RDFa Editor for Ontological Annotation

Melania Duma
Faculty of Mathematics and Computer Science
University of Bucharest
Academiei Street number 14, Bucharest
melaniaduma@gmail.com

Abstract

One of the purposes of semantic web is to provide machine understandable content and this can be achieved by annotating information. At the moment, annotations can be created manually and also automatically, both of the approaches having advantages and disadvantages. The goal of this article is to present a new semi-automatic annotation tool, which given a text will annotate words with concepts from an ontology.

1 Introduction

In the present days, World Wide Web has proven to be one of the easiest and most useful ways of gaining access to information. One of the main characteristics of this information is that it requires human intervention in order to be managed and presented. The main goal of Semantic Web is to provide a way of transforming this information, so that it is also machine comprehensible, idea that is captured in the definition provided by World Wide Web Consortium: "The Semantic Web is about two things. It is about common formats for integration and combination of data drawn from diverse sources, where on the original Web mainly concentrated on the interchange of documents. It is also about language for recording how the data relates to real world objects" (Ivan, 2011). In order to achieve the ideas presented in this definition, a need for annotation systems arises.

Document annotation can be defined as the process of providing data about the content of the documents. In some ways, the process of annotating is very similar to that of tagging, and the similarity comes from the fact that they both enrich the information by providing metadata and they both improve the search capabilities of a system. However, annotations go one step further than tags - they help create more organized metadata which can be further exploited by specialized systems.

The process of annotating documents can be done both manually, as well as automatically, both of these approaches having advantages and disadvantages.

One of the biggest advantages of manual annotation is that it has a high accuracy rate, but it has proven to be both cost and time inefficient. It also often requires for an exact procedure to be followed. At first, the human annotator must familiarize himself with the text, then proceed with the annotations. Any difficult decision regarding an annotation is then saved in a file, and a curator later reviews the specific annotations and makes any necessary changes. This whole procedure proves to be very expensive because it requires the constant training of personnel, as the level of accuracy of annotation depends drastically of the level of domain specific knowledge of the human annotator.

On the other hand, automatic annotation can offer a method of annotating in a time efficient way, while having the disadvantage that it is highly error prone, mainly because of the fact that it lacks human intervention.

In order to overcome these shortcomings, a series of semi-automatic annotation systems have been developed. Many present themselves under the form of an editor that annotates specific parts of a document (word, paragraph, section or an entire document).

Different approaches on how to perform semi-automatic annotation where implemented in a series of platforms and a survey of these platforms is presented in (Reeve and Han, 2005). AeroDAML is a pattern based system, in which nouns and relationships are connected with concepts and properties that are part of the DARPA Agent Markup Language (DAML) ontologies (Reeve and Han, 2005). Regarding the architecture, AeroDAML is made up of a series of com-

Proceedings of the Student Research Workshop associated with RANLP 2011, pages 54–59,
Hissar, Bulgaria, 13 September 2011.

ponents: a text extraction component based on AeroText, a mapping component and an annotation editor.

Armadillo is also a pattern based system, that given a set of initial words called seeds, will annotate them and extract the context surrounding these words. Based on these contexts, a set of rules is constructed, which is then used in order to discover other words surrounded by similar contexts. The process takes place again with the new words acting as seeds.

Another system that can be used for annotating documents is the KIM Platform which contains an ontology, a knowledge base, an annotation system, and an indexing and retrieval system based on the Lucene engine. During the annotation process, the token is not only mapped to a concept in the ontology but also to a reference in the knowledge base, so that word disambiguation can be provided.

MnM is a machine learning based system, based on the Lazy-NLP algorithm. The result of this algorithm is a set of rules, which are then used to tag information in the document or correct existing tags.

MUSE is another semantic annotation platform based, like KIM platform, on the GATE framework. The component that deals with semantic annotation is based on Java Annotation Pattern Engine (JAPE), which provides a grammar that helps in constructing rules. These rules are then used in order to create annotations.

Ont-O-Mat is based on S-CREAM (Semi-automatic CREAtion of Metadata), a semantic annotation framework. Information extraction in Ont-O-Mat is done with the help of Amilcare, and tagging and correction rules are created using the LP algorithm. Furthermore, the process of annotation in Ont-O-Mat is based on the PANKOW (Pattern-based Annotation through Knowledge On the Web) algorithm, which returns a set of hypothesis phrases, that are used to create annotations.

All of the systems mentioned above have been evaluated in terms of precision and recall, and the MnM and MUSE platforms have proven to have the highest performance among them, with a precision of 95%, respectively 93.5% (Reeve and Han, 2005).

