Exploiting User Queries and Web Communities in Semantic Annotation

Norberto Fernández-García, José M. Blázquez-del-Toro,
Luis Sánchez-Fernández, Vicente Luque

Telematic Engineering Department.
Carlos III University of Madrid.
{berto,jmb,luiss,vlc}@it.uc3m.es

Abstract. In order to make current Web resources understandable by computers and make possible the Semantic Web vision, we need to add semantic metadata to such Web resources. In this paper we describe the SQAPS system, which aims at providing a mean of exploiting for semantic annotation the effort of users who every day look for information on the Web. We also describe how we can take benefit of the information generated and maintained by Web Communities as Wikipedia in order to achieve our goal.

1 Introduction

In order to make current Web resources *understandable* by computers and make possible the Semantic Web vision [1], we need to add *semantic metadata* to such Web resources, we need to *semantically annotate* them.

As the current Web has a huge number of heterogeneous resources, semantic annotation can be seen as one of the core challenges for building the Semantic Web [2]. Semantic annotation has been an active area of research for several years, and different approaches to this problem can be found in the literature. But, from our point of view, no one of these approaches takes advantage of the effort of the millions of users who every day look for information on the Web.

So, in [3] we introduced the SQAPS, *Semantic Query-based Annotation, P2P Sharing*, system. The main idea behind this system is to exploit keyword-based user queries in semantic annotation of Web resources. Instead of annotating directly Web resources, as most of current systems in the state of the art suggest, we proposed a system in which users annotate their queries. The keywords in these *semantic* or *annotated* queries are sent to classical Web search engines, obtaining a vector of URLs as a result. While browsing these results, a user can say if a certain resource is relevant to his query. If so, we can associate the Web resource with the semantic query, giving implicitly a certain semantics to the Web resource. Taking this into account, SQAPS annotations can be defined as associations between an annotated query and a Web resource. These associations are represented in RDF [4] and formally described using a lightweight RDFS [5] ontology [6]. In order to share these annotations with other SQAPS users a

peer-to-peer (P2P) Distributed Hash Table (DHT) infrastructure [7] is being used.

One of the main drawbacks of the system described in [3] is that the knowledge used to annotate the queries comes from a static source: WordNet [8]. The evolution of knowledge with time was left outside of the scope of that work.

In this paper, we want to consider a more dynamic approach. Instead of using a static lexicon, we want to explore the possibility of using as knowledge source the evolutive information produced and maintained by a Web community. We will show how information from Wikipedia community [9] can be exploited in the context of SQAPS system.

The rest of this paper is organized as follows: next section introduces the SQAPS system in order to make the paper self-contained. It describes briefly the SQAPS architecture (2.1) and working model (2.2). Section 3 explains how Web Communities, and particulary Wikipedia Community, can be used in our benefit in the context of SQAPS. Section 4 briefly describes some related works from the world of semantic annotation. Section 5 introduces, with discussion purposes, a set of issues related with SQAPS system. Concluding remarks in section 6 finalize this paper.

2 The SQAPS system

In this section, we briefly introduce the SQAPS system. This description has been included in order to make the paper self-contained. More information about the system, in particular, a detailed description of SQAPS ontology can be found in [3].

2.1 System Architecture

Figure 1 shows the intended architecture of the SQAPS system. The main components of our system are:

Query Analysis Its main purpose is to allow the annotation of a keyword-based query by the user. In order to do so, the Query Analysis component divides the query into candidate terms, each of which consisting of a word or sequence of words of the query, which can represent at least a unit of meaning. In order to associate terms and meanings, and decide what are the pairs [term,meaning] interesting for the user purposes, the Query Analysis component uses the information of a Semantic Source, and asks to the user about his interests.

Semantic Source Intended to provide concrete meanings to terms. In our context, a Semantic Source should provide at least a list of terms, and for each term, a set of possible meanings with their human-readable descriptions. A unique identifier for each pair [term, meaning] should also be provided.

