ATLANTIS PRESS

Advances in Engineering Research (AER), volume 142

International Conference for Phoenixes on Emerging Current Trends in Engineering and Management (PECTEAM 2018)

Literature Survey: Analysis on Semantic Web Information Retrieval Methodologies

[1]K.Ezhilarasi, [2]G.Maria Kalavathy
[1]Research Scholar, Anna University, India
[2]Professor, St.Joseph's College of Engineering, India
k.ezhilarasi@yahoo.com, maria_kalavathy@yahoo.co.in

Abstract: **The Semantic web is the extension of an existing web that defines a standard by which, information is given in well-defined meaning and enables the machines to understand information. Many kinds of research are going in semantic web space. Researchers follow different approaches to retrieve data from the semantic web. This paper is to investigate the existing situation of the Semantic Web with the focus on effective information retrieval. Based on the architecture of semantic search engine, list of parameters (like ranking algorithm, reasoning mechanism) is framed to carry out a systematic analysis of different techniques proposed by the researchers. This survey identifies around 20 unique models that retrieve the data from the semantic web and information systems. Summary of the selected semantic search models is specified and compared them by means of "classification parameters" defined. This comparison identifies common insight, unique features and also open issues. This study can be used as a guide for future application development and research.**

Keywords: **Semantic search engine, Semantic Information retrieval, Ontology, semantic search**

I. INTRODUCTION

The Web has become an object of our daily life and the amount of information in the web is ever growing. Besides plaintexts, especially multimedia information such as graphics, audio or video has become a predominant part of the web's information traffic. But, how can we find our required information from this huge information space? Nowadays, there are many web search engines for retrieving information from the Internet. These traditional search engines retrieve and display the information based on the occurrence of words in a document, geographical location etc, instead of understanding the content by exploiting semantics.

For example, consider the user wants to find the name of all Indian researchers who have written documents on the semantic web during the last year. This kind of search is not feasible in traditional search engines, but the Semantic web has the ability to execute the search successfully since it associates formal meaning with the content.

The focus of semantic web is to make the web as machine-understandable by declaring annotations, where automated agents will be able to understand the content on the web, establish the relationship between them and take logical decisions to accomplish the complex task with minimum human interaction.

Ontology is generally defined as formal vocabulary and is considered as one of the main pillars of the semantic web. This is used to define the world by declaring the concepts and their relationships in an unambiguous way.

The Semantic Web search engines aim is to fetch the reliable, precise, relevant information what actually user needs without taking much time to traverse the irrelevant pages what traditional search engine does.

1.1 Search Process:

Till now various semantic search approaches have been published. Their application area and their realization are varied, though they have common set of ideas. So the search process included a manual keyword search on IEEE, ACM, Scopus, and Google Scholar. In the first stage we focused on the following keywords and keyword combinations: Semantic web; "{Information retrieval, Search Engine, intelligent information, search} and {prototype, model, methodology}.

For data providers, we disregarded all publications released prior to 2000. General papers (e.g. original publications on key topics) were not filtered by date. These initial searches yielded semantic search papers. Every publication was perused and summarized. Lastly, we extracted a sample set of relevant references from papers we identified as key publications.

We categorize the papers in four approaches based on functionality. Semantic Search approaches [13][25][17] can use any of the following approaches:

First Approach (Contextual Search): A contextual Approach is to disambiguate and to make queries to provide single meaning.[1][27][15]

Second Approach (Reasoning Search): The focus is on reasoning. It finds new information from the given facts.[7][10][22][23]

Third Approach (User Query Processing): Understanding of User query language. This approach's

effort is the goal of identifying the aim of people.[2][4][5][11]

Fourth approach (Ontology based search): The representation of knowledge uses ontology. The system uses the typed query by using ontology so that the search can be focused. [12][23][29][30]

Semantic search engines can mix more than one approach to fulfill different functions. There is room for a variety of search engine which means it does not fit into any type.

The filtered papers [16] are grouped into two categories:

1. Research papers that perform a search on the web.
2. Research papers that perform a search on the specific domain.

After studying the research papers, around 20 unique model/prototype are filtered. These prototypes functionalities are summarized and compared based on the classification parameters; this survey addresses the common perception, uniqueness and also open issues.

The rest of this paper is presented as follows. In section II, classification parameters framed by us to analyze the semantic search engine approaches. Section III represents overview of selected approaches with open issues. Section IV constitutes comparison of all prototype/systems based on the classification parameters. Section V comprises the scope for future research directions and finally concluded with conclusion.

II. CLASSIFICATION PARAMETERS

The architecture of conceptual-semantic search engine has components like the crawler, parser, ontology database, knowledge repository, inference engine, ranking algorithm and user interface. By considering this in mind the following parameters [14] are formulated to analyze the semantic search approaches.

1. Focus:

In this criterion, Search engines may works on information retrieval from the (Semantic) web or special purpose information systems.

2. Transparency

Regarding the user interface with semantic system features, Transparency types are as follows:

- *Transparent*: The semantic functions of the system are invisible to the user; the system appears to be an 'ordinary' search engine. Transparent systems have no means to request additional information from the user.
- *Interactive*: Interactive systems may ask the user for clarification or suggest changes to the query.
- *Hybrid*: Hybrid systems merge both interactive and transparent behavior. Normally, they act as transparent systems. If systems require user interactions means, it functions like interactive systems.

3. User Context:

The usefulness of retrieved documents always links to the user context. Many semantic search engines apply the user context to fetch the user's needed information.

- *Dynamic interaction***:** User context is extracted from user interaction dynamically. Based on the user's query and query-refinement history the system guesses about desired results. .
- *Predefined Question Category:* In this approach, queries are categorized in so-called *question-categories* that specify the user's information need. The system provides a fixed number of question-categories that are exploited during query evaluation.

4. System Design

Search engines can be designed in three possible ways they are:

- *Stand-alone search engine:* A stand-alone search engine peforms all functions by using single machine and it consists of components like the crawler, indexer and query engine etc..
- *Meta-search engine*: A meta-search engine does not have indexer,crawler and database. It distributes queries to other subordinate search-engines and combines the results, thereby provide the search result to the user.
- *Distributed Search engine:* Distributed search engine has other machines to carry out process like crawler, reasoner. It distributes all work to other machines in order to improve the scalability and performance.

5. Query Refinement:

The semantic modification of user queries is a well-known technique for information retrieval. In the area of semantic search, it often exploits information from ontologies. It plays a central role in many semantic search engines. Different techniques have been developed to increase both, recall and precision of a query. The increase of precision is often called *query disambiguation*.

The query transformation falls into three categories:

Manually: The simplest way to modify a query leaves the modification to the user. When the user enters a query, the system returns not only documents but also an appropriate part of ontology. The user navigates the ontology and reformulates his query, i.e., adds or removes query terms.