In this paper, a new RDFa editor, that aims to semi-automatically annotate data, will be introduced. This new system differentiates from the others in that it helps create valid RDFa annotations, which integrated in the content of the Web pages, will make them not only human readable, but also machine understandable. Furthermore, the full potential of these annotations can be attained later on, by creating a reasoning module which can be integrated in the platform.

The paper is structured as follows. In section 2, the main functionalities of the system are presented, together with a description of the main components that form the new editor. An evaluation of the new software is provided in section 3, while section 4 contains the conclusion and ideas for future work.

2 System Design and Challenges

2.1 System Functionalities

The aim of this article is to describe the design and implementation of an ongoing project, that aims to provide a RDFa editor for semantic annotation of documents. The aim of RDFa, which is "a way to express RDF data within XHTML, by reusing the existing human-readable data" (Birbeck and Adida, 2008), is to provide a series of attributes. These attributes can be used in order to enrich the information contained in the web pages with the help of new metadata.

Regarding the domain, the editor accepts any kind of document and ontology, as it was constructed with the purpose of being an open-domain tool.

A list of the most important functionalities that have been implemented in the new annotating platform is presented below:

- The application creates annotations based on text chunks, from a given document and a selected concept from an ontology. At the moment the system works with Nounphrases (NP)
- The ontology used for annotation can be selected by the user of the application, together with the document to be annotated
- All the annotations made to a documents are saved in a new file and respect the RDFa format
- After annotations have been made, a version of the annotated document can be viewed and saved
- After a document is opened, it is possible to also open an annotation file and merge them

- The details regarding an annotation can be viewed for every annotated NP
- Any specific annotation can be deleted at any point in time

