Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

INS

Information Systems

INformation Systems

An analysis of search-based user interaction on the Semantic Web

M. Hildebrand, J.R. van Ossenbruggen, L. Hardman

REPORT INS-E0706 MAY 2007

Centrum voor Wiskunde en Informatica (CWI) is the national research institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organisation for Scientific Research (NWO).
CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

ISSN 1386-3681

An analysis of search-based user interaction on the Semantic Web

ABSTRACT
Many Semantic Web applications provide access to their resources through text-based search queries, using explicit semantics to improve the search results. This paper provides an analysis of the current state of the art in semantic search, based on 35 existing systems. We identify different types of semantic search features that are used during query construction, the core search process, the presentation of the search results and user feedback on query and results. For each of these, we consider the functionality that the system provides and how this is made available through the user interface.

1998 ACM Computing Classification System: H.3.3; H.5.2; H.5.4
Keywords and Phrases: Semantic search; user interaction; analysis; evaluation
Note: This research was supported by the MultimediaN project funded through the Bsik programme of the Dutch Government and by the European Commission under contract FP6-027026, Knowledge Space of semantic inference for automatic annotation and retrieval of multimedia content --- K-Space.

An Analysis of Search-based User Interaction on the Semantic Web

Michiel Hildebrand, Jacco van Ossenbruggen, and Lynda Hardman*

CWI, Amsterdam, The Netherlands
firstname.lastname@cwi.nl

Abstract. Many Semantic Web applications provide access to their resources through text-based search queries, using explicit semantics to improve the search results. This paper provides an analysis of the current state of the art in semantic search, based on 35 existing systems. We identify different types of semantic search features that are used during query construction, the core search process, the presentation of the search results and user feedback on query and results. For each of these, we consider the functionality that the system provides and how this is made available through the user interface.

1 Introduction

Search based on relatively simple textual queries is an often found and frequently used feature on the Web and in many other information systems. It is generally regarded as a simple entry point into the system, after which the user can optionally engage in other, potentially more complex, interaction styles. Not surprisingly, it is also a feature found in many Semantic Web applications.

We have developed the impression that search based on simple textual queries is often regarded as a trivial aspect of a Semantic Web application — a feature that is uninteresting from a scientific perspective and draws attention away from the "real" semantic issues. If regarded as a problem at all, it is typically considered a problem to be solved by someone else, e.g. by researchers in the field of Information Retrieval or Human Computer Interaction. The topic has received relatively little attention in Semantic Web literature. At the same time, Semantic Web applications such as award-winning Flink[1] and W3C's RDF representation of WordNet[2] would obviously benefit from a simple text-based search interface, but lack this feature completely. Is this perhaps because text-based search is not that easy to provide after all? Another reason to believe this feature is underrated is that the systems that do support text-based search do this in very different ways.

This paper moves beyond anecdotal evidence to obtain a more systematic understanding of the different design dimensions that play a role in supporting

* Lynda Hardman is also affiliated with the Technical University of Eindhoven.
[1] `http://flink.semanticweb.org/`
[2] `http://www.w3.org/2006/03/wn/wn20/`

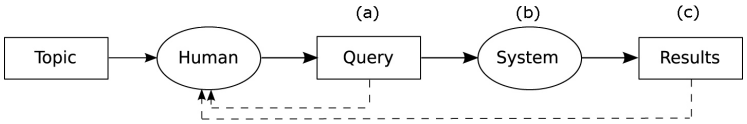

Fig. 1. High level overview of text-based query search: (a) query construction, (b) search algorithm of the system, (c) presentation of the results. Dashed lines represent user feedback. Figure adapted from [2].

text-based search on Semantic Web data. Excluding search based on structured query languages such as SPARQL [1], we focus on queries based on simple textual entry forms and "query by navigation". The next section establishes the basic search terminology used throughout the paper. Section 3 provides an in-depth analysis based on a survey of more than 30 Semantic Web applications. We argue that search is far from trivial, and list the different roles semantics currently play throughout the various phases of the search process. We discuss the main open problems in Section 4 and summarize our main conclusions in section 5.

2 Basic Search Terminology

We introduce the basic definitions used throughout the paper.

Search process — We follow the TRECVID 2007 Evaluation Guidelines [2] and divide the search process into three different phases: query construction, execution of the core search algorithm and presentation and organization of the results. We also take into account user feedback on the query and on the results (see also Figure 1).

Semantic search — We use the term semantic search when semantics are used during any of the phases in the search process.

