Extraction of Semantic Metadata for a Web 2.0 Site

By

Riemann Dorval Joseph

A thesis submitted in partial fulfillment of the requirements for the Degree of

MASTER OF SCIENCE

In
Computer Engineering
UNIVERSITY OF PUERTO RICO
MAYAGÜEZ CAMPUS
2010

Approved by:

Vega Riveros, José Fernando, PhD. President, Graduate Committee	Date
Nayda Santiago, PhD. Member, Graduate Committee	Date
Manuel Rodriguez, PhD. Member, Graduate Committee	Date
Dr Dorial Castellanos Graduate School Representative	Date

ABSTRACT

This thesis describes the architecture of a metadata extraction and a search engine using Semantic Web technology. The lack of studies about extraction of semantic metadata within PowerPoint or Open Office Presentations generates the following problems: 1) Presentations is one of the most widely used communication tools; and 2) Presenters often find themselves wasting a long time looking for information from previous presentations. Thus, we designed an application that allows: to access the information inside an Open Office Presentation and generate the metadata from this information and to go through the metadata and look for the presentations that match search criteria entered by a user. RDF (Resource Description Framework) and NLP (Natural Language Processing) are the main technologies used in our research. Tests were conducted to measure the quality of results, and then were compared to the results from other well known systems such as Google Desktop and Windows Search.

RESUMEN

Esta tesis describe la arquitectura de un motor para extraer metadatos y un motor de búsqueda usando tecnología de “Semántica Web”. Encontramos que hacen falta estudios sobre la extracción de metadatos semánticos dentro de presentaciones de “PowerPoint” u “Open Office”. La importancia de este problema estriba en dos asuntos de interés: 1) Presentaciones es una de las herramientas de comunicación ampliamente más utilizada; y 2) Los presentadores gastan con frecuencia mucho tiempo buscando una o varias diapositivas de presentaciones anteriores que podrían ser usadas en su próxima presentación. Por esta razón, hemos diseñado una aplicación que permite acceder a la información dentro de una presentación de “Open Office” y generar los metadatos correspondientes, e ir a través de ellos para buscar las presentaciones que coinciden con un criterio de búsqueda introducido por un usuario. RDF (“Resource Description Framework”) se utiliza para construir la ontología del sistema. Procesamiento de Idioma Natural (NLP por su sigla en Inglés) puede procesar y analizar la información en lenguaje natural proveniente de las presentaciones o de las solicitudes de los usuarios. Se realizaron pruebas para medir la calidad de los resultados. Los resultados se compararon con resultados de otros sistemas conocidos como “Google Desktop” y “Windows Search”.

.

Dedication

To my father Emmanus, my mother Rita, my brother, my sister, and my friends for all the unconditional support they have given me.

ACKNOWLEDGMENTS

To Dr. J. Fernando Vega-Riveros for the unconditional support and advice for carrying out this work. And I also thank the members of the Graduate Committee doctors Manuel Rodriguez and Nayda Santiago. I want to thank all my friends and especially my girlfriend for their support. Finally, I thank Hewlett-Packard Philanthropy (HP) for their financial support for this research from summer 2007 to December 2008.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

APPENDIX LIST

CHAPTER 1

INTRODUCTION

With the promising Semantic Web technologies, it is possible that not only humans but also machines can interpret and retrieve information and services. Even with the benefits of Semantic Web technologies, back to 2006 there were no adequate tools or applications, which reveal the potential of semantically encoded knowledge and services over the Semantic Web [M. Stanojevic05]. Semantic Web relies today on the use of various ontology and schema languages to represent the semantics of Web pages [M. Stanojevic05]. Concerning this case, the World Wide Web Consortium (W3C) developed alternatives to coding the information in the Web pages, in order to give greater meaning and reduce ambiguity in the information resources. One of these alternatives is the Resource Description Framework (RDF) and its extension, the Resource Description Framework Schema (RDF-S), which offers a set of primitives for the representation of knowledge models which is similar to alternatives based on marks [Gomez02]. However, to understand the content of those pages, Natural Language Understanding is necessary since the information in the documents is generally made of words or sentences.

RDF is a language for representing information on the web and describing relationships among resources in terms of named properties and values. Indeed, it is a common language for expressing information, especially metadata [Bouzeghoub04]. In order to describe the properties of the resources and their relations with other resources, the language RDF-S is used. It allows defining a taxonomy of the resources based on classes and their associated properties. A taxonomy is a particular case of an ontology where the relations are limited to class-subclass relations.

Programming languages are important in the representation of the elements of an ontology. An ontology generally is a hierarchical representation of objects and types of objects, their relations and properties. An ontology allows representing the knowledge in a domain. The knowledge provided by the ontology can be represented through languages such as RDF and RDF-S with which the metadata of the information resources is generated. The search and recovery of this information can be realized by search engines or software agents. In particular, agents present important characteristics, such as autonomy, pro-activity, and cooperation, among others.

Research on Natural Language Processing (NLP) started with the so-called rule-based methodology; however, compilation of a huge amount of grammar rules and dictionary entries is too difficult when developing practical systems [Isahara07]. Then, the trend of NLP research has been shifting to corpus-based or statistical, systems. Thanks to the rapid improvement of computer power and data storage, nowadays we can utilize huge amounts of actual linguistic data. Combining such linguistic resources with a high quality analyzer, we can extract useful linguistic information and develop practical systems for a specific domain [Isahara07].

In this thesis we set out to apply the technologies previously mentioned, in the description, representation, metadata extraction and search of Open Office Presentations within a repository. One of the main problems that we have at present is the lack of studies about extraction of semantic metadata within a PowerPoint or Open Office Presentation. This problem actually exists because of the scarcity of mechanisms or tools to access the actual contents (title text, bullet text…) inside the PowerPoint or Open Office Presentation. In other words, access to knowledge collected in the past through presentations is practically and almost inexistent.

In this investigation, we designed an application with the following responsibilities:

- Access the information inside an Open Office Presentation and generate the metadata from this information.

- Go through the metadata and look for the presentations that match a search criteria entered by a user.

The application was coded in Java using Eclipse SDK (Software Development Kit). An API (Application Programming Interface) called Jena was also used. Jena allows constructing and reading ontologies in RDF or RDF-S. One very important element of the application is the construction of an ontology based on words: nouns, verbs and adjectives. Those words that are present in the Vocabulary Ontology will be the key in defining our metadata for each Open Office Presentation. Finally, a NLP parser was used in order to analyze the information inside the presentations. The parser was developed by the Stanford Natural Language Processing group of Stanford University.

Using the technologies mentioned above will prove to be very significant in the effective extraction and retrieval of metadata inside the presentations.

1.1Motivation

Microsoft's PowerPoint is the most commonly used presentation software. In education, it is by far the most used by professors when teaching their students [Amare04]. When someone is preparing a presentation for an audience on a specific subject, sometimes it helps to have other past presentations about the same subject and use them as a guide. There are many sites on the internet that include PowerPoint presentations. When a search is realized on those sites, the match is made between the topic search and the name of the document while there are still some that go into the contents but without a semantic approach. Presentations are used to present ideas, thoughts, concepts… While the name of the document may summarize what the presentation is about, the contents of the presentation cannot be neglected. Getting into the content of the slides and extracting even more metadata for a presentation instead of limiting ourselves only to the document name is of great importance to represent the ideas, thoughts, and concepts inside the presentation. Extracting the information directly from PowerPoint presentations is not that simple. By converting the PowerPoint presentations into Open Office presentations, we can now use tools and software that allows us to access to content inside the presentations. Using the information within the presentation and then analyzing it bringing into play Natural Language Processing can create powerful metadata for a repository entirely made of Open Office presentations.

Current initiatives like *Dublin Core Metadata Initiative* (DCMI) [Beckett02], *Resource Description Framework* (RDF) [Miller98] and *Ontology Inference Layer* (OIL) [Gomer02] are being developed in order to identify and describe knowledge and information resources. It is important that the technologies of knowledge management are investigated and developed.

Those initiatives are important to facilitate the description of different resources used in the presentations that are part of the learning materials, whose knowledge contents are not being shared and accumulated effectively and explicitly by students and professors. This is our motivation to investigate and develop tools that allow storing and retrieving information from all those documents that are so frequently used in conferences, meetings and classrooms.

1.2 Objectives

The main goal of this thesis is, by using the technologies of the Semantic Web and Natural Language Processing, to first design a system capable of reading the contents of slides inside an Open Office Format Presentation in order to extract the key ideas and keywords for each slide in a repository of presentations. The key ideas and words will be stored and used as metadata.

The specific objectives of the proposed research are:

- To use Artificial Intelligence techniques to extract semantic Metadata from Open Document Format Presentations

- To store the metadata extracted from the document in a database (an RDF file)
- Build a *semantic search engine* capable of reading the database (the RDF file) and returning the addresses of the slides or presentations related to the topic to search.

1.3 Contribution

The main contributions of this thesis are:

- The development of an ontology for a set of Open Office presentations
- The development of a system capable of retrieving metadata and finding a match with the user search topic and finally returning the addresses of presentation(s) or slide(s) where the matching is positive.
- The development of a priority approach where the best matches from the search are placed first in the results returned.
- The development of an approach where matching the metadata and user search topic is taken even further by considering synonyms in the matching process.

1.4 Organization of thesis

This thesis is divided into eight chapters. The second chapter describes the State of the Art of the technologies related to this investigation. The third chapter presents the previous work that inspired us in developing our system. The fourth chapter describes the construction of our Vocabulary Ontology and the representation of the metadata in a semantic language. The fifth chapter presents the system with the extraction and retrieval of the metadata. The sixth chapter describes the methodology to realize the tests. The seventh chapter presents the results of the tests. The eighth chapter concludes the thesis.

CHAPTER 2

STATE OF THE ART

2.1 Introduction

In this chapter we will give a description of the technologies that are currently used in the Semantic Web. Section 2.2 describes the ontology and the new languages for knowledge representation. In section 2.3 the languages to write ontologies are described. In section 2.4 Natural Language Processing is illustrated.

2.2 Semantic Languages and Ontologies

In recent years, many organizations, research groups, and Internet communities have been developing technologies, tools and languages to represent knowledge and make it possible to share and re-use this knowledge. The World Wide Web Consortium (W3C) is the most representative organization developing tools and languages to represent resources on the Web. Languages such as RDF and RDF-S are used to express descriptions of resources on the Web that result into creating one or more ontologies. Other languages, developed by other organizations, are the DARPA Agent Markup Language + Ontology Inference Layer (DAML+OIL) and Web Ontology Language (OWL) that allow more sophisticated representations of Web resources. We will now present a brief description of these languages.

2.2.1 RDF

The Resource Description Framework (RDF) is an infrastructure that enables the encoding, exchange, and reuse of structured metadata. RDF is an application of XML that imposes structural constraints to provide unambiguous methods of expressing semantics. RDF additionally provides a means for publishing both human-readable and machine-processable vocabularies designed to encourage the reuse and extension of metadata semantics among disparate information communities. The structural constraints RDF imposes to support the consistent encoding and exchange of standardized metadata provides for the interchangeability of separate packages of metadata defined by different resource description communities [Miller98].

RDF has a data modeling capability which allows us to represent expressions based on a model formed by three elements:

· *Resources*: all the objects that can be expressed with RDF expressions are called resources; for example, a page Web or part of a Web page; also it can be a set of Web pages or a document.

· *Properties*: they are specific aspects, attributes or relations that describe a resource.

· *Expression*: a specific resource along with the name of a property with its value forms an RDF expression. These three elements forming an expression are called: subject, predicate and object

The following table identifies the three elements of a RDF expression for the sentence:
"Riemann is a member of FaceBook http://www.new.facebook.com/Riemann"
The different parts are presented in table 2-1:

Table 0-1 Parts of an Expression

Subject (Resource)	http://www.new.facebook.com/Riemann
Predicate (Property)	Member
Object (Literal)	Riemann

This expression can also be represented graphically. In the graphic representation, the arcs represent the properties, the resources are represented by elliptic nodes and the rectangular nodes represent the literals (known as the values of the property) as shown in Figure 2.1.

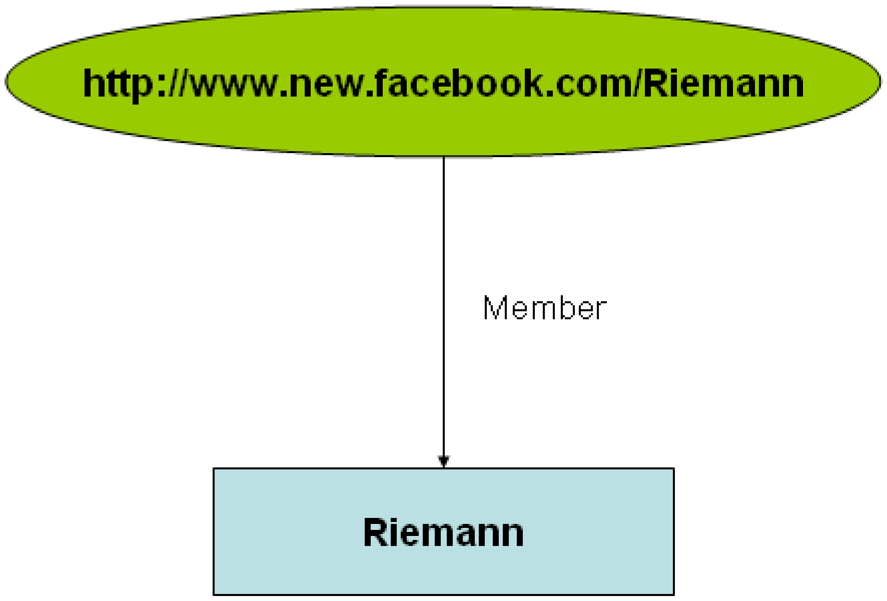


Figure 0.1 Graph of a RDF Expression

RDF provides a set of terms and syntax to represent an expression. The previous expression can be written as presented in figure 2.2:

```
<?xml version= “1.0”>
<rdf:RDF>
    <rdf:Description about=”http://www.new.facebook/Riemann”>
        <s:Member>Riemann</s:Member>
    </rdf:Description>
</rdf>
```

Figure 0.2 RDF Expression

2.2.2 RDF-S

Resource Description Framework Schema [Broekstra01] provides mechanisms to define a vocabulary for RDF data. RDF-S will allow defining attributes which will identify the characteristics of the resources, and the relations between the resources. It is possible to label the resources that will be used, to restrict possible combinations of classes, relations and to detect violations to the restrictions.

RDF-S has a set of terms which will allow us to create valid RDF-S expressions. As a result, the following terms: Class, subClassOf, and Property will allow us to build expressions about the resources (Class), to represent their properties and simultaneously represent a hierarchy of resources. The objects could be instantiated from the classes using the property `type`. The restrictions on properties can be specified using the `domain` and `range` properties. Figure 2.3 shows an example of representation in RDF-S.

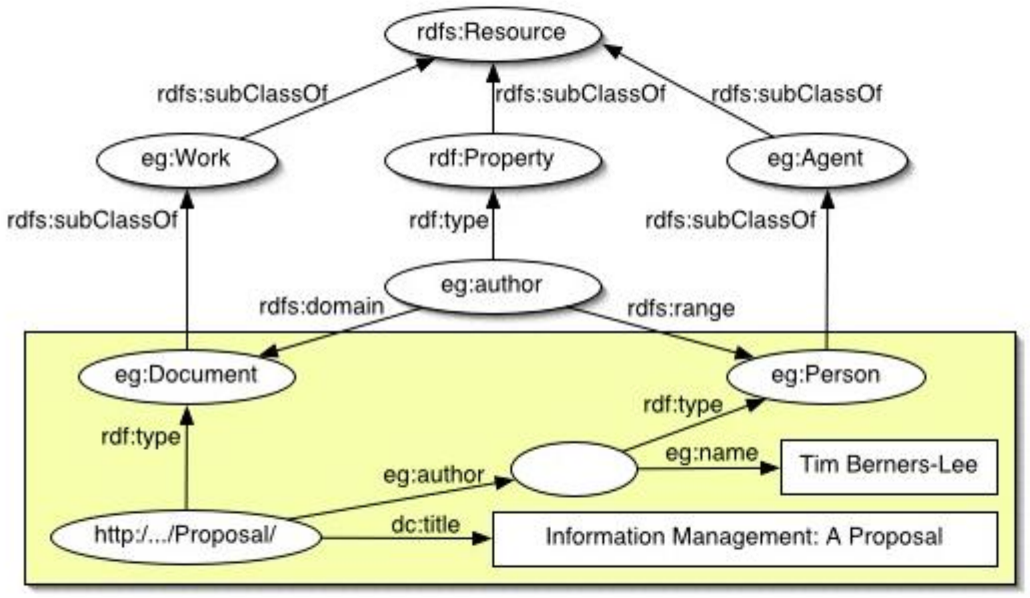


Figure 0.3 Example of a RDF Schema [Stephen02]

In the example shown in Figure 2.3 above, we have the following classes: Resource, Work, Agent, Person, author, Document. Title and name are defined as properties. We also have in the figure an instance of the class document: "http:/…./proposal".

2.2.3 Ontology Inference Layer

Ontology Inference Layer (OIL) is one of the languages to represent ontologies. OIL provides modeling primitives commonly used in frame-based approaches to ontological engineering (concepts, taxonomies of concepts, relations, and so on), and formal semantics and reasoning support found in description logic approaches [Gomez02].

OIL matches the following criteria to be efficient working with ontologies:

- It must be highly intuitive to the human user. Given the success of the frame-based and object-oriented modeling paradigms, an ontology should have a frame-like look and feel.
- It must have a well-defined formal semantics with established reasoning properties to ensure completeness, correctness, and efficiency.
- It must have a proper link with existing Web languages such as XML and RDF to ensure interoperability.

Furthermore, OIL also unifies the three important aspects that different communities provide: epistemologically rich modeling primitives as provided by the frame community, formal semantics, and efficient reasoning support as provided by description logics, and a standard proposal for syntactical exchange notations as provided by the Web community.

