# New Searching Techniques for Ontology Based Search Engines – A Conceptual Study

Sareena Rose, Asst.Professor, Department of Computer Science, Vimala College, Thrissur

**Abstract**— Search engines play a vital role in finding the relevant information across the web and make it available on the finger tips. But they seemed to be less efficient to understand the relationship between the keywords which had an adverse effect on the results it produced. Semantic search engines – only solution to this; is still an unrealized dream due to various underlying issues. The hindrances faced by semantic search engines such as annotating the documents spread across network, quality of annotation, incompatibility between the query languages stumbles the growth of semantic search engines. Reducing the time taken to search the semantic annotated documents is a highly demanded solution for these search engines. This paper focuses on a study and improvisation of searching techniques used in semantic search engines keeping time complexity as the major factor.

**Index Terms**— Ontology, OWL, RDF, Semantic search engine, Semantic annotation, WordNet.

## 1 Introduction

WORLD Wide Web is the biggest revolution which has happened to the internet technology. It still retains it pride as it serves and helps human kind indeed in many ways. Web Search engines are information retrieval systems designed to search for information stored in the web pages. The web is a huge distributed and linked mass of many resources which are poorly unstructured and unorganized. And the search engines connect man to the resources spread worldwide and quenches his hunger for knowledge. But not always the layman is happy with the results produced by the search engines and is not patient enough to browse through complete list of pages to get a relevant result.

The reason behind this is the search engines performs search based on the syntax not on semantics. The keyword based search engines fails to understand and analyze the context in which keywords are used. The situation worsens when the search phrase is a combination of keywords. The quality of the results degrades with irrelevant results of documents which contains only the part of search phrase leaving the meaning aside. A semantic search –*Tim Berner Lee's unrealized dream* - resolves this issue by analyzing the contexts and the relationships between the key words and thus producing a "quality high" and "quantity less" results. The basic idea of semantic web is to enrich the current Web with machine cognitive information about the semantics of information content.

## 2 Problem Statement

Keyword based search engines offers the best service when the user wants to know about a very general topic. But, fails when the topic is too specific and user knows little about the same. Yet another difficulty is when the user inputs a combination of keywords. It gives you a plenty of results in this case but most of them will be irrelevant to the topic leaving the user with the task of finding the right one out of it. The order in which the results are ranked also creates confusion. Underlying problem is the searching pattern. Keywords based search engine fails to understand the relation between the keywords; they consider each search input as different keywords and fetch the pages which contain it without bothering about the relationship status.

The answer to this is semantic search engines, which is based on semantics not on syntax. It works by blending conceptual with contextual meaning. But these search engines realization is quite far due to few reasons. Firstly, the semantic search engine requires semantically annotated resources as its prerequisite. It is a mightier task to semantically annotate millions of resources available on the web. And the number of documents added to web rises in an exponential manner per year. Since no one owns the web and there are no strict regulations to keep the web going it is difficult to track whether documents - both added and existing – are semantically annotated. The semantic annotation forms the crust of semantic search engines as these engines parse these annotations to produce the results. Thus the quality of the semantic annotation has a huge impact on the relevancy of the results produced. This situation highly demands the semantic annotation to be done by a domain expert rather than a programmer or a web designer. This again attributes to the slow growth of SSE as the number of documents and the domain experts are not in right proportion.

Secondly, the documents available on the Web are poorly structured or unstructured. These documents once processed for semantic search has two parts- semantic annotated part and content rich text. The searching process requires querying both these parts with different query languages which are incompatible with each other. This adds as another reason for the slow development of SSE. Thirdly, the RDF Schema and the OWL provides all the flexibility to specify rich semantic descriptions about the concepts and the relationship between the concepts. These RDF descriptions are stored in RDF store which is either implemented using a Relational database or a tree/graph like data structure. Due to the power, ease of usage, popularity of RDBMS these databases are preferred over data structures. So the storage mechanisms require the mapping of RDF to a RDBMS. These factors decelerate the development of search engines as well as retard the searching speed of them. In this era of object oriented concepts, with reusability as the prime motive; the paper focuses on optimizing the search of SSE by reusing existing resources available on the web. The paper also focuses on filtering the results by confining them to the applicable domain and thus improving the relevancy of results.

## 3 Methodology

The research work is aimed at increasing the speed and effectiveness of semantic search by semantically expanding the users query in a user friendlier manner and resolves the compatibility issues of SPARQL and SQL [1],[6] by annotating the domains into which the web pages are categorized. The model requires the semantically annotated domains, categorized web pages as prerequisites.