Query rewriting: Query rewriting is driven by the idea that a query can be optimized by the system. Three different ways, augmentation, trimming and term substitution are observed.

In the case of *augmentation*, the query is enhanced with terms that are derived from the ontological context of the original query terms, e.g., the query for 'Berners Lee' could be enhanced with 'Semantic web'.

Depending on the ontology structure (see next subsection) different semantics can be exploited.
The *trimming* of a query removes query-terms and has the opposite effect of augmentation.
Augmentation and trimming exploit that a query consisting of a *Conjunction* (AND) of terms becomes more specific with each additional term, where a query composed of a *Disjunction* (OR) becomes more general. In other words, related to the user's information need, long conjunctive queries yield high precision, where long disjunctive queries lead to the high recall.
Substitution is the process of search terms are replaced with ontologically related terms. In general, terms are substituted with synonyms, hypernyms or hyponyms from the ontology to increase recall or precision, respectively. Substitution may yield a result-set that only partially overlaps the original result set.
Graph-based: The third technique to optimize user queries requires tight coupling between the document base and the ontology. It perceives both, ontological concepts and documents as the nodes of a graph. Query terms are used to find relevant nodes in the graph. From these nodes, an algorithm traverses the graph to determine semantically related documents.

6. *Ontology structure*

Ontology-based semantic search engines rely on certain ontology structures. Ontologies are usually built from concepts, properties, constraints and possibly axioms. We observe that semantic search exploits properties only and distinguish the following cases:

- *Anonymous properties*: In the case of anonymous properties, the system disregards the name and the semantics of the property. The interrelation between two concepts indicates that they share the same context only.
- *Standard properties*: The properties are synonym_of, hypernym_of, meronym_of, instance_of and negation_of. The homonym_of property does not have to be modeled explicitly since it is equivalent to term equality. The usage of standard properties enhances semantic search capabilities. However, it also introduces dependencies on ontological structures.
- *Domain-specific properties*: Besides standard properties, a system can exploit domain-specific properties, as e.g., 'image type' in a image retrieval system.

  Ontology structure is an important criterion since it characterizes the flexibility of the search engines concerning the reuse of ontologies.

7. *Crawler*

A crawler is a program that visits Web sites and reads their pages and other information in order to create entries for indexing. The major search engines on the Web have such a program, which is also known as a "spider" or a "bot." Crawlers are programmed to visit sites that have been suggested by their proprietor as new or updated. Entire sites or specific pages can be selectively visited and indexed.

8. *Ranking Algorithm*

Search Engines use ranking algorithms to weigh different elements to determine which webpage is most appropriate to a search query. This criterion explains what type of ranking algorithm is used to arrange the result based on relevance.

9. *Reasoning Mechanism*

Reasoning mechanism allow deriving new information from existing concepts and roles that are not explored in the initial ontology. When solving a problem, one must understand the question, gather all significant facts, analyze the problem i.e. compare with previous problems (note similarities and differences), perhaps use pictures or formulas to solve the problem.
Main types of reasoning mechanism are:
Deductive Reasoning – A type of logic in which one goes from a general statement to a specific instance.
If the conclusion is not guaranteed (at least one instance in which the conclusion does not follow), the argument is said to be *invalid*.
Inductive Reasoning involves going from a series of specific cases to a general statement. The conclusion in an inductive argument is never guaranteed.

10. *Technologies used:*

The list of possible technologies may be used to develop the prototype of search engine are:
Crawler: Heritrix, MultiCrawler, BioCrawler,
Application: Apache Jena Fuseki
Semantic web Languages: RDFS, Quadruples, OWL, DAML+OIL
Database : Mysql, YARS2, db4OWL, Jena TDB,Jena SDB, Sesame,OWLLIM
Query Language: SPARQL ,SQL, RQL
Ranking and Reasoning: Apache Lucene, Protégé+HerMIT
Apart from the above, the researcher might develop their own application for doing the process.

11. *Result presentation*

How the results are presented is given in this category. Semantic search may return concept, ontology, snippets or document etc. these can be represented as text, URI with additional information like Label, comment and type etc.

12. *Open issues*

For each approach, the problems which has not yet been solved and scope for future enhancement or research are identified and specified. Issues might be related to functional (like ranking algorithms, reasoning mechanism used) or non functional (like performance, scalable and interoperable)

III. SUMMARY OF UNIQUE APPROACHES

The selected research prototype's design and its functionality are elucidated in this section. Semantic search engines are developed for the purpose of retrieving audios [9] and videos [27] also.

3.1 SHOE - Simple HTML Ontology Extension:

The architecture of SHOE [1] has following components:

Annotation:

After selecting proper ontology and using that ontology vocabulary add markup to the web pages is called as the annotation. The knowledge annotator tool is used to add SHOE knowledge to web pages. This tool has the interface to displays concepts, instances, relations and claims. The user can do editing operations on these objects.

Crawler:

Expose, a web crawler is used to search web pages with SHOE markup and store it in the knowledge repository. This crawler is traversed like a graph, where nodes are web pages and arcs are the hyperlink from those web pages. The Cost function is used for each URL where it should be placed in Queue. If unknown ontology is coming during traversal, it loads that ontology.

Knowledge Base:

Parka KB is used to store category and relation claims, as well as any new ontology information in the knowledge base (KB). Parka has the capability to answer queries on KBs with millions of assertions in seconds and provides better performance in parallel machines.

User Interface:

Parka Interface Queries: users have to draw a graph in which nodes represented constant or variable instances and arcs represented relations. To provide an answer to the query, subgraph matching on the user's graph is performed. Drawing the queries by the users is difficult and time-consuming also.

TSE Path analyzer: The user is allowed to sketch the feasible pathways of food product contamination. The user can specify the queries by selecting a few values from hierarchical lists. The results are shown in the form of a graph and detail of any node can be fetched by clicking on that node in the graph.

Issues:

Need of a general-purpose query tool that needs only minimum knowledge to use.

3.2 Inquirus 2:

It functions like a meta-search engine [2] (does not have a local database and relies on other search engines). The results returned from the other search engines are combined through combination policy and fusion policy. Ordering of results is done by fetching and analyzing individual pages and uses consistent scoring function, making the ordering problem more like that of a standard search engine.

In this architecture user preferences to the query is added. Rather than being limited solely to the use of keywords for expressing an information need, the user can provide an information need category that controls the search strategy used by the meta-search engine.

Each information need category has an associated list of sources, modification rules, and a scoring function.

Source selection:

A standard meta-search engine always uses the same source search engines: the source-selection process does not change. Meta-search engines such as SavvySearch, ProFusion, Inquirus, and MetaSEEK might not send all queries to the same search engines.

Some engines allow the user to select groups of search engines (such as "News" or "Sports"), or to select individual engines. Others attempt to map the keywords in the query to the best search engines.

Inquirus 2 does source selection based on user preferences. Preferences could be a set of sources, similar to other meta-search engines.

