Generating Tailored Texts for Museum Exhibits

Dana Dannélls

Natural Language Processing Research Unit
Department of Swedish Language
University of Gothenburg
SE-405 30 Gothenburg, Sweden
dana.dannells@svenska.gu.se

Abstract

This paper reports work that aims to generate texts in multiple languages from ontologies following the Conceptual Reference Model (CRM) ISO standard for conceptual models of museums. The rationale of this work is to increase users' knowledge and interest in the cultural heritage domain by allowing the user to select his preferable syntax presentation and influence the order of the generated information using generation techniques and Semantic Web technologies. We chose for study a small amount of logical relations represented in the ontology and wrote a grammar that is capable to describe them in natural language through user editing. We present the multilingual source authoring environment, which is built upon the grammatical framework (GF) formalism and show how it is utilized to generate multiple texts from the CRM domain ontology. The initial results comprise texts, which vary in syntax and content.

1. Introduction

During the last decade, the awareness of the need for personalization has become fundamental for cultural institutions such as museums and libraries while aim to produce textual descriptions of museum exhibits tailored to the visitor's knowledge, interests, and personal preferences, such as preferred vocabulary, syntax, sentence length etc. One of the first examples of personalization in a museum context was developed in the Intelligent Labelling Explorer (ILEX) project,[1] by using Natural Language Generation (NLG) techniques. More recently, applications within the cultural heritage (CH) domain have seen an explosion of interest in these techniques (Novello and Callaway, 2003; O'Donnell et al., 2001; Androutsopoulos et al., 2007).

The process of NLG starts from an ontology that describes a certain domain. Recently, natural language generators that are targeted towards the Semantic Web ontologies have started to emerge. A strong motivation for generating texts from ontologies is that the information represented in an ontology has a true potential to provide a large amount of text if this text is realized correctly. Gradually, the cultural heritage knowledge domain which is often characterized by complex semantic structures and large amounts of information from several different sources will benefit from the complete generation of the information delivered in the ontology.

Web ontology languages pose many opportunities and challenges for language generators. Although standards for specifying ontologies provide common representations to generate from, existing generation components are not compatible with the requirements posed by these newcoming standards. This issue has been previously addressed by developing domain-dependent authoring interfaces that are built upon an ontology and that allows it to be deployed through knowledge editing (Brun et al., 2000; Hartley et al., 2001; van Deemter et al., 2005). These interfaces are links between the ontology and the user who can manipulate the content of the document indirectly in his/her own language. An example of a template-based authoring tool that makes use of this technique within the CH domain was presented by Androutsopoulos et al. (2007). An alternative approach to template-based NLG that is particularly relevant in cases where texts are generated from logical forms in several languages simultaneously is a grammar-based approach (Bateman, 1997).

In this paper we present a multilingual source authoring tool, which is built upon the grammatical framework (GF) formalism to generate texts from the underlying semantic representation that is based on the Conceptual Reference Model (CRM) domain ontology. The authoring environment is similar to those described in Power and Scott (1998), Dymetman et al. (2000) and van Deemter and Power (2003). [2] The focus is on the process starting from a fixed semantic representation to a surface realization, with emphasis on the syntactical sentence structure, and the content variation.

The structure of the paper is as follows. In section 2 we elaborate the notion of ontology and describe both the reference ontology model and the grammar formalism that our application is built upon. Section 3 presents the grammar implementation and explains how it is utilized to generate tailored descriptions from a formal representation language. We finish with conclusions and a discussion of future work in Section 4.

2. Background

In the context of the work presented here, an *ontology* is understood as a formal model that allows reasoning about concepts, objects and about the complex relation between them. An ontology holds meta-level information about different types of entities in a certain domain and provides a structure for representing contexts, it is not human readable as it is designed to be processed by computer systems.

[1] http://www.hcrc.ed.ac.uk/Site/ILEXINTE.html

[2] The advantages of utilizing this family of domain authoring approaches that are coupled with multilingual text generation are elaborated in Scott (1999).

Examples of Web ontology-languages that have been developed by the W3C Web-Ontology working group are OWL and DAML+OIL.[3] The basis for the design of these Web technology languages based on the RDF Schema is the expressive Description Logic (DL) *SHIQ* (Horrocks et al., 2003). These languages provide extensive reasoning capabilities about concepts, objects and relationships between them.

2.1. Generating from an Ontology

In an ontology, an object may be described by semantic graphs whose nodes (concepts) represent parts of an object, and the arcs (relations) represent partial constrains between object parts. Each relation described in a logical language is binary, i.e. it connects between two nodes. In order to present a piece of information about an object represented in an ontology, multiple sentences must be formulated. It becomes valuable if these sentences that build the final text can be adapted to various contexts or users.