Knowledge Repository It stores the semantic annotations defined by the user, and also a part of the annotations that other users have defined and

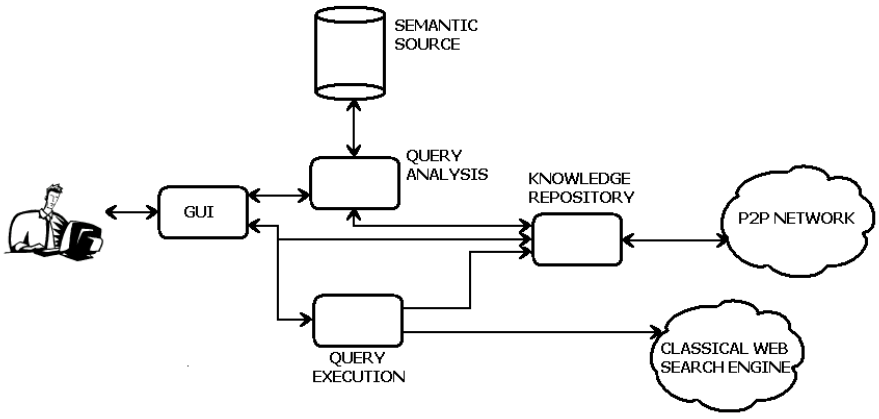

Fig. 1. SQAPS System Architecture

shared (a part of the DHT). This Knowledge Repository can be accessed by the local user, but also by other peers in the SQAPS P2P network. It also can access remote repositories of other peers.

Query Execution Receives as input the annotated user query and looks for relevant resources in the Knowledge Repository. If results are found, these are shown to the user, which can decide to look for more information or not. If no results are found, or the user requires more information, this module takes the keywords from the annotated query and sends such keywords to a classical Web search engine. The results are shown to the user, who can annotate a resource by clicking on a button. By doing so, the user states that the resource is relevant for the annotated query and a pair [URL, RDF document representing the semantic query], is inserted into the Knowledge Repository and into the annotation sharing P2P network.

P2P Network Main functionality of SQAPS system is semantic annotation creation and sharing. For this purpose, the basic functionality of the P2P network will be to allow annotation sharing. In our system, annotations are associations of URLs representing Web resources and RDF documents representing the semantic annotations of such resources. With these requirements, we have decided to rely on Distributed Hash Table (DHT) P2P networks, which offer good performance both in scalability and response time [10]. The hash of the URL of the resource being annotated would be used to decide which peer/s should store the annotation, and later, to retrieve the annotations related with a certain resource.

2.2 Working Model

In order to clarify the purposes and operation model of all the SQAPS architecture components, we provide a basic example of the intended annotation process

of our application. Figure 2 shows the main steps in this annotation process. These are:

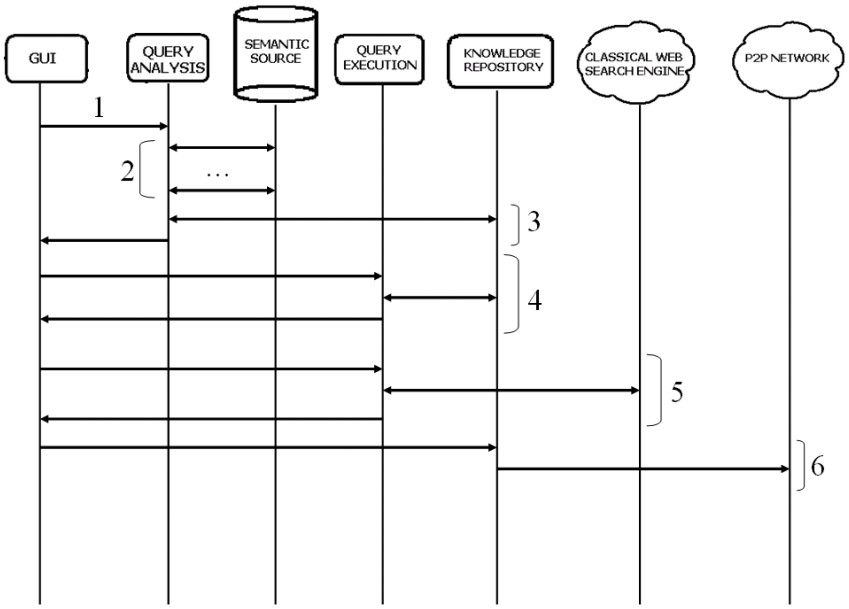

Fig. 2. SQAPS Annotation Process

1. The process starts with a user writing a keyword-based query. For instance, let us assume that the user is looking for information about Caml programming language, and he types as a query *Caml programming language*. This textual user query is sent to the Query Analysis block, which divides such query into candidate terms. For instance, the candidate terms originated in the analysis of our example query are *Caml*, *programming*, *language*, *Caml programming*, *Caml language*, *programming language* and *Caml programming language*. Sections of query between quotes are treated as a single term and not divided. As some studies suggest, typical user queries are not too long (only a 25% with three or more words) [11], so we expect that the number of terms will not be so big.
2. Once the system has the candidate terms, it looks into the Semantic Source to obtain the possible meanings associated to such terms. For instance, if we are using as Semantic Source WordNet 1.7 lexicon, as was initially our case, terms *Caml, Caml programming, Caml language* and *Caml programming lan-*

guage are not included, whereas terms *programming* (2 senses), *language* (6 senses) and *programming language* (1 sense) are included.

3. The list of possible meanings of a term can be ordered taking into account the Knowledge Repository information, which contains annotations added by the user in the past. The idea is to make the decision of user easier, including as first list items those which are expected to be more relevant for user's interests. For instance, if we look into WordNet 1.7, we can find two senses for the term *programming*:
   - Setting an order and time for planned events (scheduling).
   - Creating a sequence of instructions to enable the computer to do something (computer programming).

   If we assume that the user is a computer scientist that usually looks for information about programming languages, it is possible that he had used the term *programming* with the second sense in past queries/annotations. In such a case, this pair [term, sense] should already be in the Knowledge Repository. Then, the senses of the term *programming* would be ordered so that the first one to be shown to the user would be the one most related to the expected user's interests. Once the list of possible meanings of each term is sorted, the system displays to the user the possible interpretations of the found terms.

   Given the list of possible terms and meanings, the user selects the combination that better reflects his intention, annotating the query. Several possibilities may arise here:
   - None of the terms appears in the Semantic Source, or the senses included there are not good to reflect user intentions. In such a case, the user can simply decide not to annotate the query. This unannotated query is sent to the Query Execution component, which simply forwards the query to a classical Web search engine and returns the results. As in this case we can not generate an annotation, we have not included this situation in figure 2.
   - Part of the terms are found, and the other ones are not found, or the meanings in the Semantic Source do no reflect user intentions. Then the result will be a partially annotated query. A problem which might arise in this situation, is that it could give origin to alternative query interpretations. For instance, let's assume that the user has typed the query *Caml Light programming language*. The term *programming language* is found in the Semantic Source and the user annotates it, but he tells nothing about the other query components. Then the problem comes from the fact that the system can interpret the rest of the query in two different ways: as having two different terms, *Caml* and *Light*, or as having only one, *Caml Light*. At the moment the approach we are following is to use the information as provided by the user: if he has typed *Caml* and *Light* as different words, they are interpreted as different terms, and if he has typed *"Caml Light"* a single term will be generated. Other approaches,

like for example analyzing the textual contents of the resource being annotated, in order to find which of the interpretations is the most frequent in such contents, might be explored in the future. Another aspect which should be noted here, is that, though some of the terms of the query are annotated and some others not, we will store all the terms in the final annotation. The reason for such decision is that we expect that human users can view annotations of resources, and we think that removing unannotated terms could produce a loss of context information, which could be useful for such users.

- All the terms are found and annotated by the user.

In the last two cases, the result will be an annotated query. Let us assume that this is our situation, and the user has generated an annotated query composed by terms *Caml* (not annotated) and *programming language*. Following steps show how this annotated query is handled to generate a Web resource annotation.