2.2.4 Darpa Agent Markup Language + OIL

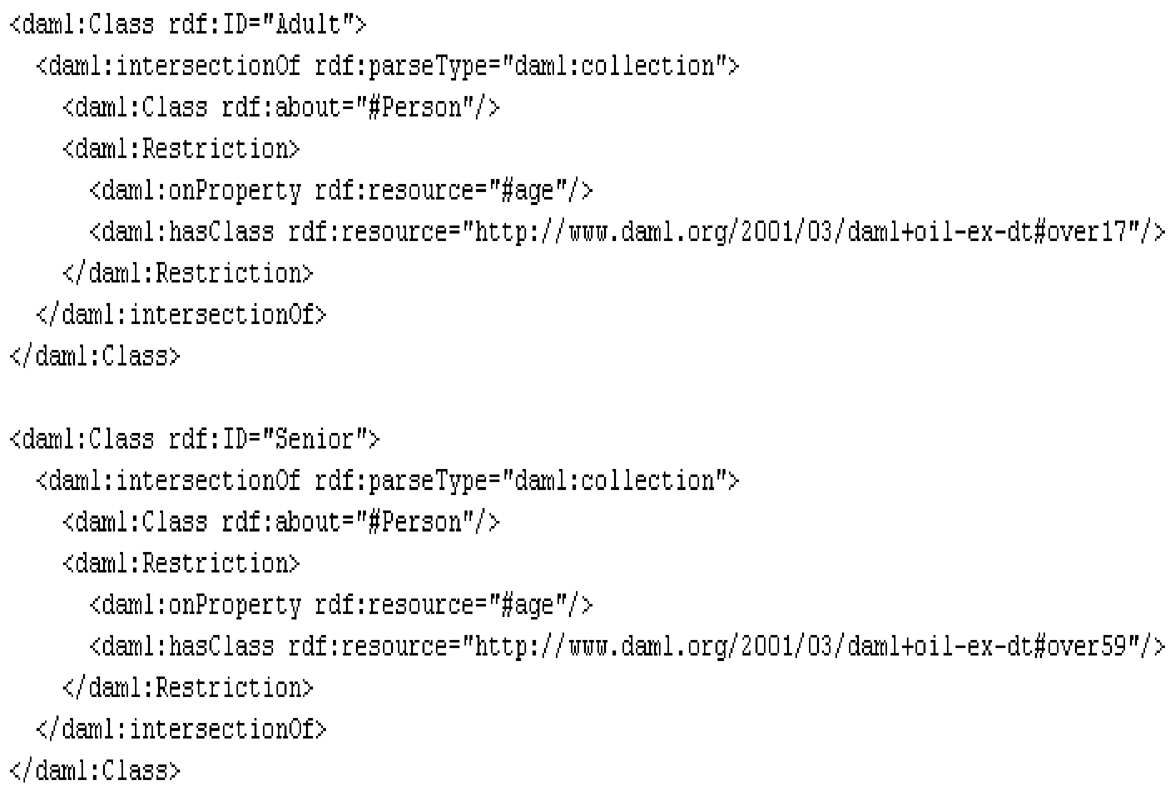
Languages are required to allow expressing the resources of the web with more details and clarity. Those descriptions can easily be used in automatic reasoning. For such a purpose, OIL and DAML+OIL ontology languages were developed. They are known as extensions of RDF-S with a rich set of primitives. The main characteristics presented by these languages are

- An underlying mapping to an expressive Description Logic (DL) provides a well defined semantics and a clear understanding of the language’s formal properties. The DL gives DAML+OIL the ability and flexibility to compose classes and slots to form new expressions.
- A machine-readable syntactic encoding in the languages of the web. RDF-S is a proposed mechanism for deploying metadata. As mentioned above, DAML+OIL

is defined as an extension of RDF-S, which makes its ontology accessible to any RDF-S aware application.

- A layered architecture, avoiding the temptation to throw everything into the core language, mixing up features that cannot be reasoned over with those that can be. As a result the limits are clear and explicit.

DAML+OIL is an important language developed by DARPA and OntoKnowledge, which presents or displays very important characteristics for the construction of ontologies, capturing knowledge and automatic reasoning. Figure 2.4 illustrates the representation of an ontology using DAML+OIL



```
<daml:Class rdf:ID="Adult">
  <daml:intersectionOf rdf:parseType="daml:collection">
    <daml:Class rdf:about="#Person"/>
    <daml:Restriction>
      <daml:onProperty rdf:resource="#age"/>
      <daml:hasClass rdf:resource="http://www.daml.org/2001/03/daml+oil-ex-dt#over17"/>
    </daml:Restriction>
  </daml:intersectionOf>
</daml:Class>

<daml:Class rdf:ID="Senior">
  <daml:intersectionOf rdf:parseType="daml:collection">
    <daml:Class rdf:about="#Person"/>
    <daml:Restriction>
      <daml:onProperty rdf:resource="#age"/>
      <daml:hasClass rdf:resource="http://www.daml.org/2001/03/daml+oil-ex-dt#over59"/>
    </daml:Restriction>
  </daml:intersectionOf>
</daml:Class>
```

Figure 0.4 Representation of an Ontology Using DAML+OIL

2.2.5 Web Ontology Language (OWL)

OWL is intended to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans. OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. As defined in Chapter 1, this representation of terms and their interrelationships is called an ontology. OWL has more facilities for expressing meaning and semantics than XML, RDF, and RDF-S, and consequently OWL goes beyond these languages in its ability to represent machine interpretable content on the Web [McGuinness04]. Figure 2.5, following the description of the OWL sublanguages, represent an example of an ontology using OWL

OWL provides three increasingly expressive sublanguages designed for use by specific communities of implementers and users.

- *OWL Lite:* It supports those users primarily needing a classification hierarchy and simple constraints. For example, while it supports cardinality constraints, it only permits cardinality values of 0 or 1

- OWL *DL*: It supports those users who want the maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). OWL DL includes all OWL language constructs, but they can be used only under certain restrictions.

- OWL *Full* is meant for users who want maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. For example, in OWL Full a class can be treated simultaneously as a collection of individuals and as an individual in its own right. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary.

```
<owl:Class rdf:about="#MusicDrama">
  <owl:equivalentClass>
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Opera"/>
        <owl:Class rdf:about="#Operetta"/>
        <owl:Class rdf:about="#Musical"/>
      </owl:unionOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>

<owl:Class rdf:about="#Opera">
  <rdfs:subClassOf rdf:resource="#MusicDrama"/>
</owl:Class>

<owl:Class rdf:about="#Operetta">
  <rdfs:subClassOf rdf:resource="#MusicDrama"/>
  <owl:disjointWith rdf:resource="#Opera"/>
</owl:Class>

<owl:Class rdf:about="#Musical">
  <rdfs:subClassOf rdf:resource="#MusicDrama"/>
  <owl:disjointWith rdf:resource="#Opera"/>
  <owl:disjointWith rdf:resource="#Operetta"/>
</owl:Class>
```

Figure 0.5 Representation of an Ontology Using OWL

2.2.6 Ontology and knowledge representation

The Web has become a major platform to provide information retrieval. In the past, people browsed the web sites to gather information they needed. In fact, processing was done manually not by computers. The rapid advance in hardware facilities and software packages has improved the function of the search engines and increased dramatically the amount of information people can access [Yang04]. One way to represent information is by using an ontology. A complete ontology contains everything to represent knowledge to be used for all application scenarios [Weishan06].

An ontology describes a set of representational primitives used to model a domain of knowledge or discourse [Gruder07]. The representational primitives are represented by classes (or sets), attributes (or properties), and relationships (or relations among class members). The definitions of the representational primitives hold information about their meaning and constraints. On a database systems approach and view, an ontology can be considered as a level of abstraction of data models, similar to hierarchical and relational models. But the ontology is proposed for modeling knowledge about individuals, their attributes, and their relationships to other individuals. Ontologies are normally specified in languages that tolerate abstraction clear of data structures and implementation strategies. Basically, ontology languages are a better suit in expressive power to first-order logic in comparison to languages used to model databases. For this reason, ontologies are said to be at the "semantic" level, whereas database schema are models of data at the "logical" or "physical" level.

Ontologies are also used in the field of Artificial intelligence. In this field an ontology is a formal and explicit description of concepts of a domain, and properties of each concept. The properties describe the characteristics and attributes of the concepts [Fridman01]. Figure 2.6 shows an ontology for fruits and vegetables.

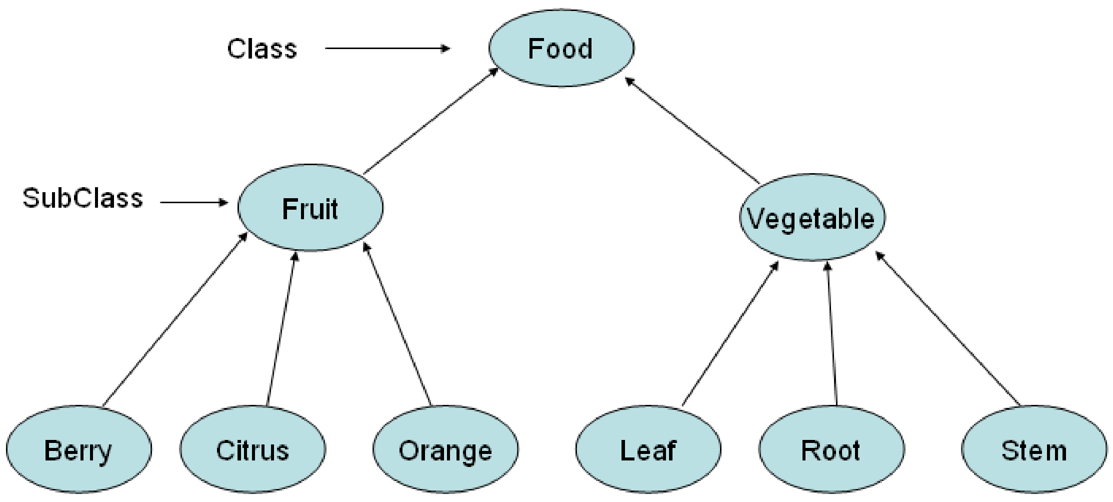


Figure 0.6 Representation of an Ontology of Fruit and Vegetable

As stated in Chapter 1, this thesis also includes the analysis of texts in Open Document Format obtained from Power Point or Open Office Impress files. For this purpose natural language processing (NLP) is used.

2.3 Natural Language Processing

Natural Language Processing is a subfield of artificial intelligence and computational linguistics. It studies the problems of automated generation and understanding of natural human languages. Natural language understanding systems convert samples of human language into more formal representations that are easier for computer programs to manipulate [M. Stanojevic05]. Natural Language Processing consists mainly of stochastic NLP and deterministic NLP. In mathematics, a model is said to be deterministic if it does not involve the concept of probability; otherwise it is said to be stochastic [Joakim02]. The stochastic NLP presents an important advantage over the deterministic NLP. When there is a grammar error, most (if not all) deterministic NLPs exit with an error code or flag, while stochastic NLP carries the analysis and produces some results. Using stochastic NLP will serve our cause better. We are working with sentences inside presentations that are likely to contain errors.

CHAPTER 3

RELATED WORK

3.1 Introduction

Studies on extraction of metadata from Open Office presentations are inexistent. Therefore, we will review in this chapter previous work carried out on the extraction and the retrieval of semantic Metadata from Web resources. Even if the methods of extraction are not directly related to Open Office presentations, their approach has been a great influence for defining our basis for the extraction of metadata from the presentations.

3.2 Support Vector Machine (SVM)

Hui Han and his group [Hui03] describe a Support Vector Machine classification-based method for metadata extraction from the header part of research papers.

The metadata extraction algorithm is divided in two major steps: line classification (SVM classification) and feature extraction. Considering a two class classification problem, the SVM attempts to find an optimal hyperplane (linear transformation kernel) to maximally separate the classes. The corresponding decision function is called a classifier. In the feature extraction, they make use of both word and line-specific features to represent their metadata. They designed a rule-based, context-dependent word clustering method with the rules extracted from various domain databases and orthographic properties of words. Their rule-based method relies on the prior knowledge embedded in domain databases.

They collect the following databases to gather a priori knowledge of the domain:

1. Standard on-line dictionary of Linux system
2. Bob Baldwin's collection of 8441 first names and 19613 last names
3. Chinese last names
4. USA state names and Canada province names
5. USA city names
6. Country names from the World Fact Book, month names, and their abbreviations

They then cluster words and bigrams based on their membership in the domain databases and their text orthographic properties. The words and bigrams in the same cluster are represented by a common feature, called word-specific feature. They define the weight of a word-specific feature as the number of times this feature appears in the sample (line).

They use a two-step algorithm for classifying text lines into a single class or multiple classes. The two steps are: the independent line classification followed by the contextual line classification. In the first step, feature vectors are generated based on the feature extraction methodology. The second step makes use of the sequential information among lines. After classifying each line into one or more classes, they extract metadata from each multi-class line based on the predicted class labels for this line. In our work, the information inside the Open Office presentations will be separated into sentences and bullets.

3.3 Text Mining Approach

In this work, Hsin [Hsin05] presents a semantics extraction method using a text mining approach. There are three basic requirements for the text mining process to be applicable to the problem. The first is that the process should be fully automatic without, or with a negligible amount of, human intervention. The second is that it should be generalizable and scalable. The last one is that it should be able to extract the real semantics of the web pages and present it in a human comprehensible way.

To obtain the metadata from a web page, first they perform a clustering process on a training set of web pages (web pages are trained by the self-organizing map (SOM), algorithm which is a neural network-based algorithm to generate a feature map, namely the keyword cluster map (KCM)). Then they apply a text mining process to the clustered result in order to obtain the metadata for each page. The clustering process is followed by a labeling process which is applied to the trained result to construct a feature map which characterizes the relationship among keywords. Finally comes the metadata generation process which is applied to the map in order to develop an ontology that is used as the metadata.

Their approach begins with a standard practice in information retrieval: the vector space model, to encode web pages with vectors, in which each element of a web page vector corresponds to a different keyword. Here they use a binary vector to represent a web page. A value of 1 for an element in a vector indicates the presence of the corresponding keyword in the web page; otherwise, a value of 0 indicates the absence of the keyword. After the encoding process, they organize the keywords into clusters by their co-occurrence similarities. Applying the SOM algorithm to the web page vectors, they actually perform a clustering process about the corpus. To decide the cluster to which a keyword belongs, they apply a labeling process to the keywords.

The labeling process establishes the association between each keyword and one of the neurons (the network consists of a regular grid of neurons). The map forms the KCM by labeling each neuron with certain words. After the labeling process, each keyword is associated to a neuron in the map. They record such associations and form the keyword cluster.

The metadata of a web page in this work consists of four parts. The first part is the *important terms* section which contains a set of high-frequency terms appearing in a given web page. The *important terms* section approach will be used to define our Vocabulary Ontology discussed later in Chapter 4. The second part is the *important themes* section which is a set of identified themes that can reflect the main interest of the web page. The third part is the *related themes* section which depicts some themes related to those in the second part and also the relationships among the themes. Finally, they include a *related pages* section which contains a set of web pages that are similar to the annotated one. The main contribution of this work is that it incorporates semantic annotation and ontology creation in a unified framework. Furthermore, their methods require no predefined ontology or human intervention, and can be applied to dynamic web pages. Although the generated metadata is rather primitive, the author believes that the method will be beneficial to the success of the Semantic Web when it migrates to the existing standards.

3.4 oSEMA

The oSEMA has the architecture of an agent-based system to retrieve information resources for a higher education environment. An ontology representing documents used by students and faculty members is also described, where RDF-S (Resource Description Framework Schema) is used. The RDF-S recommendation offers terms that allow the representation of concepts, their relationships, and their attributes, all of which form the metadata. The architecture uses three types of agent (UserAgent, SearchAgent and OntologyAgent), where each agent is in charge of different tasks such as user interaction, ontology retrieval and metadata search. The prototype called oSEMA was developed using this architecture, to recover information in a distributed environment [Dinos04].

The prototype has three agents (SearchAgent, OntologyAgent and UserAgent,). SearchAgent and OntologyAgent are defined in the main container. The OntologyAgent will be in charge of working directly on the metadata, while the SearchAgent will be in charge of conducting the searches. The last agent (UserAgent) will interact directly with the user to carry out the tasks of presentation of search results and capturing metadata.

The UserAgent is in charge of sending the messages to the other agents according to the actions that the user realizes. A user can conduct a search for information or retrieve the document ontology. This action will be translated in a message to be sent to the SearchAgent or the OntologyAgent, and those agents will process the message and send the result back to UserAgent which displays the results to the user. It can also be used to make annotations about document properties.

The OntologyAgent is in charge of accessing, modifying and updating the different domain ontologies. The messages that arrive at this agent are mainly for obtaining data from the ontology so that it is shown to the user. Some tasks were defined for this agent, for example: ObtainOntologiesBehavior that is in charge of obtaining the structure of the ontology and once the task is processed the result is sent to the agent requesting it.

SearchAgent is another one of the components of the architecture, which is in charge of looking for the metadata of one or more resources according to the coincidences that it finds among them. This agent has the SearchDocumentsBehaviour task that is in charge of processing the messages and extracting the search parameters that arrive along with the message. With these parameters the search of RDF expressions is realized before sending the result back to the agent that originated the request.

3.5 Automatic Extraction of Titles from General Documents using Machine Learning

Yunhua and his group [Yunhua05] focused their extraction on document titles since they claim that title information is useful for document retrieval. They defined some rules in the specification of the document title. The rules include “a title is usually in consecutive lines in the same format”, “a document can have no title”, “titles in images are not considered”, “a title should not contain words like ‘draft’, ‘whitepaper’, etc”, “if it is difficult to determine which is the title, select the one in the largest font size”, and “if it is still difficult to determine which is the title, select the first candidate”.

Title extraction based on machine learning consists of training and extraction. A pre-processing step occurs before training and extraction. During the pre-processing, from the top region of the first page of a Word document or the first slide of a PowerPoint document a number of units for processing is extracted. If a line only has a single format, then the line will become a unit. If a line has several parts and each of them has its own format, then each part will become a unit. Each unit will be treated as an instance in learning.

In learning, the input is a sequence of units where each sequence corresponds to a document. They take labeled units (labeled as title_begin, title_end, or other) in the sequences as training data and construct models for identifying whether a unit is title_begin, title_end, or other.