Query Modification:

To enhance the number of results relevant to a specific need, Inquirus 2 performs query modification. There are three types of query modification used:

- utilization of search engine-specific options
- Prepending terms to the query, or appending terms to the query.
- More than one modified query can be submitted for a given search engine.

Ranking Algorithm:

To incorporate multiple factors into the ordering policy, Inquirus 2 represents user preferences as an additive value function over any of the available metadata. There are two factors for each attribute: the relative weight and the attribute-value function (the mapping from the attribute's assignment to its value).

The best function for each category should be identified manually and attached with that attribute.

Open Issues:

This system did not allow the users to easily generate their own categories and automatic identification and implementation of document scoring functions.

3.3 TAP:

The Semantic Web application framework TAP [8] presented combines traditional information retrieval with semantic search. This system develops GetData Interface.

GetData is intended to be a simple query interface to network accessible data presented as directed labeled graphs. It is planned to be very easy to build, support and use, both from the perspective of data providers and data consumers. GetData is not intended to be a

complete or expressive query language a SQL, RQL or DQL.

To improve the semantic search, this system follows two steps:

1. Augment the list of documents with relevant data pulled out from the semantic web.
2. Understanding of the denotation of the search term helps to better filter and sort the list of documents retrieved.

The TAP Knowledge Base which contains about 65,000 instances of these classes is used as a lexicon to identify such searches.

To retrieve the information from the web, this system provides the wrapper for each web site (data source). GetData interface is provided with each of the site. Given the number of data sources and their distributed nature of its data sources, ABS makes extensive use of the registry and caching mechanisms provided by TAP. These entire data source together yield a Semantic Web with many millions of triples. TAP system function like meta-search engine design.

3.4 Hybrid Spread Activation [10]:

Spread activation techniques are used to find related concepts in the ontology given an initial set of concepts and corresponding initial activation values. These initial values are obtained from the results of classical search applied to the data associated with the concepts in the ontology.

The hybrid spread activation is the main part of the proposed system, occurring in the hybrid instances graph, where relations (links) have both a label that comes from the ontology definition, and a numerical weight, which comes from the weight mapping techniques. The spread activation algorithm works by exploring the concepts graph. Given an initial set of concepts, the algorithm obtains a set of closely related concepts by navigating through the linked concepts in the graph. Inferences occur naturally in this process, since the result set may contain nodes that are not directly linked to the initial set of nodes. The spread activation algorithm is domain dependent. The spread activation algorithm provides the path through which the node was obtained.

3.5 ISRA - Intelligent Semantic web Retrieval Agent [4]:

This has been developed using J2EE technologies. It uses traditional client server architecture (meta-search engine model). The client is a basic web browser, through which the user specifies search queries in natural language. The server contains Java application code and the WordNet database. The prototype also provides an interface to several search engines including Google(www.google.com),Alltheweb(www.alltheweb.com) and AltaVista. The prototype consists of three agents:

Input-Output-Parser Agent:

This is responsible for capturing the user's input, parsing the natural language query, and returning results. The agent uses "QTAG" to parse the user's input. It returns the part-of-speech for each word in the text. Based on the noun phrases (propositions) identified, an initial search query created.

WordNet Agent:

The WordNet Agent interfaces with the WordNet lexical database via JWordNet (a pure Java standalone object-oriented interface). For each noun phrase, the agent queries the database for different word senses and requests that the user select the most appropriate sense for the query. The agent extracts word senses, synonyms and hypernyms (superclasses) from the lexical database and forwards them to the query refinement agent to augment the initial query.

Query Refinement and Execution Agent:

The Query Refinement and Execution (QRE) agent expands the initial query based on word senses, and synonyms obtained from WordNet. The refined query is then submitted to the search engine using appropriate syntax and constraints, and the results returned to the user.

Issues:

Can improve the scalability and customizability of the approach, and minimize user interaction.

3.6 Librarian Agent:

Librarian agent system [5] behaves like a human librarian. This approach is based on incremental refinement of user's queries, according to the ambiguity of a query's interpretation.

The role of the Librarian Agent is

(i) to resolve the disambiguation of the queries posted by users (query management module)

(ii) to enable efficient ranking and/or clustering of retrieved answers (ranking module) and

(iii) To enable the changes in the knowledge repository regarding the users' information needs (collection management module).

Query Management module is responsible for the ambiguity measurement and for the recommendations for the refinements of a query.

This estimates the ambiguities of the initial query (so called Problem Discovery phase) in order to provide suitable modification of that query, which will decrease the number of irrelevant results or/and increase the number of relevant results (Query Refinement Phase)

Query processing involves three information sources:

- the ontology is used to determine the clarity or unambiguousness of a query
- the user's past queries help to guess the correct meaning of query terms

- the document-base is analyzed to predict the result-set size of augmented or trimmed queries

Ranking Module: It analyses the domain ontology, the underlying repository and the searching process in order to determine the relevance of the retrieved answers. Based on the relevance, the retrieved documents are displayed to the users.

Change Management module: This module is to make recommendations for the changes in the collection and in the underlying ontology. Any upgrading of knowledge repository, updating ontology is performed in this module.

3.7 *SCORE - Semantic Content Organization and Retrieval Engine*

It uses automatic classification and information-extraction techniques together with metadata and ontology information to enable contextual multi-domain searches that try to understand the exact user information need expressed in a keyword query.

The activities carried out in SCORE [3] are:

1. Defining WorldModel and Knowledgebase: Knowledge extraction agents manage the Knowledgebase by exploiting trusted knowledge sources. Different parts of the Knowledgebase can be populated from different sources. Various tools help detect ambiguities and identify synonyms. Commercial deployments of SCORE can be expedited with a predefined WorldModel and Knowledgebase.
2. Content processing: This includes classifying and extracting metadata from content. The results are organized according to the WorldModel definition and stored in the Metabase. Knowledge and content sources can be heterogeneous, internal or external to the enterprise, and accessible in push or pull modes.
3. Support for semantic applications: The semantic engine processes semantic queries, but does not currently support inference mechanisms found in AI or logic-based systems. Instead, it provides limited inference based on the traversal of relationships in the Knowledgebase. An API for building traditional and customized applications returns results as XML to facilitate GUI creation.

3.8 *TRUST – Text retrieval using semantic technologies:*

This is developed for semantic and multilingual search engine [11] capable for processing natural language questions in English, Polish, French, Portuguese and Italian.

The aim is to find sentence from a set of texts that answers questions in Natural Language. When the user framed a question, a list of pivot (key elements in that question) is displayed. For polysemous pivot, a short description of its sense is shown, the users can able select the sense that he finds adequate for his question or accept the one that suggested. The user may also choose between the local search (hard disk) and web search (hybrid both standalone and meta-search engine).