Type of results — Traditional search engines on the web typically assume the result of a search to be a set of web resources, e.g. text documents, images or video. This also holds for some Semantic Web applications. Others, however, return sets of matching URIs, sets of matching triples (an RDF sub-graph) or a combination of these. Often the behavior of a system depends on the type of result it assumes, so we make this explicit wherever this is relevant.

Overview pages and **surrogates** — We refer to presentation of the (k first) results as the *overview page*, which typically represent results using *surrogates*. For example, resulting HTML pages can be represented by their title and a text snippet, while image or video results are often represented by thumbnails.

Local view page — Surrogates typically provides links to the full presentation of the result they represent. The latter is either the full presentation of an information resource (e.g. the full HTML page that is linked from the snippet presented on the overview page), some human readable representation of metadata associated with the result, or a human readable representation of a

resulting subgraph. Following [3], we refer to the latter presentation as a *local view page*.

Syntactic matching — Syntactic search matches the query against the textual content of the resources (if applicable), the literals in the RDF metadata, the URIs in the system, or a combination of these.

Semantic matching — We use the term semantic matching for those algorithms that in addition to syntactic matching also use the graph structure of the RDF metadata and/or its semantics to find results.

3 Analysis of semantic search

We systematically scanned all proceedings of the International and European Semantic Web Conference series as well as the Journal of Web Semantics, to compile a list of end-user applications described or referred to. For each system we collected basic characteristics such as the intended purpose, intended users, the scope, the triple store and the technique or software that is used for literal indexing. The resulting document was made available online[3] and announced on three public mailing-lists[4]. Additionally, we sent personal emails to the authors of papers and developers of all included systems. This resulted in an updated version of the online document. This update was based on 15 email threads, in which additional information was provided for 11 systems and 6 additional systems within the scope of the survey were recommended, giving a total of 35 systems.

Based on the data resulting from the survey we perform a more thorough analysis of the three individual phases in the search process of Figure 1. For each of these we consider the underlying functionality and the features of the corresponding user interface. The results of this analysis are summarized in Table 1, and the table also provides the structure of the remainder of this section. We discuss query construction in section 3.1, the search algorithms in 3.2 and the presentation of the results in 3.3. Note that the examples and references merely serve as illustrations, the full analysis with references is available in the online survey.

3.1 Query Construction.

The search process starts with the user constructing a query that reflects his or her information needs. We describe the functionality for this process as provided by the systems in the survey and how this functionality is supported at the interface.

Functionality — We focus on systems that support free text queries. Constructing a query in free text requires little knowledge of the system and the data structure. The price users have to pay is ambiguity: words can have multiple meanings (lexical ambiguity) and a complex expression can have multiple

[3] `http://www.webscience.org/swuiwiki?title=Semantic_Search_Survey`

[4] `public-xg-mmsem@w3.org`, `semantic-web@w3.org`, `public-semweb-ui@w3.org`

Feature:	**Functionality**	**Interface Components**
	Query construction	
Free text input:	Keywords, natural language	Single text entry, property-specific fields
Operators:	Boolean constructs, syntactic disambiguation, semantic constraints on input/output	Application-specific syntax
Controlled terms:	Disambiguate input, restrict output, select predefined queries	Value lists, faceted browser, graph
User feedback:	Pre-query disambiguation	Semantic autocompletion
	Search algorithm	
Syntactic matching:	Exact, prefix or substring match, minimal edit distance, stemming	Not applicable
Semantic matching:	graph traversal, query expansion, spread activation, RDFS/OWL reasoning	Not applicable
	Result presentation	
Data selection:	Selected property values, class-specific template, display vocabularies	Visualized by text, graph, tagcloud, map, timeline, calendar
Ordering:	Content and link structure based ranking	Ordered list
Organization:	Clustering by property, by result path or dynamic	Tree, nested box structure, clustermap
User feedback:	Post-query disambiguation, recommendation of related resources	Facets, tagcloud, value list

Table 1. Functionality and interface support in the three phases of semantic search.

underlying structures (structural ambiguity). Ambiguous input often leads to irrelevant search results. To reduce ambiguity several systems allow additional query constructs beyond free text input: structural and semantic operators and controlled terms. We describe free text input and the additional query constructs and the role of user feedback in the process of matching free text with controlled terms.

Free text input is supported in existing systems in three ways. First, full text search allows the user to find all resources with matching textual content or metadata. In many semantic search engines full text search is the main entry point into the system [4–8]. Second, free text input can be restricted to match a value of a specific property. In faceted browsers this is the case when searching for a value within a particular facet [9–11]. Finally, systems such as Aqualog [12] and Ginseng [13] support free text input in the form of natural language expressions.

