

Hermeneus: An Architecture for an Ontology-Enabled Information Retrieval

Fabiano D. Beppler
Knowledge Engineering and
Management – EGC/UFSC
Trindade, Florianopolis, SC, Brazil
Stela Institute
Rua Prof. Ayrton R. de Oliveira, 32
Florianopolis, SC, Brazil
+55 (48) 3239-2500
fbeppler@stela.org.br

Frederico T. Fonseca
College of Information Sciences and
Technology. Pennsylvania State
University
University Park, PA 16802, USA
+1 (814) 865-6460
fredfonseca@ist.psu.edu

Roberto C. S. Pacheco
Knowledge Engineering and
Management – EGC/UFSC
Trindade, Florianopolis, SC, Brazil
Stela Institute
Rua Prof. Ayrton R. de Oliveira, 32
Florianopolis, SC, Brazil
+55 (48) 3239-2500
pacheco@stela.org.br

ABSTRACT

Ontologies improve IR systems regarding its retrieval and presentation of information, which make the task of finding information more effective, efficient, and interactive. In this paper we argue that ontologies also greatly improve the engineering of such systems. We created a framework that uses ontology to drive the process of engineering an IR system. We developed a prototype that shows how a domain specialist without knowledge in the IR field can build an IR system with interactive components. The resulting system provides support for users not only to find their information needs but also to extend their state of knowledge. This way, our approach to ontology-enabled information retrieval addresses both the engineering aspect described here and also the usability aspect described elsewhere.

Keywords

Ontology-Enabled Information Retrieval, Ontologies, Information Retrieval System Design.

1. INTRODUCTION

Ontology is a representation vocabulary that characterizes the knowledge of a domain [1]. Ontologies have been used in a variety of areas as a support for creating more intelligent systems [2]. According to Chandrasekaran et al. [3], “IR systems, digital libraries, and Internet search engines need domain ontologies to organize information and direct the search process.” Information systems are taking advantage of ontologies to perform and enhance a broad range of tasks (e.g., knowledge extraction and retrieval). Nevertheless, the use of ontologies in engineering a system is less well researched. This is the aspect suggested by Guarino [4] when he introduced the concept of ontology-driven information systems. In this paper we describe the architecture of Hermeneus, which is a framework to build IR systems that addresses how ontologies support the engineering of the system. Hermeneus uses ontology to drive the process of creating an IR system. We describe the implemented prototype that shows how a domain specialist, without knowledge in the IR field, can build an

IR system with interactive components.

Hermeneus contains four modules responsible for indexing and retrieving ontology instances, inferring additional information, and presenting visual components that allow users to interact with the system. Because we index ontology instances instead of information without context, users can formulate semantic queries (e.g., “*author: Fonseca*”) using concepts defined in the ontology. The ontology, which is graphically available, also enables users to change modes of visualization of the result set (i.e., visualize retrieved instances grouped by any concept defined in the ontology). As our approach retrieves instances (i.e., semantic content), users can refine their searches using pieces of retrieved content dynamically. For instance, when getting an insight while browsing a retrieved instance, a user can click on the instance to refine the search automatically. Furthermore, because the ontology is described in a formal language that includes axioms for specifying relationships between concepts, Hermeneus enables users to visualize and interact with additional knowledge.

In the next section, we introduce Hermeneus and in section 3 its modules and components are described. Section 4 presents a prototype developed to validate our framework. We discuss Hermeneus in section 5 and in section 6 we describe some similar works. Finally, we present our conclusions and future work.

2. HERMENEUS

The main goal of developing an IR system is to facilitate the access of the end-user to the desired information. However, building such IR systems requires background in IR. Further, designers should know how to translate the domain knowledge into terms that can lead to an implemented system. Trying to address these issues, we propose a framework that uses an ontology to drive the process of building an IR system. We called our framework Hermeneus, which in Greek means the interpreter or translator, following Kuhlthau [5] who considered that the IR system is one intermediary that should facilitate the user to move from the initial state of information need to the goal state of resolution. Hermeneus intends to allow domain specialists, without skills in IR field, to build IR systems and then make the domain knowledge accessible through a search tool. The domain specialist, after building the ontology and its knowledge base, can simply configure the modules belonging to the Hermeneus architecture. We show the general idea in Figure 1, where a domain specialist who have developed the ontology and the knowledge base can configure the required modules so as to

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'08, March 16-20, 2008, Fortaleza, Ceará, Brazil.

