DELIVERABLE 8.1

Ontology and corpus study of the cultural heritage domain
<table>
<thead>
<tr>
<th><strong>Contract No.</strong></th>
<th>FP7-ICT-247914</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project full title:</strong></td>
<td>MOLTO - Multilingual Online Translation</td>
</tr>
<tr>
<td><strong>Deliverable:</strong></td>
<td>D8.1. Ontology and corpus study of the cultural heritage domain</td>
</tr>
<tr>
<td><strong>Security (distribution level):</strong></td>
<td>Public</td>
</tr>
<tr>
<td><strong>Contractual date of delivery:</strong></td>
<td>1 Sep 2011 (M18)</td>
</tr>
<tr>
<td><strong>Actual date of delivery:</strong></td>
<td>26 Sep 2011</td>
</tr>
<tr>
<td><strong>Type:</strong></td>
<td>Deliverable</td>
</tr>
<tr>
<td><strong>Status version:</strong></td>
<td>Final</td>
</tr>
<tr>
<td><strong>Author(s):</strong></td>
<td>Dana Dannéls</td>
</tr>
<tr>
<td><strong>Task responsible:</strong></td>
<td>UGOT</td>
</tr>
<tr>
<td><strong>Other contributors:</strong></td>
<td>Ramona Enache and Mariana Damova</td>
</tr>
</tbody>
</table>
1 Introduction

This document is the first deliverable corresponding to Work Package 8 (WP8) of the European Union Seventh Framework Programme MOLTO under grant agreement FP7–ICT 247914 Case study: Cultural Heritage. The duration of WP8 is 18 months starting from March 2011.

In this section we present the goals and the purpose of WP8. In the remaining sections we report on a study of existing metadata schemas adopted by museums in Sweden (Section 2). We then describe the ontology development that was based on this study (Section 3). Afterwards follows a specification of the syntactic structures and patterns for discourse generation that is a result of corpus analysis of cultural heritage texts (Section 4.1). In Section 5 we briefly describe the ontology alignment and the museum data integration into the Museum Reason-able View. We end with a description of the grammar implementation of the ontology models in GF (Section 6).

1.1 The goals of WP8

The goals of WP8 are to:

- build an ontology-based multilingual grammar for museum information for artefacts at Gothenburg City Museum (GCM) starting from the Conceptual Reference Model (CIDOC-CRM);
- cover 15 languages for baseline functionality and 5 languages with a more complete coverage;
- build a prototype of a cross-language retrieval and representation system to be tested with objects in the museum, and automatically generate Wikipedia articles for museum artefacts in 5 languages.

The workpackage is tightly related to WP2, Grammar Developer’s Tools and WP4, Knowledge Engineering, whose task is to connect the museum data and cultural heritage ontology models to the BigOWLIM triple store (Bishop et al., 2011) and make them available via SPARQL endpoint (Eric and Andy, 2008).

1.2 Description of work

The work of WP8 is started by a study of the existing categorizations and metadata schemas adopted by the museum, as well as a corpus of texts in the current documentation which describe these objects. We will transform the CIDOC-CRM into a specific ontology aligning it with the upper-level one in the base knowledge set (WP4) and modeling the museum object metadata as a domain specific knowledge base (D8.1). Through the interoperability engine from WP4 and the IDE (Integrated Development Environment) from

\[\text{http://www.w3.org/TR/rdf-sparql-query/}\]
WP2, we will semi-automatically create the translation grammar in GF (Ranta, 2004) and further extend it (D8.2). The final result will be an online system enabling museum (virtual) visitors to use their language of preference to search for artefacts through semantic (structured) and natural language queries and examine information about them. We will also automatically generate a set of articles in the Wikipedia format describing museum artefacts in the 5 languages with extensive grammar coverage (D8.3).

This deliverable provides a study of the existing categorizations and metadata schemas of Carlotta information system that is adopted by the Gothenburg City Museum and many other museums in Sweden including: the Museum of World Culture,2 the Museum of Ethnography,3 the Museum of Mediterranean and Near Eastern Antiquities,4 and the Museum of East Asia.5 It describes the text collection of written object descriptions, the specific ontology model we have been developing, the syntactic structures and the discourse patterns that will be used in this workpackage to automatically generate Wikipedia articles for museum artefacts in 5 languages. This deliverable also presents ongoing work on aligning the ontology models and making the museum data accessible through SPARQL queries. It describes the grammar implementation for generating the information that is available in the Gothenburg City Museum database.

2 Study of museum metadata

The aim of the museum database analysis is to provide us with basic information on the metadata that is unique to artifact-based collections, with a view of producing an ontology that will capture the relation between natural language constructs and museum database structures. We hope the data study presented in this paper and the design of the ontology that follows from this study will help museums in Sweden to move from “local museum databases” towards Semantic Web standards by standardizing their collections and making them available for Semantic Web applications.

Chenhall and Vance (2010) have analyzed museum records in diverse museum databases. They found that the complexity of the structure of the information in the majority of the world’s museums lies somewhere between geology museums and art and photography museums. While the former requires simple terminology, the latter requires advanced classification categories system. They found that the emphasis placed upon objects in art museums, history museums, photography museums or natural history museums is quite different because of the functional utility of the objects, the background of what is known about the society that made and used them and the difference in what is most important to highlight. Their study revealed that despite the differences in the way information is stored, the metadata that is used to record object values is common for the majority of museums. Thus, understanding the structured information of existing systems will make

---

2http://www.varldskulturmuseet.se/
3http://www.etnografiska.se/
4http://collections.smvk.se/pls/mm/rigby.Welcome
5http://www.ostasiatiska.se/
it easier to implement a standard for diverse museum resources.