After the question is submitted, the search engines looks for text blocks containing candidate answers. The selected blocks are arranged based on proximity to the question and top ones are given to question/ answer evaluator. The most relevant text blocks are extracted. The answers are displayed arranged in descending order of their relevance.

Modules designed to implement search engine are:

Question Analysis:

Natural language questions were interpreted and transformed to Boolean query by eliminating stopping words like what, where and who. Question categories are defined to handle the questions like how, when and where. Pivot elements are extracted from the question. Pivot elements are key elements in the question that might be a word, expressions, number, date, phrase etc.

Indexing Process:

Indexation of each file was started by splitting into text blocks and each text block is parsed. For Each sentence the following information is collected:

- Relevant ontological and terminological domain found in the sentence.
- Any answer is found for defined question categories

Search Procedure:

The user is allowed to do hard disk search or web search. In case of local search, search is carried out in indexed file using search keywords along with synonyms pivot elements.

In case of web search, the question is given to meta-search engines along pivot elements. In both case results are on text blocks are given to question/ answer evaluator.

Question/ answer evaluator:

In this step, the evaluator analyzed the highest ranked text block by parsing each sentence and giving its final score to express its likelihood to answer the question. Sentences with minimum relevance are excluded. The best sentences are displayed by descendent order of their scores.

Issues:

- Improvement can be performed with word sense disambiguation, increase the use of lexical-semantic relations.
- Connection between word senses and ontology domain can be refined.
- Evaluator can be improved.

3.9 Audio data retrieval:

The audio retrieval proposed [9] is part of a special-purpose information system. It retrieves news items from a collection that is fed by broadcast audio-streams. The audio meta-data are extracted by speech

recognition and from plain-text content-descriptions that are supplied by the broadcast stations. The approach contains disjunctive query augmentation and term substitution based on domain ontology. The ontology consists of concepts, individuals and their synonyms, hypernyms and meronyms. The 'upper part' of the ontology is designed manually, while the lower-level concepts are extracted from the Yahoo hierarchy. There is no notion of user context.

3.10 Ontogator:

The key idea of the Ontogator system [6] is to combine the usage benefits of multi-facet search with the answer quality benefits of ontology-based search, together with semantic recommendations.
Ontogator uses two information sources:

1. *Domain Knowledge* consists of an ontology that defines the domain concepts and the individuals. In this paper, the domain ontology consists of some 329 promotion related concepts, such as "Person" and Building", 125 properties, and 2890 instances.
2. *Annotation Data* describes the metadata of the images represented in terms of the annotation and domain ontologies. Annotation ontology describes the metadata structure used in describing the images. It is assumed, that the subject of an image is described by associating the image with a set of RDF(S) resources of the domain knowledge, classes or instances. They occur in the image and hence characterize its content.

Based on the domain knowledge and the annotation data, Ontogator provided to the user with two services:
Multi-facet search The underlying domain ontologies are mapped into facets and facilitate multi-facet search. In our example case, there are six facets "Happenings", "Promotions", "Performances", "Persons and roles", "Physical objects", and "Places". The facets provide complementary views of the contents along different dimensions. The facets can be used for indexing the content and to help the user during information retrieval
*Recommendation system***:** After finding an image of interest by multi-facet search, the domain ontology model together with image annotation data can used to recommend the user to view other related images. The recommendations are based on the semantic relations between the selected image and other images in the repository.
The two services are connected with the information sources by tree sets of configurations or rules.
*Hierarchy rules***:** The heart of the multi-facet search engine is a set of category hierarchies by which the user expresses the queries. The hierarchy rules are a set of configurationally rules that tell how to construct the facet hierarchies from the domain ontologies.
Mapping rules: Annotations associate each image with a set of resources of the domain ontology. Mapping rules extend these direct annotations by describing the indirect relations between the images and the domain knowledge.
Mapping rules solve the problem by specifying the relations by which images are related with domain resources.
*Recommendation rules***:** The domain ontology defines not only the concepts and their hierarchical structure, but also the relations by which the actual domain classes and individuals are related with each other. Based on these relations, recommendation rules are used to find associations between an image and other images of potential interest to the user.

3.11 OWLIR - OWL Information Retrieval

OWLIR [7] is an implemented system for retrieval of documents that contain both free text and semantic markup in RDF, DAML+OIL or OWL. The framework support both retrieval-driven and inference-driven processing. Inference and retrieval should be tightly coupled; improvements in retrieval should lead to improvements in inference, while improvements in inference should lead to improvements in retrieval. The implemented system was built to solve a particular task – filtering University student event announcements.

User Interface:

A simple form-based query system allows a student to enter a query that includes both structured information (*e.g.,* event dates, types, *etc*.) and free text. The form generates a query document in the form of text annotated with DAML+OIL markup. The queries and event descriptions are processed to represent in triples, enriching the structured knowledge using local knowledge base and inference engine and swangling the triples into indexed terms. The result is a text-like query that can be used to retrieve a ranked list of events.

Ontology design:

OWLIR defines ontologies, encoded in DAML+OIL, allowing users to specify their interests in different events. These ontologies are also used to annotate the event announcements.

Text Extraction:

Event announcements in free text are converted into semantic markup by using AeroText™ system. This system extracts key phrases and elements from free text documents and translates the extraction results into a corresponding RDF triple model that uses DAML+OIL syntax. This is accomplished by binding the Event ontology directly to the linguistic knowledge base used during extraction.

Inference System:

OWLIR uses the metadata information added during text extraction to infer additional semantic relations. These relations are used to decide the scope of the search and to provide more relevant responses. OWLIR bases its reasoning functionality on the use of

DAMLJessKB. DAMLJessKB facilitates reading and interpreting DAML+OIL files, and allowing the user to reason over that information.

3.12. SWOOGLE

Swoogle [12] is designed as a system that automatically discovers SWD (Semantic Web Document) s indexes their metadata and answers queries about it.

The Swoogle architecture consists of web crawlers that discover SWDs; a metadata generator; a database that stores metadata about the discovered SWDs; a semantic relationships extractor; an N-Gram based indexing and retrieval engine; a simple user interface for querying the system; and agent/web service APIs to provide useful services.

Ontology Rank inspired by the Page Rank algorithm which ranks online documents based on hyperlinks. This algorithm uses graph that are formed by SWDs has a richer set of relations than the graph of the World Wide Web.

3.13. SWSE – Semantic Web search Engine

SWSE [22] operates over structured data and holds an entity-centric perspective on search: returns data representations of real-world entities. All process is performed in distributed systems.

Crawling: Process of retrieving the raw RDF documents from the Web is crawling. Crawler starts with a set of seed URIs, retrieves the content of URIs, parses and writes content to disk in the form of quads, and recursively extracts new URIs for crawling. Crawler supports only to traverse RDF/XML documents.

Consolidation:

The consolidation component tries to find synonymous (i.e., equivalent) identifiers in the data, and canonicalises the data according to the equivalences found. Owl: sameAs statements are extracted from the data and identify owl: sameAs triples and buffered them to a separate location.

