו משנת 1607. היא בגודל 195 על 134 סנטימטר. יצירה זו מוצגת במוזיאון הלובר.
TextPaintingIta: Portrait of Alof de Wignacourt è dipinto da Caravaggio in 1607. Misura di 195 su 134 cm. Questo dipinto è esposto al Museo del Louvre.
TextPaintingNor: Portrait of Alof de Wignacourt ble malt av Caravaggio i 1607. Det er 195 ganger 134 cm. Denne malerien er utstillt på Musée du Louvre.
TextPaintingRon: Portrait of Alof de Wignacourt este pictat de către Caravaggio în 1607. Este din 195 pe 134 cm. Acest tablou este expus în Musée du Louvre.
TextPaintingRus: Portrait of Alof de Wignacourt нарисовалося Caravaggio в 1607 году. Она с размером 195 см на 134 см. Эта картина видится в Лувр.
TextPaintingSpa: Portrait of Alof de Wignacourt está pintado por Caravaggio en 1607. Mide 195 por 134 cm. Esta pintura está expuesta en el Museo del Louvre.
TextPaintingSwe: Portrait of Alof de Wignacourt målades av Caravaggio år 1607. Den är 195 gånger 134 cm. Den här målningen är utställd på Louvren.

Figure 1: Multilingual painting descriptions

3 The Data

The data we have been experimenting with to enable multilingual descriptions of museum objects and answering to queries over them is a subset of the Gothenburg City Museum (GCM) database,[5] and a subset of the DBpedia dataset.[6] These datasets have been linked to the CIDOC-CRM [5] and the painting ontology [7]. A description of the transformation and the processing of Gothenburg City Museum data can be found in [6, 7, 8]. The processing of the DBpedia data is described in the following section.

Both datasets have been added to the Museum of Linked Open Data. They are currently part of the MOLTO knowledge infrastructure,[7] and are accessible from the online retrieval system which is described in Section 4.

The DBpedia Data

The DBpedia data was extracted from FactForge,[8] a public service developed and maintained by Ontotext. It represents a Reason-able View of the web of data and contains a segment of the Linked Open Data (LOD).[9] The dataset contains several generic knowledge sources including: Freebase,[10] Geonames,[11] Musicbrainz,[12] Wordnet,[13] CIA Factbook,[14] Lingvoj,[15] Lexvo,[16] on which inference according to OWL-Horst is performed [28] (see Section 4).

FactForge is also supplied with a reference layer, allowing efficient access and management of the heterogeneous data, by using the light-weight upper-level ontology PROTON,[17] to which the datasets of the LOD are mapped. FactForge has 15 billion RDF triples that are available for retrieval. Our goal was to extract the paintings available in FactForge along with the semantic information that was needed for the natural language description generation, including the museum it belongs to, its author, its dimensions, its type (oil painting, portrait, etc.), and the year it was created.

The above mentioned schemata do not have a *Painting* concept, therefore when retrieving the paintings from FactForge we had to analyze the information available for several paintings, and to specify what graph pattern will reliably describe the paintings

[5] `http://stadsmuseum.goteborg.se/wps/portal/stadsm/english`
[6] `http://dbpedia.org`
[7] `http://museum.ontotext.com`
[8] `http://factforge.net`
[9] `http://linkeddata.org`
[10] `http://www.freebase.com/`
[11] `http://www.geonames.org`
[12] `http://musicbrainz.org`
[13] `http://wordnet.princeton.edu/`
[14] `https://www.cia.gov/library/publications/the-world-factbook/`
[15] `http://www.lingvoj.org/`
[16] `http://www.lexvo.org/`
[17] `http://www.ontotext.com/proton-ontology`

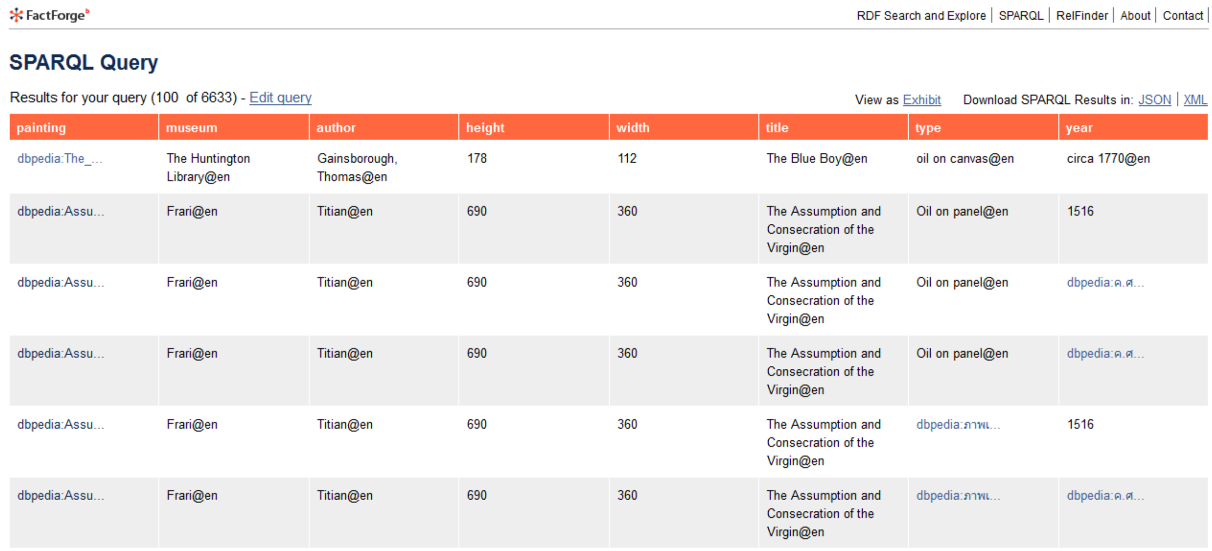
in the different datasets in FactForge. The analysis showed that the word "paintings" appeared in the comments about almost all painting objects. Thus, the formulated SPARQL query was looking for an *Artwork* concept in general, based on the Freebase predicate *visual_art.visual_artist.artworks*, and filtered out the paintings based on the mention of the word *painting* in the object of the predicate *rdfs:comment*. The SPARQL query used to retrieved the data is given bellow.

```
PREFIX fb: <http://rdf.freebase.com/ns/>
PREFIX dbpedia: <http://dbpedia.org/resource/>
PREFIX dbp-ont: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ff: <http://factforge.net/>
PREFIX painting: <http://spraakbanken.gu.se/rdf/owl/painting.owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX pext: <http://www.ontotext.com/proton/protonext#>
PREFIX ptop: <http://www.ontotext.com/proton/protontop#>
PREFIX edm: <http://www.europeana.eu/schemas/edm/>
PREFIX ore: <http://www.openarchives.org/ore/terms/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX dcterms: <http://purl.org/dc/terms/>

SELECT DISTINCT ?painting ?museum ?author ?height ?width ?title ?type ?year
WHERE {
        ?author1  fb:visual_art.visual_artist.artworks ?painting .
        ?author1 ff:preferredLabel ?author .
        ?painting ptop:isOwnedBy ?owner .
        ?owner   ff:preferredLabel ?museum .
        ?painting rdfs:comment ?comment .
        FILTER (regex(?comment, "painting"))
        ?painting dbp-prop:height ?height ;
                  dbp-prop:width ?width ;
                  dbp-prop:title ?title ;
                  dbp-prop:type ?type ;
                  dbp-prop:year ?year ;
  }
```