4. The annotated query is sent to the Query Execution system. This system first looks into the Knowledge Repository in order to find possible useful resources for user's interests. The Knowledge Repository component looks for information in its local contents and results are shown to the user as a URL list (may be empty). Currently only local contents of Knowledge Repository are explored. Of course we can think on the possibility of sending queries to other SQAPS peers implementing a kind of P2P Semantic Web Search engine facility. But, at the moment, the main functionality of SQAPS system is semantic annotation, so this facility is left for future work.
5. If no results are found, or the user requires more information, the Knowledge Repository takes the keywords from the annotated query (that is, the sequence of words *Caml programming language* in our example) and sends them to a classical Web search engine. The results of this engine are shown to the user as a list of URLs (may be empty).
6. If the list of results is not empty, the user can click on a URL, visualize a resource and, using a button in the GUI, he can associate the last semantic query to the resource being displayed. As a result, an RDF annotation is generated and inserted into the Knowledge Repository, which also inserts the annotation inside the SQAPS network in order to share it with other peers. As the only way to include an annotation is by clicking on the button, the user can decide at any moment the information to be shared, minimizing privacy problems.

3 Web Communities in SQAPS

In section 2.1 we have introduced the Semantic Source component of the SQAPS system. This Semantic Source should provide at least a list of terms, and for each term, a set of possible meanings with their human-readable descriptions. A unique identifier for each pair [term, meaning] should also be provided. The purpose of this information is to help the human user in query term annotation.

As we have said, in the first version of SQAPS, we proposed the usage of WordNet as Semantic Source. As we have also said, the problem with that approach comes from the fact that we use a static Semantic Source and knowledge is evolutive in nature: new concepts appear, others can change their meanings, etc.

Of course, new versions of WordNet can appear with time, and we can think on just updating the WordNet version of all SQAPS peers. But, from our point of view, this approach has three limitations:

- Updating the software of big amounts of distributed peers could be a problematic issue.
- We should rely in WordNet providers to get new Semantic Source versions.
- Knowledge evolves faster than new Semantic Source versions appear.

In order to address these difficulties we can think on allowing users to add new concepts to their own Semantic Sources. So, if a user is interested on a new concept, he can add it to his Semantic Source as soon as it appears. A possible problem with this approach is that different SQAPS peers will have different Semantic Sources, because personal user entries may differ from one peer to another. If we want annotations defined using personal concepts to be useful for other users, we need to define mappings between all those personal entries, which is a problematic issue.

The situation that we have just described is not a particular problem of WordNet. We can have the same problem if we use other ontologies or knowledge bases in the state of the art as Semantic Sources: we need to update their contents as knowledge evolves. Taking this into account, in this paper we want to explore a different approach: use the information generated and maintained by Web Communities as knowledge source for the query annotation process. In particular we have decided to use the information on Wikipedia [9] in our SQAPS system.

The Wikipedia project began on 2001 with the objective of producing a free content encyclopedia that could be edited by anyone. It has several interesting properties which make it attractive for our purposes:

Active community As is described in [12] Wikipedia contains nowadays approximately 1.6 million articles. During January 2005, Wikipedia had approximately 13,000 users who made at least five edits that month. Taking this information into account, we can say that Wikipedia is an active community with thousands of users, which seems to assure information maintenance.

Easily modifiable contents Wikipedia is based on Wiki [13] technology. Anybody can add new articles or modify existent ones. This can be done using a Web browser and just requires to know some basic templates and conventions. This is an advantage in relation with other possible knowledge sources as ontologies or knowledge bases, which require qualified persons (knowledge engineers) to maintain them.

Quality of contents As indicated in [12] different external analysis have shown that, though the information in Wikipedia can be modified by anybody, its contents seem to be of reasonable quality.

Broad range of topics Wikipedia was born with the objective of becoming and online encyclopedia. As a consequence, in their contents we can find information from different domains of knowledge: sports, science, literature, etc.