Ranking:

The ranking component performs links-based analysis [14] over the crawled data and derives scores indicating the importance of individual elements in the data (the ranking component also considers URI redirections encountered by the crawler when performing the links-based analysis);

Reasoning:

The reasoning component materializes new data which is implied by the inherent semantics of the input data (the reasoning component also requires URI redirection information to evaluate the trustworthiness of sources of data);

Indexing:

The indexing component prepares an index which supports the information retrieval tasks required by the user interface.

The query-processing and user interface components service queries over the index documents and displays the result.

3.14. Falcons Concept search:

This is a keyword-based ontology search engine [21] which performs following tasks:

The multithreading *crawler* dereferences URIs with content negotiation (accepting only application/rdf+xml) and downloads RDF documents, which are then parsed by Jena. The URIs newly discovered in these documents are submitted to the *URI repository* for further crawling. Each RDF triple in an RDF document and the document URI form a quadruple and is stored in the *quadruple store* implemented based on the MySQL database.

The *Meta analysis* component periodically computes several kinds of global information and updates them to the *metadata* database, e.g., which kind of entity (class/property/individual) a URI identifies and which concepts ontology contains.

The *indexer* updates a *combined inverted index*, which serves the proposed mode of user interaction, i.e., keyword search with ontology restriction. The *ranking* process is implemented based on Lucene. At indexing time, a popularity score is computed and attached to each concept. At searching time, popularity of concepts and term-based similarity between virtual documents of concepts and the keyword query are combined to rank concepts.

For each concept returned, a query-relevant structured snippet is generated from the data in the quadruple store. For each concept requested, the *browsing concepts* component loads its RDF description from the quadruple store and presents it to the user.

3.15. Watson

Watson [23] is a Semantic Web search engine providing various functionalities not only to find and locate ontologies and semantic data online, but also to explore the content of these semantic documents.

Watson performs three main activities:

1. It collects available semantic content on the web.
2. It analyses it to extract useful metadata and indexes.
3. It implements efficient query facilities to access the data.

The crawling and tracking component uses Heritrix, the Internet Archive's Crawler to discover locations of existing semantic documents.

The Validation and Analysis component is then used to create a sophisticated system of indexes for the

discovered semantic documents, using the Apache Lucene indexing system.
Based on these indexes, a core API is deployed that provides all the functionalities to search explore and exploit the collected semantic documents.
Watson provides different 'perspectives', from the most simple keyword search, to complex, structured queries using SPARQL.

3.16. GoWeb:

It combines classical keyword-based Web search with text-mining and ontologies to navigate large results sets and facilitate question answering.
GoWeb [19] is an internet search engine based on ontological background knowledge. It helps to filter potentially long lists of search results according to the categories provided by the GeneOntology (GO) and the Medical Subject Headings (MeSH). It offers an efficient search and result set filtering mechanism, highlighting and semi-automatic question answering with the ontological background knowledge. The selection of top concepts includes the occurrence frequency, the hierarchy level and, if available, a global frequency from a pre-analyzed corpus.
The workflow for GoWeb can be described as follows. The user submits a query through the search form on the GoWeb website to the server. The server preprocesses the query and sends a search request to the search service. The search service returns the first results. The first results are then annotated, highlighted (concepts and keywords), rendered and sent to the user.

3. 17. WebOWL

This is a semantic web search engine [24] for OWL data. It was built on the principles of current search engines but instead of focusing on whole pages (or whole ontologies) it focuses on the actual entities within these ontologies. It uses a ranking algorithm that assigns different ranking power to classes and individuals. The system comprises all the key components of a search engine, which are a crawler, a database, a query mechanism and a ranking algorithm.

BioCrawler
In order to discover new ontologies and refresh the data of already stored ones, WebOWL uses BioCrawler to crawl the Web. BioCrawler is an intelligent crawler that learns to recognize and remember sites that contain ontologies or link to other sites containing ontologies, thereby forming a neighborhood of semantic content. The system consists of cooperating intelligent agents running on the Jade platform.
db4OWLse Database
The database of the WebOWL search engine is an enhanced version of db4OWL, a generic OWL database currently being developed by the authors and based on the db4o object database engine(db4Objects Inc.). The system uses Jena's parser and reasoner to import data. The OWL species and type of reasoning is set via a configuration file.
OWLRank
OWLRank is an algorithm developed specifically for WebOWL and is used to determine the ranking value of OWL objects. It was inspired by PageRank and uses a similar popularity measure to determine the importance of OWL classes and individuals. OWLRank measures semantic links between classes and individuals to determine their significance.
Web Front-End
The WebOWL search engine uses a web front end to allow users to formulate queries and navigate through the results. The front-end was primarily designed as a demo for the search engine's functionality; however, its development has revealed many challenges and unanswered questions in terms of the Semantic Web's usability and appeal to Internet users.

3.18. SIRM – Semantic Information Retrieval Model
This model was designed to develop a standard RDF format of linguistic information, [30] which includes declarative specifications of a machine readable lexicon that captures morphological, syntactic, and semantic aspects of the lexical items related to ontology.
This model consists of following modules:
RiscoLex:
The semiautomatic construction of a lexical database in Portuguese for the Financial Risk domain was proposed was called as RiscoLex. This database was created based on ontology of risk and its corresponding corpus. The construction of RiscoLex is to extract the labels of classes and properties of the ontology, identify and retrieve their respective synonyms and the morphosyntactic features of each term, convert them into RDF format, and provide the lexical database with the Lemon model.
Ontology and corpus:
Document descriptors for corpus which is a descriptor to describe document contents, and people interact using natural language to infer unexpressed meanings. Ontological entities represent concepts, and inference engines automatically infer non explicit information.
Functional usage:

The user interacts in a traditional way to submit the query. The query processing standardizes the terms for the search. The lexicon-ontological knowledge includes the ontology and the RiscoLex. The corpus characterizes the database containing the documents to be retrieved. The joint indexation of the databases involved provides the lexical-semantic index, which is used in the retrieval and ranking of retrieved documents to be presented to the user.

The semantic annotation process is therefore essential to link documents to the semantic space created by the domain ontology. NLP is the main tool for document identification, comparison, and annotation.

Issues:

Consideration of peculiarity and jargons of the domain may improve the vocabulary and also it contributes to improve the search results of lexical databases with semantic IRS.

The adoption of different weighting factors, other than the tf-idf, to address the lexical-semantic indexing would be highly useful.

The databases created by ontology lexicalization could be used as tools to improve automatic summarization or automatic text writing.

The participation of lexicographers, terminologists, and linguists in the building of lexical databases could greatly contribute to the interpretation and adequacy of linguistic phenomena to the ontology environment.