In extraction, the input is a sequence of units from one document. They employ one type of model to identify whether a unit is title_begin, title_end, or other. They then extract units from the unit labeled ‘title_begin’ to the unit labeled ‘title_end’. The result is the extracted title of the document.

The unique characteristic of this approach is that they mainly utilize formatting information for title extraction. Their assumption is that although general documents vary in styles, their formats have certain patterns and it is possible to learn and utilize the patterns for title extraction. Since most presentation slides include a title, title extraction will be a necessity in our work.

CHAPTER 4

RDF METADATA FILE AND VOCABULARY ONTOLOGY FILE

4.1 Introduction

Metadata can be defined as information about information which makes it of great importance in describing objects, documents, web page resources, etc. Having information about the content, author, or legal conditions makes it easier for humans and computers to classify a resource. Metadata can be useful for:

- summarizing the meaning of the data,
- allowing users to search for the data,
- allowing users to determine if the data is what they want,
- giving information that affects the use of data (legal conditions, size, age, and so on),
- indicating relationships with other resources.

Metadata are highly structured data that describe information, the content, the quality, the condition, and other characteristics of the data. This explains the existence of metadata in documents such as: reports, papers, presentations.... In this chapter, we will describe characteristics of metadata; we will illustrate its representation through one of the Semantic Web languages, and explain the importance of using them in the construction of our ontology. The extraction and retrieval of the metadata will be discussed in the next chapter.

4.2 RDF Metadata File

Metadata are found in the information included in Open Office file presentations. For our study, we obtained some presentations that some professors of the University of Puerto Rico Mayaguez Campus had used or are still using in teaching their classes. Some of those presentations were used with the approval of the professor and others were just presentations that we had in our hands at the time.

4.2.1 Requirement of presentation document

The information that will constitute our metadata in this first approach is the texts found within the Open Office document. Graphics, pictures and videos are beyond the scope of this thesis and will be ignored within the slides of the presentation. For our case study, we have chosen Open Office presentations where the contents inside most of the slides are texts. Consequently, presentations made mostly of graphics, pictures, videos and other multimedia features have been disregarded.

Open office Impress and Microsoft Office PowerPoint presentations offer various common slide layouts. We have also limited our study to four of those layouts:

- Title slide. It corresponds to the layout of the majority of the first slides in the document. It contains: a title textbox where the user usually types the title of the document, and a subtitle textbox where the user usually types the author(s)'s information.
- Title and text. This type of slide may be the most used in a presentation. It also contains a title textbox and a additional textbox

- Title and two textboxes. This layout includes one title textbox and two additional textboxes.
- Title only. Its name says it all; it only contains a title textbox.

We have chosen only those four types of slide for our study case because they are the most commonly used in presentations. Figure 4.1 shows the different layouts

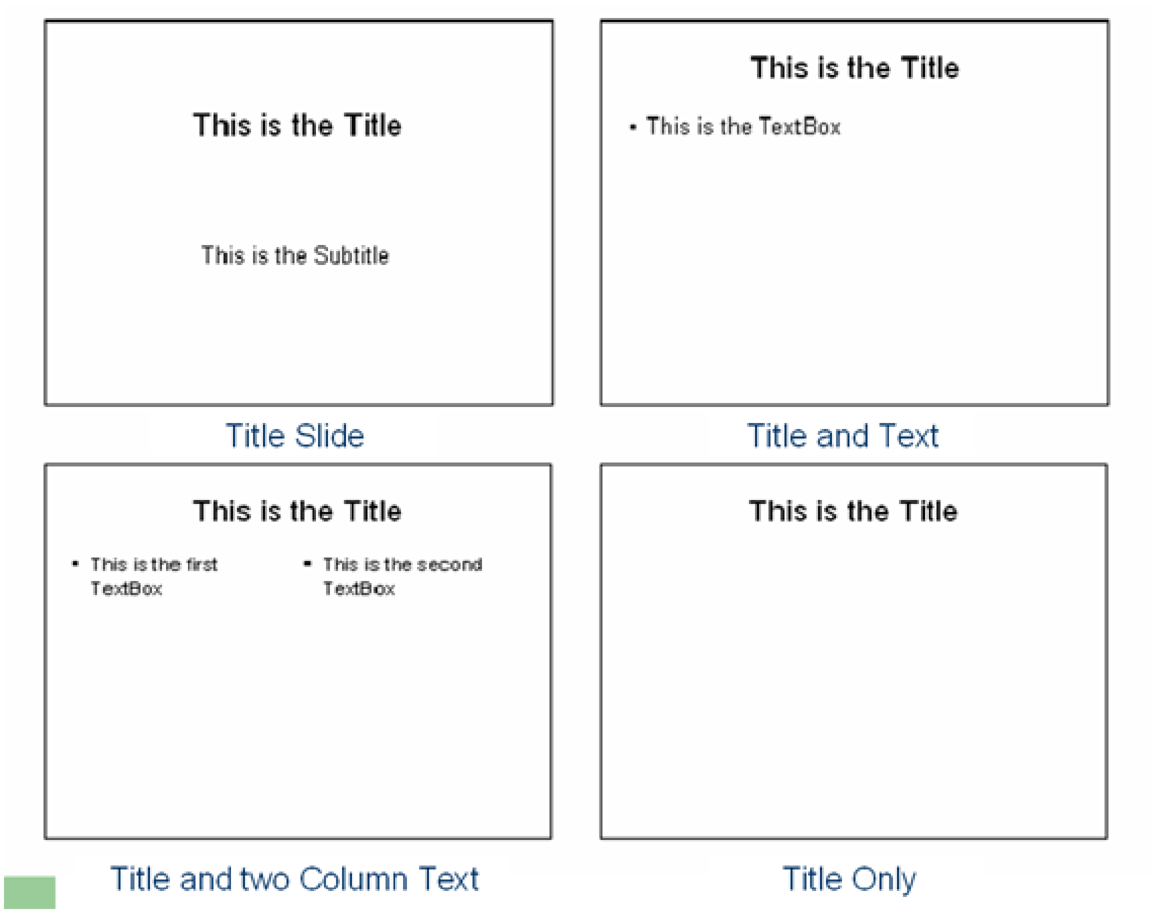


Figure 0.1 The Four Different Layouts of a Slide

4.2.2 The Identified Metadata

The metadata for the presentation consists of information extracted from the texts found inside each slide of the presentation. The texts are first divided into different sentences or phrases, and then they are analyzed before being stored as metadata.

Separation of the text is critical since we are not just extracting the information and storing it as metadata. Most of the information will be processed by a Natural Language Parser mentioned in Chapter 1 available in the following site: “http://nlp.stanford.edu/software/lex-parser.shtml”.The information inside a slide is divided into the following parts:

1. Title. The information inside of the title textbox of the slide
2. Sentences. Different sentences encountered inside textboxes, bullets or autoshapes (rectangle, circle…)
3. Bullets. They are made of sentence(s) or phrase(s).

The properties of the metadata will be determined by our Artificial Intelligence Analyzer (AI Analyzer) which will process the information from the presentations. The AI Analyzer will be discussed later in Chapter 5. The different metadata properties generated by the AI Analyzer are:

1. TitleDocument. The title of the document or the title found in the first slide of the presentation.
2. TitleSlide. The title of all the other slide(s) following the first slide.
3. NounverbSubjectObject. Sentences with a conjugated verb that have a direct or indirect object complement.
4. NounverbSubjectPrep. Sentences with a conjugated verb that have a prepositional complement.
5. NounNounSubject. Sentences where the verb “to be” is the main conjugated verb.
6. NounPhrase. Sentences without a conjugated verb.
7. NounAdjective. Sentence where noun(s) and adjective(s) are related.

Each metadata stored in our RDF Metadata File is a RDF triple where:

- The subject (resource) is the URI of the presentation
- The predicate (property) is one of the following properties mentioned above

- The object (literal) is the actual information extracted from the presentation

With the conversion and synonym approach developed in our system for extraction and retrieval of metadata, we designed a Vocabulary Ontology. The Vocabulary Ontology defines a lexicon with a limited set of nouns, verbs and adjectives.

4.3 The Vocabulary Ontology

The Vocabulary Ontology is a set of nouns, verbs and adjectives that must be present inside a sentence in order to be considered as metadata. Not every single word inside a sentence may be in our Vocabulary Ontology, but the presence of at least one of them is required. The nouns, verbs and adjectives that define our Vocabulary Ontology are determined by their frequency of use inside of the presentations that we will be used for our study. We designed an external program to extract all nouns, verbs and adjectives within the presentations and then, we were able to count them without taking into account neither the plural of the nouns nor the conjugation of the verbs. Based mostly on their frequency of appearance, we then conclude which of them will be used in our Vocabulary Ontology.

4.3.1 Construction of the Vocabulary Ontology

The construction of the Vocabulary Ontology allowed us to define a common knowledge vocabulary based on the information inside the presentations. With the identified set of words for the Vocabulary Ontology, we started to determine the classes in the ontology and their respective properties. After stating the classes, we started to construct the hierarchy of the classes. A top-down approach was applied to design the hierarchy of the classes where we began with the most general concept in the domain and broke it down into a subsequent specialization of classes as shown in Figure 4.2.

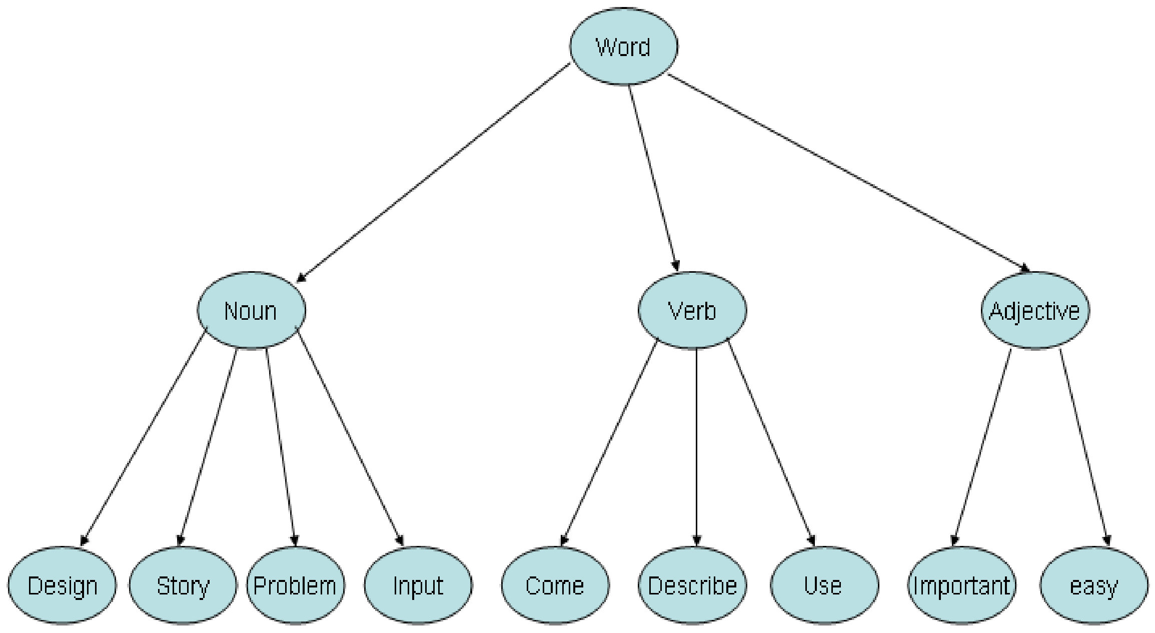


Figure 0.2 Partial Representation of the Vocabulary Ontology

We took the class *Word* as the root of the hierarchy since we know that sentences are made up of words. The root is followed by three subclasses: “noun”, “verb”, and “adjective”. We decided to put our focus only on those three subclasses because they represent the most significant information in a sentence. Figure 4.2 shows a partial representation of our Vocabulary Ontology. From this ontology we will build the ontology of the information inside the presentations.

4.3.2 Properties of the Classes

Having identified the set of words and turned them into classes is not enough to process the content of the information with the Vocabulary Ontology. Since the classes have been identified, it is now time to declare their respective properties that will help match their presence inside the presentations. Additionally, restrictions (constraints) were identified. These restrictions allow to categorize the type of value (type), the allowed values (range), the class to which the property belongs (domain), and the number of values (cardinality) of each property.

Table 0-1 Properties of the Class “Noun”

	Singular	Plural	Nsynonyms
Cardinality	1	1	Multiple (0...*)
Type	String	String	String
Range	String	String	String
Domain	Noun	Noun	Noun

In table 4-1, we have defined the following properties for the class Noun: *singular, plural, Nsynonyms,* where “singular” contains the singular value of the noun, “plural” contains the plural value of the noun, and “Nsynonyms” represents the synonym(s) of the corresponding noun.

Table 0-2 Properties of the Class “Verb”

	VBP	VBZ	VBN	VBD	VBG	Vsynonyms
Cardinality	1	1	1	1	1	Multiple (0...*)
Type	String	String	String	String	String	String
Range	String	String	String	String	String	String
Domain	Verb	Verb	Verb	Verb	Verb	Verb

In Table 4-2 we present the different properties of the class Verb:

1) VBP, defines the verb in present and future tense
2) VBZ, defines the verb in present tense, third person singular
3) VBN, defines the verb in present and past perfect tense
4) VBP, defines the verb in past tense
5) VBG, defines the verb in progressive form
6) Vsynonyms, contains the different synonym(s) of the verb

Table 0-3 Properties of the Class “Adjective”

	Adjectifvalue	ASynonyms
Cardinality	1	Multiple (0...*)
Type	String	String
Range	String	String
Domain	Adjective	Adjective

In table 4-3 we define the properties of the class Adjective. Adjectifvalue represents the string value of the adjective and Asysnonyms represents the different synonym(s) of the corresponding adjective.

4.3.3 Edition of the Vocabulary Ontology with Protégé

Protégé is a graphical tool for ontology editing and knowledge acquisition that a user can adapt to enable conceptual modeling. This open-source software is developed totally in Java. This program presents/displays a graphical screen that allows the developers to concentrate on the conceptual model rather than writing XML code, allowing them to create the different elements that comprise an ontology: class, slot, facet and others. With this tool we edited the Vocabulary Ontology that will allow us to look for the metadata inside of the presentations. Figure 4.3 shows a partial presentation of the Vocabulary Ontology as displayed on Protégé.

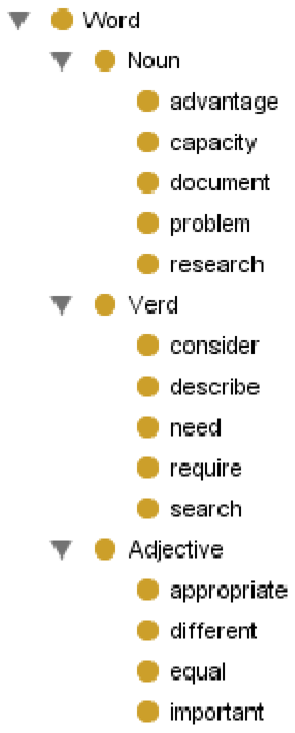


Figure 0.3 Partial Representation of the Vocabulary Ontology

In this Vocabulary Ontology, the class Word represents the root of the Hierarchy accompanied by its subclasses Noun, Verb and Adjective. The properties described in the tables above (Table 4-1, Table 4-2 and Table 4-3) were edited using Protégé.

4.3.4 Representing the Vocabulary Ontology in RDF

The XML-based semantic language "Resource Description Framework Schema" (RDF-S) provides a set of primitives describing the elements of an ontology like classes, properties and relations. With RDF-S a base vocabulary was edited, which allowed to define and label the different properties of the Vocabulary Ontology. For example, with the verb "to describe" we have the following RDF representation based on the RDF-Schema of the Vocabulary Ontology in Figure 4.4, where we show the relevant information about the subclass "describe", which is an element from the class "Verb".