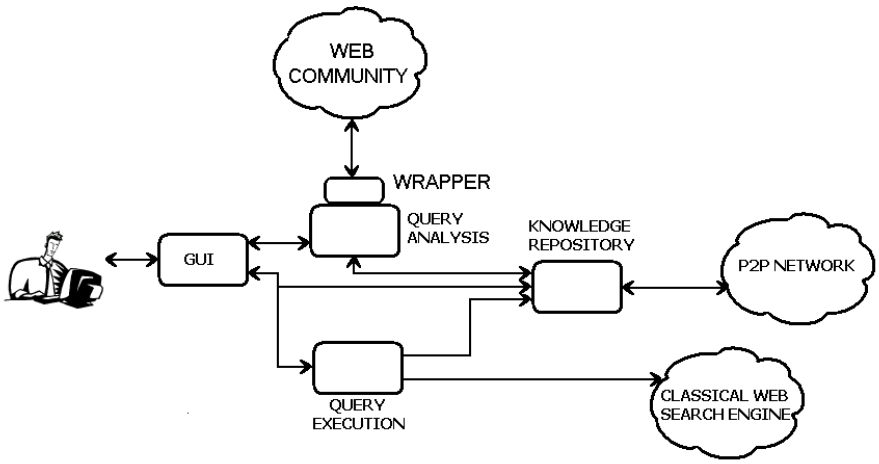

Fig. 3. SQAPS System Architecture with a Web Community as Semantic Source

In order to take advantage of the information in Wikipedia, we plan to use its information as our Semantic Source. In figure 3 we can see the SQAPS resultant system architecture. As can be seen, we will make use of wrappers to automatically process Wikipedia HTML pages. This is possible because such pages have a similar structure, which eases their automatic processing.

The objective of our wrappers is to obtain the information required for our purposes, which basically consist of:

- The term, the Wikipedia entry name.
- A description of its meaning in natural language, which will be an snippet automatically extracted from the beginning of the Wikipedia page.
- An identifier, which in our case will consist of the Wikipedia page URL.

In order to implement the wrappers which will allow us to extract information from Wikipedia pages, we are using a extended version of the WebL language [14], [15]. Figure 4 shows a simple script codified in this language. It receives as input parameters the language of the Wikipedia to be queried and the words which compose the term of interest (for example: *programming language*). Given that parameters, the script generates the URL to be accessed (*url*) and gets the page contents (*GetURL(url)*). Then it processes such contents, to extract the text snippet used as term description (*text*) and the Wikipedia entry name

(*term*). The snippet, the identifier and the entry name are shown in standard output. For instance, if we invoke the script with parameters *en programming language*, we currently get the output in figure 5.

```
import Str, Files, Forms, XPath;

var params = Rest(ARGS);

var lang  = First(params);
var words = Rest(params);
var query = "";

every word in words do
  if query == "" then
    query = word;
  else
    query = query + "_" + word;
  end;
end;

try

  var url = "http://" + lang + ".wikipedia.org/wiki/" + query;
  var Page = GetURL(url);

  var Header = Select(Elem(Page,"h1"), fun(a) a.class == "firstHeading" ? false end)[0];
  var Div = Select(Elem(Page,"div"), fun(a) a.id == "bodyContent" ? false end)[0];
  var P = Elem(Div,"p")[0];

  var text = Select(Text(P), 0, Str_IndexOf(".", Text(P)));
  var term = Text(Header);

  PrintLn("Identifier=" + url);
  PrintLn("Term=" + term);
  PrintLn("Description=" + text + "...");

catch E
  on true do
     PrintLn(E.type);
end;
```

Fig. 4. A very simple WebL Wikipedia wrapper

```
Identifier=http://en.wikipedia.org/wiki/programming_language
Term=Programming language
Description=A programming language or computer language is a standardized communication
technique for expressing instructions to a computer...
```

Fig. 5. Example of WebL script output

Of course this is just a simple script version, used to show the possible aspect of a WebL-based wrapper. The example script does not take into account things as the existence of disambiguation pages like [16] where several different meanings of a term are described. We are developing more advanced wrappers which will take this into account, trying to exploit those disambiguation pages in our own benefit.