According to Chenhall and Vance’s study, the most common museum artifact attributes are:

- Artifact Number
- Object Name
- Provenience – where did the object originate
- Price/Value
- Artist/Creatr
- Location within the museum

In the light of this study, we examined the database of the GCM and the data values we are trying to control. As a result, we designed an ontology (see Section 3) which provides one alternative for instantiating museum data values.

2.1 The Gothenburg City Museum database

Relation databases (RDB) that have traditionally functioned as internal documents tailored to the needs of museum registers, conservators, curators and scientists are the primary means in which museums document their collections. A relation database provides record attributes such as object number, name, donor, condition, material, etc. to assist those who use museum collections to locate physical objects in the museum.

We have acquired a relational database from the Gothenburg City Museum. The museum relational database is centered around the museum’s collections. It consists of tables, each table corresponds to a specific collection. Tables contain record fields with information about the available data for each object in the collection. We have been experimenting with two of the database tables which correspond to two of the museum collections, i.e. Gothenburg Industry Museum (GIM) and Gothenburg City Museum (GSM). There are 8900 museum objects described in these database tables. Each of the database tables contains 39 record attributes for describing each object. Table 1 depicts 20 of these attributes and their data values translated to English. The complete set of record fields as they appear in the database is available in Appendix A.

The Gothenburg City Museum uses the SWETERM classification system (Landahl, 1993) to document information about physical artwork objects. The Swedish classification system, although adapted after the International Committee for Documentation (CIDOC) standard, contains many concepts that are not directly available in CIDOC-CRM. For instance, there are no corresponding concepts for representing the record fields: Value, Search work, and Method of acquisition.
An elaborated metadata schema that captures record fields that are missing in existing standards was developed by the Swedish Open Cultural Heritage (SOCH). SOCH is a web service used to search and fetch data from any organization that holds information related to the Swedish cultural heritage. The idea behind SOCH is to harvest any data format and structure that is used in the museum sector in Sweden and map it into SOCH’s categorization structure.

The metadata provided by SOCH helps to intermediate data between museums in Sweden and Semantic Web applications. Although the schema satisfies domain-specific requirements, it provides little support for expressing semantic knowledge. Consequently, we decided to develop an OWL ontology that combines the SOCH metadata and the CIDOC-CRM metadata standard for transforming the museum record fields into existing standards.

### 2.2 Transformation of museum data

The development of metadata standards for transforming museum data to enable distributed construction of database resources is not new (Bearman, 1995). Some experiments with cultural heritage data transformations (Cameron, 2005; Malmsten, 2008; Bryne, 2009)
have showed that the process of transforming data from documentation tools to effective knowledge representation systems that are available on the Semantic Web requires an uniform, pre-defined meta model. Especially in the field of cultural heritage, and in particular museum databases, there has been recent initiatives for developing museum metadata that allows interoperability with already existing standards. The Museum Data Exchange 2010 project has developed a metadata publishing tool to extract data in XML.\footnote{http://www.oclc.org/research/activities/museumdata/default.htm} Brugman et al. (2008) have developed an Annotation Meta Model providing a way of defining annotation values and anchors in an annotation for multimedia resources. Other related initiatives in the context of Semantic Web are: the Amsterdam Museum Linked Open Data project,\footnote{http://www.europeana.eu/portal/thoughtlab_linkedopendata.html} aiming at producing Linked Data within the Europeana data model (Dekkers et al., 2009; Haslhofer and Isaac 2011) and the National Database Project of Norwegian University Museums (Ore, 2001) who developed an unified interface for digitalizing cultural material.\footnote{http://www.muspro.uio.no/engelsk-omM.shtml}

Although there exist automatic conversion tools to transform databases to Semantic Web standards, such as: R2O (Barrasa et al., 2004), Dartgrid (Wu et al., 2006), D2RQ (Bizer and Cyganiak, 2007) and DB2OWL (Cullot et al., 2007), the complexity of developing a standard that captures the complete information of museum artifacts have led to emphasis on manual development of these specific schemas.

\section{Modeling the Painting ontology}

The domain and application specific ontology that we describe in this section represents an explicit specification of the conceptualization of paintings and painting collections. The conceptual organization of the ontology is intended to allow recording all information that is available in the history museum databases about painting objects. The objective of the model development is that physical objects in museums are sufficiently uniform that it is possible to develop standardized systems for the storage of the events surrounding objects and their representations in written records.

The painting ontology provides a model to represent and use painting objects in the framework of the Semantic Web. The purpose of the ontology is to establish a relationship between museum objects on the basis of extended museum database fields and Semantic Web ontologies. The top-level class hierarchy corresponding to basic concepts of the Painting ontology is shown in Figure 1. The painting ontology contains 197 classes, 24 stems from CRM, 15 equivalent to SOCH, and 45 equivalent to SUMO concepts, there are 107 properties, of which 17 are subproperties of the CRM properties.\footnote{The painting ontology:\ http://spraakdata.gu.se/svedd/ontologies/painting/painting.owl}

In this section we describe the ontology requirements, the development methodology, and the remaining schemas the ontology is merged with. We also illustrate how the ontology concepts are integrated with metadata terms from different models to enable interoperability and extensibility.
Figure 1: Top-level class hierarchy of the Painting ontology.
3.1 Ontology requirements

In order to delimit the scope and elicit the required features for the painting ontology, we have defined several use cases. Below we present a summary of the possible users and their needs being of relevance in the design of the painting ontology.