Syntactic operators explicitly define the interpretation of complex search terms. Well known examples are the boolean operators AND and OR. Several

applications employ third party search libraries such as Apache Lucene, which typically provide an extensive collection of syntactic control structures[5].

Semantic operators add explicit meaning to a query. In SemSearch, for example, the user specifies to which RDFS or OWL class a result should belong. The authors illustrate this with the example `article:motta` for which the system retrieves all resources that are of type `article` and match the search term `motta` [14].

Controlled terms provide the use of predefined concepts. In QuizRDF [15] the user selects an RDF class to determine the type of the search term. Other systems provide autocompletion to support users with keyboard-based input of controlled terms [9–11, 16, 17]. A different approach is seen in DBin [18] and Haystack [19], which allow the user to select predefined queries.

User feedback on the input is useful when there are multiple controlled terms that match with the free text input. Several systems allow the user to select the intended term before it is processed by the search algorithm [6, 11, 17]. This form of user feedback allows pre-query disambiguation. In contrast, post-query disambiguation is performed on the results of the search algorithm.

Interface — The basic interface components to enter or construct a query are text entry boxes and value selection lists. These components are used in various designs, of which we mention three. If applicable, we describe the link from the interface components to the underlying data structure. Furthermore, we describe the interface aspects of user feedback on the input and several proposals for more advanced query construction.

A single text entry field is sufficient for free text input, e.g. Google and several systems that we analyzed [5–7, 15, 20]. Additional features included by some systems are selectable result types [20] and options for the search algorithm or presentation of the results [15].

Property-specific search fields support query construction guided by a specific set of possible search values [16, 21–23]. The value sets are typically defined by the range of the corresponding RDF property.

Faceted browsing allows the user to constrain the set of results within a particular facet. Typically, facets are directly mapped to properties in RDF. Alternatively, the mapping is made by projection rules. The advantage of an indirect mapping is that this allows the developer to define facets that match the user's needs while keeping the data structure unchanged [24]. Faceted browsing is applied to Semantic Web data by [9–11, 24–27] as well as by the company Siderean in the Seamark Navigator[6].

User Feedback is typically provided after the query has been entered, or dynamically during the construction of the query as a form of *semantic autocompletion*. The former method is used in Squiggle [6] and MuseumFinland [26] where the disambiguation of the matching query terms is presented after submitting the query. In semantic autocompletion the system suggests controlled terms with a label prefix that matches the text typed in already. Hyvönen and

[5] `http://lucene.apache.org/java/docs/queryparsersyntax.html`
[6] `http://www.siderean.com/`

others describe the idea of semantic autocompletion and several implementations in [17]. In faceted interfaces autocompletion is often used within a single facet [9–11, 16].

In general there is a clear need for simple interfaces, such as the single text entry field. On the other hand, interface designs that support more complex interaction styles potentially give the user more control, which is useful for the formulation of more precise information needs.

3.2 Search algorithm

All text-based search involves some form of syntactic matching of the query against textual content and/or metadata, an aspect well covered in Information Retrieval [28]. Semantic matching can extend syntactic matching by exploiting the typed links in the semantic graph. Before we describe semantic matching we briefly describe syntactic matching, focusing on the indexing functionality and the support that is already provided by the low level software on which various systems are built.

Syntactic matching — All systems in our study index the textual data in their collection for performance reasons. Which textual data is indexed, e.g. the content, the metadata or the URIs, is important for the search functionality of the system. In an ontology search engine such as Swoogle, users might want to search on URIs [4]. In annotated image collections the metadata forms the primary source for indexing [5, 6, 26]. For images that occur in web pages the contextual text provide an alternative source [6, 29]. Indices can be based on the complete word or on a stemmed version. Some interface functionalities require additional features. Autocompletion interfaces, for example, require efficient support for prefix matching. Some triple stores provide built-in support for literal indexing, for example, OpenLink Virtuoso[7] and SWI Prolog's Semantic Web library[8]. Alternatively, a search engine, such as Lucene[9], can be used together with a triple store.

Semantic matching — After syntactic matching, the structure and formal semantics of the metadata can be used to extend, constrain or modify the result set. Note that in a connected RDF graph, any two nodes are connected by a path in this graph. Naive approaches to semantic search are computationally too expensive and increase the number of results dramatically. Systems thus need to find a way to reduce the search space and to determine which semantically related resources are really relevant.

Inspired by the semantic continuum described by Ushold [30], we distinguish three levels of semantic matching: graph traversal, explicit use of thesauri relations and inferencing based on the formal semantics of RDF, RDFS and OWL.