With this query we retrieved over 44 thousand painting entities from the entire FactForge dataset. An extract of this data is shown in Figure 2.

The Forest platform which provides the Graphical User Interface (GUI) for both FactForge and MOLTO knowledge infrastructure allows the data to be presented in the user interface, but also to be exported as JSON (JavaScript Object Notation) or XML (Extensible Markup Language format). The data exported in the XML format was used to


FactForge | RDF Search and Explore | SPARQL | RelFinder | About | Contact

SPARQL Query

Results for your query (100 of 6633) - Edit query | View as Exhibit | Download SPARQL Results in: JSON | XML

painting	museum	author	height	width	title	type	year
dbpedia:The_...	The Huntington Library@en	Gainsborough, Thomas@en	178	112	The Blue Boy@en	oil on canvas@en	circa 1770@en
dbpedia:Assu...	Frari@en	Titian@en	690	360	The Assumption and Consecration of the Virgin@en	Oil on panel@en	1516
dbpedia:Assu...	Frari@en	Titian@en	690	360	The Assumption and Consecration of the Virgin@en	Oil on panel@en	dbpedia:ค.ศ...
dbpedia:Assu...	Frari@en	Titian@en	690	360	The Assumption and Consecration of the Virgin@en	Oil on panel@en	dbpedia:ค.ศ...
dbpedia:Assu...	Frari@en	Titian@en	690	360	The Assumption and Consecration of the Virgin@en	dbpedia:ภาพเ...	1516
dbpedia:Assu...	Frari@en	Titian@en	690	360	The Assumption and Consecration of the Virgin@en	dbpedia:ภาพเ...	dbpedia:ค.ศ...

Figure 2: Data extracted from FactForge

produce the data pool to exemplify the grammar to ontology interoperability in the cultural heritage domain that is presented in this deliverable. In the following section, we provide a more detailed summary of our work with making the DBpedia data accessible through the knowledge infrastructure and available in GF.

3.1 Creation of unified reference layer

In order to make the DBpedia data accessible through the MOLTO knowledge infrastructure, it required some preprocessing, cleaning, and mapping to the Painting ontology for data consistency.[18] This unification was needed to use a consistent SPARQL queries from where NL descriptions could be generated.

Firstly, we attempted to clean data noise and results that would make a single painting reappear in the query results. Then, we transformed year and size strings into only numbers. This was necessary because some year strings contained a mixture of literal and numerical data, containing words with meanings such as *around the year* and *approximately*. For each painter, museum and painting we had a single representation in the data. We used a unified function that truncated Uniform Resource Identifiers (URIs) to unique identifiers (IDs). For example, <http://dbpedia.org/resource/A_Burial_At_Ornans> was truncated to *A_Burial_At_Ornans*, Salvador Dalí became *Salvador_Dal_*. This ID was searched in the rest of the data, to find unique paintings and unify them under the Painting ontology. For different URIs pointing to the same painting, we used the *owl:sameAs* predicate to keep the data linked in the other graphs in the LOD cloud. Example follows for A_Burial_At_Ornans.

[18] svn://molto-project.eu/wp8/d8.3/painting.owl

Table 2: Internal data format

```
entry {
    ident = "GSM940059Obj",
    title = return "Sigrid Heurlin",
    painter = return "Saga Wallin",
    ptype = ["oil painting", "portrait"],
    colour = [],
    size = return "70 545",
    material = return "wood",
    year = return "1960",
    museum = return "Gothenburg City Museum"
}
```

```
<http://spraakbanken.gu.se/rdf/owl/painting.owl#A_Burial_At_Ornans>
owl:sameAs <http://dbpedia.org/resource/A_Burial_At_Ornans> .
<http://spraakbanken.gu.se/rdf/owl/painting.owl#A_Burial_At_Ornans>
owl:sameAs <http://mpii.de/yago/resource/A_Burial_At_Ornans> .
<http://spraakbanken.gu.se/rdf/owl/painting.owl#A_Burial_At_Ornans>
owl:sameAs
<http://www4.wiwiss.fu-berlin.de/flickrwrappr/photos/A_Burial_At_Ornans> .
```

A corresponding representations of titles in the abstract syntax is the following (see also Section 2.2):

```
TA_Burial_At_Ornans : Title ;
```

The concrete syntaxes gave the RDF labels in different languges as representation of the painting. Materials and painting types were extracted from painting's text description and normalized to the existing grammar types. Painters' names were reverted; starting with first- and ending with family- name.

3.2 The GF data

The same data that was retrieved from FactForge and the GCM that has been made available through OWLIM (Section 4), has been made available in GF through an internal database from where we automatically created the grammar (Section 2.2). An example of how this data was presented locally in a Haskell database is given in Table 3.2.

There are 15350 entities represented in the above form in the GF database. These correspond to approximately 153500 triples in the painting datasets, which are available through the MOLTO knowledge infrastructure. The GF database covers 662 titles of paintings, 116 painter-names and 104 museum-names.