- The ontology should provide authors who are developing applications for information retrieval, information integration, knowledge management, e-learning etc. an off-the-shelf ontology that can be used for storage, exchange of data and reasoning;
- It will help ontologists to map, marge, match concepts and data with multiple ontologies;
- It should be flexible enough to allow grammarians to choose any content from it and realize it in natural language;
- We will provide an open content allowing any author to use and edit the ontology;
- The ontology will accommodate the many different web-browsers and ontology editors that are widely used by engineers and human editors to manipulate, edit and/or automatically extract cultural heritage content;
- It will help terminologists to develop class taxonomies and populate ontologies accordingly;
- The ontology will allow computer users and on-line museum visitors to navigate and browse for cultural heritage information using their own language.

Defining how classification schemes that describe museum collections are to be handled and standardized is an on-going integral transformative process that will allow to embrace the above user needs.

3.2 The ontology development methodology

The painting ontology is an application ontology. It is aligned to the domain specific CIDOC-CRM ontology which formalizes shared concepts within the CH knowledge domain. While the domain ontology supports tasks such as semantically annotating service capabilities and contents, the application ontology is designed to support integration and interoperability with other schemata. For example, the abstract classes delivered by the CIDOC-CRM cannot directly be used to document the value and current state of a specific artwork such as painting. But the generic nature of the model’s classes and properties allows easy integration with additional ontology models that do support documentation of this kind of information about museum entities.

The ontology design follows the central elements in museum database including: people, places, things, and events. The main reference model of the painting ontology is the OWL 2
implementation of the CRM.\textsuperscript{11} The additional models that are currently integrated in the ontology are: SOCH, Merge and Mid-Level-Ontology from SUMO.\textsuperscript{12} The painting ontology was constructed manually using the Protégé editing tool.\textsuperscript{13}

Integration of the ontology concepts are accomplished by using the OWL construct: \textit{intersectionOf} as specified below. This is also illustrated in Figure 2.

\begin{verbatim}
<owl:Class rdf:about="&painting;Painting">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <rdf:Description rdf:about="&ksasok;item"/>
      <rdf:Description rdf:about="&milo;PaintedPicture"/>
    </owl:intersectionOf>
  </owl:Class>
</owl:Class>

<owl:Class rdf:about="&painting;Painting">
  <owl:equivalentClass>
    <owl:intersectionOf rdf:parseType="Collection">
      <rdf:Description rdf:about="&ksasok;item"/>
      <rdf:Description rdf:about="&milo;PaintedPicture"/>
    </owl:intersectionOf>
  </owl:Class>
</owl:Class>

The schemas that are stated in the above example are denoted with the following prefixes: painting ontology (&painting), SOCH (&ksasok), Mid-Level-Ontology (&milo) and CIDOC-CRM ontology (&core). In this example, the class \textit{Painting} is defined in the painting ontology as a subclass of \textit{E22_Man-Made_Object} class from the CIDOC-CRM ontology and is an intersection of two classes, i.e. \textit{item} from the SOCH schema and \textit{PaintedPicture} from the Mid-Level Ontology. The example above specifies that the members of the class \textit{Painting} are those that exist as members of both \textit{item}, and \textit{PaintedPicture}, in other words, a painting object is a painted picture item.

![Figure 2: Integration of the painting ontology concepts with other models](image)

Discovering correspondences between terms in different ontologies is a crucial process. Although there exists a number of tools supporting semi-automatic ontology integration (Chalupsky, 2000; Dou et al., 2002), strategies for dealing with heterogeneous concepts

\textsuperscript{11}http://purl.org/NET/cidoc-crm/core
\textsuperscript{12}http://www.ontologyportal.org/
\textsuperscript{13}http://protege.stanford.edu/
are not well developed, therefore integration processes are still largely conducted by hand. Below we specify the ontology models that are integrated with the painting ontology.

3.2.1 The CIDOC-CRM

The International Committee for Documentation Conceptual Reference Model (CIDOC CRM) that was accepted by ISO in 2006 as ISO21127 (Crofts et al., 2008), is one of the most widely used standards that has been developed to facilitate the integration, mediation and interchange of heterogeneous cultural heritage information. The CIDOC-CRM class hierarchy consisting of about 90 classes and 148 properties (Doerr, Ore and Stead 2007; Doerr 2005).

The CIDOC CRM, independent of any specific application, is primarily defined as an interchange model for integrating information in the cultural heritage sector.

3.2.2 The Swedish Open Cultural Heritage (SOCH)

SOCH functions as an exchange/aggregator where data from many local databases are made searchable and visible to the public and to the research community. It provides about 100 metadata elements that are used by the service.

More than 20 museums in Sweden have made their collections available through the SOCH service. By integrating the SOCH data schema in the ontological framework we gain automatic access to these collections in a semantically interoperable way.

3.2.3 The Suggested Upper Merged Ontology (SUMO)

The Suggested Upper Merged Ontology (SUMO) is a general upper-level ontology which has been mapped with WordNet (Pease and Fellbaum, 2009; Niles and Pease 2003). It is based on the syntax and the semantics of the Knowledge Interchange Format (KIF).\(^\text{14}\)

The museum database analysis showed there is a considerable overlap with similar concepts from SUMO which links the domain specific concepts with domain independent information such as information about buildings, colors, material etc.