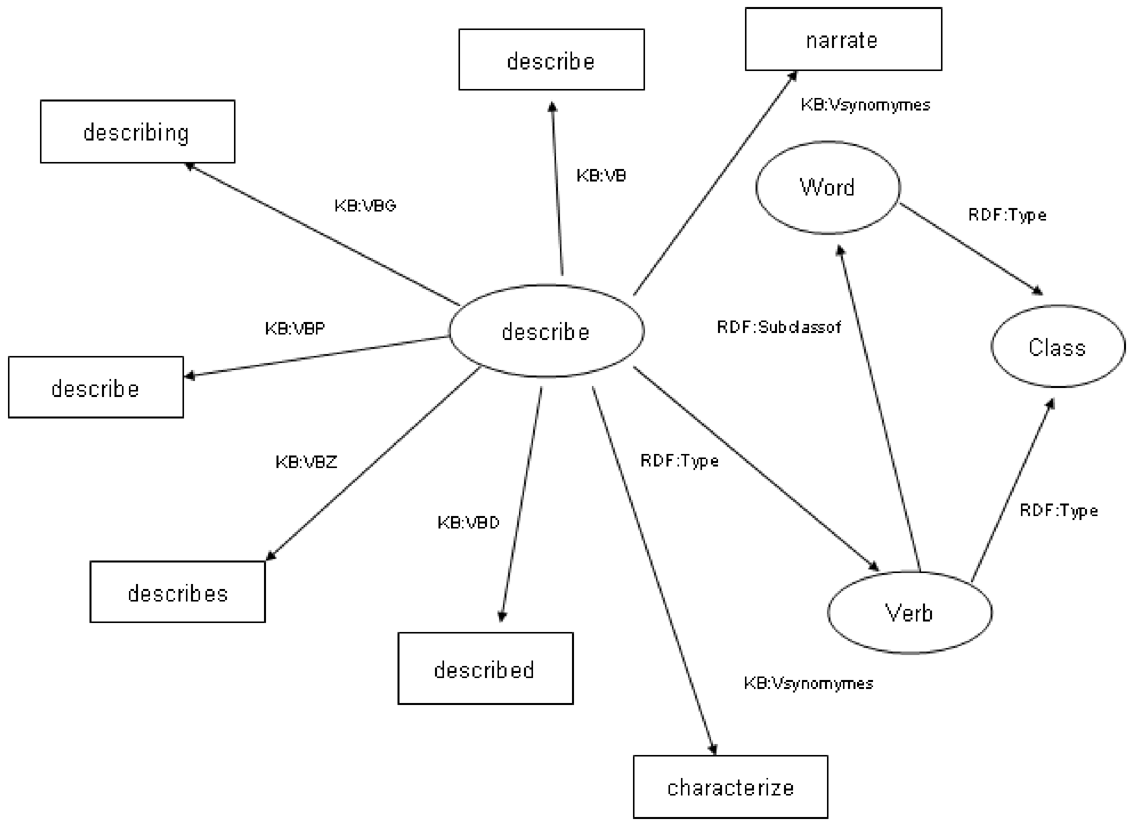
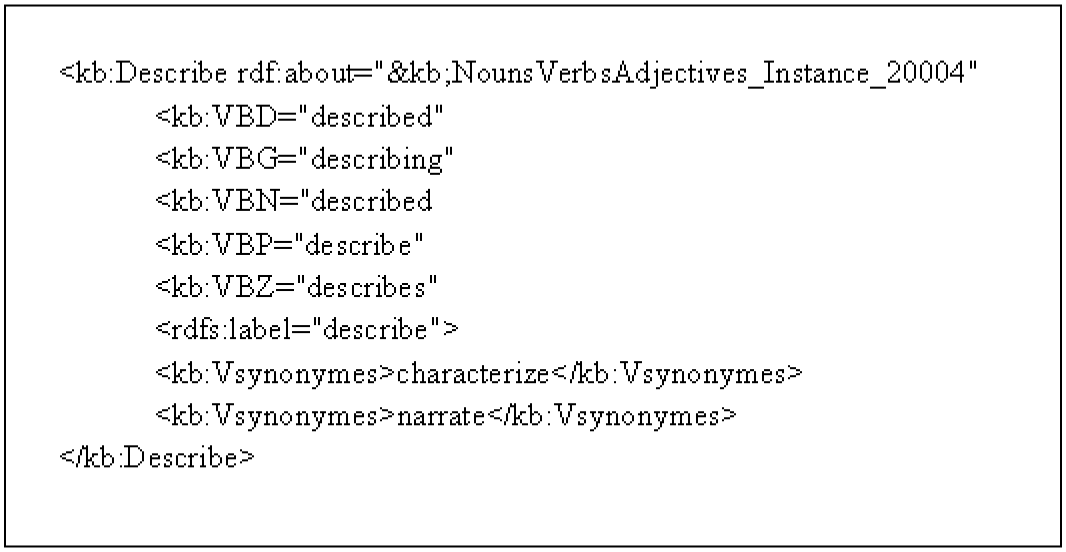


Figure 0.4 RDF Representation of the Subclass “describe” from the Vocabulary Ontology

Figure 4.5 represents the description in RDF language of the example above.



```
<kb:Describe rdf:about="&kb;NounsVerbsAdjectives_Instance_20004"
          <kb:VBD="described"
          <kb:VBG="describing"
          <kb:VBN="described
          <kb:VBP="describe"
          <kb:VBZ="describes"
          <rdfs:label="describe">
          <kb:Vsynonymes>characterize</kb:Vsynonymes>
          <kb:Vsynonymes>narrate</kb:Vsynonymes>
</kb:Describe>
```

Figure 0.5 RDF Language Description of “Describe”

In the next chapter, we will represent the different methods and tools used for the extraction and retrieval of metadata.

CHAPTER 5

EXTRACTION AND RETRIEVAL OF METADATA

5.1 Introduction

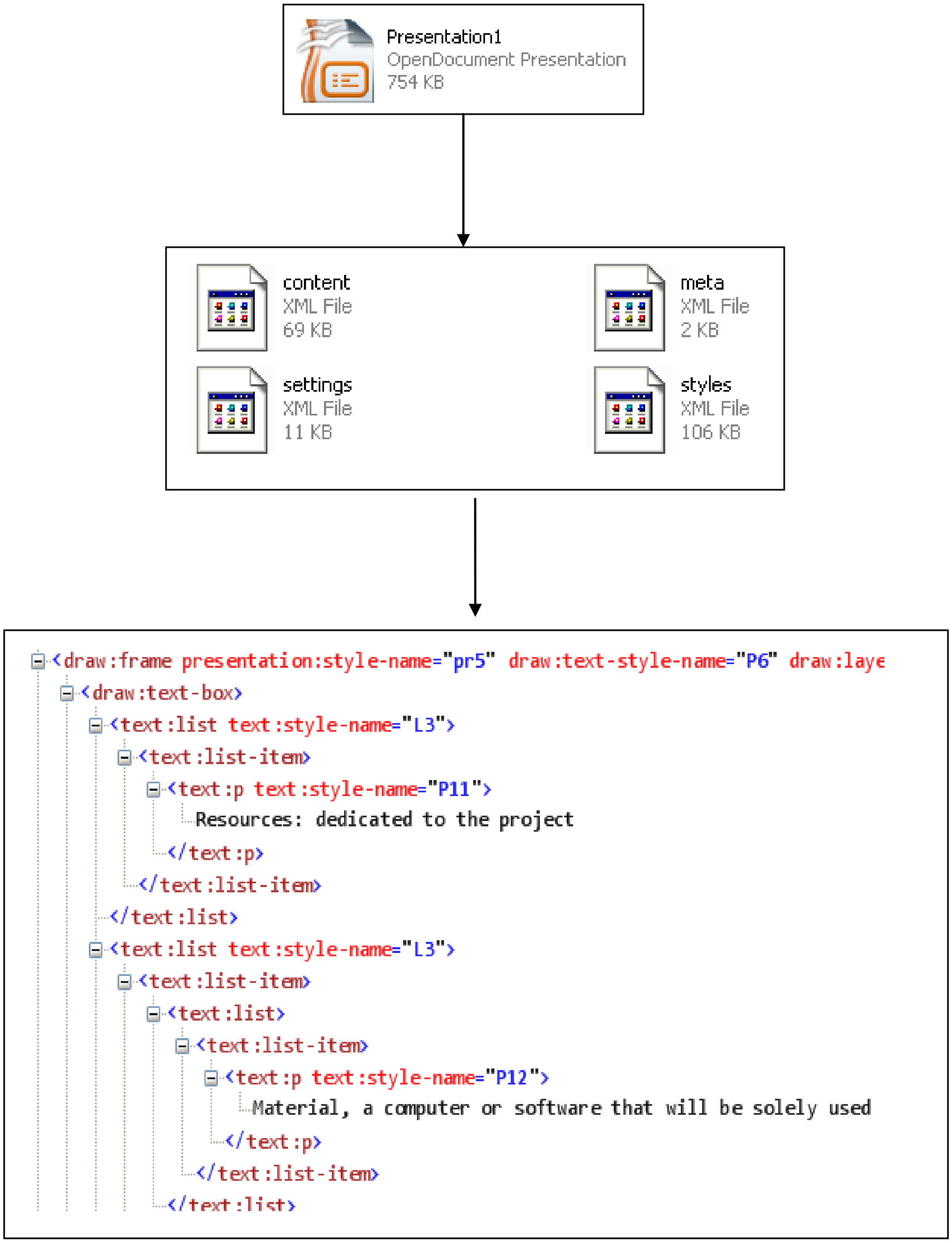
Extracting metadata means to reach into available sources, dragging out the metadata, and to stack it up neatly so other processes can exploit it [Kathleen04]. Metadata extraction can be handled in a couple of ways. The first way is to just grab the metadata whenever you need it. The second way is to make the metadata extraction a process that has its own conditions. In our case, we decided to use the second way.

Information retrieval deals with how people find information and how tools can be constructed to help in that process. Since the advent of the Web, these tools have become known as search engines [Andrick05]. Those search engines share a common process: Metadata retrieval.

In this chapter we will represent the methods and tools used for the extraction and retrieval of Metadata. First we will present the Open Office Format presentation. Then we will write about the different tools used in our application: Jena, Stanford NLP parser. Finally we will discuss the extraction and retrieval of the metadata

5.2 Extraction of XML File from Open Office Presentation

Each Open Office presentation is made of a compilation of different XML files and other folders. The following XML files are named: meta, content, settings and style. From all the XML files mentioned above, the only one that concerns us in our research is the “content.xml”. The file “content.xml” is the file that has all the information we need in a presentation.



```
<draw:frame presentation:style-name="pr5" draw:text-style-name="P6" draw:laye
  <draw:text-box>
    <text:list text:style-name="L3">
      <text:list-item>
        <text:p text:style-name="P11">
          Resources: dedicated to the project
        </text:p>
      </text:list-item>
    </text:list>
    <text:list text:style-name="L3">
      <text:list-item>
        <text:list>
          <text:list-item>
            <text:p text:style-name="P12">
              Material, a computer or software that will be solely used
            </text:p>
          </text:list-item>
        </text:list>
```

Figure 0.1 Extraction of a XML File from Open Office Presentation

In Figure 5.1, the upper part represents the file of an Open Office presentation "Presentation1".In the Open Office Format, four sub-files are compressed inside an Open Office presentation. After extracting the content from the file we have the following four XML sub-files in the middle: *content.xml*, *styles.xml*, *meta.xml* and *settings.xml.* The `settings.xml` file contains information intended for use exclusively by OpenOffice.org. The `meta.xml` file contains information about the document itself. The `styles.xml` file contains information about the styles that are used in the document. The *content.xml* file contains the information inside the document. Finally, the bottom exemplifies a partial information form the file "content.xml". Having the information of the presentation document inside the "content.xml" is nothing more than a step. Next we need to be able to access the information from the XML file.

5.3 Jena: Semantic Web Framework

Jena is a Java framework for building Semantic Web applications. It provides a programmatic environment for RDF, RDFS, and OWL, SPARQL and includes a rule-based inference engine [McBride02].

Jena is a Java API which can be used to create and manipulate RDF. Jena has object classes to represent graphs, resources, properties and literals. The interfaces representing resources, properties and literals are called Resource, Property and Literal respectively. In Jena, a graph is called a model and is represented by the Model interface. The Jena Framework includes:

- A RDF API
- Reading and writing RDF in RDF/XML, N3 and N-Triples
- An OWL API
- In-memory and persistent storage
- SPARQL query engine.

SPARQL is an RDF query language. We will not be using it in our program since it returns exact matches and our search engine presents the results using a percentage evaluation matching which will be discussed later on in Section 5.6.2.

5.4 Stanford NLP Parser

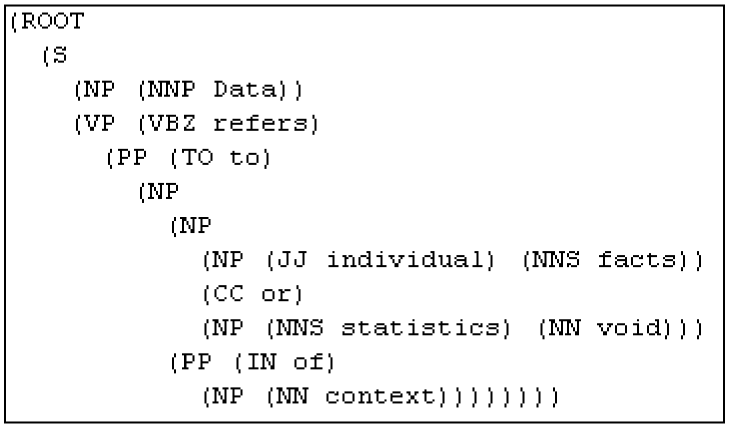
A natural language parser is a program that works out the grammatical structure of sentences, by defining for instance, which groups of words go together (as "phrases") and which words are the subject or object of a verb. Probabilistic parsers use knowledge of language gained from hand-parsed sentences, trained to produce the most likely analysis of new sentences. Although these statistical parsers may make some mistakes, they still commonly work rather well. The development of NLP parsers was one of the biggest breakthroughs in natural language processing in the 1990s [Stanford03].

The parser we are using for our research and program is the Stanford NLP Parser. This package is a Java implementation of probabilistic natural language parsers, both highly optimized PCFG (Probabilistic Context-Free Grammar) and lexicalized dependency parsers, and a lexicalized PCFG parser.

The software can also be used simply as an accurate unlexicalized stochastic context-free grammar parser. This approach forms a good performance statistical parsing system. A GUI is provided for viewing the phrase structure tree output of the parser [Stanford03].

To illustrate the capabilities of the Stanford Parser we will take a sentence from one of our Open Office presentations and parse it from our application. The sentence is: “Data refers to individual facts or statistics void of context”. The parser gives us the following major output:

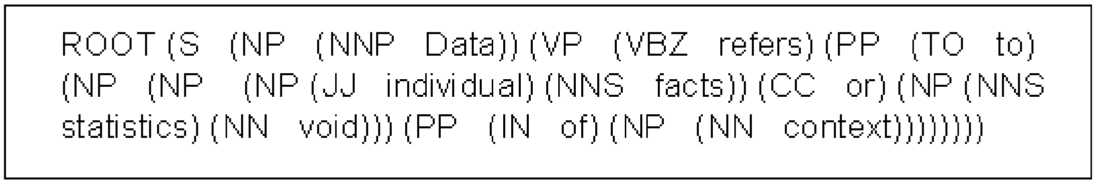
1) A NLP tree representation output of the sentence as shown in Figure 5.2



```
(ROOT
  (S
    (NP (NNP Data))
    (VP (VBZ refers)
      (PP (TO to)
        (NP
          (NP
            (NP (JJ individual) (NNS facts))
            (CC or)
            (NP (NNS statistics) (NN void)))
          (PP (IN of)
            (NP (NN context)))))))
```

Figure 0.2 A NLP Tree Representation of a Sentence

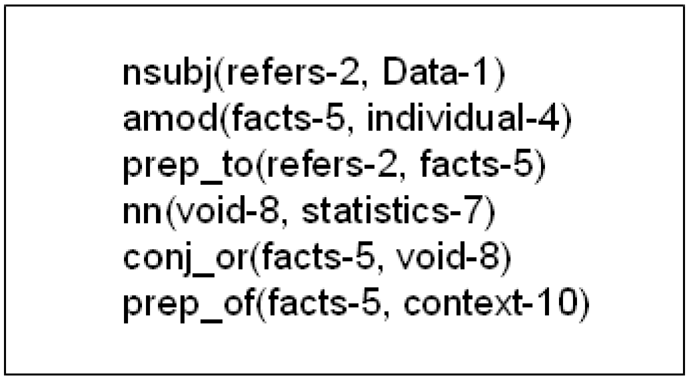
2) A string output for the tree representation of the sentence as shown in Figure 5.3



```
ROOT (S (NP (NNP Data)) (VP (VBZ refers) (PP (TO to)
(NP (NP (NP (JJ individual) (NNS facts)) (CC or) (NP (NNS
statistics) (NN void))) (PP (IN of) (NP (NN context)))))))
```

Figure 0.3 String Representation of the Tree

3) A relational representation between the words as shown in 5.4



```
nsubj(refers-2, Data-1)
amod(facts-5, individual-4)
prep_to(refers-2, facts-5)
nn(void-8, statistics-7)
conj_or(facts-5, void-8)
prep_of(facts-5, context-10)
```

Figure 0.4 Relational Representation of the Sentence

This is one of many other possible outputs from the Stanford Parser depending on the sentences and the words used to make up sentences. Of all three figures (Figure 5.2, Figure 5.3 and Figure 5.4), the illustration shown in Figure 5.4 is our main concern since it shows the relations between the words with each other and also our Metadata will be deduced from the relations of the words that forms the sentences. Our secondary concern is the “string representation of the tree” where we can extract the properties of the words (NN for noun, NNS for noun in plural, NNP for proper noun, JJ for adjective…).

5.5 Extraction of Metadata

In this section we will go deeper into the representation of the metadata in a RDF file. In this section we describe the main classes responsible for

- Reading the content from the Open Office presentation
- Analyzing the content through our AI Analyzer
- Storing the evaluated content if it meets the requirements

5.5.1 Reading the content from Open Office presentations.

As we said earlier in the chapter, extracting the information directly from the Open Office requires extracting the “content.xml” file from the presentation file so that we can then access the information written by the user in the presentation. Every new “content.xml” file is saved in a folder which has the name of its Open Office document. The folder containing all the folders with the file “content.xml” will be our Open Office presentation repository. This being done, we can now start accessing the information inside the presentations.

In order to access the information inside of the presentations, we first access the file “content.xml” from the first folder in the repository. Using the Eclipse SDK software and the Jena API, we were able to separate all the information from the presentation into a list of nodes where each node represents the information of a slide. The following list is named “listofSlides”. We then create a class named “ThePageLayout”. ThePageLayout is responsible for extracting and separating all the information inside each slide of a presentation.

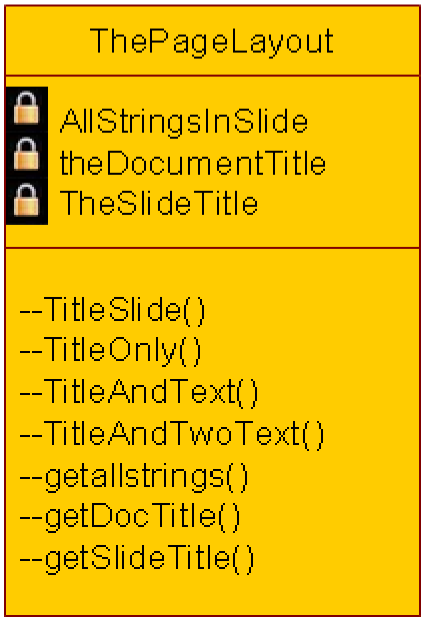


Figure 0.5 Structure of the Class ThePageLayout

Figure 5.5 shows the structure of the *ThePageLayout* class and the methods used to extract the information inside.

1. The *TitleSlide* method is responsible for getting the information from a slide whose layout’s attribute name matches “title slide”. This type of slide contains two textboxes. The first one has the title which is saved in the variable “*TheSlideTitle*”. The second one has the subtitle information that is separated into sentences and bullets, and then saved in the “AllStringsInSlide” variable.
2. The *TitleOnly* method is responsible for getting the information from a slide whose layout’s attribute name matches “title only”. Beside the title of the slide, this type of slide is likely to have “shape” figures in it that contain more information in the slide.