Table 3: Translated museum names from Wikipedia

Bulgarian	26
Catalan	63
Danish	33
Dutch	81
Finnish	40
French	94
Hebrew	46
Italian	94
German	99
Norwegian	50
Romanian	27
Russian	87
Spanish	89
Swedish	58

The pre-processing of the data required cleaning of redundant identifiers, elimination of entries that contained Chinese and other unsupported languages, replacement of dimension triples with one string, and replacement of empty painter/artist names with the string *unknown*.

The strings assigned to paining titles, painters and museum names are by default the original strings as they appear in the data. However, because without translations of the name entities the results can become artificial and for some languages ungrammatical, we run a script that translates the museum name-entities from Wikipedia automatically. In the cases where no translation was found, the original string, as it appears in the data was used. From Wikipedia we extracted entity pairs. This was done by curling for Web pages where a museum name appears and extracting translations from them. We first extracted all the retrieved entities for each language. The lists of extracted pairs were further reduced to remove duplicated and ambiguous entries. This process resulted in *lexPairs-LANG* file for each language. When the grammar data *DataGrammarLANG* is created, translations of museum-names are extracted from these files. Unfortunately, the amount of the translated entities was not equal for all languages. The distribution of the translated terms is given in Table 3

3.3 Challenges

The majority of the challenges in the production of the paintings data pool stemmed from the very nature of Linked Open Data. The data in the LOD cloud are notoriously noisy and inconsistent. The multilingual labels from the FactForge datasets and more precisely from DBpedia, are not always available in all supported languages, and one can discover mistakes in them. Another problem was that not all paintings or art objects are uniformly described with the same set of characteristics. This resulted in the lack of some information that we needed for the proper generation of the natural language descriptions in the answers. For instance, some paintings were missing a title or a painter name. As the GF grammars required this information in order to generate a description, we replaced titles with id numbers and empty painter names with the string *unknown* or removed the entry. Moreover, some of the paintings appeared twice in the result set retrieved by the FactForge SPARQL query. This occurred because some of the predicates representing them were presented with different strings and triggered two RDF triples. These repetitions were identified and removed.

To summarize, although DBpedia in its large pool of data provides access to multilingual content, it is inconsistent. Many of the entries it contains are missing translations. There is a mixture of numeric and string literals. There are many duplications, most of them occur because the same ID appears in different languages. The content of the data is verbose, for example place-names and museum-names are represented with one string, for example: "Rijksmuseum, Amsterdam", instead of two different strings linked by two separate concepts, i.e. *Museum* and *Place*. This kind of inconsistent data representation made the translation of entries harder because there was no match of those strings in the Wikipedia pages.

4 The retrieval system

The MOLTO cultural heritage retrieval system prototype is an overlay of the molto-web generic project,[19] that almost overlaps with the Knowledge Representation Infrastructure (KRI) prototype developed in WP4.[20] Another overlay is molto-patents prototype.[21]

While researching for paintings data for the prototype, we found it necessary to experiment with federated SPARQL queries, that retrieve data from remote SPARQL end points. Hence, we migrated OWLIM 5,[22] and Forest 1.4.[23] As a result, federated SPARQL queries were enabled and allowed exploring different remote or local sources of data from DBpedia, GIM, and GSM.

OWLIM supports OWL Horst which is an extension of RDFs. It is based on ter Horst, where he defines RDFs extensions toward rule support, describing a dialect of OWL, which makes use of RDF rule entailment (R-entailment) [1]. Each rule has a set of premises, which conjunctively defines the body of the rule. The premises are sequences of RDF statements, where variables can take any of the three positions. The head of the rule also has one of

[19] `svn://molto-project.eu/wp4/projects/molto-web`

[20] `http://molto.ontotext.com/`

[21] `http://molto-patents.ontotext.com/` developed in WP7.

[22] OWLIM (`http://www.ontotext.com/owlim`) is a commercial RDF database management system, developed by Ontotext.

[23] Forest is a Web-based framework for management of RDF datastores and semantically annotated documents, internal product of Ontotext.

more consequences, also in the form of RDF statements binding variables from the premises. Thus, OWL Horst language has the following characteristics which are supported by the retrieval system:

1. It is a proper (backward compatible) extension of RDFs, which allows to use the classes of RDFs with OWL Horst reasoning

2. It allows rule extensions without Description Logic (DL) related constraints, because it is based on R-entailment formalism

3. Its complexity allows greater scalability compared to other approaches combining DL ontologies with rules.

4.1 System workflow

The architecture of the system is shown on Figure 3. The retrieval workflow is described in the current section.

First, the GUI suggests few possible queries to the user in their selected languge. Autocomplete is available for all possible queries, including paintings, authors, museums, materials, colours, painting types, etc. The user creates and submits a query. It is passed to the Java backend, where a GF process is started (or already runs). It is a GF process for the *QueryPainting.pgf* executable, which comprises of all query grammars, including the SPARQL one. Consequently, through GF, the natural languge is translated to SPARQL. An example of a call to GF is as follows:

```
14:39:11.425 [qtp1392834697-37] INFO  c.o.m.r.ResultMappingMuseum
 - write to GF:
 p -lang=QueryPaintingSpa "mostra toda la informaci\'{o}n sobre todas las
 pinturas al \'{o}leo" | l -lang=QueryPaintingSPARQL
```

The result is a SPARQL query that is executed over OWLIM from where a set of triples is returned. For some queries that require a single word answer, text generation is not supported yet. An example for such a scenario can be seen on Figures 7 and 8.