Linking the ontology concepts to SUMO concepts provides us not only extended knowledge about the ‘world’ but also access to English lexical units.

4 Corpus study

This section describes the corpus analysis that was carried out as a part of the requirements for the language generation component.

The purpose of the text analysis was to identify and describe the syntactic constructions that are necessary for generating museum object descriptions. Based on these syntactic constructions, a grammar will be implemented to automatically generate these constructions from the ontology.

\(^{14}\text{http://www.csee.umbc.edu/kse/kif/}\)
4.1 Languages and corpora

To identify the syntactic constructions that characterize the CH domain, we have collected parallel texts from Wikipedia in two languages: English and Swedish. In total, we analyzed 40 parallel texts that are available under the category *Painting*. Additionally, we performed a selection of files containing object descriptions from digital libraries that are available through museums online databases. The majority of the Swedish descriptions were taken from the World Culture Museum,\(^\text{15}\) the majority of the English descriptions were collected from the Met Museum.\(^\text{16}\) Table 2 provides a data statistics from these two text collections.

<table>
<thead>
<tr>
<th></th>
<th>English</th>
<th>Swedish</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of tokens</td>
<td>42792</td>
<td>27142</td>
</tr>
<tr>
<td>Number of sentences</td>
<td>1877</td>
<td>2214</td>
</tr>
<tr>
<td>Avg. sentence length</td>
<td>24 (tokens)</td>
<td>13 (tokens)</td>
</tr>
<tr>
<td>Avg. description length</td>
<td>5 (sentences)</td>
<td>6 (sentences)</td>
</tr>
</tbody>
</table>

Table 2: Data statistics from the text collections.

4.2 Syntactic structures

A part of our study is to discover the patterns of regularity that exist in the grammatical systems of individual languages. Below we summarize some of the syntactic structures that our analysis revealed. These are the most frequent structures for describing museum objects.

4.2.1 English

- PN → NP
  Van Gogh

- Det → CN → NP
  The portrait

- NP → Adv → NP
  The bell in London

- V2 → Adv → VP
  displayed at the Paris Salon
  painted by Jamie Wyeth

- V2 → Adv → VP
  displayed here suggest the hand of an artist

\(^{15}\)See [http://collections.smvk.se/pls/vkm/rigby.welcome](http://collections.smvk.se/pls/vkm/rigby.welcome)

\(^{16}\)See [http://www.metmuseum.org](http://www.metmuseum.org)
• V2 → NP → VP
displays painting of tulip bearing her signature

4.2.2 Swedish

• PN → NP
  Van Gogh

• Det → CN → NP
  Målningen

• NP → Adv → NP
  en målning av Gustaf Cederström

• V2 → Adv → VP
  återfinns på Museo Reina Sofía

• VP → Adv → VP
  finns här sedan 1967

• V2 → NP → VP
  utförd 1886 på Dalarö

4.3 Discourse patterns

We attempted to identify how ontology statements are aggregated in the discourse using a discourse strategy approach. A discourse strategy is an approach to text structuring through which particular organizing principles for a text are defined. This approach is based on the observation that people follow certain standard patterns of discourse organization for different discourse goals in different domains. Through linguistic analysis we observed how the domain representation is encoded in the structured object descriptions that we have collected. We then followed the discourse structure to learn how the ontology statements are composed in English and Swedish.

Below we summarize the discourse patterns and the semantic concepts that we will focus on in the generation phase. Text strings are marked with the corresponding ontology concept they are associated with in the ontology.

4.3.1 English

[PAINTING CREATOR DATE SIZE COLOR LOCATION]
Guernica painting was painted by Pablo Picasso creator in 1937 date. It is of size 349 by 776 cm size and is painted in white, black and gray color. The painting is displayed at Museo Reina Sofia in Madrid location.

[PAINTING TYPE CREATOR LOCATION DATE]
Guernica painting is an oil painting type by the Spanish artist Pablo Picasso creator. The painting was displayed at Museo Reina Sofia location in 1937 date.

[Painting type location creator size]
The Last Supper painting is a mural painting type in Milan location created by Leonardo da Vinci creator. It measures 450 by 870 centimeters size.

[Painting type creator place date location]
The Massacre painting at Chios location is an oil painting type by the French artist Eugène Delacroix creator. The painting was completed and displayed at the Salon place of 1824 date and presently hangs at the Musée du Louvre in Paris location.

[Painting type date creator]
Valdemar Atterdag holding Visby to ransom painting is an oil on canvas, signed in 1882 date by the Swedish historical painter Carl Gustaf Hellqvist creator (1851 – 1890).

[Painting type creator date size]
Olympia painting is an oil on canvas painting type by Édouard Manet creator. Painted in 1863 date, it measures 130.5 by 190 centimetres (51 x 74.8 in) size.

4.3.2 Swedish

[Painting creator date size location]
Guernica painting är en målning av Pablo Picasso creator från 1937 date. Målningen mäter 349 × 776 cm size. Den återfinns numera på Museo Reina Sofia i Spaniens huvudstad Madrid location.

[Painting type location creator date]
Skapelsen av Adam painting är en takfresk type i Sixtinska kapellet location, målad av Michelangelo creator cirka 1511 date.

[Painting type creator date]
Sagoprinsessan painting är en oljemålning type av John Bauer creator från 1904 date.

[Painting type creator date location]
Frälsningsarmén painting är en målning av Gustaf Cederström creator från 1886 date. Orignalet finns på Göteborgs konstmuseum location.