The model proposed works in three different phases. The first phase is context extraction phase which receives the input from the User Interface and extracts the different contexts in which the keyword can be used. The second phase is Related Entities Extraction phase which finds out the concepts which are related to the keywords. The third phase semantically expands the query keyword in terms of the related entities extracted by the previous phase. Also this is the searching phase which searches the inverted database to retrieve most relevant options and presents the results to the user.

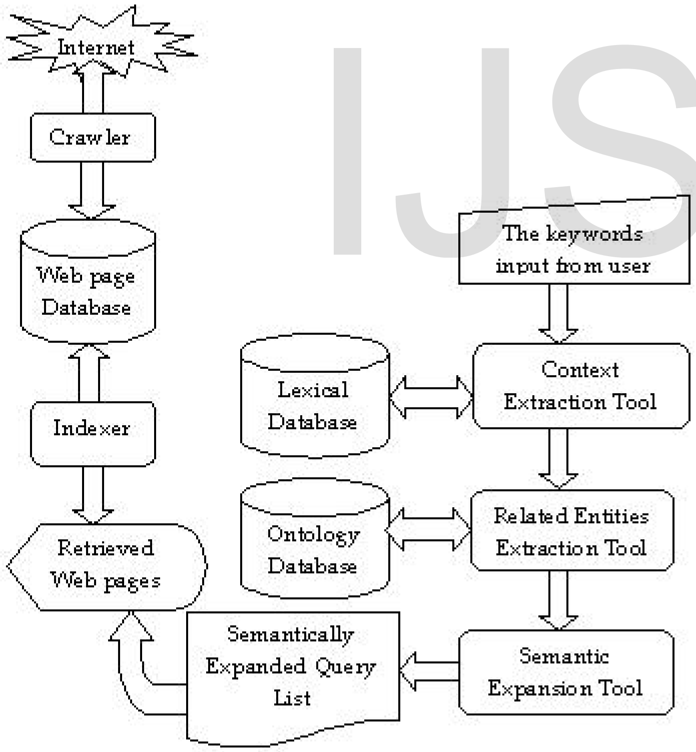

Fig 1. Working Model of proposed Semantic search Engine

### 3.1 Phase 1 - Context Extraction Phase

A word in English language can be used in different contexts depending on the way in which it is used (Noun, Verb, Adjective). Lexical semantics is the study of how and what the words of a language denote. Words may either be taken to denote things in the world, or concepts, depending on the particular approach to lexical semantics. Lexical semantics covers theories of the classification and decomposition of word meaning, the differences and similarities in lexical semantic structure between different languages, and the relationship of word meaning to sentence meaning and syntax. Lexical relations is defined as patterns of association that exist between lexical items in a language using tools like synonymy, antonymy (opposites), hyponymy and hypernymy - and to a certain degree homonymy as well -are used in this field.

Lexical database is chosen as the tool for the first phase because of the following reasons.

1. Sense disambiguation is crucial for Information Retrieval.
2. It is the best available resource for extracting the contexts of keywords.
3. The database can be modified to include the new terms or relations whenever required.
4. Ease of use, understandability

WSD is an application of Artificial Intelligence and is used to identify the meaning of words. WSD can be viewed as a classification task: word senses are the classes, and an automatic classification method is used to assign each occurrence of a word to one or more classes based on the evidence from the context and from external knowledge sources where systems are expected to disambiguate all open-class words in a text (i.e., nouns, verbs, adjectives, and adverbs). This task requires wide-coverage systems such as dictionary and thesaurus. So this is also known as knowledge-based or dictionary-based WSD. This helps to exploit knowledge resources (such as dictionaries, thesaurus, ontologies, collocations,) to infer the senses of words in context. These methods usually have lower performance than their supervised alternatives, but they have the advantage of a wider coverage, because of the use of large-scale knowledge resources. Using knowledge based WSD in IR systems creates a drastic impact on its performance by making the sense of the word clear which directly helps in confining the search to that particular domain.

```
Algorithm : ContExt(Keyword k)
Input: The keyword k
Output: The selected context related to the keyword: SC.

1) Set k as the keyword entered by the user.
2) Generate IndexWord for keyword k as NOUN.
3) Process IndexWord to generate synsets and store in
   IN_D document set.
4) Display IN_D to the user.
5) Store the user selected one in SC.
```

The essential idea of this algorithm is to automatically extract the senses related to the keyword when it is used NOUN. We omit VERB, ADJECTIVE and ADVERB for the time being assuming more relevance goes to former states of word. Retrieving the synonyms of the keyword as NOUN almost covers all the contexts related to it. In the procedure it sets the keyword as the input and retrieves the IndexWord. An IndexWord is a single word and part of speech can be used to lookup a Synset object. It then extracts the synset for which the keyword appears as noun. Then it processes every set of senses and thus obtains the different contexts of the given keyword. It then displays the resultant set and prompting the user to select the context. The output is the context selected by the user is passed on to the next phase as input.