A more interesting use-case is when we request painting descriptions, for example with a query like "show me everything about all portraits". Then, the resulting SPARQL has the following form:

```
PREFIX painting: <http://spraakbanken.gu.se/rdf/owl/painting.owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT distinct ?painting ?title ?material ?author ?year ?length ?height
       ?color ?museum
WHERE   { ?painting rdf:type painting:OilPainting ;
rdfs:label ?title ;
```

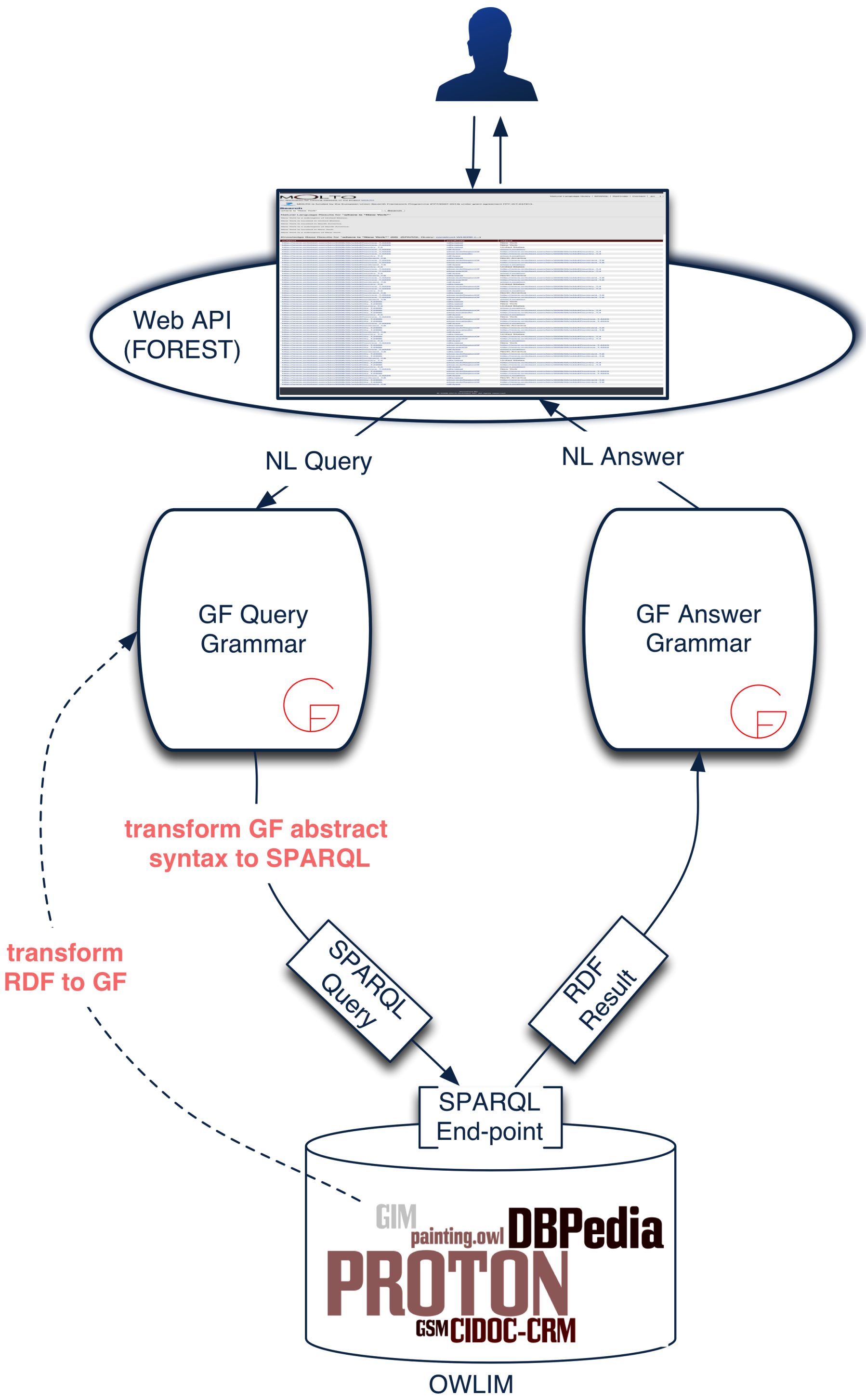


Figure 3: Natural languge-GF-SPARQL-RDF-Natural languge workflow

```
painting:createdBy ?author;
painting:hasMaterial ?material;
painting:hasCurrentLocation ?museum;
painting:hasCreationDate ?date;
painting:hasDimension ?dim  .
?author rdfs:label ?painter .
?date painting:toTimePeriodValue ?year . ?dim painting:lengthValue ?length ;
painting:heightValue ?height . ?museum rdfs:label ?loc .
} LIMIT 200
```

A naming convention for the query parameters is used in order to retrieve all entities from the resulting bindings. Queries that will result in a coherent description contain the following parameter names: *?painting ?title ?material ?author ?year ?length ?height ?color ?museum*. Mapping all data to one multi-layered ontology [2], in our case the Painting ontology, allows this consistency. Each returned binding of the above parameters is turned from an URI string value to a parameter in the GF abstract syntax. For example:

```
?museum = <http://spraakbanken.gu.se/rdf/owl/painting.owl#
                             Derby_Museum_and_Art_Gallery>
```

is translated to

```
MDerby_Museum_and_Art_Gallery
```

Optimally, we use the same code for stripping URI to GF entity name (painter, painting, museum) in the call to and from GF using the data from the RDF results. We perform this similar conversion for the bindings whose data is an URI containing the strings: ?title ?material ?author ?color ?museum. Next, we form an abstract syntax tree representation that is send to GF for generating a painting description. For example, to get a Spanish translation a request would look like:

```
l -unlextext -lang="TextPaintingSpa" DPainting
(PTitle TA_Philosopher_Lecturing_on_the_Orrery) PJoseph_Wright
NoPaintingType NoColours (MkSize (SIntInt 203 147)) (MkMaterial MCanvas)
(MkYear (YInt 1990)) (MkMuseum MDerby_Museum_and_Art_Gallery)
```

We have a second GF process that we run in the backend. This process runs the *TextPainting.pgf* executable and returns painting descriptions. This way, we describe every painting with a single call to the *DPainting* function as described in Section 2.5. Examples of the retrieved entities and their natural language descriptions are shown in Figure 4.