[Painting type creator date place]
Sommarnöje painting är en akvarell type av Anders Zorn creator; utförd 1886 date på Dalarö location.

[Painting type date creator location]
Nattvarden *painting* är en muralmålning *type* av Leonardo da Vinci *creator* cirka 1495-1498 *date* i klostret Santa Maria delle Grazie, Milano *location*.

[**painting creator date size location**]

Segern vid Narva *painting*, målning av Gustaf Cederström *creator*, påbörjad 1907 *date*, färdigställd 1910 *date*. Tavlan mäter 295 × 395 cm *size* och innehås av Nationalmuseum i Stockholm *location*.

## 5 Ontologies alignment and querying

As a part of the work of WP4, the group in Ontotext have constructed the Museum Reasonable View with the data from the Gothenburg City Museum database (see Section 2.1), and made it a part of the Reason-able View. The process of integrating the Gothenburg City Museum data and the Painting ontology (see Section 3) into the Museum Reason-able View is described in details in Damova (2011) and in Damova and Dannélls (2011).

### Access to the museum’s data through queries

A complementary way of arriving at the museum’s knowledge through questions. Orna and Pettitt (2010) name a number of typical queries about museum collections. Below follow some query examples:

- What is in the collections ?
- Why was it collected ?
- Where did it come from ?
- Where is it now ?

The Museum Reason-able View that has been loaded in OWLIM and is being accessible via SPARQL endpoint allows to formulate these kind of queries.\(^{17}\) A description of the complete framework including the generation process that is described in the next section is provided in Dannélls et al. (2011).

## 6 Encoding the ontologies in GF

The capabilities of GF as a host-language for ontologies were investigated in Enache and Angelov (2010), where SUMO, the largest open-source ontology was translated to GF. It was shown that the type system provides a robust framework for encoding classes, instances and relations. The same basic implementation design that was used for encoding SUMO in GF is applied in this work for encoding the CIDOC-CRM OWL 2 ontology and the

\(^{17}\)The access to the museum data is available from: [http://museum.ontotext.com/SPARQL](http://museum.ontotext.com/SPARQL)
Painting ontology. Using this implementation design, classes and sub classes are encoded as functions in the GF grammar. Ontological information about individuals (i.e, museum data values) is encoded in GF as axioms, external to the grammar functions. For example, a representation of the instance *BigGardenObj* is defined as follows:

```plaintext
fun BigGardenObj : Ind Painting ;
```

Where *Painting* was defined previously as a class and the dependent type *Ind* is used to encode class information of instances. The abstract representations of this GF implementation for the two ontologies are provided in Appendix B and C.

The remaining information about *Big Garden* from the ontology is encoded as a set of axioms with the following syntax:

```plaintext
isPaintedOn (el BigGradenObj) (el Canvas)
createdBy (el BigGardenObj)(el CarlLarsson)
hasCreationDate (el BigGardenObj) (el (year 1937))
```

The wrapper function *el* is used to make the above-mentioned coercion, where the two types, along with the inheritance object that represents the proof that the coercion is valid are not visible here, since GF features implicit arguments.

**Verbalization of the ontology axioms**

GF differentiates between domain dependent and domain independent linguistic resources, as it is designed to be applicable both to natural and to formal languages. One abstract grammar can have several corresponding concrete grammars; a concrete grammar specifies how the abstract grammar rules should be linearized in a compositional manner. Multilingual functional grammatical descriptions permit the grammar to be specified at a variety of levels of abstraction, which is especially relevant for ontology verbalization.

The natural language generation implementation in GF is based on composeable templates. We obtain the verbalization of classes and templates automatically, mainly based on their Camel-Case representation. Below follows a few English sentence examples that we are able to generate based on the ontology axioms:

- *Big Garden is a painting*
- *Big Garden is painted on canvas*
- *Big Garden is painted by Carl Larsson*
- *Big Garden was created in 1937*

Since the parser uses the resource library grammars, the result sentence will be syntactically correct, regardless of the arguments we use it with. To generate a coherent text that follows the discourse patterns, as specified in Section 4.2, more work is needed, since a grammatically correct verbalization is not possible based only on the ontology information.
7 Summary

Current data standards for storing museum information do not address all the requirements for storing information about museum objects. We observed there are some possible directions for accessing museum knowledge outside the museum application. Since we are concerned with building an application that is capable of generating natural language from existing museum knowledge, we have developed an ontology that aggregates existing metadata schemas for expressing knowledge about paintings and painting collections. Through text analysis we identified the most common syntactic constructions and discourse patterns that are relevant for generating museum object descriptions within the framework of GF.

The proposed ontology model and the museum data that we have acquired from the Gothenburg City Museum database have been integrated into the Museum Reason-able View to allow convenient querying of museum objects from multiple datasets. We have described the grammar implementation for the language generation component through which multilingual descriptions about these museum objects will be generated.

Our future work includes: translation of the museum data values, implementation of discourse patterns for English, Finnish, French, German and Swedish and evaluation of the generation results using native speakers of the language.

References


Prud’hommeaux Eric and Seaborne Andy. SPARQL. the query language for RDF, January 2008. URL \url{http://www.w3.org/TR/rdf-sparql-query/}. W3C Recommendation.