### 3.2 Phase 2 Related Entities Extraction Phase (REE)

REE is the second phase in the model proposed. It works on extracting the entities relating to the context of the keyword selected by the user in the previous phase. The input of this phase is the output of previous phase and output of this phase is semantically expanded query list which will be fed to the next phase as input.

**Pre-requisite and tools used are**

1. Pre-categorized documents which are classified using text categorization methods which employs semantic methods.
2. A very well semantically annotated domains
3. Ontology database mapped on to RDBMS [4],[8]

Automatic text categorization[10] is a task of assigning one or more pre-specified categories to an electronic document based on its content. There are two approaches for text categorization. First is knowledge engineering approach in which expert's knowledge about the categories is directly encoded in the system either declaratively or in the form of procedural classification rules. The other is Machine Learning approach in which general inductive processes builds a classifier by learning from a set of pre classified examples. Text categorization is very useful in text indexing, document sorting, and text filtering and Web page categorization. The automatic text categorization method when employed in semantic search model reduces the overhead of search engines in finding out the relevant document from a pool of heterogeneous documents database. Also this organizes the knowledge in a very efficient way so that it can be conveniently and effectively used by anyone for any kind of IR purposes.

Domain in the context of the search model can be defined as a cluster of document whose contents are either similar or related to each other. Domains can be hierarchically organized and the documents belonging to these domains can be identified using text categorization methods as explained above. These domains need to be semantically annotated as the next step in this proposed search model. Semantic annotations in a document are additional information that identifies or defines a concept in semantic model in order to describe a part of the document. RDF (Resource Description Framework) [7] offers a framework to model hierarchies of classes and properties. RDF was developed to provide a standard way to model, describe and exchange information about "resources" and also serves as a base for higher level language that describes ontologies. In RDF, a statement links two resources. The statement is viewed as sentences that have subject-verb-object structure.

Eg. I live in Bangalore

(subject:I) ( predicate: live in ) ( object: Bangalore )

By semantically annotating the domains we solve the incompatibility between query languages, reduces errors, saves time as annotation requires highly skilled labor. Now the semantically annotated domains can be queried by SPARQL and the text content using SQL. The incompatibility issues won't arise since the domains are stored separate from the content rich web pages. This solves a major issue but a good classification algorithm is required here to categorize new and existing web pages into various domains. If we annotate only the domains keeping the document as it is, it reduces the overhead of annotating every existing document. This will in turn save the time of domain expert. Secondly, the semantic annotation demands high knowledge of the domains which may require the assistance of a domain expert. It is not feasible to seek assistance of the domain expert to annotate every single resource on the web classified to be in this domain. Another reason is since no one owns the Web and there is no regulatory body to control the functioning of web it may not be possible to track whether each document added to web is semantically annotated or to check whether semantic annotation is properly done. The exponential rise in number of resources added to the web makes it still more impractical. These overheads can be totally avoided by this method. We can make use of one or n experts help for the same so that the quality of annotation can be preserved as well as the errors can be brought down to a very lower level.

Explosive growth of RDF data on the Web drives the need for novel database systems, called RDF stores that can efficiently store and query large RDF datasets. Most existing RDF stores, including Jena, Sesame, 3store, KAON, RStar, OpenLink Virtuoso, DLDB, RDFSuite, DBOWL, PARKA, and ProvRDF, use a relational database management system (RDBMS) as a backend to manage RDF data[7],[5]. The main advantage of the RDBMS-based approach is that a mature and vigorous relational query engine with transactional processing support can be reused to provide major functionalities for RDF stores. Relational databases stores RDF as triples in tables as subject, predicate and object as their fields.

Predicate which takes four different values is only considered in this search model:

1. Instance of: The members or instances of a class are referred to as individuals in OWL. This relation is analogous to the "instance of" relationship in OOPS. The mapping process changes all the "rdf:type" relation in OWL to "instance of" on RDBMS. For instance, Daisy rdf:type owl:Flower is mapped to RDBMS as Daisy (Subject) instance of(Predicate) flower(Object).