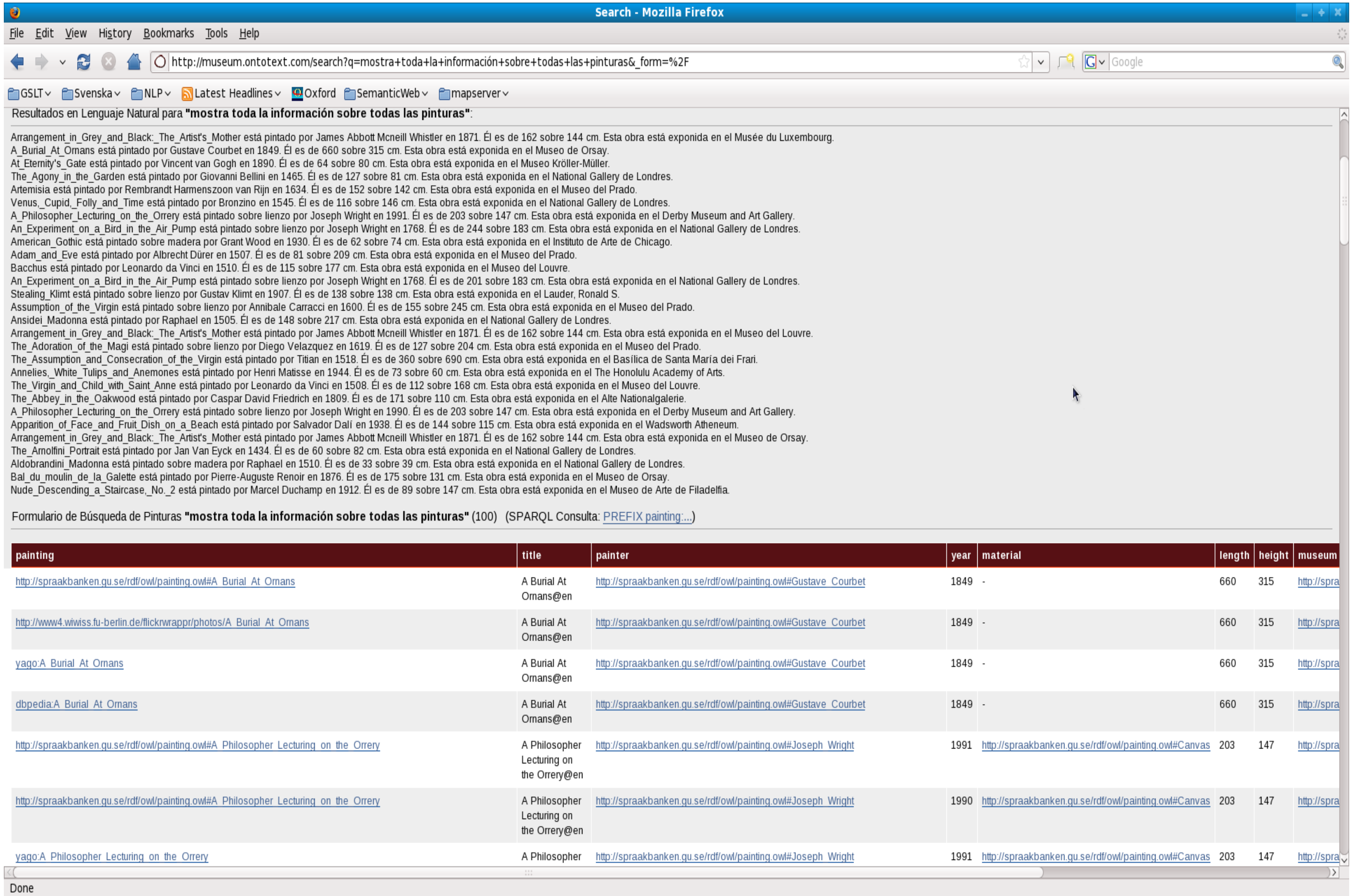


Figure 4: Results in Spanish to - "mostra toda la información sobre todas las pinturas al óleo" ("show me everything about all oil paintings")

4.2 Multilingual interface

It was a challenge to process NL queries and answers in the form of coherent painting descriptions in 15 natural languges and also SPARQL. We experienced the usual encoding issues, such as conversion to utf-8, both on frontend and backend side, for example in tomcat settings,[24] and data preprocessing and processing. We also had to deal with some NL-to-SPARQL generation obstacles that were due to the novelty nature of the SPARQL generation approach. For example, it was very important to concatenate strings in a certain order and with the right syntax, i.e. end of a triple with a full stop, separate a sequence of categories with semi columns, etc.

4.2.1 NL queries user interface

The system supports queries in 12 natural languages: Bulgarian, Catalan, Dutch, English, Finnish, French, German, Italian, Romanian, Russian, Spanish and Swedish. Painting descriptions are also available in Danish, Hebrew and Norwegian. Some of the currently supported user interface (UI) sample queries are the corresponding ones of those shown in Figure 5.

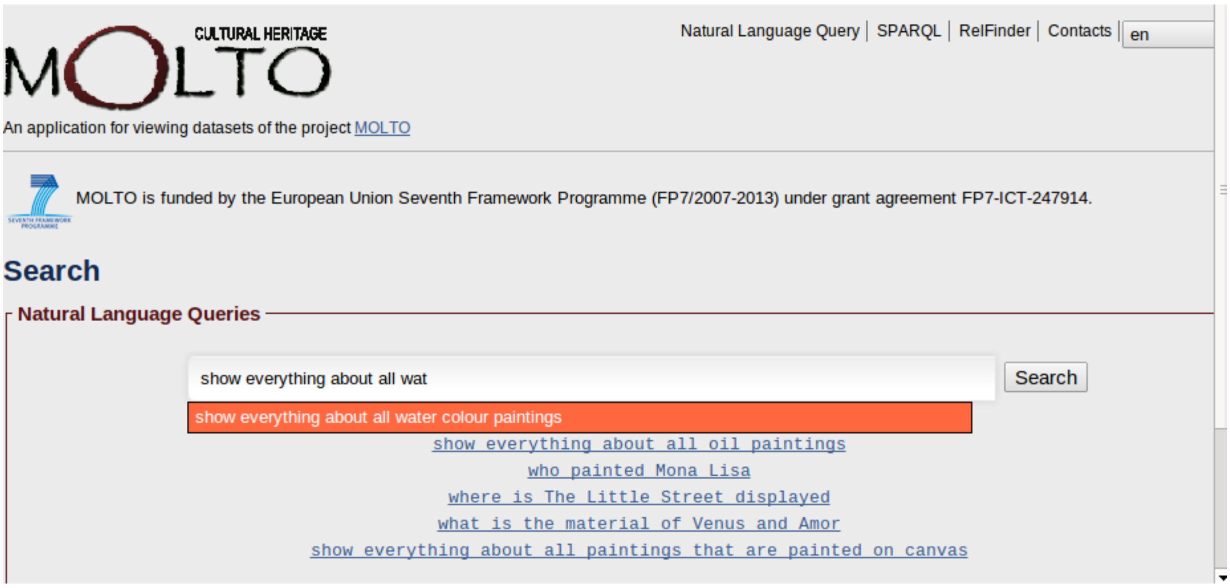


Figure 5: Example queries in English with autocomplete

Two examples of search queries and answers from the Spanish and the Bulgarian interfaces can be seen in Figures 4, 6 and 7, 8. When inspecting the Spanish answer in Figure 4, we discover that on the reply, we find both painting descriptions for each row of the result of the query and the table of semantic data results. The top seven resulting painting descriptions are listed below.