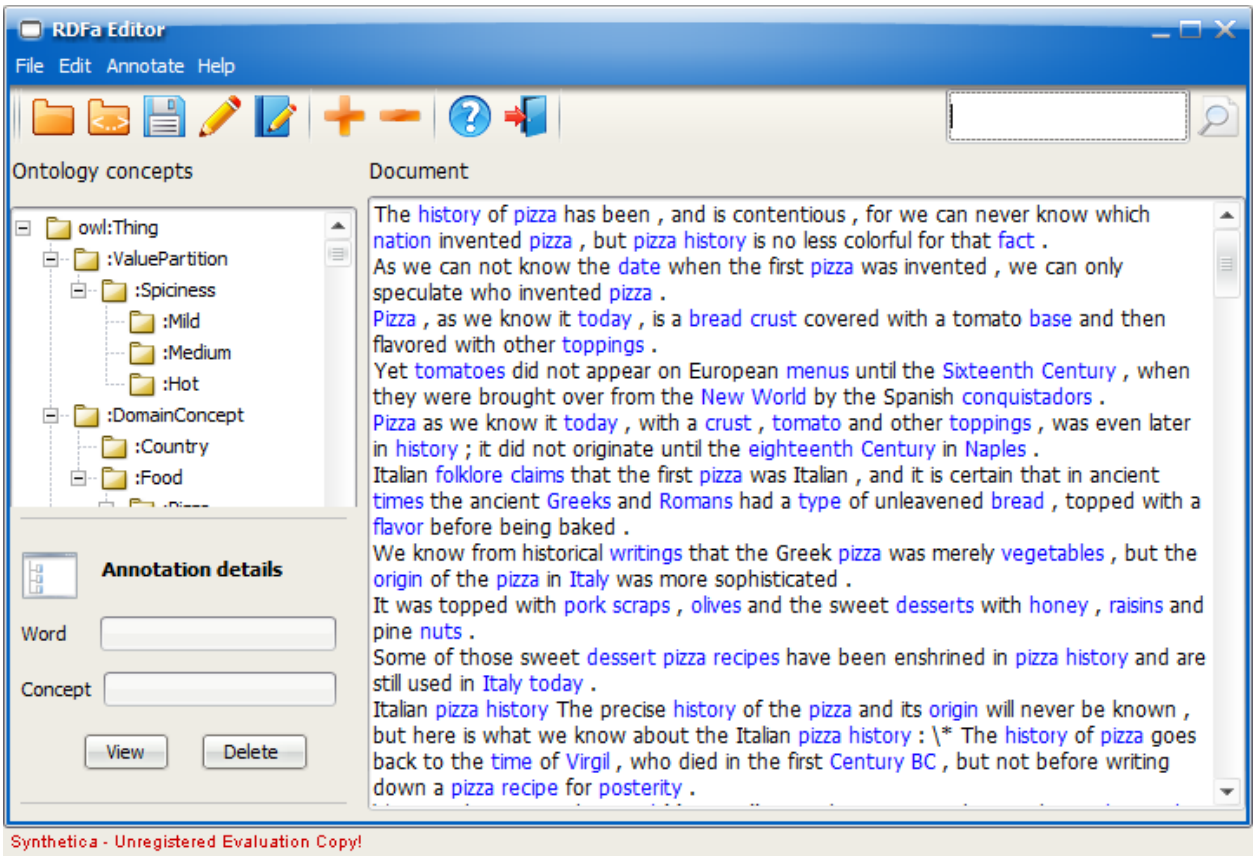

Fig. 1: The interface of the system

2.2 System Components

Regarding its architecture, the system is made up of a series of different components, that combined achieve the goal of semi-automatically annotating a document (Figure 2).

The Ontology Manager component, used for importing and presenting the ontology, was created with help of Jena, a Java framework used for writing Semantic Web applications and Pellet, an open-source Java based OWL DL reasoner. The Pellet API provides a class, that given an ontology, it presents the class hierarchy under the form of a tree component that can be easily integrated in the application.

The Document component deals with the management of the document that will be annotated. Thus, a document is opened and then the text is retrieved and passed to the Document Parser component.

The Document Parser component is concerned with the parsing of the document, which results in the creation of a list of tokens. In order to determine which of the tokens are Nounphrases, a part of speech tagger was needed. Therefore, the tagger selected was Stanford Log-linear Part-Of-Speech Tagger developed by the Stanford Natural Language Processing Group. This particular tagger has been chosen because it was developed in Java and it was easy to be integrated into the rest of the application, but also because of its high performance rate.

The tagger provides two models for the English language: bidirectional-distsim-wsj-0-18.tagger and left3words-wsj-0-18.tagger.

In the development of the new system the first of the options above was chosen mainly because, even though it is slower than the other model, it has an increased accuracy (97.32% over 96.97%).

The Annotation component of the application deals with the automatic annotation when a noun has been selected from the text and a concept has also been selected from the ontology. In order to achieve this, a lemmatizer was needed, which has the goal "to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form" (Manning et al., 2008).

The use of a lemmatizer was preferred to that of a stemmer, mainly because while the stemmer uses only heuristics in order to determine a base form of a word, a lemmatizer uses vocabularies and morphological analysis in order to determine the dictionary form of an word. Thus, due to the way in which it is constructed, a lemmatizer could help increase the accuracy of annotations.

WordNet is constructed as a lexical database for the English language. It provides software tools that support automatic text analysis and one of these tools is the lemmatizer. The lemmatizer was used to automatically annotate all the nouns that share the same lemma with the selected noun.

All the annotations made are saved in a file in XML format. Every annotation is characterized by a noun, represented by the interval indicating the position it occupies in the text, and by the concept used in the annotation, specified by the RDFa attribute @about.

Resource Description Framework (RDF) is a language whose goal is to create statements regarding Web sources. In order to create the statements, RDFa, a recommendation of W3C, uses a set of attributes composed of a few XHTML attributes together with some newly created ones (Birbeck and Adida, 2008).

Among the latter ones, the @about attribute is used in order to define the subject of the data. Because of the fact that the system annotates only nouns, it was assumed that all the nouns represent the subject of the data, and therefore they have all been annotated using the @about attribute.

The file, in which the annotations are saved, can be later opened again and used in order to organize the existing annotations. More precisely, once the text document is displayed, and an annotation file is opened, the annotations from this file are retrieved and then processed.

The processing of an annotation requires the extraction of the concept it references and the interval which characterizes the position of a noun. Once this interval is retrieved, the position is searched in the current document and the corresponding noun is highlighted.