3. The *TitleAndText* method is responsible for getting the information from a slide whose layout's attribute name matches "title and text". This type of slide contains two textboxes. The first one has the title which is saved in the variable "*TheSlideTitle*". The second one has other information that is separated into sentences and bullets, and then saved in the "*AllStringsInSlide*" variable.
4. The *TitleAndTwoText* method is responsible for getting the information from a slide whose layout's attribute name matches "title and two texts". This type of slide contains three textboxes. The first one has the title which is saved in the variable "*TheSlideTitle*". The second and third one have other information that is separated into sentences and bullets, and then saved in the "*AllStringsInSlide*" variable.

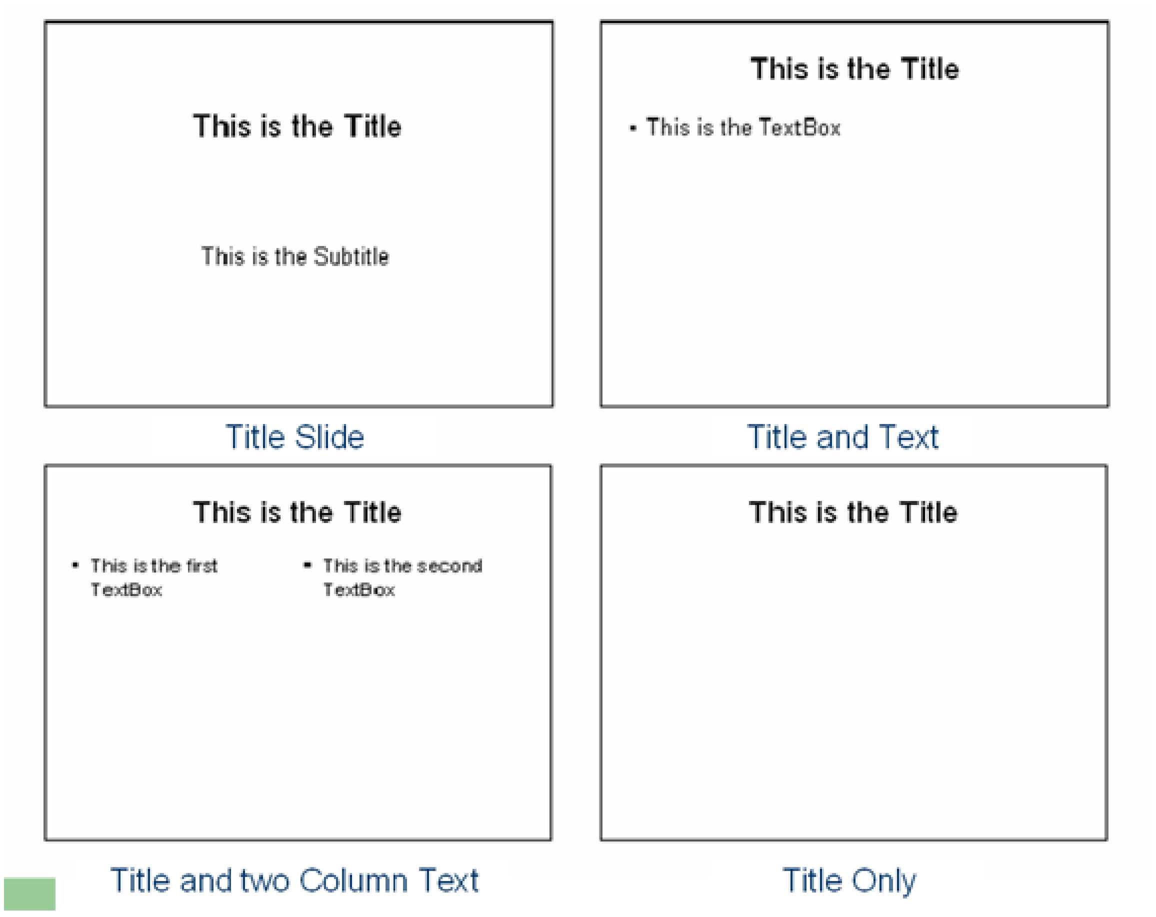


Figure 0.6 The Four Different Layouts of a Slide

Figure 5.6 shows the four different layouts.

.To summarize all, the extraction of information inside each slide follows a set of rules that depends on the layout of the slide. There exists also other types of slides but they will not be analyzed since they contain graphics and pictures whose analysis is outside the scope of our thesis

5.5.2 AI Analyzer

After reading and saving the content of a slide, it is time now to analyze the extracted information and save the following metadata in the RDF file. The information for each slide consists of: a variable that contains the title of the slide (*theDocumentTitle* or *TheSlideTitle*) and another variable that contains a list of all the sentences and bullets from the slide (*AllStringsInSlide*). Now we create the RDF resource of the slide.

For the first slide only of each presentation, the RDF resource value corresponds to the URI of the presentation. This RDF resource has only one property: *TitleDocument* and the value of the property is the title of the slide.

For all the other slides, the RDF resource value corresponds to the URI of the presentation plus the slide number (URI/slide#). The first property of the resource is *TitleSlide* and its value is the title of the slide. The next step is to analyze each sentence and bullet through our AI Analyzer.

The variable *AllStringsInSlide* is an *ArrayList*. The information from the slide is made of sentences and bullets. All sentences beside the title are saved in the variable *AllStringsInSlide* . Then, each bullet is separated into sentences which will be also saved in the variable *AllStringsInSlide*. Each sentence from the variable *AllStringsInSlide* is analyzed one after the other by our AI Analyzer.

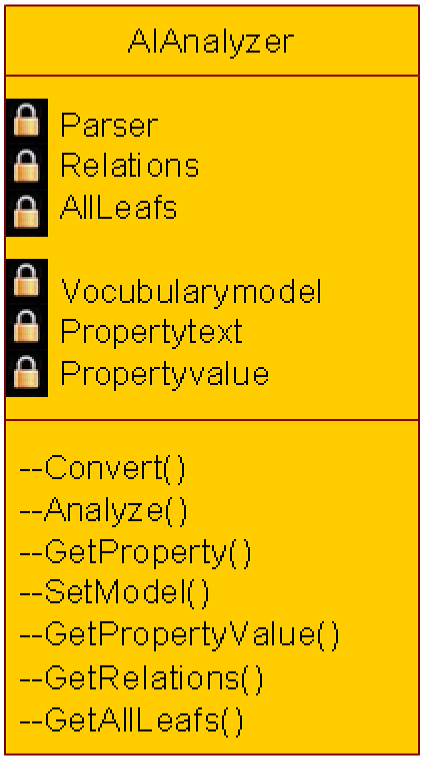


Figure 0.7 Structure of the Class AIAnalyzer

Figure 5.7 shows the structure of the Class AIAnalyzer and the methods used to analyze the information extracted from the presentations.

The variable *Parser* is a Class variable which is in charge of parsing the text with the Stanford Parser. It returns two main variables: an arraylist of all the relations between the words from the parser and another arraylist that contains all the words within the text with their assigned properties. The arraylists will be saved in the variables *Relation* and *AllLeafs* respectively.

The method *Convert* is used to:

1. Get rid of plural in nouns that appear in our Vocabulary Ontology that contains only singular nouns.
2. Get rid of some conjugations (present tense third person, past tense, past participle tense) while making sure not to change the structure of the sentence. In other words, if both the original and the converted sentence go through the parser, we will have the same relational presentation.

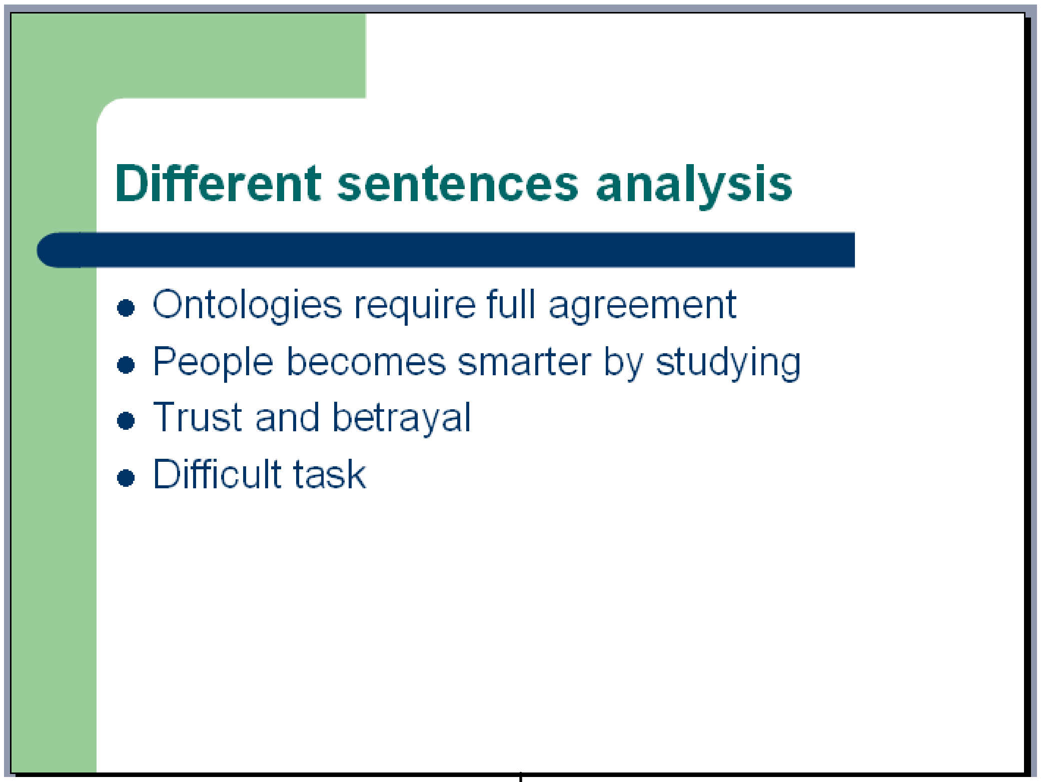
Once the sentence is converted, it will go through the Analyze method. The Analyze method is the one responsible for analyzing our sentences and determining the metadata to be stored. We stated earlier in the previous chapter that every sentence will be analyzed by our Artificial Intelligence Analyzer (AI Analyzer), but not all of them will result into metadata. A sentence needs to have at least one word from our Vocabulary Ontology before being considered as metadata. Each sentence will have a unique property. Using our variable "*Relations*" which contains all the relations in the sentence, we determine the property of the sentence in the following priority order:

- *NounverbSubjectObject*. Sentence with a conjugated verb that has a direct or indirect object complement.
- *NounverbSubjectPrep*. Sentence with a conjugated verb that has a prepositional complement.
- *NounNounSubject*. Sentence where the verb "to be" is the main conjugated verb.
- *NounPhrase*. Sentence without a conjugated verb. They can be either phrases or incomplete sentences
- *NounAdjective*. Sentence where noun(s) and adjective(s) are related. They can be either phrases or incomplete sentences

Once the property of the converted sentence is determined, it is saved in the variable "*Propertytext*". The only operation left is to establish the metadata. The final metadata is generated by getting rid of unimportant information inside the converted sentence mostly words like definite and indefinite articles, prepositions, etc. The converted sentence is then saved in the *Propertyvalue* variable. The RDF triples is created and stored in the RDF Metadata File where:

- The subject (resource) is the URI/slide#
- The predicate (property) is the *Propertytext*
- The object (literal) is the *Propertyvalue*

Figure 5.8 exemplifies the extraction of the metadata and the structure of the metadata in the RDF file.



Different sentences analysis

- Ontologies require full agreement
- People becomes smarter by studying
- Trust and betrayal
- Difficult task

```
<rdf:Description rdf:about="C:/Documents and Settings/Riemann Dorval/My
Documents/Test/Different sentences analysis/Slide2">
   <j.0: NounAdjective >Difficult task </j.0: NounAdjective >
   <j.0: NounPhrase >Trust betrayal </j.0: NounPhrase >
   <j.0:NounVerbSubjectPrep>People becomes smarter studying
</j.0:NounVerbSubjectPrep>
   <j.0:NounVerbSubjectObj>ontology require full agreement
</j.0:NounVerbSubjectObj>
   <j.0:TitleSlide>Different sentences analysis </j.0:TitleSlide>
 </rdf:Description>
```

Figure 0.8 RDF Structure of Extracted Information from Slides

5.5.3 Metadata Extraction Summarization

To summarize the extraction of the metadata process, we have the Figure 5.9

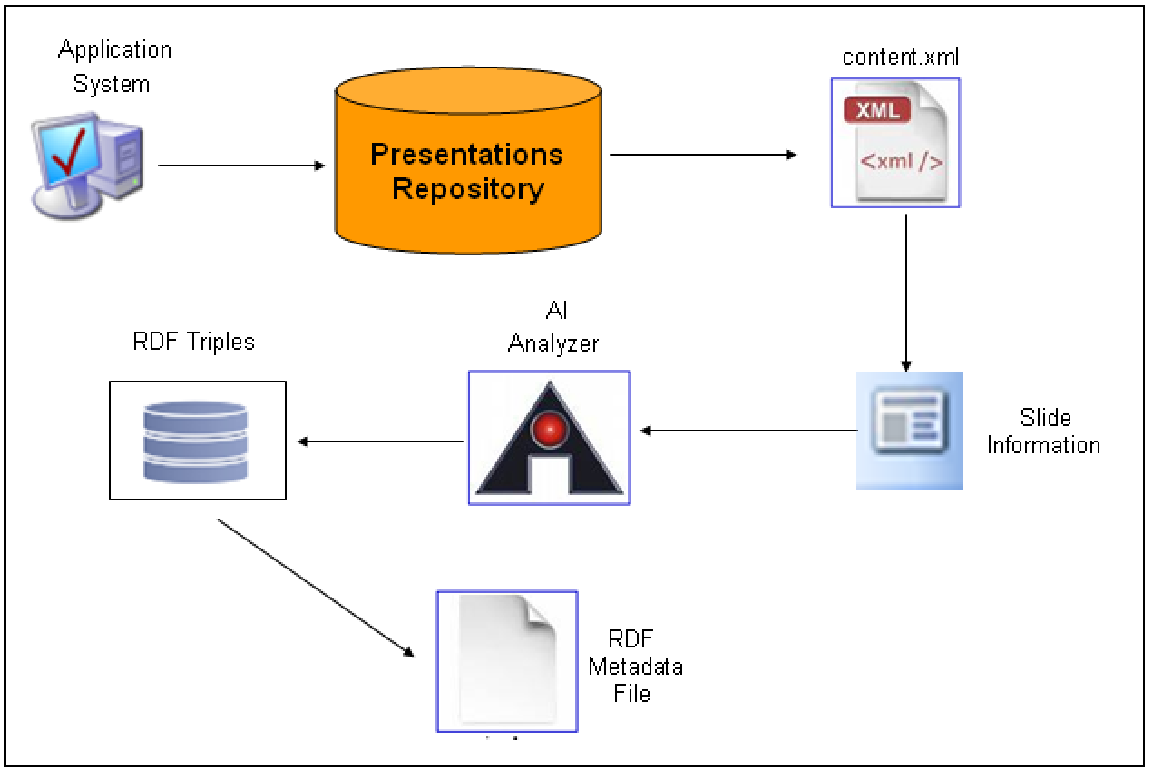


Figure 0.9 Summarization of Metadata Extraction

The application starts by accessing the repository of the presentations where the "*content.xml*" file from each presentation is stored. *ThePageLayout* class will read the information inside the slides one after the other. The information is separated and stored in their corresponding variables (see section 5.5.1). The variables are now analyzed through our AI Analyzer, then the RDF triples are generated and saved in the RDF Metadata file (see section 5.5.2)

5.6 Retrieval of Metadata

Once we have our RDF file with the metadata from the presentations inside of our repository, we can now make a search among the presentations. In this section we present the class responsible for searching the metadata and matching it with the user's search topic.

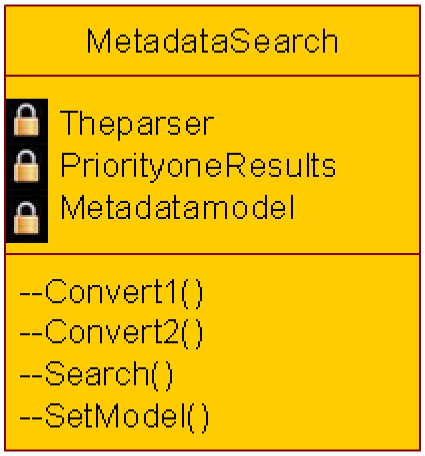


Figure 0.10 Structure of the Class MetadataSearch

The variable *Theparser* is a class type variable. *TheParser* is responsible for parsing the user's search topic. The *PriorityoneResults* is an arraylist variable which has all the results where the user's search topic has a minimum of 75% matching success with the metadata. The percentage of the matching success is calculated through an algorithm that will be described later in the chapter. The *Convert1* method is the same as the Convert method described in the previous section. The *Convert2* method does the same as the Convert1, but it takes synonyms into account. The *Convert2* method verifies if the nouns, verbs and adjectives are synonyms from our Vocabulary Ontology and then replace them by their base value.

After the user's search topic is parsed and converted, the Search method is called. The Search method is in charge of going through the RDF file that contains the metadata and finding the best possible match with the user's search topic. The different properties of our metadata are: *TitleDocument, TitleSlide, NounverbSubjectObject, NounverbSubjectPrep, NounNounSubject, NounPhrase* and *NounAdjective*. In order to match those properties with the user's search topic, we decided to classify the user's search topic into two main characteristics: is it a question topic or not?

5.6.1 User's Search Topic Analysis

Finding out whether or not a sentence is a question is carried out with the Stanford Parser. Table 5.1 shows different sentences and the results of their Tree and Relational Representation.