There has been successful attempts to generate from ontologies (Wilcock, 2003; Wilcock and Jokinen, 2003; Bontcheva and Wilks, 2004; Bontcheva, 2005). Wilcock (2003) and Wilcock and Jokinen (2003) have shown how RDF/XML generation approach can be extended so that the information embedded in the ontology can be exploited to generate texts from Web ontology-languages such as DAML+OIL and OWL without the need for a lexicon. Bontcheva (2005) demonstrated how to minimize the effort when generating from Web ontology-languages while being more flexible than ontology verbalisers. Some of the difficulties reported by these authors concern lexicalization and in establishing context variations.

2.2. The CRM Ontology

One initiative to enable an ontology in the context of the cultural heritage is the Conceptual Reference Model domain ontology. The International Committee for Documentation of the International Council of Museums Conceptual Reference Model (CIDOC-CRM)[4] is a core ontology and ISO standard for the semantic integration of cultural information with library archive and other information (Doerr, 2005). The primary role of the CRM is to enable information exchange and integration between heterogeneous sources of cultural heritage information.

The central idea of the CIDOC-CRM is that the notion of historical context can be abstracted as things and people. It concentrates on the definition of relationships rather than classes to capture the underlying semantics of multiple data and meta structures. It tends to provide an optimal analysis of the intellectual structure of cultural documentation in logical terms, which is available in several formats such as RDF and OWL that have hardly been explored yet. The work described in this paper is based on the OWL version of the ontology.[5]

[3] http://www.w3.org/TR/

[4] http://cidoc.ics.forth.gr/

[5] http://cidoc.ics.forth.gr/OWL/cidoc_v4.2.owl

2.3. The Grammatical Framework (GF)

The Grammatical Framework (Ranta, 2004) is a functional grammar formalism based on Martin-Löf's type-theory (Martin-Löf, 1973) implemented in Haskell.[6] GF focuses on language independent semantic representations. It 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 constructing a detailed mapping from semantics to form. This aspect is crucial for natural language generation to work. What makes the grammar suitable for generating from ontologies and in particular from OWL, is that it allows multiple inheritance. GF has three main module types: abstract, concrete, and resource. Abstract and concrete modules are top-level, in the sense that they appear in grammars that are used at runtime for parsing and generation. They can be organized into inheritance hierarchies in the same way as object-oriented programs. The main advantage with converting the ontology to GF is that we can make use of the rich type system in the concrete syntax for capturing morphological variations. Our approach is based on the idea suggested by Khegai et al. (2003) who utilized GF to automatically generate multiple texts from semantic representations. The source authoring environment deploys similar techniques to those introduced in Power and Scott (1998), Dymetman et al. (2000) and van Deemter and Power (2003).

3. Generating from the Ontology

We chose for study a small amount of logical relations represented in the ontology and wrote a grammar that is capable to describe them in natural language through user editing. The following code is a fragment taken from the ontology we employed. The code states that the class *PaintingP9091* must have at least one value *TypeValue* on property *has_type*; the individual *TypeValue* is an instance of the class *cidoc:E55.Type*[7] and has two property values: "tool" and "painting".

```
<owl:Class rdf:about="PaintingP9091">
 <owl:Restriction>
  <owl:onProperty rdf:resource="&cidoc;P2F.has_type"/>
   <owl:hasValue rdf:resource="#TypeValue"/>
 </owl:Restriction>
</owl:Class>
<owl:Thing rdf:about="#TypeValue">
  < rdf:type rdf:resource="&cidoc;E55.Type"/>
  <Tool rdf:datatype="&xsd;string">tool
  </Tool>
  <Painting rdf:datatype="&xsd;string">painting
```

[6] Haskell is a standardized purely functional programming language with non-strict semantics. Similar to Lisp and Scheme.

[7] The notation &cidoc; is used instead of the whole namespace,i.e http://cidoc.ics.forth.gr/OWL/cidoc_v4.2.owl#

</Painting>
</owl:Thing>

The above fragment exemplifies the representation of the classes and relationships that are utilized by the grammar. In the grammar implementation, classes are represented as categories; properties are functions (rules) between two categories, where each property links between two classes; individuals are lexical categories (strings). Below is a representation of the *mkObject*, which corresponds to a function that links between the classes of an *Object*:

mkObject:ObjectNodeI→ObjectNodeII→ObjectNodeIII→Object;

In this example the *Object* category corresponds to *PaintingP9091*. Each *ObjectNode* is a class, according to the above ontology representation, *ObjectNodeI* corresponds to the cidoc class *cidoc:E55.Type*. It is followed by *ObjectNodeII*, i.e. *cidoc:E52.Time-Span* and *ObjectNodeIII*, i.e. *cidoc:E21.Person*, as shown below.