Copyright 2008 ACM 978-1-59593-753-7/08/0003...\$5.00.

generate an IR system. Basically, all modules need to have access to the ontology and the knowledge base. The inference module requires additional configuration due to the necessity of defining rules in order to extract additional information. After configuring such modules, semantic indexes are built automatically. The resulting IR system intends to provide an interactive environment allowing users not only to search the domain knowledge but also to develop their information needs.

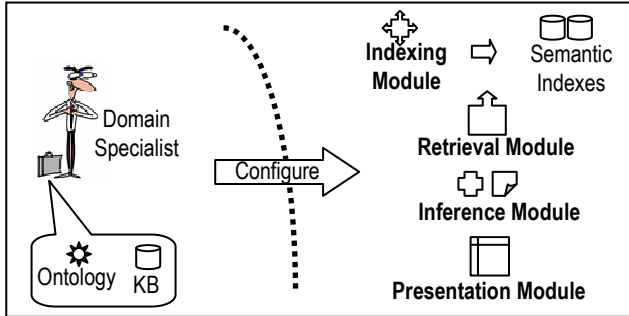


Figure 1. Domain specialist configures Hermeneus

3. THE FRAMEWORK ARCHITECTURE

We propose a framework that uses an ontology to drive the process of creating an IR system. In addition, the ontology also supports the construction of interactive components. According to Guarino [4], “ontologies refer to an engineering artifact, constituted by a specific vocabulary used to describe a certain reality, plus a set of explicit assumptions regarding the intended meaning of the vocabulary words.” As an engineering artifact, ontologies may be used to generate or validate information systems components [6]. Our framework contains four modules: *indexing*, *retrieval*, *inference*, and *presentation*. The ontology and its knowledge base should be created by the domain specialist – the actor who has the required knowledge about the domain. It is important to emphasize that the ontology should be defined according to specific principles and rules widely stated in the literature [4, 7, 8].

3.1 Indexing module

The *indexing module* uses an ontology and its knowledge base to create automatically semantic indexes composed of ontology instances. Basically, this module reads the ontology so as to identify its concepts and relations – a class and its respective properties define a concept. Then, the *indexing module* loads the instances stored in the knowledge base in order to index their content. Inverted index is the most common structure for indexing a text collection so as to speed up the searching task [9]. We adapted the inverted index structure to store semantic information. This approach adds a new dimension to the (semantic) inverted index structure that allows the identification of terms for specific contexts quickly. Furthermore, Hermeneus creates a semantic index for every concept defined in the ontology – each index stores instance contents specifically about one concept and the concepts it has direct relation with. These indexes, besides supporting semantic queries, enable users to visualize the result set in different perspectives (e.g., visualize the result set grouped

not only by institutions and its researchers but also by researchers and their respective institutions).

3.2 Retrieval module

The *retrieval module* is responsible for retrieving ontology instances from the semantic indexes as a response to a user need. Before being sent to the user, retrieved instances are ranked. The ranking process is another important feature of the *retrieval module*. We use the similarity measure called *coordinate matching* [10]. As this module handles semantic queries and the indexes contain the context for each term, we simply restrict possible retrieved instances to those that contain exactly the semantic content described in a semantic query. Because we use just the subset of instances that hold the semantic expressions, we did not change the original similarity measure. A semantic query is composed of concept description and respective terms (e.g., “*author:pacheco*”). Figure 2 describes in EBNF the format of queries applied to a search.

```

search ::= query | semantic_query
query ::= term { term }
semantic_query ::= concept ":" term { term } ";" { concept ":" term { term } ";" }
concept ::= <classes described in the ontology>
term ::= letter { letter }-| digit { digit }-
letter ::= "a" | "b" | ... | "z"
digit ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"