[24]http://tomcat.apache.org/

Figure 6: Example query in spanish – "mostra toda la información sobre todas las pinturas al óle"("show me everything about all oil paintings")

1. Diana_and_Actaeon está pintado sobre lienzo por Titian en 1556. Mide 202 por 185 cm. Esta pintura está exponida en el Sutherland, Francis Egerton, 7th Duke of.
2. Christ_in_the_House_of_Martha_and_Mary está pintado sobre lienzo por Diego Velazquez en 1618. Mide 104 por 63 cm. Esta pintura está exponida en el National Gallery de Londres.
3. Belisarius_Begging_for_Alms está pintado sobre lienzo por Jacques-Louis David en 1781. Mide 312 por 288 cm. Esta pintura está exponida en el Palais des Beaux-Arts de Lille.
4. A_Philosopher_Lecturing_on_the_Orrery está pintado sobre lienzo por Joseph Wright en 1991. Mide 203 por 147 cm. Esta pintura está exponida en el Derby Museum and Art Gallery.
5. The_Coronation_of_the_Virgin está pintado sobre lienzo por Diego Velazquez en 1644. Mide 124 por 176 cm. Esta pintura está exponida en el Museo del Prado.
6. An_Experiment_on_a_Bird_in_the_Air_Pump está pintado sobre lienzo por Joseph Wright en 1768. Mide 244 por 183 cm. Esta pintura está exponida en el National Gallery de Londres.

7. Bathsheba_at_Her_Bath está pintado sobre lienzo por Rembrandt Harmenszoon van Rijn en 1654. Mide 142 por 142 cm. Esta pintura está exponida en el Museo del Louvre.

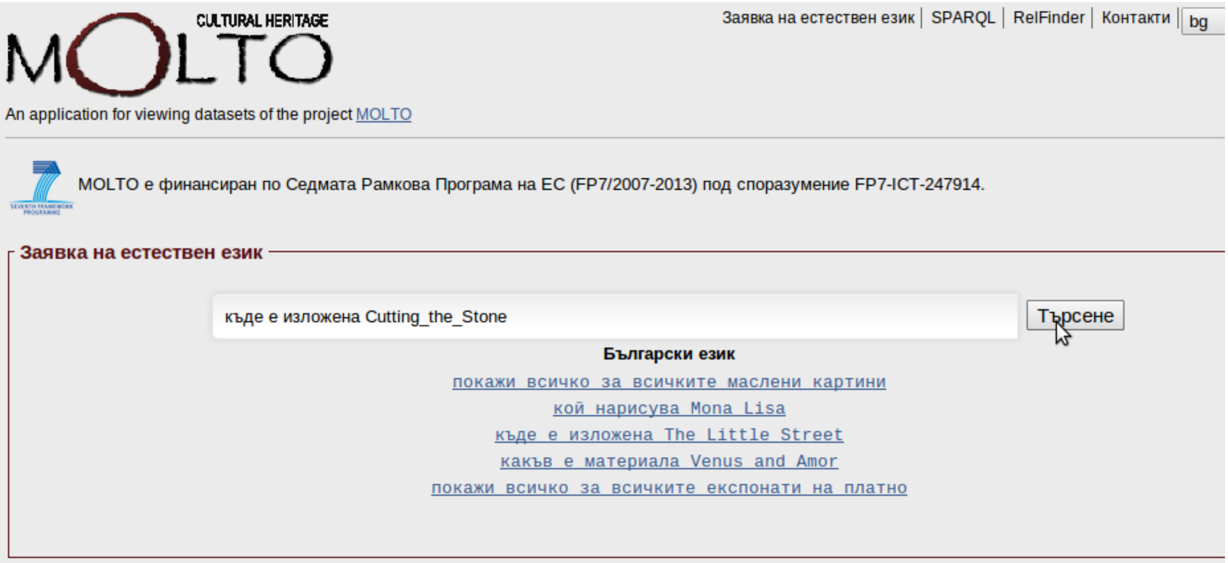


Figure 7: Example query in Bulgarian, asking on "where is Cutting_the_Stone displayed"

The answer to the query taken from the Bulgarian example, Figures 7 and 8, reveals there is only one triple returned. It can be explored further through the RDF end-point.

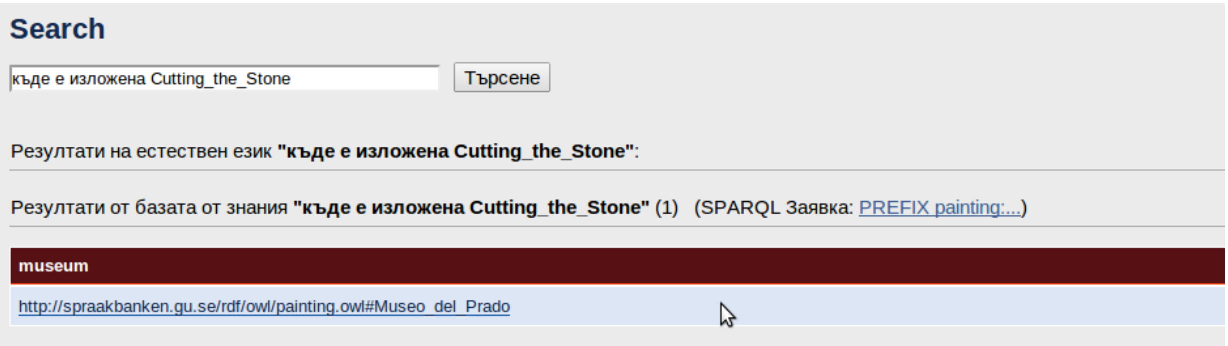


Figure 8: Answer for "where is Cutting_the_Stone displayed"

4.2.2 RDF browsing view

The prototype allows exploring RDF graphs by providing actual links to each entity in the graph. It also shows the participants in the triples where the entity is respectively one of the following: (1) object, (2) predicate, (3) object.