2. A kind of: The inheritance in OWL can be represented using the "rdfs:subclassof" syntax. For instance owl:Mammal rdf:type owl:Class

   owl:Human rdf:type owl: Class rdfs:subclassof owl:Mammal.

   This implies that

   1. Human is a specialization of Mammal.
   2. Human has all the attributes and properties of mammal.
   3. Also any restrictions on Mammal class will be inherited by Human.

   The mapping process changes all the "rdfs:subclassof" relation in OWL to "a kind of" on RDBMS. The above example can be mapped to Human (subject) a kind of (predicate) Mammal (Object).

3. Also known as: The synonyms are represented in this model as 'aka' whose abbreviation is "also known as". owl:equivalent Class is a built-in property that links a class description to another class description. The meaning of such a class axiom is that the two class descriptions involved have the same class extension (i.e., both class extensions contain exactly the same set of individuals).

   ```
   <owl:Class rdf:about="#Cancer">
   <owl:equivalentClass rdf:resource="#Malignant_neoplasm"/>
   </owl:Class>
   ```

   The above statement implies that “Cancer” and “Malignant Neoplasm” is equivalent to each other and hence “Malignant Neoplasm” is considered as a synonym for “Cancer” and can be recorded as a tuple in ontology database with values Cancer(subject) aka (Predicate) Malignant Neoplasm

   (Object)

4. A part of: OWL does not provide any built-in primitives for part-whole relations (as it does for the subclass relation), but contains sufficient expressive power to capture most of the common cases. It can be represented in OWL using the “owl:inverseof” property which states that existence of property relationship in one direction implies that another relationship exists in the opposite, or inverse direction.

   ```
   <owl:ObjectProperty rdf:ID="hasPart">
   <rdf:type rdf:resource="&owl;FunctionalProperty" />
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="isaPart of">
   <owl:inverseOf rdf:resource="#hasPart" />
   </owl:ObjectProperty>
   ```

   The part of – whole relationship is represented in this model as “a part of”. For instance, Tyre is a part of car is equivalent to the statement Car has part Tyre. This can be recorded as a tuple in ontology database with values Tyre (subject) apartof (Predicate) Car (Object).

“Also known as” has the highest priority followed by “instance of”, “a kind Of”, and “a part of” predicates in this phase of search model proposed. The related entities will be extracted and ranked only on the above said order. The reasons which attributes to the selection of the above said predicates are:

1. ‘Also known as‘has the highest priority among predicates because the relation between keywords and synonyms has higher degree of closeness than any other predicates. Equivalent classes have the same extensions and their members are somewhat similar. Different words exist in English language which shares the similar meaning. The search will be more fruitful if the search engine can fetch results which have not only the keywords but its synonyms also.

2. ‘Instance of‘ is given priority 2 in this search model. The real world objects are modeled in OWL as members of classes. Members or individuals in OWL are the instances of classes. Moreover rather than concepts its members are frequently used as the physical existence of the concepts happens through its entities, concepts just remain as abstraction. In such a scenario, the ‘instance of‘ plays a major role in retrieving relevant results. For example, if the query keyword is DBMS then it will be necessary to include Oracle, SQL, MS Access also as these are instances of DBMS.

3. ‘A kind of‘and ‘a part of‘share almost the same priority while extracting the entities in this phase. ‘A kind of‘represents the inheritance or subclass relationship which can be used to retrieve the results with a specialized version of keywords. For instance, if the keyword used is ‘cricket‘ in the context of game its inherited classes such as T20, Test, One day also have the same significance in the set of results. ‘A part of‘ which represents part of – whole relationship is helpful in retrieving the attributes and properties of class while searching.

Algorithm 2 : RelEntExt(String Context, String Keyword)
*Input: Keyword and the context selected by the user.*
*Output: RE_S, A set of related entities extracted.*

Procedure:
Items used by the procedure:

- O_S, which temporarily stores the extracted related entities of each iteration.
- D_S, which stores the list of domains. Let D_S = {D1, D2, …………, Dn} where Di i=1 to n are the domains selected which are matching to the context.

```
Steps:
1. Set O_S= F.
2. For every domain Di in D_S,
        I Retrieve subjects where object is the keyword and
        predicate is ‘aka‘ and add these to semantic query list
        RE_S.
        II Retrieve subjects where object is the keyword and
        predicates are ‘instance of‘and ‘apart of‘. Add these to
        O_S temporary output list.
        III. Retrieve objects where subject is the keyword and
```

```
        predicates are 'a kind of'. Add these to O_S temporary
        output list.
3. Output O_S to the user.
4. If the user is satisfied go to Step # 5
   Else
        I. Read the new keyword set K= new Keyword.
        II. Call RelEntExt(Context, K).
5. Output RE_S.
6. Stop.
```