# A Record fields in the GCM database

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FALT1</td>
<td></td>
</tr>
<tr>
<td>PREFIX</td>
<td></td>
</tr>
<tr>
<td>FOREMLSNUMMER</td>
<td></td>
</tr>
<tr>
<td>UNERNUMMER</td>
<td></td>
</tr>
<tr>
<td>SAKORD</td>
<td></td>
</tr>
<tr>
<td>KLASS_1</td>
<td></td>
</tr>
<tr>
<td>KLASS_2</td>
<td></td>
</tr>
<tr>
<td>ANTAL</td>
<td></td>
</tr>
<tr>
<td>TILLVERKARE</td>
<td></td>
</tr>
<tr>
<td>TILLVERKNINGSTID</td>
<td></td>
</tr>
<tr>
<td>MODELL_MARKE</td>
<td></td>
</tr>
<tr>
<td>TILLBEHOR</td>
<td></td>
</tr>
<tr>
<td>LANGD</td>
<td></td>
</tr>
<tr>
<td>BREDD</td>
<td></td>
</tr>
<tr>
<td>HOJD</td>
<td></td>
</tr>
<tr>
<td>DIAM</td>
<td></td>
</tr>
<tr>
<td>VIKT</td>
<td></td>
</tr>
<tr>
<td>BESKRIVNING</td>
<td></td>
</tr>
<tr>
<td>HISTORIA</td>
<td></td>
</tr>
<tr>
<td>MATERIAL</td>
<td></td>
</tr>
<tr>
<td>KONSERVERING</td>
<td></td>
</tr>
<tr>
<td>FORVARVAT_FAN</td>
<td></td>
</tr>
<tr>
<td>FORVARVSDATUM</td>
<td></td>
</tr>
<tr>
<td>TIDIGARE_INVENTARIENUMMER</td>
<td></td>
</tr>
<tr>
<td>PLACERING</td>
<td></td>
</tr>
<tr>
<td>KOLLINUMMER</td>
<td></td>
</tr>
<tr>
<td>NEGATIVNUMMER</td>
<td></td>
</tr>
<tr>
<td>BILAGOR</td>
<td></td>
</tr>
<tr>
<td>REGISTRERINGSDATUM</td>
<td></td>
</tr>
<tr>
<td>SIGNATUR</td>
<td></td>
</tr>
<tr>
<td>FALTANDRING_SENASTE_INVENTERINGSDATUM</td>
<td></td>
</tr>
<tr>
<td>STOLDBEGARLIGHET</td>
<td></td>
</tr>
<tr>
<td>FORSARKRINGSVARDE</td>
<td></td>
</tr>
<tr>
<td>SAMTIDIGT_FORVARVAT</td>
<td></td>
</tr>
<tr>
<td>FOTOGRAFERINGSDATUM</td>
<td></td>
</tr>
<tr>
<td>FALT35</td>
<td></td>
</tr>
<tr>
<td>FALT36</td>
<td></td>
</tr>
<tr>
<td>BILSOKFALT</td>
<td></td>
</tr>
</tbody>
</table>
B The CIDOC-CRM OWL 2 in GF

abstract Cidoc = Basic ** {

cat CRM_Literal;