```

Figure 2. EBNF describing a query in a search

3.3 Presentation module

The *presentation module* configures an interface that takes advantage of the ontology in order to create an interactive environment for IR. This module contains three components: *ontology navigator*, *retrieved instances*, and *additional information*. The *ontology navigator component* presents the ontology in a graphic way, which allows users to navigate in the domain knowledge. The navigator also assists the composition of semantic queries and allows changes to the visualization of the result set. This last feature enables users to look at retrieved instances grouped by a specific concept, called *central concept*. In addition, the *ontology navigator* uses distinct shape to facilitate the identification of the *central concept* and it uses colors for the identification of the other concepts and their respective values. The second component, called *retrieved instances*, organizes the retrieved instances in order to allow users to compose semantic queries dynamically through mouse clicks using pieces of such instances. As a result, instead of formulating semantic queries manually when getting insights with retrieved information, users can simply click on the desired content and a search refinement is performed automatically. The last one, called *additional information component*, enables users to visualize and interact with additional information about a specific instance. Users can select a retrieved instance to see additional information. Moreover, such information can also be used to refine a search – similarly to the *retrieved instances component*. The *additional information component* is directly related to the *inference module*, which uses inference techniques and rules pre-configured to extract further information.

3.4 Inference module

The *inference module* extracts additional information from retrieved instances using the ontology and its knowledge base through inference and pre-configured rules. As the domain specialist has the knowledge about the domain, s/he should configure the rules. According to Golbreich [11] and Hyvönen et al. [12], rules are useful to represent the deductive part of the knowledge. Hermeneus accepts personalized rules that can be built in the ontology and instance level. Ontology-level rules are applied directly on the ontology structure. This feature allows the definition of new relationships between concepts or the specification of different restrictions. For example, in an ontology describing only direct family relationships (e.g., grandfather, mother, father, son), ontology-level rules could define indirect relationships (e.g., uncle, aunt, nephew, niece, cousin). Instance-level rules, also known as query-level rules [11], enable the extraction of information from ontology instances (i.e., knowledge base). Instance-level rules can also be formulated to take advantage of ontology-level rules. For example, in the family example above, we can configure ontology-level rules to specify indirect relationships, so that instance-level rules could easily retrieve the cousins and uncles of a specific person.

4. PROTOTYPE DESCRIPTION

We implemented a prototype to show the viability of our framework using information about paper's citations from the Journal of the America Society for Information Science and Technology. We conceived an ontology (Figure 3) composed of five classes that define *paper*, *author*, *institution*, *journal*, and *keyword* concepts. Subsequently, we built the knowledge base with the ontology instances. All instances were constructed with content explicit available in the citations except *keyword* instances. In order to create *keyword* instances, we used the Information Extraction technique called Named Entity Recognition (NER) [13], which extracts entities from the *title* and *abstract* fields. We chose Web Ontology Language (OWL) to represent our ontology and the Jena Toolkit [14] to build the knowledge base, configure the inference engine, and describe and interpret ontology-level rules (Jena Rule Language) and instance-level rules (SPARQL).

The *indexing module* reads automatically the ontology and the knowledge base in order to create the semantic indexes. Our prototype has five indexes – one for each concept defined in the ontology. Each index stores instances about a specific concept and instances of direct related concepts (e.g., the *author* index