Graph traversal takes only the structure of the graph into account. Several techniques are in use to constrain graph search algorithms. In Tap, constraints

[7] `http://www.openlinksw.com/`
[8] `http://www.swi-prolog.org/packages/semweb.html`
[9] `http://lucene.apache.org/`

define which relations to traverse for the instances of a particular class [7]. Alternatively, a weighted graph search algorithm may constrain the possible path structures and path length. Such an algorithm requires the assignment of weights to the edges in the graph, where the weights reflect the importance of the corresponding RDF relations. In e-Culture, weights are manually assigned to RDF relations [5]. SemRank automatically computes weights based on statistics derived from the graph structure [31]. Spread activation [32] is another computationally attractive technique for graph traversal, which can incorporate weights as well as the number of incoming links.

Thesaurus relations are sometimes used for query expansion. With the acceptance of SKOS [33] as a standard representation for thesauri, semantic matching with hierarchical broader term (BT) and narrower term (NT) and the associative related term (RT) can be implemented in a generic way. The Squiggle framework is an example in which this is done [6]. Facet browsers typically rely on hierarchical thesauri relations to restrict their result sets [11, 26]. Within the FACET project the integration of thesauri in the search process is studied extensively. This resulted in a demonstrator as well as a proposal for a semantic expansion service [34], which in turn formed the basis for the experimental SKOS API[10].

RDFS/OWL reasoning can also influence the search results. Several systems support RDFS subsumption once an RDF class is selected in the interface [14, 18, 35, 36]. In Dose, specialization and generalization over the subclass hierarchy is used dynamically according to the number of search results [37]. Some systems support partial OWL reasoning, and process, for example, only the OWL identity relations. In Flink [38] and SWSE [8] these are extensively used to model the identity between extracted entities.

3.3 Presentation of Results

We describe how explicit semantics are used to extend the baseline functionality in the presentation of search results, and the techniques that are used to visualize the results in the interface. As a baseline we consider the presentation of search results by popular search engines such as Google: the selected information for presentation is the URI or label of the result, surrogates of the content (e.g. text snippets or image thumbnails) and, optionally, additional information such as the file size. The results are typically organized in a plain list and ordered by relevance. We describe, for each aspect, how additional semantics are used to extend this baseline.

Functionality — We consider three aspects of the presentation: *selecting* what data to present, *organizing* the results and *ordering* the results. In addition, we discuss the function of user feedback on the results.

Selecting what to present — This issue is tightly bound to the question what the search engine considers to be a "result". If the result is a Web page or other information resource, traditional surrogates are typically used. When the search result is a set of URIs referring to nodes in an RDF graph, or a set of RDF triples,

[10] `http://www.w3.org/2001/sw/Europe/reports/thes/skosapi.html`

systems need to invent new ways to represent the results in their overview page. In most systems we studied, the decision on what (meta)data is used for the surrogates is hardwired into the system. QuizRDF supports template definitions for each RDF class [15]. Dbin [18] create templates for specific user tasks and domains. Display vocabularies such as Fresnel [39], as used by Longwell [9], provide full control over what data to select for presentation and how to present it.

Organizing the results — Semantics can also play a role in grouping semantically similar results together in the presentation, a feature commonly referred to as *clustering*. Assuming that users are interested in the results of only one cluster, clustering can also be considered as a form of post-query disambiguation. In our study we found several forms of clustering. In many systems, the values of a particular property are used to group the result set on common characteristics within a particular dimension. In [7] results with similar types are clustered together. In faceted browsers similar behavior is found, systems described in [9, 11, 26] all support clustering on the values of a particular facet. Noadster uses concept lattices to determine dynamically which properties to use for a given result set [3]. In e-Culture [5], the RDF path between the literal that is syntactically matching the query and the result may span more than one property. Clustering the results on these paths illustrate the interpretations of the query.

Ordering of results — The order of the search results can determined with different techniques. Ranking of results based on relevance is a well covered topic in Information Retrieval [28]. Numerous algorithms have been developed, evaluated and applied in successful applications. Term frequency-inverse document frequency (tf-idf) is an often used syntactic measure to determine the importance of a word based on the number of occurrences in a document relative to the number of occurrences in the entire collection. Many systems in our study use Lucene, which provides ranking based on tf-idf. In addition to textual content, the link structure is another source for ranking. Swoogle uses a variant of PageRank [40] to measure the relevance of RDF documents. PageRank was adapted to compensate for different types of relations that link RDF documents and terms [4]. In SWSE [41] a variant of PageRank based on the principle of focussed subgraphs [42] is used.