--------
-- CONCEPTS
--------

fun E1_CRM_Entity : Class;
fun E2_Temporal_Entity : Class;
fun E2_Temporal_Entity_Class : SubClass E2_Temporal_Entity E1_CRM_Entity;
fun E3_Condition_State : Class;
fun E3_Condition_State_Class : SubClass E3_Condition_State E2_Temporal_Entity;
fun E4_Period : Class;
fun E4_Period_Class : SubClass E4_Period E2_Temporal_Entity;
fun E5_Event : Class;
fun E5_Event_Class : SubClass E5_Event E4_Period;
fun E6_Destruction : Class;
fun E6_Destruction_Class : SubClass E6_Destruction E4_End_of_Existence;
fun E7_Activity : Class;
fun E7_Activity_Class : SubClass E7_Activity E5_Event;
fun E8_Acquisition : Class;
fun E8_Acquisition_Class : SubClass E8_Acquisition E7_Activity;
fun E9_Move : Class;
fun E9_Move_Class : SubClass E9_Move E7_Activity;
fun E10_Transfer_of_Custody : Class;
fun E10_Transfer_of_Custody_Class : SubClass E10_Transfer_of_Custody E7_Activity;
fun E11Modification : Class;
fun E11Modification_Class : SubClass E11Modification E7_Activity;
fun E12_Production : Class;
fun E12_Production_Class : SubClass E12_Production (both E11Modification E83Beginning_of_Existence);
fun E13_Attribute_Assignment : Class;
fun E13_Attribute_Assignment_Class : SubClass E13_Attribute_Assignment E7_Activity;
fun E14_Condition_Assessment : Class;
fun E14_Condition_Assessment_Class : SubClass E14_Condition_Assessment E13_Attribute_Assignment;
fun E15_Identifier_Assignment : Class;
fun E15_Identifier_Assignment_Class : SubClass E15_Identifier_Assignment E13_Attribute_Assignment;
fun E16_Measurement : Class;
fun E16_Measurement_Class : SubClass E16_Measurement E13_Attribute_Assignment;
fun E17_Type_Assignment : Class;
fun E17_Type_Assignment_Class : SubClass E17_Type_Assignment E13_Attribute_Assignment;
fun E18_Physical_Thing : Class;
fun E18_Physical_Thing_Class : SubClass E18_Physical_Thing E72_Legal_Object;
fun E19_Site : Class;
fun E19_Site_Class : SubClass E19_Site E18_Physical_Thing;
fun E20_Biological_Object : Class;
fun E20_Biological_Object_Class : SubClass E20_Biological_Object E19_Site;
fun E21_Person : Class;
fun E21_Person_Class : SubClass E21_Person (both E20_Biological_Object E39_Actor);
fun E22_Man_Made_Object : Class;
fun E22_Man_Made_Object_Class : SubClass E22_Man_Made_Object (both E19_Site E24_Physical_Man_Made_Thing);
fun E23_Physical_Man_Made_Thing : Class;
fun E23_Physical_Man_Made_Thing_Class : SubClass E23_Physical_Man_Made_Thing (both E19_Site E71_Man_Made_Thing);
fun E24_Condition_Assessment_of_Existence : Class;
fun E24_Condition_Assessment_of_Existence_Class : SubClass E24_Condition_Assessment_of_Existence (both E19_Site E71_Man_Made_Thing);
fun E25_Man_Made_Feature : Class;
fun E25_Man_Made_Feature_Class : SubClass E25_Man_Made_Feature (both E23_Physical_Man_Made_Thing E26_Physical_Feature);
fun E26_Physical_Feature : Class;
fun E26_Physical_Feature_Class : SubClass E26_Physical_Feature E18_Physical_Thing;
fun E27_Site : Class;
fun E27_Site_Class : SubClass E27_Site E24_Physical_Man_Made_Thing;
fun E28_Conditional_Object : Class;
fun E28_Conditional_Object_Class : SubClass E28_Conditional_Object E71_Man_Made_Thing;
fun E29_Design_or_Procedure : Class;
fun E29_Design_or_Procedure_Class : SubClass E29_Design_or_Procedure E73_Information_Object;
fun E30_Right : Class;
fun E30_Right_Class : SubClass E30_Right E89_Propositional_Object;
fun E31_Document : Class;
fun E31_Document_Class : SubClass E31_Document E73_Information_Object;
fun E32_Authority_Document : Class;
fun E32_Authority_Document_Class : SubClass E32_Authority_Document E31_Document;
fun E33_Linguistic_Object : Class;
fun E33_Linguistic_Object_Class : SubClass E33_Linguistic_Object E73_Information_Object;
fun E34_Inscription : Class;
fun E34_Inscription_Class : SubClass E34_Inscription (both E33_Linguistic_Object E37_Mark);
fun E35_Title : Class;
fun E35_Title_Class : SubClass E35_Title (both E33_Linguistic_Object E41_Appellation);
fun E36_Visual_Item : Class;
fun E36_Visual_Item_Class : SubClass E36_Visual_Item E73_Information_Object;
fun E37_Mark : Class;
fun E37_Mark_Class : SubClass E37_Mark E36_Visual_Item;
fun E38_Graph : Class;
fun E38_Graph_Class : SubClass E38_Graph E36_Visual_Item;
fun E39_Actors : Class;
fun E39_Actors_Class : SubClass E39_Actors E77_Persistent_Item;
fun E40_Legal_Body : Class;
fun E40_Legal_Body_Class : SubClass E40_Legal_Body E74_Group;
fun P99_dissolved : El E68_Dissolution -> El E74_Group -> Formula ;

fun P100_was_death_of : El E69_Death -> El E21_Person -> Formula ;

fun P101_had_a_general_use : El E70_Thing -> El E55_Type -> Formula ;

fun P102_has_title : El E71_Man_Name_Thing -> El E35_Title -> Formula ;

fun P103_was_intended_for : El E71_Man_Name_Thing -> El E55_Type -> Formula ;

fun P104_is_subject_to : El E72_Legal_Object -> El E30_Right -> Formula ;

fun P105_right_held_by : El E72_Legal_Object -> El E39_Actor -> Formula ;

fun P106_is_composed_of : El E90_Symbolic_Object -> El E90_Symbolic_Object -> Formula ;

fun P107_has_current_or_former_member : El E74_Group -> El E39_Actor -> Formula ;

fun P108_has_produced : El E12_Production -> El E77_Persistent_Item -> Formula ;

fun P109_has_current_or_former_curator : El E78_Collection -> El E39_Actor -> Formula ;

fun P110_augmented : El E79_Part_Addition -> El E24_Physical_Man_Made_Thing -> Formula ;

fun P111_added : El E79_Part_Addition -> El E18_Physical_Thing -> Formula ;

fun P112_diminished : El E80_Part_Removal -> El E18_Physical_Thing -> Formula ;

fun P113_removed : El E80_Part_Removal -> El E18_Physical_Thing -> Formula ;

fun P114_is_equal_in_time_to : El E2_Temporal_Entity -> El E2_Temporal_Entity -> Formula ;

fun P115_finishes : El E2_Temporal_Entity -> El E2_Temporal_Entity -> Formula ;

fun P116_starts : El E2_Temporal_Entity -> El E2_Temporal_Entity -> Formula ;

fun P117_occurs_during : El E2_Temporal_Entity -> El E2_Temporal_Entity -> Formula ;

fun P118_overlaps_in_time_with : El E2_Temporal_Entity -> El E2_Temporal_Entity -> Formula ;

fun P119_meets_in_time_with : El E2_Temporal_Entity -> El E2_Temporal_Entity -> Formula ;

fun P120_occurs_before : El E2_Temporal_Entity -> El E2_Temporal_Entity -> Formula ;

fun P121_overlaps_with : El E53_Time -> El E53_Time -> Formula ;

fun P122_borders_with : El E53_Time -> El E53_Time -> Formula ;

fun P123_resulted_in : El E81_Transformation -> El E77_Persistent_Item -> Formula ;