Table 0-1 Tree and Relational Presentation of Some Sentences

Sentence	**Tree Presentation**	**Relational Presentation**
What does data refer to	`(SBARQ (WHNP (WP What)) (SQ (VBZ does) (NP (NNS data)) (VP (VB refer) (PP (TO to)))))`	`dobj(refer-4, What-1)` `aux(refer-4, does-2)` `nsubj(refer-4, data-3)` `prep(refer-4, to-5)`
What refers to individual facts	`(SBARQ (WHNP (WP What)) (SQ (VP (VBZ refers) (PP (TO to) (NP (JJ individual) (NNS facts))`	`nsubj(refers-2, What-1)` `amod(facts-5, individual-4)` `prep_to(refers-2, facts-5)`
How does a car work	`(SBARQ (WHADVP (WRB How)) (SQ (VBZ does) (NP (DT a) (NN car)) (VP (VB work)))))`	`advmod(work-5, How-1)` `aux(work-5, does-2)` `det(car-4, a-3)` `nsubj(work-5, car-4)`
knowledge management	`(NP (NN knowledge) (NN management))`	`nn(management-2, knowledge-`
difficult tasks in life	`(NP (NP (JJ difficult) (NNS tasks)) (PP (IN in) (NP (NN life)))))`	`amod(tasks-2, difficult-1)` `prep_in(tasks-2, life-4)`

The Stanford Parser allows to tell apart whether a user's sentence is a question or not. The sentence is a question when the head of the Tree Presentation is "SBARQ". This is not the only important information given by the parser. In case the question is asked with the pronoun "what", we have two possibilities: the pronoun is an object complement or the pronoun is the subject of the verb. As we can see in Table 5.1, the relation between the pronoun and the verb illustrates the two possibilities. In the first question from Table 5.1, we have the following relation *"dobj(refer-4, What-1)"* . This relation tells us that the pronoun is an object complement of the verb. In the second question from

Table 5.1, we have the following relation "*nsubj(refers-2, What-1)"*. This relation tells us that the pronoun is the subject of the verb.

To summarize, the user's sentence is first analyzed in order to determine what type of sentence he entered. Then, the sentence is converted into the "*matching sentence*" through the *Convert1* method. The "*matching sentence*" is the sentence that will be matched with the metadata from our RDF Metadata file. Using the examples in Table 5.1, Table 5.2 shows the "*matching sentences*" that result from the analysis and conversion processes during the retrieval of metadata.

Table 0-2 Matching Sentences from User's Sentences

User Sentence	**Matching Sentence**
What does data refer to	data refer to
What refers to individual facts	refer to individual fact
How does a car work	a car work
Knowledge management	knowledge management
difficult tasks in life	difficult task in life

5.6.2 The Matching Process

Earlier in the section we said that the *PriorityoneResults* is an arraylist variable which has all the results where the user's search topic has a minimum of 75% matching success with the metadata. The algorithm for the matching process is shown in Figure 5.11:

```
Initialize totalmatch to zero

Initialize Pos to zero

Initialize startingPos to zero

totalwords equal to quantity of words in the matching sentence

While there is still a word waiting to be matched
          Pos equal to startingPos
          If word is matched with other word from metadata
          sentence at position Pos
               Increment totalmatch by 1
               startingPos equal Pos+1
          Else
               Increment Pos by 1

Percentage is equal to matching totalmatch divided by totalwords
```

Figure 0.11 Matching Algorithm

If the percentage is superior or equal to 75%, the resource of the metadata is saved in the *PriorityoneResults*.

In Section 5.5.2 we followed a specific priority order to determine the property of the metadata if the sentence was not a title. The priority order is*:* *NounverbSubjectObject*, *NounverbSubjectPrep*, *NounNounSubject*, *NounPhrase* and *NounAdjective*. The "*matching sentence*" does not need to be matched with all the triples in our RDF Metadata file because of the priority order in storing the metadata. Table 5.3 shows the different metadata properties from the RDF Metadata file to which the "*matching sentence*" will be matched with using the examples in both Table 5.1 and Table 5.2

Table 0-3 Matching Properties and Matching Sentences

User Sentence	Matching Sentence	Matching Properties
What does data refer to	data refer to	TitleDocument, TitleSlide, NounverbSubjectObject, NounverbSubjectPrep
What refers to individual facts	refer to individual facts	TitleDocument, TitleSlide, NounverbSubjectObject and NounverbSubjectPrep
How does a car work	a car work	TitleDocument, TitleSlide, NounverbSubjectObject and NounverbSubjectPrep
knowledge management	knowledge management	TitleDocument, TitleSlide, NounverbSubjectObject, NounverbSubjectPrep, NounNounSubject and NounPhrase
difficult tasks in life	difficult task in life	TitleDocument, TitleSlide, NounverbSubjectObject, NounverbSubjectPrep, NounNounSubject, NounPhrase and NounAdjective

5.6.3 Retrieval of Metadata Summarization

To summarize the extraction of the metadata process, we have Figure 5.12

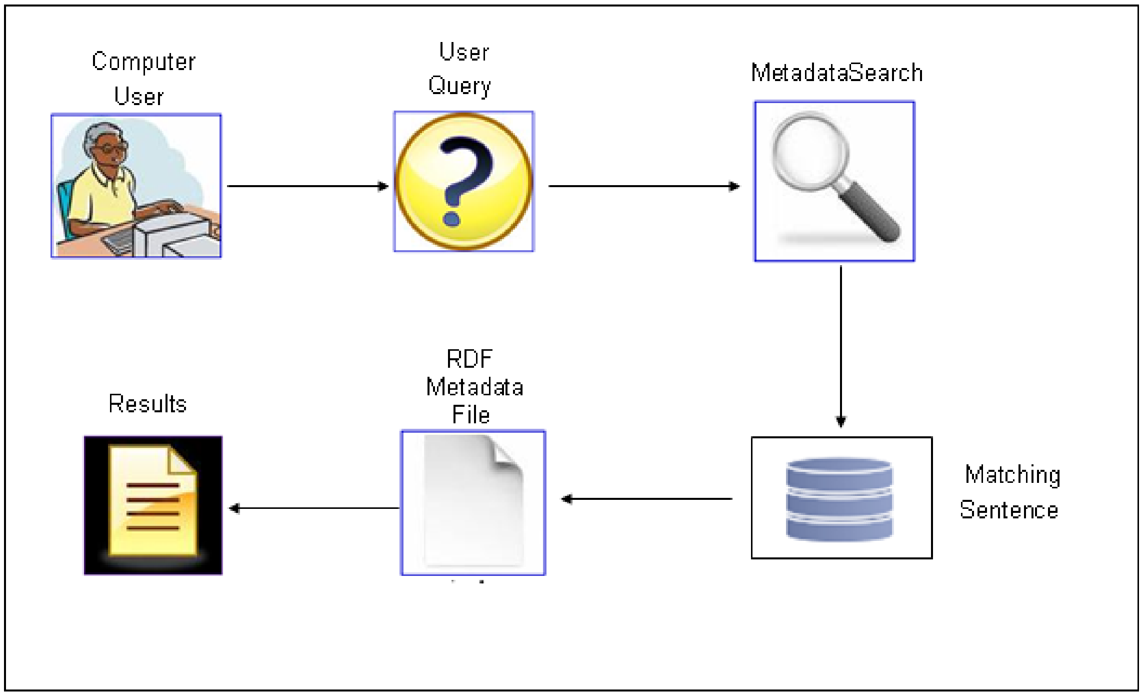
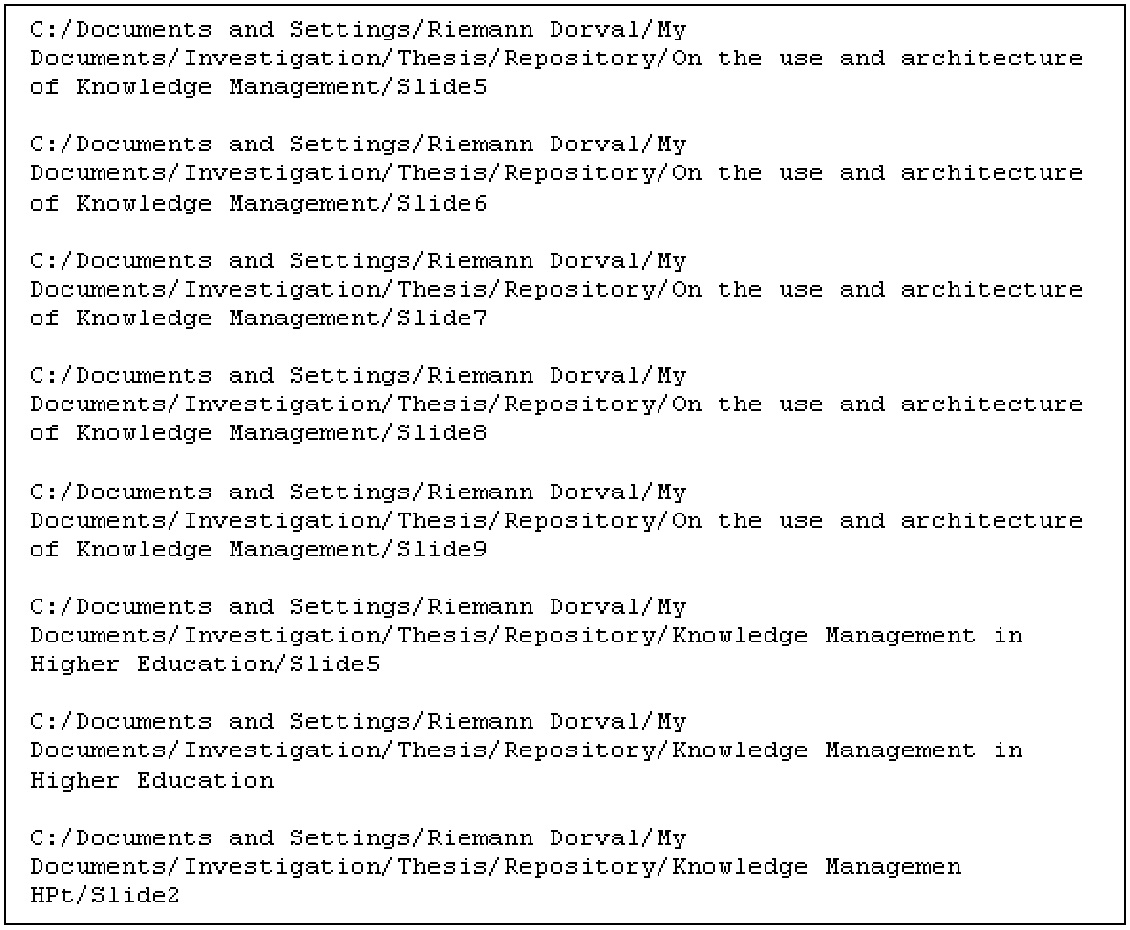


Figure 0.12 Summarization of Retrieval of Metadata

The retrieval of the metadata starts by the user entering the "*query sentence*". The Metadata Search class will analyze the user's "*query sentence*" and change it into the "*matching sentence*" that is related to the query (see Section 5.6.1). The "*matching sentence*" will now be compared with the associated triples that are in the RDF Metadata file. A matching percentage will be generated from the matching process. If the matching percentage is high enough (greater or equal to 75%), the resource of the corresponding RDF triple will be saved in the *"PriorityoneResults"* arraylist (see Section 5.6.2). When the "*matching sentence"* has been compared to all of its associated metadata, the results will be displayed to the user. If the "*query sentence*" is "Knowledge Management". The application returns the following partial results: Figure 5.13



```
C:/Documents and Settings/Riemann Dorval/My
Documents/Investigation/Thesis/Repository/On the use and architecture
of Knowledge Management/Slide5

C:/Documents and Settings/Riemann Dorval/My
Documents/Investigation/Thesis/Repository/On the use and architecture
of Knowledge Management/Slide6

C:/Documents and Settings/Riemann Dorval/My
Documents/Investigation/Thesis/Repository/On the use and architecture
of Knowledge Management/Slide7

C:/Documents and Settings/Riemann Dorval/My
Documents/Investigation/Thesis/Repository/On the use and architecture
of Knowledge Management/Slide8

C:/Documents and Settings/Riemann Dorval/My
Documents/Investigation/Thesis/Repository/On the use and architecture
of Knowledge Management/Slide9

C:/Documents and Settings/Riemann Dorval/My
Documents/Investigation/Thesis/Repository/Knowledge Management in
Higher Education/Slide5

C:/Documents and Settings/Riemann Dorval/My
Documents/Investigation/Thesis/Repository/Knowledge Management in
Higher Education

C:/Documents and Settings/Riemann Dorval/My
Documents/Investigation/Thesis/Repository/Knowledge Managemen
HPt/Slide2
```

Figure 0.13 Partial Results from a User Query

By analyzing the results in Figure 5.12 we can see that our results do not only point to the physical address of the full presentation but also to a specific slide within the presentation. Also, if there is more than one slide matching the same search criteria within the same presentation, they all appear in our results.

In this chapter we described how the application extracts metadata from the presentations and how metadata is retrieved and matched with the user's sentence when a user search occurs. In the next chapter we will describe the methodology to conduct different tests in order to evaluate our application.

CHAPTER 6

METHODOLOGY

6.1 Introduction

The metadata represents the information inside the Open Office presentations and is saved in a RDF file. The metadata contains statements of the form subject-predicate-object. To search the metadata we used a metadata retrieval process which was described in section 5.6, which allowed us to search the metadata related to the user's search topic.

In order to evaluate the accuracy and reliability of the retrieval of metadata process, it is necessary to carry out some tests to compare our system with other known search engines. In this chapter we present the methodology for the tests.

6.2 Purpose of the Test

The purpose of the text is to evaluate the accuracy and reliability of the retrieval of metadata. For that purpose we compare the results of our application with the results of two other well known search engines: Windows Search and Google Desktop.

6.3 Description of Methodology

In this section, we describe the tests that are done to compare and evaluate the metadata retrieval process.

6.3.1 Equipment and Tools

To carry out the tests, a computer with the following technical specifications was used.

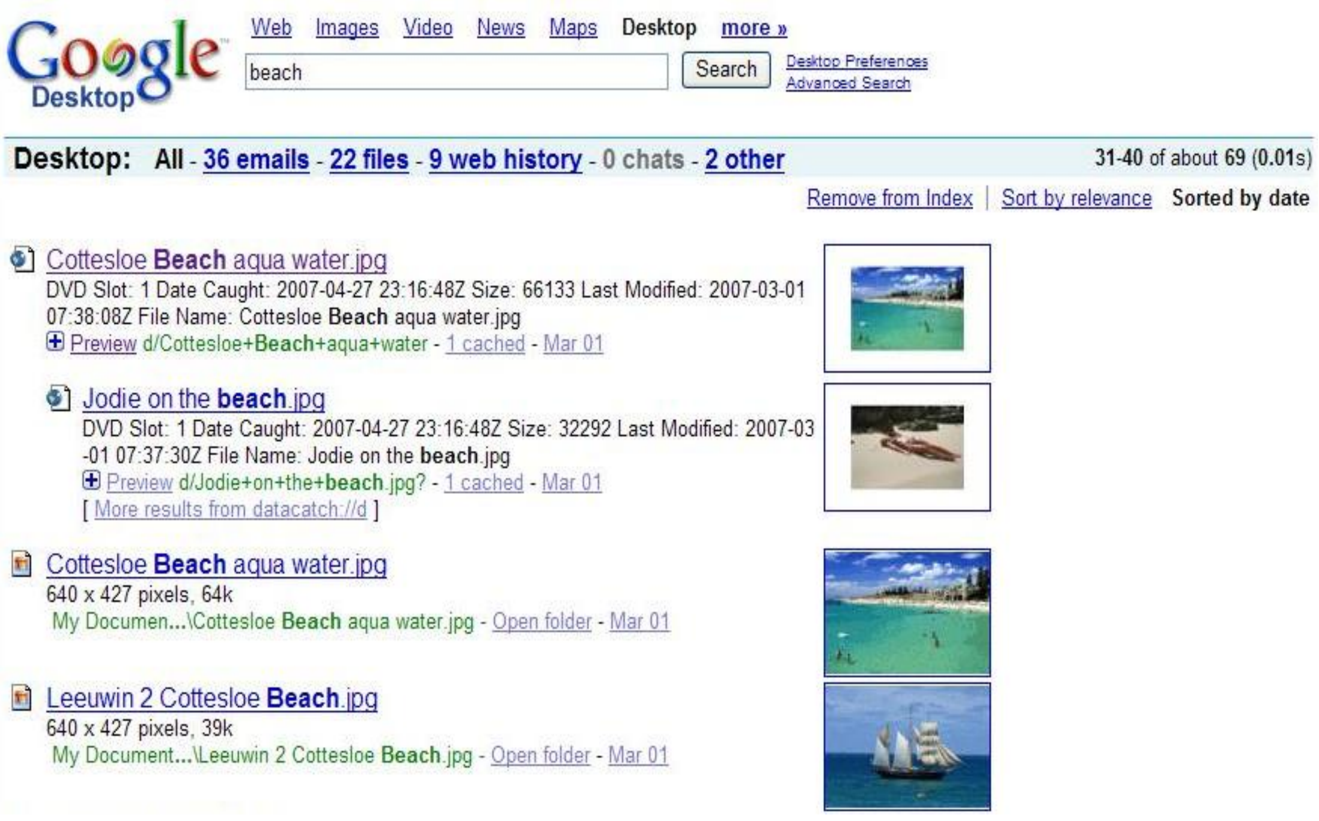
Table 0-1 Description of the Computer Used for the Tests

Description	Operative System
Dell Inspiron I6400 Intel® Core™2 CPU T7200 2.00GHz 997 MHz, 1 GB of RAM	Window XP Professional

The application, the presentations repository and the RDF file are all saved in the computer. To run our application we used the following software:

- Eclipse SDK(Software Development Kit)
- Java 1.5.1

The first application compared with our system is Google Desktop. Google Desktop makes searching the user’s computer as easy as searching the web with Google. It is a desktop search application that provides full text search over email, files, music, photos, chats, Gmail, web pages that you've viewed, and more. By making the user’s computer searchable, Desktop puts the information easily within reach and frees the user from having to manually organize the files, emails and bookmarks. When you do a Desktop search, you will go to a page showing the most relevant search results (http://desktop.google.com/features.html#overview). Each result includes the file name and a brief snippet with the search terms highlighted.