{Type} *instance_of* ObjectNodeI
{Time-Span} *instance_of* ObjectNodeII
{Person} *instance_of* ObjectNodeIII

Consequently, individuals such as "tool" and "painting" are terminals and are declared in the concrete syntax. In the next sections we describe the abstract and the concrete representations

3.1. The Abstract Representation

The abstract syntax is a context-free grammar where each rule has a unique name. An abstract rule in GF is written as a typed function. The categories and functions are specified in GF by *cat* and *fun* declarations. Below is a fragment of the grammar:

cat

Object ;ObjectNodeI ; Type ;
ObjectNodeII ; Time-Span ;
ObjectNodeIII ; Person ;

fun

HasType_This : Type → ObjectNodeI;
HasType_Here : Type → ObjectNodeI;
HasType_Template : Type → ObjectNodeI;
HasTimeSpan: Time-Span → ObjectNodeII;
CarriedOutBy_Painting: Person → ObjectNodeIII;
CarriedOutBy_Tool: Person → ObjectNodeIII;

The abstract syntax gives a structural description of a part of the domain. It has several advantages, one of which is the ability to utilize the same categories differently depending on the semantic complexity of the context. Here we declared three functions for the *ObjectNodeI* to achieve context variations, though very simple ones. Similarly, we declared two functions for the *ObjectNodeIII*, however, the difference between *CarriedOutBy_Painting* and *CarriedOutBy_Tool* is the choice of the verb in the linearization rule. The verb *painted by* is applied when the subject is the noun *painting*, but the verb *created by* is applied when the subject is the noun *tool*, in cases when the object is an instance that belongs to the category *Person*.

3.2. The Concrete Representation

Each category and function introduced in the abstract syntax has a corresponding linearization type in the concrete syntax. Linearization rules are declared differently for each target language. In addition, each concrete syntax also contains grammatical parameters and grammar rules, which are used to ensure grammatical correctness for each language, in our case English and Swedish. An example of linearization rules taken from the English concrete syntax is the following:

lin

CarriedOutBy_Painting obj = {s = det ! obj.num ++ cop ! obj.num ++ "painted by" ++ obj.s ; num=obj.num};

Painting = {s = "painting" ; num = sg} ;
Painting = {s = "paintings" ; num = pl} ;

Grammatical features are supported by GF and the agreement between the pronoun and the verb is enforced in the generated sentences. The variable *obj* represents a terminal string. The parameter *num* is an abbreviation for the parameter type "number", it contains the inherent number that can be either singular (sg) or plural (pl). The operation *det* is a determiner, and the operation *cop* is copula verb.

3.3. The Authoring Environment

Figure 3 illustrates the source authoring environment. The left-side window shows the abstract syntax tree, which represents the *Object* structure. The large window positioned to the right is the linearization area, the editing focus is presented as the highlighted metavariable *?3*. The bottom area shows the context-dependent refinement for the *ObjecNodeIII*, there are two possible relations to choose from.

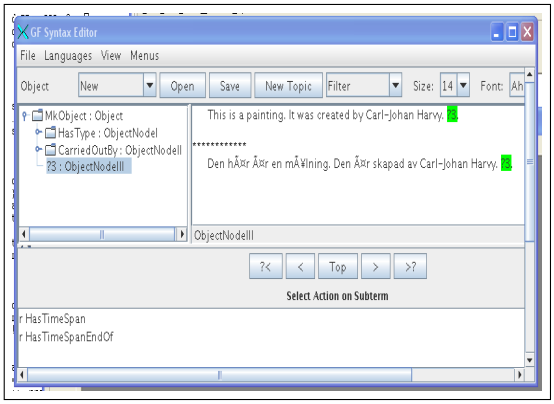

Figure 1: The GF source authoring environment.

The authoring tool that is built upon the GF grammar makes it possible to generate the following texts:

English
(1) Here we have a painting. It was painted by Carl-Johan Harvy. It was made in 1880.

(2) This is a tool. It was made in 1880. It was created by Carl-Johan Harvy.

(3) On the second floor of the history museum we have paintings. They were created by Siri Derkert. They were produced in Italy.

Swedish

(1) Här har vi en målning. Den är målad av Carl-Johan Harvy. Den är gjord på 1880 talet.

(2) Det här är ett redskap. Det är gjort på 1880 talet. Det är tillverkat av Carl-Johan Harvy.

(3) På andra våningen i historiska museet har vi målningar. De är tillverkade av Siri Derkert. De är producerade i Italien.