User Feedback — In our study, we did not encounter typical IR user feedback where the matching and ranking algorithms is influenced by the user's feedback. We mainly encountered user feedback to disambiguate, specialize, generalize or expand the result set. Most systems support expansion of a query by adding a keyword or by selecting a value from a property field or facet. In several systems, post-query disambiguation of free text input is supported through the selection of an RDF type [8, 15, 36, 43]. Alternatively, queries can be specialized or generalized with concepts from narrower or broader thesaurus relations [6, 11, 26]. An unwanted side effect of query refinement is the risk of ending up with no results. This can be avoided by restricting the user beforehand to use only those terms that lead to results. This is one of the principles behind faceted browsing interfaces [44].

We observed that domain-specific applications use the semantics to organize the search results into clusters. Domain-independent search engines typically rely on ranking techniques for effective presentation of the search results.

Interface — Most systems provide a straight-forward interface that directly reflects the structure of the selected data and how it is organized. Typical examples include numbered lists for a linearly ranked set of results or visual grouping of clustered results in nested box layout structures. Since RDF is represented as a graph, visualizing the data as a graph may seem a straightforward choice. However, from a user interface perspective, "'big fat graphs" quickly become unmanageable [45], with only a few exceptions including the visualization of social networks between small groups of people [38]. Other visualization techniques (see [46] for a more extensive overview) we encountered include:

- *Tagclouds* that indicate the importance of textual metadata with variations in the font size. OpenAcademia [21] visualizes the concepts related to research publications and search. DBpedia.org [43] presents the available RDF types of the search results.
- *Clustermaps* that visualize the overlap between classes of instances, without needing an explicit concept representing this overlap [47]. For example in AutoFocus a clustermap visualizes the results of individual constraints as well as result sets that satisfy multiple constraints [25].
- *Data type-specific* visualizations are used in several systems to present space and time on a geographical map, timeline or calendar. The Simile timeline[11], several map visualization tools and Google Calendar can be used through publicly available APIs. Hence, we do not list the individual systems that make use of these.
- *Local view pages* provide a detailed presentation of the metadata associated with single URI. Systems such as Tabulator [48] and Disco [49] are based on the notion of a *concise bounded description*or CBD[12] and present the statements where the current focus URI is a subject. Others systems' local view pages may contain all statements in which the URI occurs either as a subject or object. Sesame's URI explorer pages[13], Noadster [3] and E-culture [5] also include statements where the URI plays the role of the property.

4 Discussion

In the previous section we summarized the wide range of semantic search features encountered in the three phases of the search process as realized by the (primarily research prototype) systems included in our study. A next step for the community could be to systematically include the usage of semantics in each of the three phases of every semantic search engine. This, however, does not guarantee that the resulting system would actually improve search results for the end-user. In

[11] `http://simile.mit.edu/timeline/`
[12] `http://www.w3.org/Submission/CBD/`
[13] `http://www.openrdf.org/`

this section, we take a step back and discuss to what extent it is feasible to isolate the individual processes that depend on semantics and to work towards the development of mature, off-the-shelf components that support specific aspects of the semantic search process. In this way, developers are not faced with having to build a single gargantuan system, but can select mature components depending on their users' needs.

Lack of evaluation of semantic search algorithms —

With a few notable exceptions [20, 50], the search algorithms analyzed in this study are not or only briefly evaluated on the quality of their search results for end-users. Note that, for example in Information Retrieval, there is a long tradition of evaluating the quality of retrieval systems. Conference series such as TREC and INEX contribute to an (evolving) community consensus about which dimensions to evaluate, and how to measure a system's performance on that dimension. It is safe to say that within the Semantic Web community, we have not yet developed a similar consensus about the use of explicit semantics to improve search, and how to evaluate and to compare semantic search systems.

A tradition of evaluation, however, seems to be maturing in other areas of Semantic Web research. Examples include the Ontology Alignment Evaluation Initiative (OAEI) for evaluation of alignment methods, the EON workshops for evaluation of ontologies, and the workshops on Scalable Semantic Web Knowledge Base Systems (SSWS) for benchmarking the scalability of (SPARQL) triple stores. Several systems in our study already use off-the-shelf components to implement their triple store (most notably Sesame) and text retrieval component (most notably Lucene), and we believe it is no coincidence that these components implement precisely those aspects of the application for which mature benchmarks and measurements are available.