Google Desktop
Web Images Video News Maps Desktop more »
beach
Search
Desktop Preferences
Advanced Search
Remove from Index | Sort by relevance Sorted by date
Preview
Open folder

Desktop: All - 36 emails - 22 files - 9 web history - 0 chats - 2 other 31-40 of about 69 (0.01s)

Cotteslore **Beach** aqua water.jpg
DVD Slot: 1 Date Caught: 2007-04-27 23:16:48Z Size: 66133 Last Modified: 2007-03-01 07:38:08Z File Name: Cotteslore **Beach** aqua water.jpg
Preview d/Cotteslore+**Beach**+aqua+water - 1 cached - Mar 01

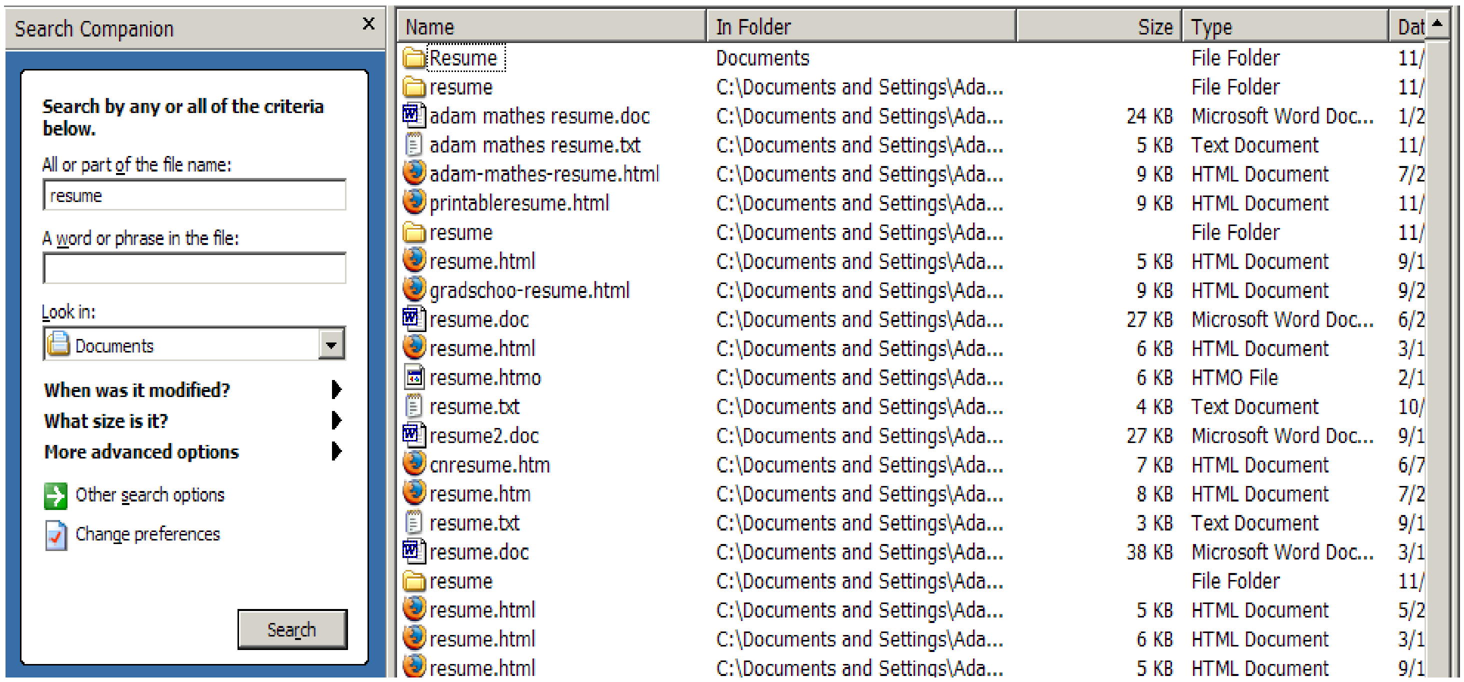
Jodie on the **beach**.jpg
DVD Slot: 1 Date Caught: 2007-04-27 23:16:48Z Size: 32292 Last Modified: 2007-03-01 07:37:30Z File Name: Jodie on the **beach**.jpg
Preview d/Jodie+on+the+**beach**.jpg? - 1 cached - Mar 01
[More results from datacatch://d]

Cottesloe **Beach** aqua water.jpg
640 x 427 pixels, 64k
My Documen...\Cottesloe **Beach** aqua water.jpg - Open folder - Mar 01

Leeuwin 2 Cottesloe **Beach**.jpg
640 x 427 pixels, 39k
My Document...\Leeuwin 2 Cottesloe **Beach**.jpg - Open folder - Mar 01

Figure 0.1 Representation of a Google Desktop Search Result

The second application compared with our system is Windows Search. Windows Search makes it easy to search for files and folders, printers, people, and other computers on a network; and it is a convenient starting point for searching for information on the Internet. Windows Search also has an indexing service that maintains an index of all the files on the computer, making searches faster. When using Windows Search, the user can specify several search criteria. For example, the user can search for files and folders by name, type, and size. The user can find a file based on when he/she last worked on it or search for files containing specific text (http://www.microsoft.com/ resources/documentation/ windows/xp/all/proddocs/en-us/find_overview.mspx?mfr=true)



Search Companion

Search by any or all of the criteria below.

All or part of the file name: resume

A word or phrase in the file:

Look in: Documents

When was it modified?

What size is it?

More advanced options

Other search options

Change preferences

Search

Name	In Folder	Size	Type	Dat
Resume	Documents		File Folder	11/
resume	C:\Documents and Settings\Ada...		File Folder	11/
adam mathes resume.doc	C:\Documents and Settings\Ada...	24 KB	Microsoft Word Doc...	1/2
adam mathes resume.txt	C:\Documents and Settings\Ada...	5 KB	Text Document	11/
adam-mathes-resume.html	C:\Documents and Settings\Ada...	9 KB	HTML Document	7/2
printableresume.html	C:\Documents and Settings\Ada...	9 KB	HTML Document	11/
resume	C:\Documents and Settings\Ada...		File Folder	11/
resume.html	C:\Documents and Settings\Ada...	5 KB	HTML Document	9/1
gradschoo-resume.html	C:\Documents and Settings\Ada...	9 KB	HTML Document	9/2
resume.doc	C:\Documents and Settings\Ada...	27 KB	Microsoft Word Doc...	6/2
resume.html	C:\Documents and Settings\Ada...	6 KB	HTML Document	3/1
resume.htmo	C:\Documents and Settings\Ada...	6 KB	HTMO File	2/1
resume.txt	C:\Documents and Settings\Ada...	4 KB	Text Document	10/
resume2.doc	C:\Documents and Settings\Ada...	27 KB	Microsoft Word Doc...	9/1
cnresume.htm	C:\Documents and Settings\Ada...	7 KB	HTML Document	6/7
resume.htm	C:\Documents and Settings\Ada...	8 KB	HTML Document	7/2
resume.txt	C:\Documents and Settings\Ada...	3 KB	Text Document	9/1
resume.doc	C:\Documents and Settings\Ada...	38 KB	Microsoft Word Doc...	3/1
resume	C:\Documents and Settings\Ada...		File Folder	11/
resume.html	C:\Documents and Settings\Ada...	5 KB	HTML Document	5/2
resume.html	C:\Documents and Settings\Ada...	6 KB	HTML Document	3/1
resume.html	C:\Documents and Settings\Ada...	5 KB	HTML Document	9/1

Figure 0.2 Representation of a Windows Search Result

6.3.2 Testing Scenarios

To perform the tests, we designed two scenarios to evaluate and compare the accuracy and reliability of our application. The difference between the two scenarios is that in one of them our application does not include synonyms while in the other the application does include synonyms.

Metadata in our RDF file have one of the following properties: *TitleDocument*, *TitleSlide*, *NounverbSubjectObject*, *NounverbSubjectPrep*, *NounNounSubject*, *NounPhrase* and *NounAdjective*. It is important that we make different searches where we are able to take advantage of the metadata properties and compare our results with the results of the two other applications. Each sentence will generate results from all three applications: our application, Google Desktop and Windows Search. Since we are comparing accuracy, only the results returned by the variable "*PriorityoneResults*" (the arraylist containing the results where the matching percentage is greater or equal to 75%, see Chapter 5, section 5.6.2) from our application will be considered in both scenarios.

6.3.2.1 First Scenario

In the first scenario we make different searches where we use sentences from the presentations and also questions related to the sentences from the presentations. We also decided not to include synonyms in this scenario. The different properties that will be taken into account in this scenario are: *TitleDocument, TitleSlide, NounverbSubjectObject, NounverbSubjectPrep* and *NounPhrase.* This leaves us with the following types of search.

- **TitleDocumen**t: we will use sentences made of the entire or partial Open Office presentations' name
- **TitleSlide**: we will use sentences made of exact titles used from different presentations
- **NounverbSubjectObject:** we will use question sentences that start with the pronoun "what", where a pronoun can be the verb subject or the verb object complement. Most questions will be short sentences
- **NounverbSubjectPrep:** we will use question sentences that start with how. Most questions will be short sentences.
- **NounPhrase:** we will use noun phrase.

6.3.2.2 Second Scenario

The second scenario is a reproduction of the first scenario where three of the five properties mentioned in the first scenario will be taken into account: *NounverbSubjectObject, NounverbSubjectPrep* and *NounPhrase*. The sentences used in the second scenario are slightly different in terms of word than the sentences used in the first scenario. But if we compare the meaning of both sentences, they are the same. Most of the sentences from the first scenario contain at least one word from our Vocabulary Ontology. Some of the words that are present in our Vocabulary Ontology will be replaced by a synonym. The new sentence will be the one used to find the results.

6.4 Document Repository and the RDF Metadata File

Our repository of Open Office documents holds 20 Open Office presentations. The RDF Metadata file is the file that contains the metadata from all 20 presentations. Previously in Chapter 5, Figure 5.8 we had shown how the metadata is structured through our RDF Metadata file. For the comparison of the applications, only the results pointing to this repository from the results returned by both Google Desktop and Windows Search will be considered. In the next chapter we present the results.

CHAPTER 7

RESULTS

This chapter presents the results from the tests performed in the two scenarios described in the previous chapter.

7.1 Scenario 1

The repository for this scenario contains 20 Open Office presentations. The RDF Metadata file contains the metadata extracted from all the presentations using the methods described in Chapter 5. The extracted metadata contains information from the presentations but also the URI of the presentations from the repository and the slide number in appropriate cases. Different searches will be made using our application and the results will be compared to the results from other well known system such as Google Desktop and Windows Search. The searched sentences or phrases will be selected from the sentences of the presentations from the repository. The different sentences or phrases that will be taken into account in this scenario are: title of document, title of slide, sentence with direct object complement, sentence with prepositional complement, and noun phrase. For each type of search, a table will display the results of the experiment. The table will show the following results:

- The total number of results returned by the application where the *matching percentage* (Chapter 5 section 5.6.2) is greater or equal to 75%.
- The number of results from the Google Desktop system and the number of common results with our application

- The number of result from the Windows Search system and the number of common results with our application

1) Title of Document: In this section we use sentences made of the entire or partial Open Office presentations' name. The sentences used for this section are:

1. Knowledge Management
2. Creativity and Innovation
3. Semantic Web

Table 0-1 Scenario 1: Title Document Results

Experiment number	Application results	Google Desktop results / common application results	Windows Search results/ common application results
1st sentence	14	6 / 4	4 / 4
2nd sentence	1	1 / 1	1 / 1
3rd sentence	12	7 / 1	1 / 1

2) Title of slide: In this section we use sentences made of the exact slide titles used from different presentations. The sentences used for this section are:

1. Semantic search
2. The world wide web
3. Teamwork challenges

Table 0-2 Scenario 1: Title of Slide

Experiment number	Application results	Google Desktop results / common application results	Windows Search results/ common application results
1st sentence	4	7 / 0	2 / 0
2nd sentence	1	2 / 1	3 / 1
3rd sentence	2	2 / 0	2 / 0

3) Direct object complement: we will use question sentences that start with the pronoun "what", where a pronoun can be the verb subject or the verb object complement. Questions will be short sentences.

1. What do ontologies require
2. What is knowledge discovery
3. What does a Statement object provide
4. What do semantics involve
5. What represents general system inference skills

Table 0-3 Scenario 1: Direct object complement Results

Experiment number	Application results	Google Desktop results / common application results	Windows Search results/ common application results
1st sentence	1	0 / 0	0 / 0
2nd sentence	1	1 / 0	0 / 0
3rd sentence	1	0 / 0	0 / 0
4th sentence	1	2 / 0	0 / 0
5th sentence	1	0 / 0	0 / 0

4) Prepositional complement: we will use question sentences that start with how. Questions will be short sentences.

1. How can conflict be positive

2. How does information need to be processed
3. How does data in the semantic web need to be made
4. How to make teaching more effective
5. How does a distributed agent architecture work

Table 0-4 Scenario 1: Prepositional Complement Results

Experiment number	Application results	Google Desktop results / common application results	Windows Search results/ common application results
1st sentence	1	1 / 0	2 / 0
2nd sentence	1	1 / 0	2 / 0
3rd sentence	1	0 / 0	0 / 0
4th sentence	1	0 / 0	0 / 0
5th sentence	2	0 / 0	0 / 0

5) Noun phrase: we will use noun phrase. Phrases will be short

1. Metadata and Ontology
2. Meeting facilitation
3. Communication skills
4. Developed distributed agent
5. Information resource

Table 0-5 Scenario 1: Noun Phrase Results

Experiment number	Application results	Google Desktop results / common application results	Windows Search results/ common application results
1st sentence	3	2 / 0	0 / 0
2nd sentence	1	1 / 0	1 / 0
3rd sentence	1	3 / 0	1 / 0
4th sentence	2	2 / 0	0 / 0
5th sentence	7	5 / 0	3 / 0

7.2 Scenario 2

In this scenario three of the five properties mentioned in the first scenario will be taken into account: *NounverbSubjectObject, NounverbSubjectPrep* and *NounPhrase*. The sentences used in the second scenario are slightly different in terms of words than the sentences used in the first scenario, but their meanings are equivalent.

1) Direct object complement: we will use question sentences that start with the pronoun "what", where a pronoun can be the verb subject or the verb object complement. Questions will be short sentences

1. What does ontology necessitate
2. What does a Statement object supply
3. What do semantics implicate

Table 0-6 Scenario 2: Direct Object Complement Results

Experiment number	Application results	Google Desktop results / common application results	Windows Search results/ common application results
1st	1	0 / 0	0 / 0
2nd	1	0 / 0	0 / 0
3rd	1	0 / 0	0 / 0

2) Prepositional complement: we will use question sentences that start with "how". Questions will be short sentences.

1. How can dispute be positive
2. How does in the semantic web info need to be made
3. How to make teaching more effective

Table 0-7 Scenario 2: Prepositional Complement Results

Experiment number	Application results	Google Desktop results / common application results	Windows Search results/ common application results
1st	1	0 / 0	0 / 0
2nd	1	1 / 0	0 / 0
3rd	1	0 / 0	0 / 0

3) Noun phrase: we will use noun phrases. Phrases will be short

1. Reunion facilitation
2. Communication aptitude
3. Information asset

Table 0-8 Scenario 2: Noun Phrase Results

Experiment number	Application results	Google Desktop results / common application results	Windows Search results/ common application results
1st	1	0 / 0	0 / 0
2nd	1	0 / 0	0 / 0
3rd	7	0 / 0	0 / 0

7.3 Discussion

As we can see from the Table 7.1 and Table 7.2, when the user query is the name of the document or a title from a slide of the presentations, all three applications return a considerable amount of results. It is likely to find common results from all three applications. Additional results shown in Table 7.9 shows a great accuracy from all three applications when dealing with document title.

Table 0-9 Results from all Three Applications Using Document Title

User query: Knowledge Management	
Our Application Results	1. C:../My Documents/…/On the use and architecture of Knowledge Management/Slide5 2. C:../My Documents/.../On the use and architecture of Knowledge Management/Slide6 3. C:../My Documents/.../On the use and architecture of Knowledge Management/Slide7 4. C:../My Documents/.../On the use and architecture of Knowledge Management/Slide8 5. C:../My Documents/.../On the use and architecture of Knowledge Management/Slide9 6. C:../My Documents/.../Knowledge Management in Higher Education/Slide5 7. C:../My Documents/.../Knowledge Management in Higher Education 8. C:../My Documents/.../Knowledge Managemen HPt/Slide2 9. C:../My Documents/.../Knowledge Managemen HPt/Slide6 10. C:../My Documents/.../On the use and architecture of Knowledge Management 11. C:../My Documents/.../Knowledge Managemen HPt 12. C:../My Documents/.../On the use and architecture of Knowledge

	Management/Slide2 13. C:../My Documents/.../Knowledge Management 14. C:../My Documents/.../On the use and architecture of Knowledge Management/Slide4
Google Desktop Results	1. C:../My Documents/.../Knowledge Management 2. C:../My Documents/.../Knowledge Management in Higher Education 3. C:../My Documents/.../Knowledge Management HPt 4. C:../My Documents/.../IT Workshop 5. C:../My Documents/.../Engineering and project mgmt 6. C:../My Documents/.../On the use and architecture of Knowledge Management
Windows Search Resutls	1. C:../My Documents/.../ Knowledge Management 2. C:../My Documents/.../ Knowledge Management in Higher Education 3. C:../My Documents/.../ Knowledge Management HPt 4. C:../My Documents/.../On the use and architecture of Knowledge Management

When the user query is a question (Table 7.3 and Table 7.4), our application was able to return at least one result while the other applications, Google Desktop and Windows Search, were not able to generate any results for 6 out of 10 questions. As we can see in Table 7.10, our application is able to return accurate results. The results point to the presentation followed by the slide number where the desired information is stored.