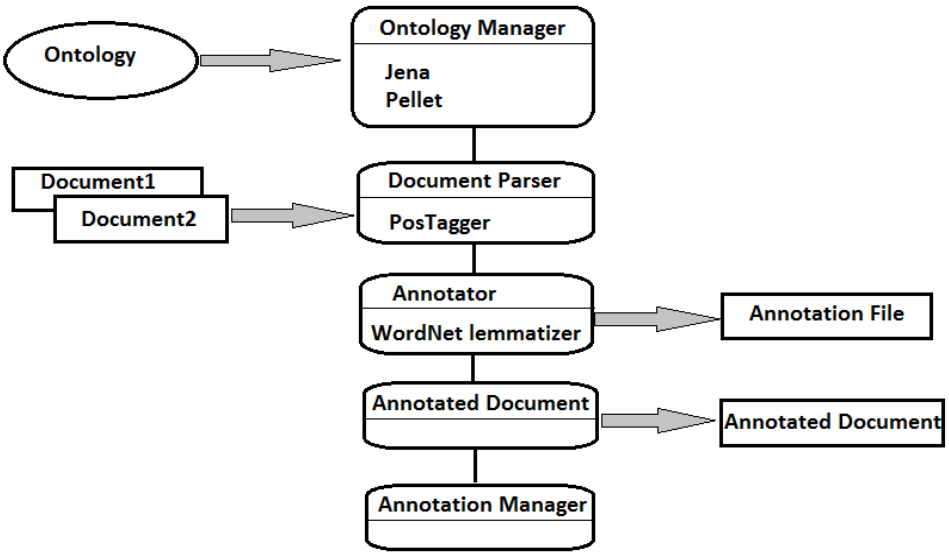

Fig. 2: Components of the system

The Annotated Document component provides the possibility to view the annotated file. In order to achieve this, the list of annotations specific to the current document is merged with the text and they form a new document which can be viewed and saved.

In more details, the interval, which describes the position of the noun in the text, is determined for every annotation. The ends of the interval are then searched in the text and the RDFa attribute is inserted in the corresponding position. These insertions are done in a new document so that the original one remains unaffected by these changes.

The Annotation Manager component deals with the management of annotations. It provides a way to view and analyze the concept which was used during the annotation of a noun and also to delete the specified annotation if proven inaccurate.

3 Evaluation

In this section an evaluation of the new system is presented. In order to achieve this, the metrics used for evaluation will be defined. Recall is defined as the number of accurate annotation divided by the number of all the annotations made by a human annotator, while precision is defined by the number of accurate annotations divided by the number of accurate plus the number of inaccurate annotations generated by the annotation platform.

The new system was tested using a document base created with the help of 35 Web pages. The Web pages were parsed and the content was extracted and saved in text files. The ontology used in the evaluation was the pizza.owl ontology and it was chosen because it is constructed in a way that allows correct extraction of the tree component using Jena and Pellet.

In order to evaluate the new tool, a number of factors were taken into consideration like the time required by the annotation process and the accuracy of the annotations decided by a human annotator.

The time is in direct correlation with the length of the document being processed and annotated, and it ranges from 0.8 seconds for a 500 words document to 4.6 seconds for a 3000 words document. This has been measured on a 3 GB RAM, 2.4. GHz machine.

In the evaluation stage, the application was tested on a number of 35 documents written in English, having a length on average of 1700 words. After the automatic annotation of the documents, 1427 annotation were generated.

In order to determine the accuracy of the annotations, one human annotator verified every annotation. Next, the accurate and inaccurate annotations were counted in order to compute the measures of precision and recall. So, the precision for this set of documents was determined to be 78.3% and the recall to be 69.1%.

These results could be explained by the part-of-speech tagger used, which has an accuracy of 97.32% and also by the accuracy of the WordNet lemmatizer, but also by the fact that at the moment the system, does not use word disambiguation techniques.

4 Conclusion and Future Work

The goal of the new editor is to provide a way of annotating documents using concepts from an ontology. In the same time, the annotations made, are valid under the RDFa recommendation. In this way, the main goal of Semantic Web, that of presenting information so that computers can understand and utilize it, is achieved.

One of the current limitations of the application is that it only annotates Nounphrases. The annotation process makes use of only a taxonomy extracted from the ontology. In the future, the annotation process could be extended so that adjectives are also annotated.

Another limitation is that the current version of the software supports only documents written in English and having the .txt extension. In the future, the system will be improved, so that a spidering component is added. It will extract content directly from the web pages, which will be afterwards annotated.

At the moment, when a noun is annotated, all the nouns that share the same lemma with the selected token, are also annotated with the same concept. This process could be further extended in the near future so that synonyms of the selected noun, obtained with the help of synsets from WordNet, could be also, automatically, annotated with the same concept.

Another question that arises is if it would be possible to store the annotation files on a server so that other persons would have access to them, which will surely broaden the perspective of the application. The application would then become a client-server application. This would raise some issues, that would have to be taken in consideration, like concurrent changes of an annotation file.

Another improvement that could be made is to increase the speed of the document parsing. This can be achieved by altering the implementation of the parsing algorithm, so that it becomes more time efficient.

As a conclusion, the new system presented has achieved its goal - it provides a new way of semi-automatically annotating documents. The annotations constructed are based on the RDFa recommendation, and in this way they can be further used by specialized systems. This proves to be one of the main advantages of the new platform. Nonetheless, improvements can and will be made to the system, that will help make it more efficient and flexible.

Acknowledgments

I would like to express my gratitude to Prof. dr. Monica Tataram, University of Bucharest and to Prof. dr. Walther von Hahn and Prof. dr. Cristina Vertan, University of Hamburg, for their supervision, help and guidance.