4 Related Work

The field of semantic annotation has been an active area of research for long time. In the state of the art we can find several interesting proposals, like for example: SEAN [17], a system for automatic semantic annotation of content-rich template-based HTML documents; SemTag [18] designed to provide automatic semantic annotation of large amounts of documents, using information obtained from TAP [19] knowledge base; Annotea [20], a manual, annotation system which allows users define annotations of XHTML or XML resources and share such annotations using web-based annotation servers; AeroDAML [21] which automatically annotates documents using natural language processing technologies; CREAM [22] which allows users to manually generate annotations of Web resources by typing, by selecting pieces of text from these resources, or by associating to the resource elements in a knowledge base; S-CREAM [23] evolution of CREAM to allow semiautomatic annotation of resources using natural language processing techniques; PANKOW [24] an automatic system which uses linguistically-based regular expressions, and statistics from Google [25] queries, to identify instances of a concept in a text; SMORE [26] which allows the manual semantic markup of HTML documents using ontologies as knowledge sources; the COHSE Annotator [27] which allows users to select text in a resource and associate to such text a concept or instance in an ontology; MnM [28] which allows the automatic or semiautomatic annotation of text-based Web resources using natural language processing tools; the SHOE Knowledge Annotator [29] which allows the addition of annotations in SHOE language [30] to HTML documents, or [31], where the authors propose a system for deep annotation. In that system, SQL queries to a database, used to generate dynamic Web pages, can be annotated by the Web site provider. These SQL query annotations are later used to generate Web page annotations.

Apart from semantic annotation tools, collaborative bookmarking systems as [32] are also related with SQAPS approach to semantic annotation. These systems allow to bookmark interesting results from a Web searching process and share such results with other users of the system, exploiting the effort of Web users in a similar way to SQAPS.

As far as we know, the main differences of SQAPS approach with respect to all these works are:

- These systems do not exploit the annotation of keyword-based user queries in annotating Web resources, as proposed by SQAPS system.
- Exploiting the information maintained by Web communities, like Wikipedia, as knowledge source for the annotation process is also not suggested by these works.

5 Discussion

The approach that we have introduced in previous sections is still a work in progress. From our point of view it has some negative aspects which are described in this section for discussion purposes:

– The semantics offered by Wikipedia is not very formal. Wikipedia can not be currently seen as an ontology or knowledge base. Basically it offers the semantics of a semantic network of linked topics, being such links between topics untyped. From our point of view, we have here a tradeoff between the degree of formalization of our Semantic Source and the easiness of modifying it: the more formal the Semantic Source, the richer the semantics that it offers, but the more difficult is to maintain that knowledge. In any case, we can find in the state of the art proposals for making Wikipedia a more formal source without making much more difficult the edition process [33]. This could be a way of addressing this limitation.
– If the structure of Wikipedia pages changes, we need to update our wrappers, so software maintenance is still required. But, from our point of view, knowledge evolves faster than Wikipedia pages structure, so the updating process in this case seems less critical than in the WordNet one.
– We have not a local Semantic Source, but a remote one. We need Web connection and the Wikipedia being operative in order to search our new Semantic Source. But, given that we also need Web connection and a Web search engine being operative in order to be able to annotate a Web page, we consider that this is not a critical drawback.

6 Conclusions and Future Lines

In this paper we have described the SQAPS, *Semantic Query-based Annotation, P2P Sharing*, system. The main idea behind this system is to exploit keyword-based user queries in semantic annotation of Web resources. Instead of annotating directly Web resources, as most of current systems in the state of the art suggest, we have proposed a system in which users annotate their queries. In order to share these keyword-based semantic annotations with other SQAPS users, a DHT P2P infrastructure is used.

From our point of view, this alternative approach to annotation of Web resources has the advantage that can exploit the effort of the millions of users who every day look for information on the Web. Additionally it supports the annotation of different kinds of resources. This is so because in the context of the SQAPS system, an annotation is an association of a certain URL, representing a Web resource, and a semantic query. As resources can be multimedia files and not only text or HTML, in principle the annotation of multimedia items is supported, but, of course, only in the case that those items can be retrieved using a keyword-based query and a classical Web search engine.