Starting an evaluation track on semantic search is, however, far from trivial. Even assuming that there is sufficient critical mass in our community to do so, precisely because current systems vary so widely, agreeing on what to measure will be difficult. Whatever the outcome of the discussion, we will need a publicly available, representative test corpus. Many of the systems in our study, however, have been tested with either unrealistic "toy" data sets or with real world data that is not publicly available. Comparing the performance of systems using Semantic Web technology against systems using other IR approaches would require a corpus that can be used fairly by all systems, or a corpus that is available in various formats. For example, one could compare the performance of state of the art IR systems on the XML version of Wikipedia[14] against that of Semantic Web systems on an RDF version[15].

Lack of user evaluation of the interfaces — A similar argument applies to the interface components found in our study. For none of the systems we could find proper user evaluations that would stand the criteria commonly found in the HCI community. Again, setting up user evaluations for Semantic Web ap-

[14] INEX'07 is using Wikipedia as an XML test corpus, see `http://inex.is.informatik.uni-duisburg.de/2007/`.

[15] E.g. `http://dbpedia.org/`.

plications is, in general, not an easy task. The need for a baseline to compare the new user interfaces is a problem here too. In many situations, Semantic Web applications provide new functionality that is not available in the current system of the user, or the new application provides comparable functionality, but to a (semantically) integrated data set that is currently distributed over several isolated applications. In both situations, fair comparison of the user interface of the new application with the old is almost impossible. User interface testing in a situation where it is unclear whether the underlying semantic application provides any real additional value to the user at all, makes the problem only worse. To start a tradition of user testing and evaluation of Semantic Web applications[16], one could start with more basic studies to test the effect of similar but different user interfaces on the same Semantic Web application, deferring the more difficult question as to whether a particular (semantically enriched) user interface of a particular Semantic Web application is better than the interface of its non-semantic counterpart.

Lack of APIs and middleware support — Many of the features discussed are implemented by the application layer with low level support from the triple store. For example, the semantic autocompletion discussed relies on giving suggestions while the user types. This requires extremely fast look-ups to find the set of suggestions based on literal prefix matching. Graph-based search algorithms are also notoriously computationally expensive and require low level support that is not provided by every triple store. This makes it hard to distribute search components independently of the triple store or text retrieval engine used. Note that even if middleware tools do provide the required extra functionality, the lack of common APIs is still a show stopper if the functionality discussed requires more than the standard (SPARQL-based) APIs provide.

5 Conclusions

We analyzed the state of the art in text-based query search as currently implemented in Semantic Web applications. We have identified the various roles that semantics play in query construction, the core search algorithm and presentation of search results. Our study shows that many systems already support aspects of semantic search. For some, it is even the main entry point into the underlying data. The quality of the search functionality and its user interface thus has a significant impact on the overall usability of these applications. Working towards generic libraries and publicly available services that support the individual processes will help developers to incorporate more semantic search fucntionality into applications. To improve the uptake of Semantic Web applications by end users, we feel our community should strive to make this research area more mature and start to develop methods for objective and systematic evaluation of the functionality and interface aspects of end-user applications.

[16] The SWUI community is trying to establish this, see `http://swui.semanticweb.org/`

Acknowledgments

We like to thank Željko Obrenović, Raphaël Troncy, Lloyd Rutledge and Alia Amin for their feedback on the paper. Lora Aroyo for her comments on the setup of the public survey. Sören Auer, Andreas Harth, Eyal Oren, Kyumars Sheykh Esmaili, Honghan Wu, Kotis Kostas, David Huynh, John Davies, Giovanni Tummarello, Eero Hyvoönen, Kingsley Idehen, Georgi Kobilarov, Bonino Dario, Eero Hyvönen, Vanessa Lopez and Irene Celino for their contribution to the survey. This research was supported by the MultimediaN project funded through the BSIK programme of the Dutch Government and by the European Commission under contract FP6-027026, Knowledge Space of semantic inference for automatic annotation and retrieval of multimedia content — K-Space.