The algorithm 2 takes the output of the context extraction phase and the keyword entered by the user as input. It uses a list to store related entities extracted in each iterations also a list to store the domains which are selected as corresponding to the context. For every domains selected, the algorithm retrieves the subjects whose predicate is 'aka' and its object as the keyword. Also this step retrieves the object whose predicate is 'aka' and its subject as the keywords. The synonyms thus retrieved are stored in the final output document for the use in next phase. In the next step it is the turn of extracting all subjects which are related to its object by the predicate 'instance of' and 'a part of'. The reason behind retrieving subject using predicate is all the members of class occupies subject field when represented in OWL. 'A part of' is considered as the next predicate because this predicate in OWL represents the subject as specialization of the concept (objects).

In the following step the reverse process happens, all the objects are retrieved which are connected to its subject by 'a kind of' predicate. 'a kind of' predicate represents subclass relationship in OWL. All the generalized entities can be selected in this step for the given keyword by retrieving all the objects connected to this subject. This is given a last priority assuming that the specialization and instances have a significant role when compared to generalization.

The entities thus selected and stored in temporary output list are displayed to the user. The user can go for further refinement of extraction if he wishes to do so. Otherwise the RE_S (the set of related entities) is given as the output of this phase. If the user continues the above procedure is called recursively to find a new set of related entities by replacing the keyword with the newly selected one. The RE_S now contains a set of keywords selected by the user and its corresponding synonyms. RE_S can be denoted as {S1, S2, S3,…..,Sn} where n denotes the number of iterations the procedure recursively called where each Si = {Ki, Syni, Syn2, Syn3,………..,Synn}. Each element in the set Si are combined by Boolean Operator 'OR' and each Si will be combined by Boolean 'AND' operator.

This is the resultant set which contains semantically expanded query.

### 3.3Phase III - Searching the Web

The third phase is responsible for searching in the heterogeneous pool of web documents. This phase input is not just a keyword but a set of related terms connected by Boolean operators. The output of this phase is the net output for the whole search process which will be displayed to the user. The tools used are Google Advanced Search Interface, Indexer and crawler used by search engines.

## 4 Experimental Results

The search model had made use of

- WordNet Version 2.0 [7]
- Java Wordnet Library Functions (JWNL)
- Semantically annotated Domain Cancer
- Protégé 4.0
- RDBMS
- Google Advanced Search

For analysis purpose, the keyword "Blood Cancer" was selected. After the processing of first and second phase the output was semantically expanded query combined with OR and AND operators.

*(Blood cancer OR haematological Malignancy OR Leukemia OR Multiple Myeloma)*

*AND*

*(Cancer OR Malignant Neoplasm OR oncology).*

This was given as the query phrase in Google Advanced Search which yielded around 446,000 results in .43 seconds. And for the comparison purpose the term "Blood cancer" was given in the normal search which yielded 94,900,000 results in .05 seconds

The comparison between the two results can be recorded in tabular format as follows:

Table 1.

| Parameters | Semantic Search | Normal Search | Difference |
|---|---|---|---|
| Time(seconds) | .43 | .05 | .38 |
| No: of results | 446,000 | 94,900,000 | 94,554,000 |
| Relevant results | 133 | 53 | 80 |
| Irrelevant results | 17 | 97 | 80 |
| Precision | 88.667% | 35.333% | 53.334% |

The Recall Ratio cannot be used in this comparison as we are using Google database and we don't have an accurate number of documents existing in these databases in the above context. The analysis of results is done by taking a sample set of first 150 results of both search engines.

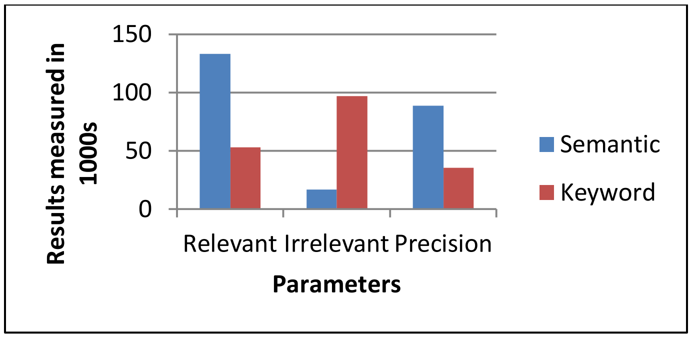

Fig 2. Performance Comparison of Semantic and keyword based search engines

The performance analysis shows that "Normal search" outweighed "Semantic Search" in terms of time taken to retrieve and no: of results retrieved. There is a huge difference in number in both the cases. But when the relevant and irrelevant results are taken into account "Normal Search" is nowhere near the "Semantic Search" in terms of its performance. The precision rate for semantic search engine when steeps high to nearly 90% the keyword based search engine stumbled down to 35%. The difference of the precision rate will gradually increase if increase the sample set size.