fun P124_transformed : El E81_Transformation -> El E77_Persistent_Item -> Formula ;

fun P125_used_object_of_type : El E7_Activity -> El E55_Type -> Formula ;

fun P126_used_constituent : El E15_Identifier_Assignment -> El E41_Appellation -> Formula ;

fun P127_used_attribute_to : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P128_was_based_on : El E83_Type_Creation -> El E1_CRM_Entity -> Formula ;

fun P129_exemplifies : El E1_CRM_Entity -> El E55_Type -> Formula ;

fun P130_represents : El E36_Visual_Item -> El E1_CRM_Entity -> Formula ;

fun P131_assigned : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P132_assigned_attribute_to : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P133_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P134_used_constituent : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P135_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P136_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P137_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P138_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P139_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P140_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P141_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P142_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P143_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P144_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P145_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P146_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P147_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;

fun P148_assigned_attribute : El E13_Attribute_Assignment -> El E1_CRM_Entity -> Formula ;
C The Painting ontology in GF

abstract Painting =
    Basic **
open (C=Cidoc),
    (ML=MidLevelOntology),
    (S=Merge)
in {

-----------
-- CONCEPTS
-----------

fun ResourcePlace : Class;
fun ResourceCollection : Class;
fun ResourceItem : Class;
fun ResourceItemClass : Class;
fun ResourceItemColor : Class;
fun ResourceItemImage : Class;
fun ResourceItemMaterial : Class;
fun ResourceItemMeasurement : Class;
fun ResourceItemName : Class;
fun ResourceItemNumber : Class;
fun ResourceItemTechnique : Class;
fun ResourceItemTitle : Class;
fun ResourceNameAuth : Class;
fun PresentationOrganizationShort : Class;
fun MidLevelOntologyOilMedium : Class;
fun S_AbstractClass : SubClass S.AbstractPaintingRepresentedEntity;
fun SUMO_Causes : Class;
fun SUMO_CausesClass : SubClass SUMO_CausesPaintingPurpose;
fun S_CommunicationClass : SubClass S.CommunicationPaintingPurpose;
fun S_ContentDevelopmentClass : SubClass S.ContentDevelopmentPaintingPurpose;
fun S_DirectingClass : SubClass S.DirectingPaintingPurpose;
fun S_EducationalProcessClass : SubClass S.EducationalProcessPaintingPurpose;
fun S_Foot : Class;
fun S_FootClass : SubClass S.FootPaintingUnitOfMeasure;
fun S_FruitOrVegetableClass : SubClass S.FruitOrVegetablePaintingConcreteThing;
fun S_GraphElementClass : SubClass S.GraphElement S.GraphElement;
fun S_GraphElementAbstract : SubClass S_AbstractClass;
fun S_GraphElementFoot : SubClass S_FootClass;
fun S_GraphElementObject : SubClass S_GraphElementClass;
fun S_GraphElementObjectAbstract : SubClass S_GraphElementAbstract;
fun S_GraphElementObjectFoot : SubClass S_GraphElementObject;
fun S_GraphElementObjectFootAbstract : SubClass S_GraphElementObjectFoot;
fun S_GraphElementObjectFootObject : SubClass S_GraphElementObjectFoot;
fun S_GraphElementObjectFootObjectAbstract : SubClass S_GraphElementObjectFootObject;
fun S_GraphElementObjectFootObjectFoot : SubClass S_GraphElementObjectFootObject;
fun S_GraphElementObjectFootObjectFootAbstract : SubClass S_GraphElementObjectFootObjectFoot;
fun S_GraphElementObjectFootObjectFootObject : SubClass S_GraphElementObjectFootObjectFoot;
fun S_GraphElementObjectFootObjectFootObjectAbstract : SubClass S_GraphElementObjectFootObjectFootObject;
fun S_GraphElementObjectFootObjectFootObjectFoot : SubClass S_GraphElementObjectFootObjectFootObject;
fun S_GraphElementObjectFootObjectFootObjectFootAbstract : SubClass S_GraphElementObjectFootObjectFootObjectFoot;
fun S_GraphElementObjectFootObjectFootObjectFootObject : SubClass S_GraphElementObjectFootObjectFootObjectFoot;
fun S_GraphElementObjectFootObjectFootObjectFootObjectAbstract : SubClass S_GraphElementObjectFootObjectFootObjectFootObject;
fun S_GraphElementObjectFootObjectFootObjectFootObjectFoot : SubClass S_GraphElementObjectFootObjectFootObjectFootObject;
fun S_GraphElementObjectFootObjectFootObjectFootObjectFootAbstract : SubClass S_GraphElementObjectFootObjectFootObjectFootObjectFoot;
fun S_GraphElementObjectFootObjectFootObjectFootObjectFootFoot : SubClass S_GraphElementObjectFootObjectFootObjectFootObjectFoot;
fun S_GraphElementObjectFootObjectFootObjectFootObjectFootFootAbstract : SubClass S_GraphElementObjectFootObjectFootObjectFootObjectFootFoot;
fun S_GraphElementObjectFootObjectFootObjectFootObjectFootObject : SubClass S_GraphElementObjectFootObjectFootObjectFootObjectFoot;
fun S_GraphElementObjectFootObjectFootObjectFootObjectFootObjectAbstract : SubClass S_GraphElementObjectFootObjectFootObjectFootObjectFootObject;
fun S_GraphElementObjectFootObjectFootObjectFootObjectFootObjectFoot : SubClass S_GraphElementObjectFootObjectFootObjectFootObjectFootObject;