References

1. W3C: SPARQL Query Language for RDF. W3C Candidate Recommendations are available at http://www.w3.org/TR (2006) Edited by Eric Prud'hommeaux and Andy Seaborne.
2. TRECVID organizers: Guidelines for the TRECVID 2007 Evaluation. http://www-nlpir.nist.gov/projects/tv2007/tv2007.html (2007)
3. Rutledge, L., van Ossenbruggen, J., Hardman, L.: Making RDF Presentable – Integrated Global and Local Semantic Web Browsing. In: The Fourteenth International World Wide Web Conference, Chiba, Japan, IW3C2, ACM Press (2005) 199–206
4. Ding, L., Pan, R., Finin, T., Joshi, A., Peng, Y., Kolari, P.: Finding and Ranking Knowledge on the Semantic Web. In Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A., eds.: 4th International Semantic Web Conference, ISWC 2005. Lecture Notes in Computer Science, Galway, Ireland, Springer (2005) 156–170
5. Schreiber, G., Amin, A., van Assem, M., de Boer, V., Hardman, L., Hildebrand, M., Hollink, L., Huang, Z., van Kersen, J., de Niet, M., Omelayenjko, B., van Ossenbruggen, J., Siebes, R., Taekema, J., Wielemaker, J., Wielinga, B.: MultimediaN E-Culture Demonstrator. In: The Semantic Web - ISWC 2006. (2006) 951–958
6. Celino, I., Valle, E.D., Cerizza, D., Turati, A.: Squiggle: a Semantic Search Engine for indexing and retrieval of multimedia content. In: Proceedings of the First International Workshop on Semantic-enhanced Multimedia Presentation Systems (SEMPS 2006), Athens, Greece (2006) 20–34
7. Guha, R., McCool, R., Miller, E.: Semantic Search. In: Proceedings of the 12th international conference on World Wide Web, Budapest, Hungary (2003) 700–709
8. DERI: SWSE. http://www.swse.org/ (2005-2007)
9. SIMILE: Longwell RDF Browser. http://simile.mit.edu/longwell/ (2003-2005)
10. m.c. schraefel, Smith, D.A., Owens, A., Russell, A., Harris, C., Wilson, M.L.: The evolving mSpace platform: leveraging the Semantic Web on the Trail of the Memex. In: Proceedings of Hypertext 2005, Salzburg (2005) 174–183
11. Hildebrand, M., van Ossenbruggen, J., Hardman, L.: /facet: A Browser for Heterogeneous Semantic Web Repositories. In: The Semantic Web - ISWC 2006. (2006) 272–285

12. Lopez, V., Pasin, M., Motta, E.: AquaLog: An Ontology-portable Question Answering System for the Semantic Web. In: Proceedings of the European Semantic Web Conference ESWC 2005, Creete, Grece (2005) 546–562
13. Bernstein, A., Kaufmann, E., Kaiser, C., Kiefer, C.: Ginseng: A Guided Input Natural Language Search Engine for Querying Ontologies. In: Jena User Conference, Bristol, UK. (2006)
14. Lei, Y., Uren, V., Motta, E.: SemSearch: A Search Engine for the Semantic Web. In: 15th International Conference on Knowledge Engineering and Knowledge Management Managing Knowledge in a World of Networks (EKAW 2006), Podebrady, Czech Republic (2006)
15. Davies, J., Weeks, R.: QuizRDF: Search Technology for the Semantic Web. In: Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 4 - Volume 4. (2004)
16. Kiryakov, A., Popov, B., Terziev, I., Manov, D., Ognyanoff, D.: Semantic Annotation, Indexing, and Retrieval. Journal of Web Semantics **2**(1) (2004) 49–79
17. Hyvönen, E., Mäkelä, E.: Semantic Autocompletion. In: Proceedings of the first Asia Semantic Web Conference (ASWC 2006), Beijing (2006)
18. Tummarello, G., Morbidoni, C., Nucci, M.: Enabling Semantic Web communities with DBin: an overview. In: Proceedings of the Fifth International Semantic Web Conference ISWC 2006, Athens, GA, USA (2006)
19. Quan, D., Karger, D.R.: How to Make a Semantic Web Browser. In: The Thirteenth International World Wide Web Conference, New York City, IW3C2, ACM Press (2004)
20. Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R.S., Peng, Y., Reddivari, P., Doshi, V.C., , Sachs, J.: Swoogle: A Search and Metadata Engine for the Semantic Web. In: Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management, Washington, D.C., USA (2004) 652–659
21. Mika, P.: OpenAcademia. http://www.openacademia.org/ (2006)
22. Heflin, J., Hendler, J.: Searching the Web with SHOE. In: Artificial Intelligence for Web Search. Papers from the AAAI Workshop, Menlo Park, CA (2000) 35–40
23. Metaweb: Freebase. http://www.freebase.com/ (2007)
24. Suominen, O., Viljanen, K., Hyvönen, E.: User-centric Faceted Search for Semantic Portals. In: Proceedings of the European Semantic Web Conference ESWC 2007, Innsbruck, Austria (2007)
25. Fluit, C., Sabou, M., van Harmelen, F.: Supporting User Tasks through Visualisation of Light-weight Ontologies. In Staab, S., Studer, R., eds.: Handbook on Ontologies in Information Systems. Springer-Verlag (2003) 415–434
26. Hyvönen, E., Junnila, M., Kettula, S., Mäkelä, E., Saarela, S., Salminen, M., Syreeni, A., Valo, A., Viljanen, K.: MuseumFinland — Finnish museums on the semantic web. Journal of Web Semantics **3**(2-3) (2005) 224–241
27. Oren, E., Delbru, R., Decker, S.: Extending Faceted Navigation for RDF Data. In: 5th International Semantic Web Conference, Athens, GA, USA (2006) 559–572
28. Baeza-Yates, R.A., Ribeiro-Neto, B.A.: Modern Information Retrieval. ACM Press / Addison-Wesley (1999)
29. Group, X.: Falcon-S. http://www.falcons.com.cn/ (2005-2006)
30. Uschold, M.: Where are the semantics in the semantic web? AI Magazine **24**(3) (2003) 25–36
31. Anyanwu, K., Maduko, A., Sheth, A.: SemRank: Ranking Complex Relationship Search Results on the Semantic Web. In: Proceedings of the 14th international conference on World Wide Web, Chiba, Japn, ACM Press (2005) 117–127