Table 0-10 Results from our Application when User Query Is a Question

User Query	Application Results
What do ontologies require	`C:/../My Documents/.../Semantic Web/Slide32`
What represents general system inference skills	`C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide21`
How does a distributed agent architecture work	1. `C:/../My Documents/.../Knowledge Managemen HPt/Slide22` 2. `C:/../My Documents/.../Semantic Web/Slide51`
How to make teaching more effective	`C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide4`

When the user query is a noun phrase (Table 7.5), our application and Google Desktop were both able to generate a considerable amount of results. Results from Table 7.11 show a greater accuracy and better results by our application. Our application points directly to the slide number of the presentation while both Google Desktop and Windows Search point only to the URI of the presentation

Table 0-11 Results from all Application where User Query Is a Noun Phrase

User query: Information Resource	
Application Results	1. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide17` 2. `C:/../My Documents/.../Semantic Web/Slide18` 3. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide22` 4. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide25` 5. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide13` 6. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide14` 7. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide16`
Google Desktop Results	1. `C:/../My Documents/.../`On the use and architecture of Knowledge Management 2. `C:/../My Documents/.../`SemanticWeb 3. `C:/../My Documents/.../`Knowledge Management 4. `C:/../My Documents/.../`Knowledge Management in Higher Education 5. `C:/../My Documents/.../`Assessing the Environment
Windows Search Results	1. `C:/../My Documents/.../`On the use and architecture of Knowledge Management 2. `C:/../My Documents/.../`SemanticWeb 3. `C:/../My Documents/.../`Knowledge Management

When dealing with synonyms, (Table 7.6, Table 7.7 and Table 7.8), our application was still able to return results even when some words where changed inside the sentence and replaced by their synonyms. The Table 7.12 below shows that the results returned by our application in the first scenario are the same than the results in the second scenario. Also the results from: Table 7.6, Table 7.7 and Table 7.8, shows that the other two applications, Google Desktop and Windows Search were not able to generate any results for 8 out of 9 queries.

Table 0-12 Results while Taking Synonyms into Account

User Query	Application Results
What does ontology necessitate	`C:/../My Documents/.../Semantic Web/Slide32`
What does ontology require	`C:/../My Documents/.../Semantic Web/Slide32`
Information asset	1. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide17` 2. `C:/../My Documents/.../Semantic Web/Slide18` 3. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide22` 4. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide25` 5. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide13` 6. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide14` 7. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide16`
Information resource	1. `C:/../My Documents/...On the use and architecture of Knowledge Management/Slide17` 2. `C:/../My Documents/.../Semantic Web/Slide18` 3. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide22` 4. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide25` 5. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide13` 6. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide14` 7. `C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide16`
How to make teaching more productive	`C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide4`
How to make teaching more effective	`C:/../My Documents/.../On the use and architecture of Knowledge Management/Slide4`

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

In this research we built an application capable of extracting metadata from Open Office Presentations using a stochastic NLP. Also this application includes a metadata retrieval process responsible for matching a user search with the metadata stored in the RDF file. In this study, we compared the results from our metadata retrieval process with the results of two other well known search engines: Google Desktop and Windows Search.

8.1 Conclusions

An ontology generally is a hierarchical representation of objects and types of objects, their relations and properties. The ontology can be represented in different languages such as RDF, RDF-S. The application includes two types of files: RDF Metadata file and Vocabulary Ontology file.

Our RDF Metadata file provides a representation of the information from the Open Office presentations where each metadata is a RDF triple. The information is analyzed through an AI Analyzer. The AI Analyzer contains a NLP parser that processes the information. The AI Analyzer analyzes the parser results and determines the subject, the property and the object of the RDF triple to be stored in the RDF Metadata file.

The semantic language Resource Description Framework Schema (RDF-S) provides a set of terms that allowed building our Vocabulary Ontology. The Vocabulary Ontology is composed of different words: nouns, verbs and adjectives. In the extraction of metadata process, this Vocabulary Ontology was used to extract the basic information about the contents of the Open Office presentation by getting rid of plural from nouns and the verbs conjugation inside a sentence before being saved as metadata. Our system only requires a Vocabulary Ontology and not any domain-specific ontology which increases the flexibility with regard to other semantic web applications that are specialized in a particular topic or domain.

We carried out different tests in order to evaluate the accuracy and reliability of our application. The tests were divided into two scenarios, one without and one with synonyms. The tests were conducted to compare the results from our application with results of other two search engines: Google Desktop and Windows Search. While our application returns the physical address of the presentation and the slide number, the comparing search engines only return the physical address of the presentation when a match is found. When taking synonyms into account as in the second scenario, two different sentences can be made of different words but they have the same meaning. So it is not just matching words, but matching meanings which leads to a more accurate search pointing not only to the physical address of the full presentation but pointing to a specific slide within a presentation.

Our investigation presents a RDF file with information extracted from Open Office presentations that considers the format of the presentations and each sentence and bullet inside the slides of a presentation. Our work also includes the use of synonyms in retrieving metadata.

8.2 Future Work

Within the main future work we propose:

- The extension of the system for Microsoft PowerPoint files (.ppt and .pptx files)

- The extension and representation of the RDF Metadata file in a more expressive semantic language, such as the Web Ontology Language (OWL)). The language provides additional vocabulary, relations and constraints to give greater expressiveness to the elements that are part of an ontology. OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. Using this language, better semantic representation and construction of a more powerful metadata file can be achieved.

- Exploiting further the *relational analysis* provided by the Stanford Parser in order to obtain and use more complex semantic properties and relations of the sentences. These properties will not be measured by the words in the sentence but rather by the relationships of the words inside the sentence. It will also be interesting to investigate the parser with information for other types of questions, *e.g.*, questions starting with adverbs not addressed in this thesis, e.g. *"Why"* and *"Where"*

- Extend the information extraction from presentations beyond the four types of slides analyzed in this thesis (Title slide, Title and Text, Title and Two Text, Title only), but for all the types of slides that are used in a presentation, e.g. diagrams, chart, pictures, etc.

- Making use of a lexicon and thesaurus, while making a search, can bring a lot of problems. For instance, the main problem we had dealing with synonyms was that a word can have five synonyms and it is not time-efficient to make five different searches. During our investigation we came across an online thesaurus with a very comprehensive vocabulary: http://wordnet.princeton.edu/ . WordNet is a large lexical database of English. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms. Wordnet is available freely and publicly for download. Using this web service or the downloaded application we propose the following methods to be investigated:

1. Having different synonyms pointing to one word called the *"base word"*

2. Before storing the information, reduce the information into *"base words"*, i.e., replacing each word with their base word inside the sentence

3. When making a search, convert the user's query into "base word(s) query" and then match the "base word(s) query" in the file containing the metadata.

BIBLIOGRAPHY

[Al-Khalifa06] Al-Khalifa, H.S.; Davis, H.C.; “FolksAnnotation: A Semantic Metadata Tool for Annotating Learning Resources Using Folksonomies and Domain Ontologies”. Innovations in Information Technology, 2006 Nov. 2006 Page(s):1 – 5

[Aleman-Meza05]Aleman-Meza, B.; Halaschek-Weiner, C.; Arpinar, I.B.; “Ranking complex relationships on the semantic Web”. Cartic Ramakrishnan; Sheth, A.P.;Internet Computing, IEEE Volume 9, Issue 3, May-June 2005 Page(s):37 – 44

[Amare04] Amare Nicole; “Technology for Technology’s Sake: The Proliferation of *PowerPoint*.” *Proceedings of the International Professional Communication Conference*. Minneapolis, MN: September, 2004. 61-63

[Bouzeghoub04]Bouzeghoub, A.; Defude, B.; Ammour, S.; Duitama, J.-F.; Lecocq, C.; “A RDF description model for manipulating learning objects”. Advanced Learning Technologies, 2004. Proceedings. IEEE International Conference on
30 Aug.-1 Sept. 2004 Page(s):81 – 85

[Ceravolo07]Ceravolo, P.; Damiani, E.; Viviani, M.; “Bottom-Up Extraction and Trust-Based Refinement of Ontology Metadata”. Knowledge and Data Engineering, IEEE Transactions on Volume 19, Issue 2, Feb. 2007 Page(s):149 – 163

[Cheonshu05] Cheonshu Park; Joochan Shon; “A study on the web ontology processing system”. Advanced Communication Technology, 2005, ICACT 2005. The 7th International Conference on Volume 2, 0-0 0 Page(s):1035 – 1038

[Christopher99] Christopher D. Manning, Hinrich Schütze: Foundations of Statistical Natural Language Processing, MIT Press (1999)

[Cruz04] Cruz, I.R.; Huiyong Xiao; Feihong Hsu; “An ontology-based framework for XML semantic integration”. Database Engineering and Applications Symposium, 2004. IDEAS '04. Proceedings. International 7-9 July 2004 Page(s):217 – 226

[Dinos04] Dinos Larry; “Architecture of a system based on Agents for the retrieval of Metadata RDF on the basis of an Ontology of documents” 2004. Thesis M.S. Universidad de Puerto Rico, Mayagüez, P.R.

[Enhong05]Enhong Chen; Gaofeng Wu; “An ontology learning method enhanced by frame semantics”. Multimedia, Seventh IEEE International Symposium on 12-14 Dec. 2005 Page(s):8 pp

[Gomez02] Gómez-Pérez, A. y O. Corcho. 2002. Ontology Languages for the Semantic Web. Intelligent Systems, IEEE, Volume: 17, Issue: 1, Pages: 54-60.

[Gruber07] Tom Gruber; “Ontology”.http://tomgruber.org/writing/ontology-definition-2007.htm

[Hsin05] Hsin-Chang Yang; Chung-Hong Lee, “Automatic Metadata Generation for Web Pages Using a Text Mining Approach”. Web Information Retrieval and Integration, 2005. WIRI '05. Proceedings. International Workshop on Challenges in 08-09 April 2005 Page(s):186 - 194

[Huajun05]Huajun Chen; Zhaohui Wu; Yuxin Mao; “Rewriting queries using views for RDF-based relational data integration”. Tools with Artificial Intelligence, 2005. ICTAI 05. 17th IEEE International Conference on 14-16 Nov. 2005 Page(s):5 pp.

[Hui03]Hui Han; Giles, C.L.; Manavoglu, E.; Hongyuan Zha; Zhenyue Zhang; Fox, E.A.; “Automatic document metadata extraction using support vector machines” Digital Libraries, 2003. Proceedings. 2003 Joint Conference on 27-31 May 2003 Page(s):37 – 48

[Isahara07] Isahara, Hitoshi; “Resource-based Natural Language Processing”. Natural Language Processing and Knowledge Engineering, 2007. NLP-KE 2007. International Conference on Aug. 30 2007-Sept. 1 2007 Page(s):11 – 12

[Joakim02] Joakim NIVRE “On Statistical Methods in Natural Language Processing”. School of Mathematics and Systems Engineering, Växjö University, SE-351 95 Växjö, Sweden

[Kapetanios95] Kapetanios, E.; Kramer, R.; “A knowledge-based system approach for scientific data analysis and the notion of metadata”. Mass Storage Systems, 1995. 'Storage - At the Forefront of Information Infrastructures', Proceedings of the Fourteenth IEEE Symposium on 11-14 Sept. 1995 Page(s):274 – 283

[Kathlenn04] Kathleen Dollard; “Code Generation in Microsoft.Net”. Apress, published January 19 2004, page(s) 1-22.

[Klein02] Klein, M.; “Interpreting XML documents via an RDF schema ontology”. Database and Expert Systems Applications, 2002. Proceedings. 13th International Workshop on 2-6 Sept. 2002 Page(s):889 – 893

[Kyong03] Kyong-Ho Lee; Yoon-Chul Choy; Sung-Bae Cho; “Logical structure analysis and generation for structured documents: a syntactic approach” Knowledge and Data Engineering, IEEE Transactions on Volume 15, Issue 5, Sept.-Oct. 2003 Page(s):1277 – 1294

[McBride02] McBride; Hewlett-Packard Labs, “Jena: a semantic Web toolkit” : Internet Computing, IEEE Nov/Dec 2002 Volume: 6, Issue: 6 Page(s): 55- 59

[M. Stanojevic05] M. Stanojevic; S. Vranes; “A Natural Language Processing for Semantic Web Services”. Computer as a Tool, 2005. EUROCON 2005.The International Conference on Volume 1, 2005 Page(s):229 - 232

[Miller98] Miller Eric; “An Introduction to the Resource Description Framework” http://www.dlib.org/dlib/may98/miller/05miller.html

[Min-Yuh05] Min-Yuh Day; Tzong-Han Tsai; Cheng-Lung Sung; Cheng-Wei Lee; Shih-Hung Wu; Chorng-Shyong Ong; Wen-Lian Hsu; “A knowledge-based approach to citation extraction”. Information Reuse and Integration, Conf, 2005. IRI -2005 IEEE International Conference on. 15-17 Aug. 2005 Page(s):50 – 55

[Mutton03] Mutton, P.; Golbeck, J.; “Visualization of semantic metadata and ontologies” Information Visualization, 2003. IV 2003. Proceedings. Seventh International Conference on 16-18 July 2003 Page(s):300 – 305

[Nagao05] Nagao, M.; “Natural language processing and knowledge”. Natural Language Processing and Knowledge Engineering, 2005. IEEE NLP-KE '05. Proceedings of 2005 IEEE International Conference on 30 Oct.-1 Nov. 2005 Page(s):1

[Payne94] Payne, J.S.; Stonham, T.J.; Patel, D.; “Document segmentation using texture analysis ”. Pattern Recognition, 1994. Vol. 2 - Conference B: Computer Vision & Image Processing., Proceedings of the 12th IAPR International. Conference on Volume 2, 9-13 1994 Page(s):380 - 382 vol.2

[Peregudov04] Peregudov, A.F.; Glasman, K.F.; “Enrichment of semantic metadata based on interactions with user associations” Consumer Electronics, 2004 IEEE International Symposium on 2004 Page(s):604 – 608

[Potok02] Potok, T.E.; Elmore, M.T.; Reed, J.W.; Samatova, N.F.; “An ontology-based HTML to XML conversion using intelligent agents”. System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on 7-10 Jan 2002 Page(s):1220 – 1229

[Staab01] Staab, S., A. Maedche, S. Handschuh. “An Annotation Framework for the Semantic Web”. In: S. *Ishizaki (ed.), Proceedings of The First International Workshop on MultiMedia Annotation. January. 30 - 31, 2001. Tokyo, Japan*

[Sheth02] Sheth, C. Bertram, D. Avant, B. Hammond, K. Kochut, Y. Warke, “Semantic Content Management for Enterprises and the Web” IEEE Internet Computing, July/August 2002, pp. 80-87.

[Simon97] Simon, A.; Pret, J.-C.; Johnson, A.P.; "A fast algorithm for bottom-up document layout analysis". Pattern Analysis and Machine Intelligence, IEEE Transactions on Volume 19 , Issue 3, March 1997 Page(s):273 - 277

[Stanford03] The Stanford Natural Language Processing Group, http://nlp.stanford.edu/software/lex-parser.shtml

[Steinacker01] Steinacker, A., A. Ghavam y R. Steinmetz. 2001. Metadata Standards for Web-Based Resources. Multimedia, IEEE, vol: 8, issue: 1, pp: 70-17.

[Stephen02]Stephen Buswell, Dan Brickley, Brian Matthews; "Semantic Web Advanced Development for Europe". Integration with XML Technology. Project Number:IST-2001-34732

[Stephens04]Stephens, R.T.; "Utilizing metadata as a knowledge communication tool". Professional Communication Conference, 2004. IPCC 2004. Proceedings. International 2004 Page(s):55 – 60

[Stuart03] Stuart Russel, Peter Norvig; "Artificial Intelligence, A modern Approach" Pearson Education, INC 2003, Upper Saddle River, New Jersey 07548, Page 818

[T. A.00] T. A. Mostek, K. D. Forbus, and C. Meverden, "Dynamic Case Creation and Expansion for Analogical Reasoning", Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, Pages: 323 – 329, Year of Publication: 2000, ISBN: 0-262-51112-6.

[Tikk04] Tikk, D.; Kardkovacs, Z.T.; Andriska, Z.; Magyar, G.; Babarczy, A.; Szakadat, I.; "Natural language question processing for hungarian deep web searcher". Computational Cybernetics, 2004. ICCC 2004. Second IEEE International Conference on 2004 Page(s) :303 – 308

[Wu07] Wu, R.; "Semantic metadata in enterprise integration" SoutheastCon, 2007. Proceedings. IEEE March 2007 Page(s):233 - 236

[Yahaya06] Yahaya, N.A.; Buang, R.; "Automated Metadata Extraction from Web Sources". Web Intelligence and International Agent Technology Workshops, 2006. WI-IAT 2006 Workshops. 2006 IEEE/WIC/ACM International Conference on 18-22 Dec. 2006 Page(s):176 – 179

[Yang04] Yang, S.J.H.; Shao, N.W.Y.; Lan, B.C.W.; Irene Chen; "An ontology based content model for Web services description". Services Computing, 2004 (SCC 2004) Proceedings. 2004 IEEE International Conference on 15-18 Sept. 2004 Page(s):245 - 252

[Yunhua05]Yunhua Hu; Hang Li; Yunbo Cao; Meyerzon, D.; “Automatic extraction of titles from general documents using machine learning”; Digital Libraries, 2005. JCDL '05. Proceedings of the 5th ACM/IEEE-CS Joint Conference on 7-11 June 2005 Page(s):145 - 154

[Weishan06] Weishan Zhang; Kunz, T.; “Product Line Based Ontology Development for Semantic Web Service”. Service-Oriented System Engineering, 2006. SOSE '06. Second IEEE International Workshop Oct. 2006 Page(s):183 - 188

[Wilde07] Wilde, Erik; “Declarative Web 2.0”. Information Reuse and Integration, 2007. IRI 2007 IEEE International Conference on 13-15 Aug. 2007 Page(s):612 – 617

Appendix

Appendix A Acronyms

AI	Artificial Intelligence
DAML+OIL	*Darpa Agent Markup Language + Ontology Inference Layer*
DCMI	*Dublin Core Metadata Initiative*
FIPA	*The Foundation for Intelligent Physical Agents*
HTML	*Hypertext Markup Language*
KCM	*Keyword Cluster Map*
KQML	*Knowledge Query and Manipulation Language*
NLP	*Natural Language Processing*
OIL	*Ontology Inference Layer*
oSEMA	*Ontology Semantic Agent*
OWL	*Ontology Web Language*
RDF	*Resource Description Framework*
RDF-S	*Resource Description Framework Schema*
RDQL	*RDF Data Query Language*
SOM	*Self Organizing Map*
SVM	*Support Vector Machine*
WWW	*World Wide Web*
W3C *World*	*Wide Web Consortium*