3.19. IBRI-CASONTO

This is a search engine [29] for College of Applied Sciences (CAS), Sultanate of Oman. The system is based on the RDF dataset as well as Ontological graph. This engine is developed for two languages Arabic and English.

User Interface:

It provides three parts of searching which are Keyword Searching, SPARQL Expert and CAS queries. CAS Queries includes a set of predefined queries based on our Arabic and English ontology. SPARQL Expert, which requires an expert of writing SPARQL Query because it forces the user to write a manual query. The Keyword Searching that retrieves the results based on the full-text matching of the query.

Indexing:

TDB indexing is built on the Fuseki for Jena TDB dataset is used. The indexing process in Lucene is used for keyword searching, consists of a chain of logical steps after gain access to the original content you need to search.

Searching:

Keyword-Based search:

It is done by the support of Apache Lucene, which provides with the access to the Lucene indexes. This type of searching get the matched keywords without understand the concept behind it.

Semantic-based search:

Semantic Searching of IBRI-CASONTO is supported by Apache Jena Fuseki. It provides a SPARQL server that can use the Jena TDB for persistent storage. In addition, it provides with the SPARQL protocols for query, update and rest update over the HTTP. Moreover, the SPARQL query offers the searching over the triple-store and retrieves the needed results.

Storage:

Jena TDB is used to store triples because it is a component of Jena for RDF storage and query. It supports the full range of Jena APIs. The MySQL as RDBMS is used for the keyword searching purpose.

Inference:

The inference is used to discover new relationships between the data that modeled as a set of defined relationships between the resources. It works as automatic procedures that deriving additional information by generating new relationships based on the ontology dataset. There are several automated reasoners, which can plug-in inside the ontology environment such as Protégé (Pallet, FaCt++, HerMiT, etc).

HerMiT is an open source that is already plug-in in protégé 4.3 and it is a perfect reasoner for ontologies, which is written in OWL. This reasoner is based on a novel ‘‘hypertableau” calculus that delivers efficient reasoning than any known algorithm.

3.20 IRSCSD - Information retrieval system for computer science domain

The main objective of this research is design and development of semantic web-based system [28] for integrating ontology towards domain-specific retrieval support like [26].

Methodology involves the following stages:

- First Stage involves the designing of framework for semantic web-based system.
- Second stage builds the prototype for the framework using Protégé tool.
- Third Stage deals with the natural language query conversion into SPARQL query language using Python-based QUEPY framework.
- Fourth Stage involves firing of converted SPARQL queries to the ontology through Apache’s Jena API to fetch the results.

IV. COMPARISON

Comparison of unique approaches that are developed till now are represented in table (Table: 1, 2, 3 and.4) based on classification parameters.

V. FUTURE RESEARCH DIRECTIONS IN SEMANTIC WEB SEARCHING

The realization of semantic search engine faces challenges in [22]:

- The system must scale to large amounts of data.
- The system must be robust in the face of heterogeneous, noisy, impudent, and possibly continuing data collected from a large number of sources.
- Providing reliable, most relevant information to the user without taking much time.

The semantic web information retrieval process consists of major tasks like ranking, query processing, reasoning and indexing. The scope for future enhancements/research in each task is given separately.

Ranking:

Ranking in Web search engines depends on many factors, ranging from globally computed ranks to query-dependent ranks to location, preferences, and history of the searcher. Same traditional search engine ranking approaches are used by many researchers. Only a few performs ontology-based document ranking in semantic search. Apart from these approaches, scope for doing research in the ranking algorithm:

- Inclusion of some additional signals into the ranking procedure is area for further research, especially in the face of complex database-like queries and results beyond the simple list of objects.
- The question of finding appropriate mathematical representation in current ranking of RDF graphs leads to new ranking algorithms.
- Mathematically calculated weights can be assigned for entities, links in RDF graph, thereby devising new weightage scheme for ranking algorithm which might lead to most relevant results on top.

Indexing:

The main directions for future work in indexing would be to identify an intersection of queries for which optimized indexes can be built in a scalable manner, and queries which offer greater potential to the UI.

Investigation of compression techniques and other low-level optimizations may further increase the base performance.

Explore new methods for increment update so that the system is constantly building the new index while the old is being queried against;

Query Processing:

With respect to current query-processing capabilities, our underlying index structures have proven scalable. Open research question here is how to optimize for top-k querying in queries involving joins; joins at large-scale can potentially lead to the access of large-volumes of intermediary data, used to compute a final small results size; thus, the question is how top-k query processing can be used to immediately retrieve the best results for joins.

Extending the query processing to handle more complex queries is a topic of importance when considering extension and improvement of the current spartan UI.

In order to fully realize the potential benefits of querying over structured data, we need to be able to perform optimized query processing.

Reasoning:

To investigate some backward-chaining (runtime) approaches which complement a partial materialization strategy.

Combination of ranking into the reasoning process can be considered in order to improve the precision.

The general scopes of semantic search that always need future research are [25] [16]:

1. *Identify the objective of user*

This plays a considerable role in the intelligent semantic search engine. Still there is scope for identifying the intention of the user, by seeing the keywords used and previous search keywords history use some mining techniques.

2. *Individual user patterns can be extrapolated to the global users.*

In search engines that presented disambiguation to search terms, a user could type in a search word that was ambiguous (e.g., Java) and search engine would return a list of options (programming language, coffee, island in the South Seas), thereby search engine can be interactive.

3. *Inaccurate queries.*

The users use typically domain specific knowledge and they don't include all the potential synonyms and the variations in search query. Users have a problem but they aren't sure how to phrase their query.

4. *User Friendly UI*
5. *Providing Reliable data*

The above are considered as improvements in user context and user interface design for the semantic search engine.

Main functional requirements need to be fulfilled and improved in the semantic search engine are:

High recall and High precision:

Consistently A few intelligent semantic search engines are unable to show their significant performance in improving the precision and lowering the recall.

Adaptability:

Many systems require a certain ontology structure, i.e., they rely on custom-tailored ontologies. It can cope with arbitrary ontologies but provide weaker semantic capabilities. It is an open problem how systems may adapt themselves to *existing* ontologies, i.e., ontologies that have been designed with a different purpose. This is not only important concerning the reuse of ontologies but also as regards the interoperability between knowledge-based systems in general. The system adaptability is an important step towards domain-independent semantic search engines.

Performance/scalability. We only found few work on the performance of systems. On the market, semantic search engines have to compete with standard search engines. They may introduce only a little overhead compared to standard solutions. Consequently, they need efficient implementation regarding indexing time, index space and response time.

VI. CONCLUSION

In this work, classification parameter is introduced for comparing semantic search engines approaches. With regard to the classification scheme, common ideas, their advantages and drawbacks are explained. Furthermore, research and application-development issues are identified that are not addressed by current systems. From this survey, there is a large number of promising approaches to semantic document retrieval can be learnt.