An additional feature of the SQAPS system version introduced here is that it exploits the information generated and maintained by Wikipedia Web community for the annotation process. This has the advantage that the source of knowledge evolves with time instead of being static, as it was in the former SQAPS version. Additionally, we are planing as future line to integrate the Wikipedia edition process with SQAPS, so as soon as a user detects that some of the terms in his query are not included in Wikipedia he can have the possibility of adding the new contents by himself.

On the negative side, one of the main drawbacks of our system is the dependence on user collaboration. In relation with this, from our point of view, integrating the annotation activities with habitual user actions, such as Web search, should be a point in favor. Moreover, if we take into account that queries are not too long, and that query processing is partly done by the system, we expect that the work finally done by the users will imply a low overhead compared to classical keyword-based search.

In any case, we need to test the concept behind our system and its usability. In order to do so we are currently working on a first basic prototype of the SQAPS system. It is being developed as a standalone Java application, using Jena [34] as RDF(S) management system (for the Knowledge Repository), JXTA 2 [35] framework for the initial P2P network implementation, and the Google Web APIs [36] in order to connect our system to a classical Web search engine. Of course if the results of this prototype testing are promising, we also plan in the near future to integrate our application with popular Web browsers using plug-ins or other Web browser extension technologies.

Acknowledgements

This work has been partially funded by the *Ministerio de Educación y Ciencia de España*, as part of the Infoflex Project, TIC2003-07208.

References

1. Berners-Lee, T.; Hendler, J.; Lassila, O.; The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities. Scientific American, May 2001.
2. Handschuh, S. and Staab, S. (Eds.); Annotation for the Semantic Web. Ed. IOS Press, ISBN 1-58603-345-X, 2003.
3. Fernández-García, N.; Sánchez-Fernández, L.; Blázquez-del-Toro, J.; Larrabeiti, D.; An Ontology-based P2P System for Query-based Semantic Annotation Sharing. In Ontologies in P2P Communities Workshop colocated with ESWC 2005, Heraklion, Crete, May 2005.
4. Resource Description Framework (RDF). Available at: http://www.w3.org/RDF/.
5. RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation 10 February 2004. Available at: http://www.w3.org/TR/rdf-schema/.
6. The SQAPS System Ontology. Available at: http://www.it.uc3m.es/berto/SQAPS/SQAPSOnt.rdf.xml.
7. Distributed Has Table. Wikipedia: The Free Encyclopedia. Available at: http://en.wikipedia.org/wiki/Distributed_hash_table.
8. WordNet, a lexical database for the English language. Available at: http://wordnet.princeton.edu/.
9. Wikipedia: The Free Encyclopedia. Available at: http://en.wikipedia.org/.
10. Harren, M.; Hellerstein, J.M.; Huebsch, R.; Loo, B.T.; Shenker, S.; Stoica, I.; Complex Queries in DHT-based Peer-to-Peer Networks. Available at: http://www.cs.rice.edu/Conferences/IPTPS02/191.pdf.