References

Ben Adida, Mark Birbeck, Shane McCarron and Steven Pemberton. 2008. *RDFa in XHTML - Syntax and Processing.* June 2011 <http://www.w3.org/TR/rdfa-syntax/>.

Mark Birbeck and Ben Adida. 2008. *RDFa Primer.* June 2011 <http://www.w3.org/TR/xhtml-rdfa-primer/>.

Sam Chapman, Alexiei Dingli and Fabio Ciravegna. 2004. *Armadillo: Harvesting Information for the Semantic Web.* 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 25-27 July 2004, Sheffield, UK.

Stephen Dill, Nadav Eiron, David Gibson, Daniel Gruhl, R. Guha, Anant Jhingran, Tapas Kanungo, Kevin S. Mccurley, Sridhar Rajagopalan and Andrew Tomkins. 2003. *A Case for Automated Large Scale Semantic Annotation.* Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 1, No. 1: 115-132.

Siegfried Handschuh, Steffen Staab and Fabio Ciravegna. 2002. *S-CREAM Semi-automatic CREAtion of Metadata.* Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web.

Herman Ivan. 2011. *W3C Semantic Web Activity*. July 2011. <http://www.w3.org/2001/sw/>.

Herman Ivan. 2009. *W3C Semantic Web Frequently Asked Questions.* July 2011. <http://www.w3.org/RDF/FAQ>.

Atanas Kiryakov, Borislav Popov, Dimitar Manov, Damyan Ognyanoff, Rosen Marinov and Ivan Terziev. 2004. *Automatic Semantic Annotation with KIM.* 3rd International Semantic Web Conference (ISWC2004).

Nadzeya Kiyavitskaya, Nicola Zeni, James R. Cordy, Luisa Mich and John Mylopoulos. 2005. *Semi-Automatic Semantic Annotations for Web Documents.* Proceedings of SWAP 2005, 2nd Italian Semantic Web Workshop.

Paul Kogut and William Holmes. 2001. *AeroDAML Applying Information Extraction to generate DAML annotations from Web Pages.* Proceedings of K-CAP 2001.

Jacob Köhler, Stephan Philippi, Michael Specht, and Alexander Rüegg. 2006. *Ontology based text indexing and querying for the semantic web*. Journal Knowledge-Based Systems archive Vol. 19 No. 8

Lothar Lemnitzer and Paola Monachesi. 2007. *Keyword extraction for metadata annotation of Learning Objects.* Workshop on Natural Language Processing and Knowledge Representation for eLearning environments

Lothar Lemnitzer, Cristina Vertan, Alex Killing, Kiril Simov, Diane Evans, Dan Cristea and Paola Monachesi. 2007. *Improving the search for learning objects with keywords and ontologies*. EC-TEL 2007 - Second European Conference on Technology Enhanced Learning.

Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze. 2008. *Introduction to Information Retrieval.* Cambridge University Press, Cambridge, UK.

Frank Manola and Eric Miller. 2004. *RDF Primer.* June 2011 <http://www.w3.org/TR/rdf-primer/>.

Matthew Petrillo and Jessica Baycroft. 2010. *Introduction to manual annotation.* July 2011. <http://gate.ac.uk/wiki/IntroToManualAnnotation_April2010.pdf>.

Borislav Popov, Atanas Kiryakov, Angel Kirilov, Dimitar Manov, Damyan Ognyanoff and Miroslav Goranov. 2003. *KIM - Semantic Annotation Platform*. Proceedings of the International Semantic Web Conference.

Borislav Popov, Atanas Kiryakov, Damyan Ognyanoff, Dimitar Manov and Angel Kirilov. 2004. *KIM – a semantic platform for information extraction and retrieval*. Natural Language Engineering, Vol. 10, No. 3-4: 375-392.

Lawrence Reeve and Hyoil Han. 2005. *Survey of Semantic annotation platforms.* Proceedings of the 20th Annual ACM Symposium on Applied Computing, Web Technologies and Applications track.

Antonio Sanfilippo, Stephen Tratz, Michelle Gregory, Alan Chappell, Paul Whitney, Christian Posse, Patrick Paulson, Bob Baddeley, Ryan Hohimer and Amanda White. 2006. *Automating Ontological Annotation with WordNet.* Proceedings of SemAnnot 2005

David Vallet, Miriam Fernandez and Pablo Castells. 2005. *An Ontology-Based Information Retrieval Model.* Proceedings of Extended Semantic Web Conference.

Semantic Annotation. June 2011. <http://www.ontotext.com/kim/semantic-annotation>.