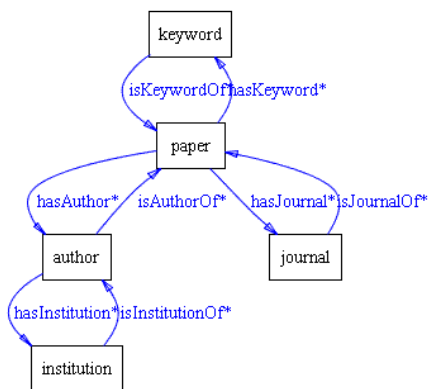


Figure 3. Ontology about papers

contains information about *author*, *paper*, and *institution* instances). The *retrieval module* just needs to know the file location of the ontology (i.e., OWL file) and of the indexes. The *ontology navigator component*, belonging to the *presentation module*, just needs to know the ontology's file location so as to present it graphically. The *retrieved instances* and *additional information* components, also belonging to the *presentation module*, communicate directly to the *retrieval* and *inference* modules respectively. We used the Jena inference engine and configured the rules in the *inference component*. Our prototype contains one ontology-level rule that creates a new relation between *author* and *keyword* concept; therefore, we can easily find the keywords of a specific author. Figure 4 (top) shows an example of ontology-level rule expressed in Jena Rule Language and an instance-level rule (bottom) that extracts information from the instances using the new relation created by the ontology-level rule.

```

[rule1: (?paper NS#hasAuthor ?author) (?paper NS#hasKeyword ?keyword)
-> (?author NS#authorsKeyword ?keyword)]
SELECT ?keywordname WHERE { ?keyword :keywordname ?keywordname .
?author :authorsKeyword ?keyword .?author :name 'Frederico Fonseca' . }
  
```

Figure 4. Example of ontology- and instance-level rules

We implemented a few instance-level rules used to extract additional information from the knowledge base. In addition, as our approach allows the visualization of retrieved instances according to a *central concept*, we configured different instance-level rules for some concepts. When the *central concept* is *paper* the *additional information component* presents additional information about the authors and their institutions; any other *central concept* additional information specifically about the paper of a selected instance is shown. The *presentation module* arranges the interactive components in a way that allow users to visualize and interact with all of them in the same screen. Figure 5 shows the screen with four different visualization areas. The first one at the top contains the text field where basic and semantic queries can be typed. The second one on the left shows the ontology – the *ontology navigator component* – where users can visualize and interact with concepts and their relations. This component also assists users to create and visualize (i.e., the yellow circle) semantic queries and to choose a *central concept* (i.e., square shape) to view the result set in different perspectives. The third visualization area on the center presents the retrieved instances allowing users to compose new semantic queries dynamically through mouse-clicks on each piece of the retrieved content. And finally the fourth area on the right – the *additional*

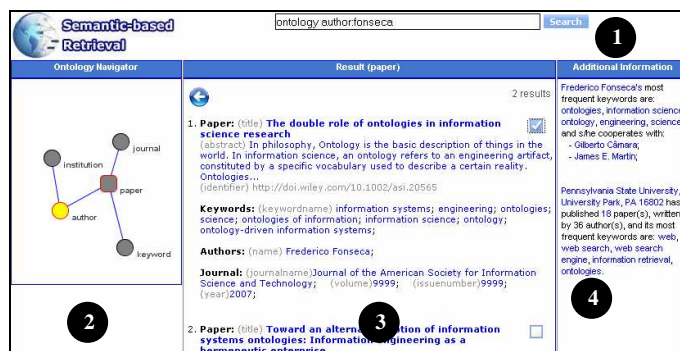


Figure 5. Result screen showing: (1) place to type queries; (2) ontology navigator component; (3) retrieved instances component; and (4) additional information component

information component – displays additional information about a retrieved instance chosen by the user.

5. DISCUSSION

We presented our framework Hermeneus that uses an ontology to drive the process of creating an IR system. The ontology also supports the construction of interactive components. In the following sections we discuss how the ontology can be used to build the required modules of an IR system and how the interactive components provide support for users not only to find their information needs but also to extend their state of knowledge.

5.1 Building ontology-based IR systems

Our goal was to create a higher-level framework to build IR systems by domain specialists without skills in IR research. Therefore, domain specialists can focus solely on the domain knowledge and not concern with implementation details. In addition, the aim was to shorten the time of developing IR applications while also improving their usability and usefulness. Because domain specialists can build the IR system themselves, our approach also avoids problems related to communication between stakeholders (e.g., domain specialist, designer, application developer, user-interface designer) normally involved in a process of creating an IR system. Our framework is in accordance with Hendry and Harper [15] statement who suggests that “an architecture can reduce the cognitive distance between concepts in a problem domain and the software abstractions need in a program.”