REFERENCES

[1]. Heflin, Jeff, and James Hendler. "Searching the Web with SHOE." *AAAI-2000 Workshop on AI for Web Search*. 2000.

[2]. Glover, Eric J., et al. "Web Search---Your Way." *Communications of the ACM* 44.12 (2001): 97-102.

[3]. Sheth, Amit, et al. "Managing semantic content for the Web." *IEEE Internet Computing* 6.4 (2002): 80-87.

[4]. Burton-Jones, Andrew, et al. "A heuristic-based methodology for semantic augmentation of user queries on the web." *Conceptual Modeling-ER 2003* (2003): 476-489.

[5]. Stojanovic, Nenad. "On the role of the Librarian Agent in ontology-based knowledge management systems." *J. UCS*9.7 (2003): 697-718.

[6]. Hyvönen, Eero, Samppa Saarela, and Kim Viljanen. "Ontogator: combining view-and ontology-based search with semantic browsing." *information retrieval* 16 (2003): 17.

[7]. Mayfield, James, and Tim Finin. "Information retrieval on the Semantic Web: Integrating inference and retrieval." *Proceedings of the SIGIR Workshop on the Semantic Web*. 2003.

[8]. Guha, Ramanathan, and Rob McCool. "TAP: a Semantic Web platform." *Computer Networks* 42.5 (2003): 557-577.

[9]. Khan, Latifur, Dennis McLeod, and Eduard Hovy. "Retrieval effectiveness of an ontology-based model for information selection." *The VLDB Journal—The International Journal on Very Large Data Bases* 13.1 (2004): 71-85.

[10]. Rocha, Cristiano, Daniel Schwabe, and Marcus Poggi Aragao. "A hybrid approach for searching in the semantic web." *Proceedings of the 13th international conference on World Wide Web*. ACM, 2004.

[11]. Amaral, Carlos, et al. "Design and Implementation of a Semantic Search Engine for Portuguese." *LREC*. 2004.

[12]. Finin, Tim, et al. "Swoogle: Searching for knowledge on the Semantic Web." *Proceedings of the National Conference on Artificial Intelligence*. Vol. 20. No. 4. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[13]. Esmaili, Kyumars Sheykh, and Hassan Abolhassani. "A categorization scheme for semantic web search engines." *Computer Systems and Applications, 2006. IEEE International Conference on.*. IEEE, 2006.

[14]. Li, Yufei, Yuan Wang, and Xiaotao Huang. "A relation-based search engine in semantic web." *IEEE Transactions on Knowledge and Data Engineering* 19.2 (2007).

[15]. Tummarello, Giovanni, Renaud Delbru, and Eyal Oren. "Sindice. com: Weaving the open linked data." *The Semantic Web*. Springer, Berlin, Heidelberg, 2007. 552-565.

[16]. Mangold, Christoph. "A survey and classification of semantic search approaches." *International Journal of Metadata, Semantics and Ontologies* 2.1 (2007): 23-34.

[17]. Dong, Hai, Farookh Khadeer Hussain, and Elizabeth Chang. "A survey in semantic search technologies." *Digital Ecosystems and Technologies, 2008. DEST 2008. 2nd IEEE International Conference on*. IEEE, 2008.

[18]. Lamberti, Fabrizio, Andrea Sanna, and Claudio Demartini. "A relation-based page rank algorithm for semantic web search engines." *IEEE Transactions on Knowledge and Data Engineering* 21.1 (2009): 123-136.

[19]. Dietze, Heiko, and Michael Schroeder. "GoWeb: a semantic search engine for the life science web." *BMC bioinformatics*10.10 (2009): S7.

[20]. Barbieri, Davide, et al. "Deductive and inductive stream reasoning for semantic social media analytics." *IEEE Intelligent Systems* 25.6 (2010): 32-41.

[21]. Qu, Yuzhong, and Gong Cheng. "Falcons concept search: A practical search engine for web ontologies." *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 41.4 (2011): 810-816.

[22]. Hogan, Aidan, et al. "Searching and browsing linked data with swse: The semantic web search engine." *Web semantics: science, services and agents on the world wide web* 9.4 (2011): 365-401.

[23]. d'Aquin, Mathieu, and Enrico Motta. "Watson, more than a semantic web search engine." *Semantic Web* 2.1 (2011): 55-63.

[24]. Batzios, Alexandros, and Pericles A. Mitkas. "WebOWL: A Semantic Web search engine development experiment." *Expert Systems with Applications* 39.5 (2012): 5052-5060.

[25]. Sudeepthi, G., G. Anuradha, and M. Surendra Prasad Babu. "A survey on semantic web search engine." *IJCSI International Journal of Computer Science Issues* 9.2 (2012): 241-245.

[26]. Kamath, S. Sowmya, et al. "A semantic search engine for answering domain specific user queries." *Communications and Signal Processing (ICCSP), 2013 International Conference on*. IEEE, 2013.

[27]. Farhadi, Babak, and M. B. Ghaznavi-Ghoushchi. "Creating a novel semantic video search engine through enrichment textual and temporal features of subtitled YouTube media fragments." *Computer and Knowledge Engineering (ICCKE), 2013 3th International eConference on*. IEEE, 2013.

[28]. Bansal, Ritika, and Sonal Chawla. "Design and development of semantic web-based system for computer science domain-specific information retrieval." *Perspectives in Science* 8 (2016): 330-333.

[29]. Sayed, Awny, and Amal Al Muqrishi. "IBRI-CASONTO: Ontology-based semantic search engine." *Egyptian Informatics Journal* (2017).

[30]. Schiessl, Marcelo, and Marisa BRÄSCHER. "Ontology lexicalization: Relationship between content and meaning in the context of Information Retrieval." *Transinformação* 29.1 (2017): 57-72.

TABLE 1
COMPARISON OF UNIQUE APPROACHES BASED ON CLASSIFICATION PARAMETERS

Prototype /Project	SHOE	Inquirus2	TAP	Hybrid Spread Activation	ISRA
By	Jeff Heflin and James Hendler	Eric J. Glover et al	R. Guha and Rob McCool	Cristiano Rocha	Andrew Burton-Jones,
Focus	WWW	WWW	WWW	WWW	WWW
System Design	Stand-alone	Meta Search Engine	Metasearch model	Stand-alone	Meta search Engine
User context	Used TSE Path analyser as UI	*Predefined Question Category*	Dynamic interaction and *Predefined Question Category*	Not done	Dynamic interaction and *Predefined Question Category*
Transparency	Transparent	Transparent	Hybrid	Transparent	Interactive
Query Refinement	Manual	Prepending and appending related terms with the query		Instance graph based	Carried using query refinement agent which uses Wordnet
Ontology structure	Hypernym		Anonymous	Domain specific	Hypernym Synonym
Crawler	Expose Web Crawler	Not applicable	GetData interface+ABS	Traditional search engine method	None
Ranking Algorithm	No ranking mechanism	Ordering Policy Uses additive function for each category like topical relevance	TAP search interface	-	Relevant results from different engines are combined
Reasoning Mechanism	Parka KB	Combination policy	TAP knowledge base	New Inferences are find out while traversing the instance graphs	Nil
Result Presentation	Graphs with URL	URL + Text	Graphs, Nodes , URI, description	Nodes , documents (path that travel in instance graphs)	URLs and 'snippets' provided from the web pages
Technologies used	Knowledge annotator, Expose, Parka KB,PIQ, TSE Path analyzer		TAP Knowledge base TAPache	J2EE, Lucene	J2EE, WordNet
Open Issues	Need of a general-purpose query tool that needs only minimum knowledge to use	did not allow the users to easily generate their own categories and automatic identification and implementation of document scoring functions.	Need to create generalized framework. Methodology need to be developed to understand the search term	No weight mapping formulas. Evaluation scheme can be devised.	Can improve the scalability and customizability of the approach, and minimize user interaction.