These results once again prove the need of semantic search by the search engines. It's high time that we change the way by which we search.

## 5 Future Work

The search model proposed in this research is devised to work only with one keyword. This method if implemented in a wide manner will be unable to meet user's requirements as the user is quite aggressive and greedy in their demands when it comes to search engines. They need relevant information in their fingertips no matter the subject is descriptive or objective. In such a scenario, the model proposed may not be sufficient enough to cater the needs. The first and foremost task in the refinement of this model is to make it work for multiple words. If a query is having multiple words or if it is a sentence, instead of directly feeding it into context extraction phase, it has to be processed using natural language processing methods. The queries have to be categorized according to the contexts in which they might occur. For example, the queries such as "differences between procedural and OOPS", "procedural versus OOPS", "similarity between procedural and OOPS" has to be categorized into a common group "compare - concept1, concept2". The "compare" set can take any of these values, {difference, versus, similarity}, and concept1 and concept2 can be any two concepts. Similarly, the phrases like "I cordially invite", "I will obliged" can be categorized into text type of queries. Once the type of queries is categorized, it can be decided whether we need to extract the contexts. The context extraction will be only good enough if the query contains any concepts. The other query types can be directly fed into rest of phases for further processing. However, normal search engines can yield best results for the text type queries, as we don't need a semantic search pattern for this. In a whole, we can assume that the refined model should categorize the queries before they go ahead in processing it.

Secondly, we can concentrate on automating the mapping of ontology to RDBMS. It will be easier to carry out this task using machine rather than consuming numerous man hours. Finally, a new algorithm to rank the resultant pages will serve this search engine better. The PageRank algorithm patented by GOOGLE used currently by this search model works on different metrics but most important one is considered to be the number of hits a page receives. The algorithm for this search model has to be reframed for the semantic search model such that pages are ranked on the following metrics: term frequency, its weightage, relevancy of terms which can be computed by finding the synonyms frequency of the respective terms and their related entities. It should be totally free from the number of hits. Such an algorithm can show justice to the search engine by not hiding the relevant links in back pages.

## References

[1] Artem Chebotko, Shiyong Lu, and Farshad Fotouhi, "*Semantics Preserving SPARQL-to-SQL Translation*", Data & Knowledge Engineering 2009.

[2] Chris Sherman, "*Google Power: Unleash the full potential of Google*", Mc Graw Hill Osborne 2005.

[3] Fritz Schneider, Nancy Blachman, Eric Fredricksen, "*How to do everything with Google,*" Mc Graw Hill Osborne 2004.

[4] Guobing Zou Bofeng Zhang Yanglan Gan Jianwen Zhang, "*An Ontology-based Methodology for Semantic Expansion Search*", Fifth International Conference on Fuzzy Systems and Knowledge Discovery 2008.

[5] Hakia Team, hakia "*Semantic Search Technology – making sense of the worlds Information*". White paper Jan 2010.

[6] John Hebeler, Matthew Fisher, Ryan Blace, Andrew Perez – Lopez, "*Semantic Web Programming*", Wiley India 2009.

[7] Karin K. Breitman, Marco Antonio Casanova, Walt Truszkowski, "*Semantic Web: Concepts, Technologies and Applications*", Springer Verlag Berlin Heidelberg London 2007.

[8] Lizhen Li, Zhifeng Dong, Keming Xie, "*Ontology of general concept for Semantic Searching*", Second International Conference on Computer Modeling and Simulation 2010.

[9] Luiz André Barroso, Jeffrey Dean, Urs Hölzle, "*Web Search For A Planet:The* Google Cluster Architecture", IEEE Computer Society 2003.

[10] Maciej Janik and Krys Kochut, "Training-less Ontology-based Text Categorization", European Conference on Information Retrieval (ECIR) March 30th 2008.

[11] WordNet: An Electronic Lexical Database. The MIT Press (1998)