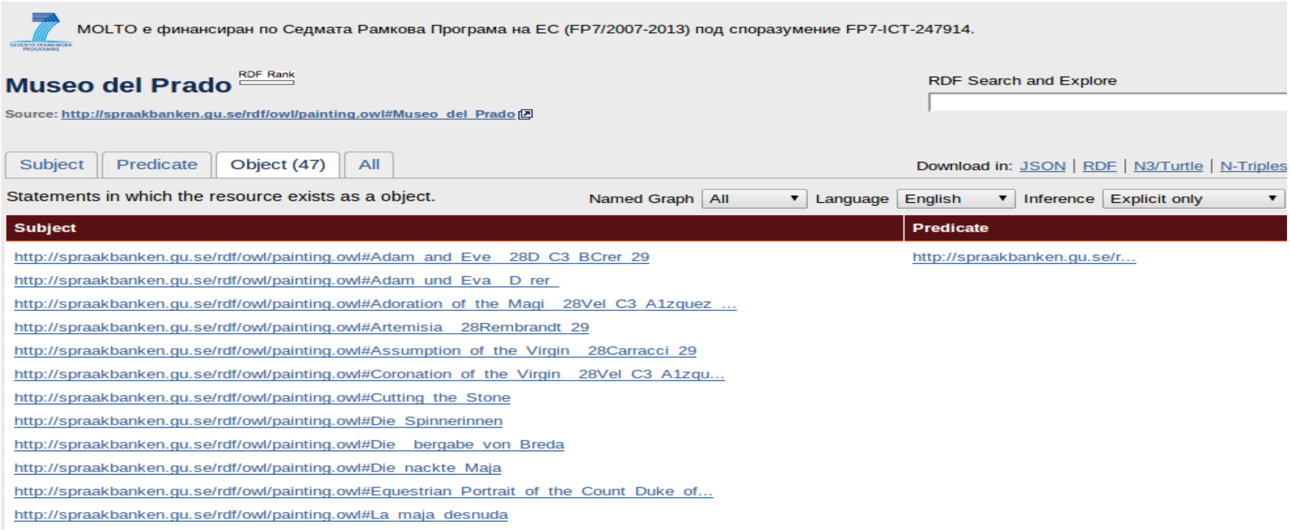


Figure 9: View of RDF facts where “Museo del Prado” is an object

In a continuation of the search of the example given in Figure 8, we can see all RDF triples, in which “Museo del Prado” is an “object” as illustrated in Figure 9. All the listed objects are related to this museum through the relating predicate *hasCurrentLocation*. We can also further explore each of them separately.

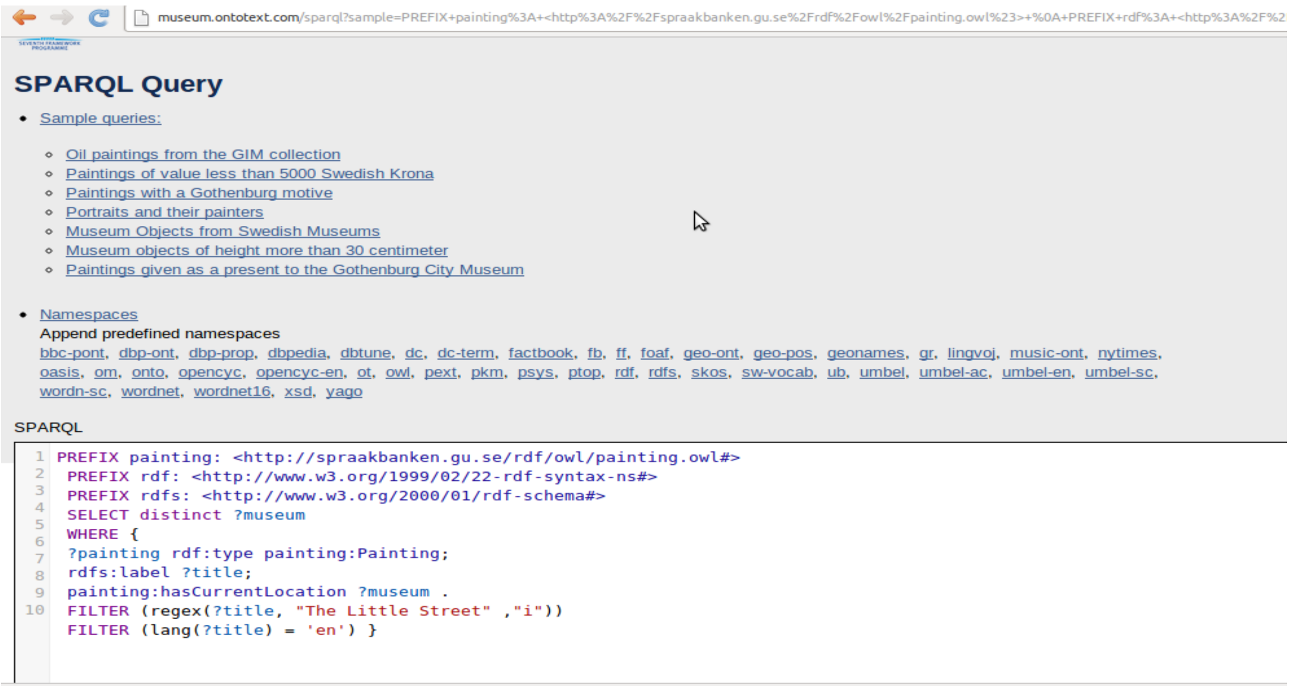


Figure 10: SPARQL view

4.2.3 SPARQL endpoint

Finally, our prototype provides a public SPARQL endpoint, against which one could explore the knowledge base with manually written SPARQL queries. In Figure 10 we see the resulting query of the natural languge question from Figure 7. It was retrieved by the link to "SPARQL Query" that is visible in Figure 8.

5 Conclusions

We present an ontology-based multilingual application grammar developed in the Grammatical Framework and a cross-language retrieval system that uses this grammar for generating museum object descriptions in the Semantic Web.