TABLE 2
COMPARISON OF UNIQUE APPROACHES BASED ON CLASSIFICATION PARAMETERS

Prototype /Project	Librarian Agent	SCORE	TRUST	Audio Data Retrieval	Ontogator
By	Nenad Stojanovic	Amit P. Sheth	Carlos Amarol	Latifur Khan	Eero Hyv¨onen
Focus	Information system Library Portal	Information system	WWW, local hard disk search	Audio data of football sports	IS for image retrieval
System Design	Standalone	Stand-alone	Hybrid	Stand-alone	Stand-alone
User context	*Interactive*	None	Interactive	None	Content based browser
Transparency	Transparent	Transparent	Transparent	Transparent	Transparent
Query Refinement	Conj. Augumentation Query Refinement	Manual	Natural Language Processing	Disjoint Augment substitution	Substitution
Ontology structure	-	Domain specific	Ontology concepts are not used indepth	Domain dependent ontology for football game	Domain specific Annotation ontology
Crawler	-	Extractor Toolkit Main memory indexing		Meta data created for broadcast audio stream	
Ranking Algorithm	Based on relevance	-	Score given based on relevance	Vector space model, TF and IDF calculation	Multi-facet search
Reasoning Mechanism		Inferences through relationships	Lemma, inflection, Parsing the text to find for relevant for the user question	Axioms in ontology used as inference	Implemented using mapping rules recommendation rules and hierarchy rules
Result Presentation	Documents +ontology	XML	Text blocks	Audio data	Images +descriptions of images. description of resource can be viewed Topic maps
Technologies used		-	Not given	Not specified	SWI Prolog, Protégé, RDQL
Open Issues	No dedicated ranking algorithm		No user context queries, so filtering the search is difficult. Instances used in test are limited, which cannot reveal the real feasibility.	No user context queries, Instances used in test are limited, which cannot reveal the real feasibility.	

TABLE 3
COMPARISON OF UNIQUE APPROACHES BASED ON CLASSIFICATION PARAMETERS

Prototype /Project	OWLIR	Swoogle	SWSE	Falcons	Watson
By	James Mayfield and Tim Finin	Tim Finin, et al	Aidan Hogan et al	Yuzhong Qu Gong Cheng	Mathieu d'Aquin and Enrico Motta
Focus	WWW	WWW	WWW	WWW	WWW
System Design	Meta-search engine model	Distributed	Distributed	Stand-alone	Stand-alone
User context	*Form based query interface*	*Predefined Question Category*	Separate UI to handle all type of queries	None	Watson API
Transparency	Transparent	Transparent	Transparent	Interactive	Transparent
Query Refinement	Manual	Manual	Manual	Ontology selection	-
Ontology structure	Event Ontology	Generic Ontology	OWL rule based ontology	Hypernym	Web Ontology
Crawler	AeroText used for text extraction	Google crawler Focused Crawler Swoogle Crawler	Separate crawler algorithm is used		Internet Archive Crawler
Ranking Algorithm	-	Ontology Rank	Link based Ranking	Concept Ranking	-
Reasoning Mechanism	DAMLJessKB		Authoritative reasoning		Description logic
Result Presentation	Documents, semantic markup	URIs, Literals, type Classes, Relationships, ontologies used etc	Entity snippets, description.	Query Relevant structure snippet, URI, Label, type RDF description, Ontology metadata	URI, Labels, comment
Technologies used	WONDIR, DAMLJessKB, AeroText	Jena, MySQL	Berkely DB, Distributed Architecture	Jena, MySQL, Apache Lucene	Heritrix, Apache Lucene Jena
Open Issues		More concerned with more traditional document-search over ontologies.		improve the method of snippet generation in order to better present ontology structures.	do not include components for consolidation or reasoning. Instead it focus on providing APIs to external services

TABLE 4
COMPARISON OF UNIQUE APPROACHES BASED ON CLASSIFICATION PARAMETERS

Prototype /Project	GoWeb	Webowl	SIRM	IBRI-CASONTO	IRSCSD
By	Heiko Dietze and Michael Schroeder	Alexandros Batzios and Pericles A. Mitkas	Marcelo SCHIESSL Marisa BRÄSCHER	Awny Sayed Amal Al Muqrishi	Ritika Bansal, Sonal Chawla
Focus	WWW	WWW Tested Pets Ontology	WWW	IS for College of Applied Sciences	IS
System Design	Meta	Stand-alone	Stand-alone	Stand-alone	Stand-alone
User context	Dynamic interaction and *Predefined Question Category*	*Predefined Question Category Through Web UI*		Interactive, specialized interface	Dynamic Interaction
Transparency	Hybrid	Transparent	Transparent	Transparent	Interactive
Query Refinement	-	Query By Example	Graph-based	Ontological Graphs	Natural Language Query into SPARQL using tool
Ontology structure	GO,MeSH	Domain specific	Lexicon model for ontologies, OntoRisk - Domain specific	Domain specific	Domain specific
Crawler	-	BioCrawler	Labels, synonyms are fetched from ontology convert into rdf stored in db	-	-
Ranking Algorithm	Filter based part-of and is-a relationships of ontology and relevance	OWLRank	Solr – term weighting is based on tf-idf algorithm	Default	Default
Reasoning Mechanism	OWL	Jena Reasoner Forward chain method	Lexical semantic indexing OntoRisco – axioms	HerMit	HerMit
Result Presentation	Snippets, abstract form	Classes, URI	documents	URI, Entities	Entities
Technologies used	BioCreAtIvE	Jade, Jena db4o object database engine MySQL	Protégé , Python+NLTK, Apache Jena Fuseki, RDFLib	Protégé+Hermit Apache Lucene Jena TDB MySQL	Protégé, Python +QUEPY framework Apache's Jena Fuseki
Open Issues			The implementation of different weighting factors, other than the tf-idf, to address the lexical-semantic indexing would be highly useful.		Till now not implemented for real-time data .