The difference between the first and second sentence is the order in which the *ObjectNodeII* and the *ObjectNodeIII* appears, this is done with the help of the *variants* function that allows for syntactic variations by reordering the linearized categories. The third sentence illustrates a typical example of a combined template and grammar based generation, e.g. the fixed sentence: "On the second floor of the history museum" that has been prewritten.

4. Conclusions and Future Work

In this paper we have presented a multilingual grammar-based approach, the aim of which is to generate exhibit descriptions following the CRM domain ontology. We chose for study a small amount of logical relations represented in an ontology and have started to examine the capabilities of utilizing a grammar to bridge between ontology representations and different users.

We suggest an approach to support a user on receiving information based on the Semantic Web in the cultural heritage domain and show how the GF authoring tool, which allows users to choose the content and the form of the output text can be utilized to generate texts from CIDOC CRM. Future work will focus on ontology studies and on particular problems of generating for cultural heritage. We are also planning to utilize the Resource Grammar Library that has been developed to provide the linguistic details for application grammars on different domains. This will be a step towards high quality summary generation. Our goal is to built a grammar that reflects the ontology structure and supports all the OWL features to allow the user to interact with the full ontology.

5. References

I. Androutsopoulos, J. Oberlander, and V. Karkaletsis. 2007. Source authoring for multilingual generation of personalised object descriptions. *Natural Language Engineering*, 13(3):191–233.

J. A. Bateman. 1997. Enabling technology for multilingual natural language generation: The kpml development environment. *Natural Language Engineering*, 3(15–55).

K. Bontcheva and Y. Wilks. 2004. Automatic report generation from ontologies: the miakt approach. In *Nineth International Conference on Applications of Natural Language to Information Systems (NLDB)*. Manchester.

K. Bontcheva. 2005. Generating tailored textual summaries from ontologies. In *Second European Semantic Web Conference (ESWC'05)*, Crete.

C. Brun, M. Dymetman, and V. Lux. 2000. Document structure and multilingual authoring. In *In Proc. of First International Natural Language Generation Conference (INLG)*, Mitzpe Ramon, Israel, June.

M. Doerr. 2005. The cidoc crm, an ontological approach to scheme heterogeneity. In Dagstuhl Seminar, editor, *Semantic Interoperability and Integration*, pages 1862–4405.

M. Dymetman, V. Lux, and A. Ranta. 2000. Xml and multilingual document authoring: Convergent trends. In *In Proceedings of COLING*. Saarbrucken, Germany.

A. Hartley, D. Scott, J. Bateman, and D. Dochev. 2001. Agile – a system for multilingual generation of technical instructions. In *In MT Summit VIII, Machine Translation in the Information Age*, pages 145–150.

I. Horrocks, P.F. Patel-Schneider, and F. van Harmelen. 2003. From shiq and rdf to owl: The making of a web ontology language. *Journal of Web Semantics*, 1(1):7–26.

J. Khegai, B. Nordström, and A. Ranta. 2003. Multilingual syntax editing in gf. pages 453–464, Mexico. Springer.

P. Martin-Löf. 1973. An intuitionistic theory of types: Predicative part. In H. E. Rose and J. C. Shepherdson, editors, *In Proc. of Logic Colloquium '73, Bristol, UK*, volume 80, pages 73–118.

A. Novello and C. Callaway. 2003. Porting to an Italian surface realizer: A case study. In *Proc. of the 9th European Workshop on NLG*, pages 71–78.

M. O'Donnell, J. Oberlander C. Mellish, and A. Knott. 2001. Ilex: an architecture for a dynamic hypertext generation system. *Natural Language Engineering*, 7(3):225–250.

R. Power and D. Scott. 1998. Multilingual authoring using feedback texts. In *17th International Conference on Computational Linguistics*, pages 1053–1059.

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

D. Scott. 1999. The multilingual generation game: Authoring fluent texts in unfamiliar languages. In *Proc. of the 16th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1407–1411.

K. van Deemter and R. Power. 2003. High-level authoring of illustrated documents. *Natural Language Engineering.*, 9(2):101–126.

K. van Deemter, E. Krahmer, and M. Theune. 2005. Real versus template-based natural language generation: a false opposition? *Computational Linguistics*, 31(1):15–23.

G. Wilcock and K. Jokinen. 2003. Generating responses and explanations from rdf/xml and daml+oil. In *Knowledge and Reasoning in Practical Dialogue Systems IJCAI-2003*, pages 58–63.

G. Wilcock. 2003. Talking owls: Towards an ontology verbalizer. In *Human Language Technology for the Semantic Web and Web Services*, pages 109–112.