11. Li, J.; Loo, B.T.; J.M. Hellerstein, J.M.; Kaashoek, M.F.; On the Feasibility of Peer-to-Peer Web Indexing and Search. In 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03), February 2003, Berkeley, CA, USA.
12. Wikipedia. Wikipedia: The Free Encyclopedia. Available at: http://en.wikipedia.org/wiki/Wikipedia.
13. Wiki. Wikipedia: The Free Encyclopedia. Available at: http://en.wikipedia.org/wiki/Wiki.
14. Kistler, T.; Marais, H.; WebL - A programming language for the Web. In Proceeding of the 7th International World Wide Web Conference, pp. 259-270. Computer Networks and ISDN systems 30, 1998.
15. Luque Centeno, V.; Delgado Kloos, C.; Sánchez Fernández, L.; Fernández García, N.; Intelligent Automated Navigation through the Deep Web. In Proceedings of Second International Atlantic Web Conference, AWIC 2004, pp. 125-134, LNAI 3034. Cancun, Mexico, May 2004.
16. Bush. Wikipedia: The Free Encyclopedia. Available at: http://en.wikipedia.org/wiki/Bush.
17. Mukherjee, S.; Yang, G.; Ramakrishnan, I.V.; Automatic Annotation of Content-Rich HTML Documents: Structural and Semantic Analysis. The Semantic Web - ISWC 2003, LNCS 2870, pp 533-549.
18. Dill, S.; Eiron, N.; Gibson, D.; Gruhl, D.; Guha, R.; Jhingran, A.; Kanungo, T.; Rajagopalan, S.; Tomkins, A.; Tomlin, J.A.; Zien, J.Y.; SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation. WWW2003 Conference, May 2003, Budapest, Hungary.
19. TAP: Building the Semantic Web. Available at: http://tap.stanford.edu/.
20. Kahan, J.; Koivunen, M-R.; Prud'Hommeaux, E.; Swick, R.R.; Annotea: An Open RDF Infraestructure for Shared Web Annotations. In WWW10 Conference, May 1-5 2001, Hong Kong.
21. Kogut, P.; Holmes, W.; AeroDAML: Applying Information Extraction to Generate DAML Annotations from Web Pages. First International Conference on Knowledge Capture (K-CAP 2001). Workshop on Knowledge Markup and Semantic Annotation, Victoria, B.C. October 21, 2001.
22. Handschuh, S.; Staab, S.; Authoring and Annotation of Web Pages in CREAM. Proceedings of the 11th International World Wide Web Conference, WWW 2002, Honolulu, Hawaii, May 7-11, 2002. ACM Press.
23. Handschuh, S.; Staab, S.; Ciravegna, F.; S-CREAM: Semi-automatic CREAtion of Metadata Proceedings of the European Conference on Knowledge Acquisition and Management - EKAW-2002. Madrid, Spain, October 1-4, 2002.
24. Cimiano, P.; Handschuh, S.; Staab, S.; Towards the Self-annotating Web. In the 13th International World Wide Web Conference, WWW 2004, New York, USA, May 17-22, 2004
25. Google Web Search Engine. Available at: http://www.google.com.
26. Kalyanpur, A.; Hendler, J.; Parsia, B.; Golbeck, J.; SMORE - Semantic Markup, Ontology, and RDF Editor. Available at: http://www.mindswap.org/papers/SMORE.pdf.
27. Bechhofer, S.; Goble, C.; Carr, L.; Kampa, S.; COHSE: Semantic Web gives a Better Deal for the Whole Web? Poster presentation at ISWC International Semantic Web Conference, Sardinia, June 2002.
28. Vargas-Vera, M.; Motta, E.; Domingue, J.; Lanzoni, M.; Stutt, F.; Ciravegna, F.; MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup. In 13th International Conference on Knowledge Engineering and Management (EKAW 2002), Springer Verlag, 2002.

29. SHOE Knowledge Annotator. Available at: http://www.cs.umd.edu/projects/plus/SHOE/KnowledgeAnnotator.html.
30. SHOE: Simple HTML Ontology Extensions. Available at: http://www.cs.umd.edu/projects/plus/SHOE/index.html.
31. Handschuh, S.; Staab, S.; Volz, R.; On Deep Annotation. In 12th International World Wide Web Conference, WWW2003, Budapest, Hungary, May 2003.
32. Kanawati, R.; Malek, M.; A Multi-agent System for Collaborative Bookmarking. The CEUR Workshop Proceedings Website. Available at: http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-59/8Kanawati.pdf.
33. Krötzsch, M.; Vrandecic, D.; Völkel, M.; Wikipedia and the Semantic Web - The Missing Links. In Proceedings of Wikimania 2005: The First International Wikimedia Conference, Frankfurt am Main, Germany, August 2005. Available at: http://www.aifb.uni-karlsruhe.de/WBS/mak/pub/wikimania.pdf.
34. Jena - A Semantic Web Framework for Java. Available at: http://jena.sourceforge.net/index.html
35. JXTA 2: A high-performance, massively scalable P2P network. Available at: http://www-106.ibm.com/developerworks/java/library/j-jxta2/
36. Google Web APIs Home. Available at: http://www.google.com/apis/