The presented application covers semantic data from the Gothenburg City Museum database and DBpedia. The grammar enables descriptions of paintings and answering to queries over them, covering 15 languages for baseline functionality.

We described as well a prototype of a cross-language retrieval and representation system that has been tested with the same data, using tools from WP4 and WP7.

References

[1] Baader, F., Calvanese, D., McGuinness, D. L., Nardi, D., and Patel-Schneider, P. F., Eds. *The Description Logic Handbook: Theory, Implementation, and Applications* (2003), Cambridge University Press.

[2] Bouayad-Agha, N., Casamayor, G., Mille, S., Rospocher, M., Saggion, H., Serafini, L., and Wanner, L. From Ontology to NL: Generation of multilingual user-oriented environmental reports. *Lecture Notes in Computer Science 7337* (2012).

[3] Brickley, D., and Guha, R. *RDF Vocabulary Description Language 1.0: RDF Schema*. W3C, 2004. `http://www.w3.org/TR/rdf-schema/`.

[4] Cimiano, P., Buitelaar, P., McCrae, J., and Sintek, M. LexInfo: A declarative model for the lexicon-ontology interface. *Web Semantics 9* (March 2011), 29–51.

[5] Crofts, N., Doerr, M., Gill, T., Stead, S., and Stiff, M. *Definition of the CIDOC Conceptual Reference Model*, 2008.

[6] Damova, M. *Data Models and Alignment*, May 2011. Deliverable 4.2. MOLTO FP7-ICT-247914.

[7] Damova, M., and Dannélls, D. Reason-able View of Linked Data for cultural heritage. In *Proceedings of the third International Conference on Software, Services and Semantic Technologies (S3T)* (2011).

[8] Dannélls, D. *D.8.1 Ontology and corpus study of the cultural heritage domain*, 2011. Deliverable of EU Project MOLTO Multilingual Online Translation.

[9] Dannélls, D. *Multilingual text generation from structured formal representations*. PhD thesis, Department of Swedish, University of Gothenburg, Gothenburg, Sweden, 2012.

[10] Dannélls, D. On generating coherent multilingual descriptions of museum objects from semantic web ontologies. In *Proceedings of the Seventh International Natural Language Generation Conference (INLG 2012)* (Utica, IL, May 2012), Association for Computational Linguistics, pp. 76–84.

[11] Dannélls, D., Ranta, A., and Enache, R. *D.8.2 Multilingual grammar for museum object descriptions*, 2012. Deliverable of EU Project MOLTO Multilingual Online Translation.

[12] Dannélls, D., Damova, M., Enache, R., and Chechev, M. A Framework for Improved Access to Museum Databases in the Semantic Web. In *Recent Advances in Natural Language Processing (RANLP). Language Technologies for Digital Humanities and Cultural Heritage (LaTeCH)* (2011).

[13] Dannélls, D., Damova, M., Enache, R., and Chechev, M. Multilingual online generation from semantic web ontologies. In *Proceedings of the World Wide Web Conference (WWW2012)* (Lyon, France, 2012).

[14] Davies, R. EuropeanaLocal – its role in improving access to Europe's cultural heritage through the European digital library. In *Proceedings of IACH workshop at ECDL2009 (European Conference on Digital Libraries)* (Aarhus, September 2009).

[15] Dekkers, M., Gradmann, S., and Meghini, C. *Europeana Outline Functional Specification for development of an operational European Digital Library*, 2009. Europeana Thematic Network Deliverable 2.5.

[16] Doerr, M., Gradmann, S., Hennicke, S., Isaac, A., Meghini, C., and van de Sompel, H. The Europeana Data Model (EDM). In *WORLD LIBRARY AND INFORMATION CONGRESS: 76TH IFLA GENERAL CONFERENCE AND ASSEMBLY 10-15 August 2010, Gothenburg, Sweden* (Aug. 2010).

[17] Ferré, S. SQUALL: A controlled natural language for querying and updating RDF graphs. In *CNL* (2012), pp. 11–25.

[18] Garlik, S. H., and Andy, S. *SPARQL 1.1 Query Language*, March 2013. W3C Recommendation.

[19] Group, W. O. W. *OWL Web Ontology Language Overview*, December 2012. W3C Recommendation.

[20] Hirst, G. *Ontology and the lexicon*. Springer Verlag, Berlin Germany, 2004.

[21] Lassila, O., and Swick, R. R. *Resource Description Framework (RDF). Model and Syntax Specification*, 2 1999.

[22] McCrae, J., de Cea, G. A., Buitelaar, P., Cimiano, P., Declerck, T., Gómez-Pérez, A., Gracia, J., Hollink, L., Montiel-Ponsoda, E., Spohr, D., and Wunner, T. Interchanging lexical resources on the Semantic Web. *Language Resources and Evaluation 46*, 4 (2012), 701–719.

[23] Ore, C.-E. S. The norwegian museum project, access to and interconnection between various resources of cultural and natural history. In *European Conference on Research and Advanced Technology for Digital Libraries ECDL* (2001).

[24] Ranta, A. Grammatical Framework, a type-theoretical grammar formalism. *Journal of Functional Programming 14*, 2 (2004), 145–189.

[25] Ranta, A. The GF resource grammar library. *The on-line journal Linguistics in Language Technology (LiLT) 2*, 2 (2009).

[26] Ranta, A. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Studies in Computational Linguistics. CSLI, Stanford, 2011.

[27] Schalley, A. C., and Zaefferer, D., Eds. *Ontolinguistics*. Trends in Linguistics. Studies and Monographs. Mouton de Gruyter, Berlin, 2007.

[28] ter Horst, H. J. Combining RDF and Part of OWL with Rules: Semantics, Decidability, Complexity. In *Proceedings of The Semantic Web ISWC* (Heidelberg, 2005), vol. 3729 of *LNCS*, Springer Berlin, pp. 668–684.

[29] Unger, C., and Cimiano, P. Pythia: Compositional meaning construction for ontology-based question answering on the Semantic Web. In *Natural Language Processing and Information Systems - 16th International Conference on Applications of Natural Language to Information Systems, NLDB 2011, Alicante, Spain, June 28-30, 2011. Proceedings* (2011), vol. 6716 of *Lecture Notes in Computer Science*, Springer, pp. 153–160.