32. Rocha, C., Schwabe, D., de Aragao, M.: A hybrid approach for searching in the semantic web. In: Proceedings of the 13th International World Wide Web Conference, New York, NY, USA (2004) 374–383
33. W3C: SKOS Core Guide, W3C Working Draft. W3C Candidate Recommendations are available at http://www.w3.org/TR (2005) Edited by Alistair Miles and Dan Brickley.
34. Binding, C., Tudhope, D.: KOS at your Service: Programmatic Access to Knowledge Organisation Systems. Journal of Digital Information **4**(4) (2004)
35. Auer, S., Dietzold, S., Riechert, T.: OntoWiki -A Tool for Social, Semantic Collaboration. In: Proceedings of the Fifth International Semantic Web Conference ISWC 2006, Athens, GA, USA (2006)
36. Duke, A., Glover, T., Davies, J.: Squirrel: An Advanced Semantic Search and Browse Facility. In: Proceedings of the European Semantic Web Conference ESWC 2007, Innsbruck, Austria (2007)
37. Bonino, D., Corno, F., Farinetti, L., Bosca, A.: Ontology Driven Semantic Search. WSEAS Transaction on Information Science and Application **1**(6) (2004) 1597–1605
38. Mika, P.: Flink: Semantic Web Technology for the Extraction and Analysis of Social Networks. Journal of Web Semantics **3**(2-3) (2005) 211–223
39. Bizer, C., Lee, R., Pietriga, E.: Fresnel — A Browser-Independent Presentation Vocabulary for RDF. In: Proceedings of the Second International Workshop on Interaction Design and the Semantic Web, Galway, Ireland (2005)
40. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project (1998)
41. Hogan, A., Harth, A., Decker, S.: ReConRank: A Scalable Ranking Method for Semantic Web Data with Context. In: 2nd Workshop on Scalable Semantic Web Knowledge Base Systems. (2006)
42. Kleinberg, J.M.: Authoritative sources in a hyperlinked environment. Journal of the ACM **46**(5) (1999) 604–632
43. Berlin, F.U.: Search DBpedia.org. http://dbpedia.org/search/ (2007)
44. Yee, K.P., Swearingen, K., Li, K., Hearst, M.: Faceted Metadata for Image Search and Browsing. In: CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems, Ft. Lauderdale, Florida, USA, ACM Press (2003) 401–408
45. m.c. schraefel, Karger, D.: The Pathetic Fallacy of RDF. In: The 3rd International Semantic Web User Interaction Workshop (SWUI06). (2006)
46. Geroimenko, V., Chen, C.: Visualizing the Semantic Web — XML-based Internet and Information Visualization. Springer-Verlag (2003)
47. Fluit, C., Sabou, M., van Harmelen, F.: Ontology-based Information Visualization. In: Visualizing the Semantic Web: XML-based Internet and Information Visualization. Springer-Verlag (2003) 36–48
48. Berners-Lee, T., Chen, Y., Chilton, L., Connolly, D., Dhanaraj, R., Hollenbach, J., Lerer, A., Sheets, D.: Tabulator: Exploring and Analyzing linked data on the Semantic Web. In: The 3rd International Semantic Web User Interaction Workshop (SWUI06), Athens, Georgia, USA (2006)
49. Bizer, C., Gauß, T.: Disco - Hyperdata Browser. http://sites.wiwiss.fu-berlin.de/suhl/bizer/ng4j/disco/ (2007)
50. Sure, Y., Iosif, V.: First results of a Semantic Web Technologies Evaluation. In: Proceedings of the Common Industry Program, University of California, Irvine (2000)