Hermeneus builds automatically the required modules of an IR system taking into account the ontology structure and its knowledge base. Hermeneus does not depend on a specific IR model. We defined its modules to support the creation of interactive components. For example, Hermeneus creates a semantic index for each concept defined in the ontology, so users can visualize the result set in different perspectives. Because this grouping task is performed during the indexing process, there is no impact on performance when changing modes of visualization *on the fly*. Additionally, because Hermeneus deals with semantic queries, it retrieves information more precisely. According to Vallet et al. [16], ontologies can overcome the limitations of keyword-based search and provide better recall and precision.

We developed Hermeneus to be modular, flexible, configurable, and extensible. Its architecture is composed of four modules and three components that facilitate to adapt to specificities required by a domain. For example, a designer can change how to visualize the retrieved instances adjusting the *retrieved instances component*. Domain specialists can configure rules and specify how the additional information should be presented. In our prototype, for instance, we extended the relations between concepts configuring ontology-level rules and instance-level rules to extract more valuable information directly from the ontology and its knowledge base. Furthermore, because Hermeneus is modular, new features can be added to the architecture easily. For example, a designer can develop a new ranking algorithm and change only the *retrieval module*; or, s/he can add a new similarity measure and allow domain specialists to choose which one to use when configuring the framework.

5.2 Interactive components

Searching is often an exploratory activity where users apply knowledge, intuition, and strategies together with tools to find the desired information [17]. Our approach uses ontology to provide interactive components that assist users not only during the search process but also developing their information needs. The *ontology navigator component*, for instance, enables users to navigate in the ontology concepts and their relationships. Fluit et al. [18] states that the added value of visualizing an ontology graphically lies in its expressivity because the vocabulary of the domain is easy to detect. As users interact with the ontology, they can compose more valuable queries. This component also allows define the type of retrieved instances selecting a *central concept*, so users can have different perspectives of the result set for the same query. McGuinness [19] states that ontology can be used to derive further views of knowledge, particularly in the exploration of new navigation purposes. For example, our prototype retrieved 11 instances about *paper* concept when searching for “*semantic web*” terms (the *central concept* is *paper*); if one wants to see just the papers that mention the “*ontology*” instance, related to the *keyword* concept, s/he just needs to select as the *central concept* the *keyword* concept. With this view, the user can easily identify the “*ontology*” instance, related to the *keyword* concept, and the respective list of *paper* instances.

The *retrieved instance component*, which shows the retrieved instances, allows users to select pieces of such instances to refine their searches. For example, searching for the “*search*” term in our prototype, we received 172 instances; clicking on “*web search*” content, belonged to a retrieved instance, related to the *keyword* concept (i.e., our query is “*search keyword: web search;*”), gave as a result only 61 instances. As users articulate their information needs more dynamically performing this task continuously, they can find the desired information more quickly. In the prior example, if we click on “*search engine*” content, also related to the *keyword* concept (i.e., our query is “*search keyword: web search; keyword: search engine;*”) we receive just 32 instances – from 172 to 32 instances with two clicks. Users can also visualize and interact with additional information for each retrieved instance. This is the role of the *additional information component* that presents information extracted by the *inference module*. Hatala et al. [20] state that “ontologies and rules provide an excellent platform for building a highly-responsive context-aware interactive application.” In our prototype, for instance, users can easily see the author’s most frequent keywords, the list of people who cooperate with a specific person, and the amount of published papers related to an institution. The information available in this module can also be used as an exploratory activity, for users can interact with such information and refine their queries dynamically. As users learn more about their problem space when interacting with retrieved content [5, 21], their information needs often modifies – our components give support for this transition.

6. RELATED WORK

There are some works applying ontology and IR research so as to develop better IR systems, but none of them are similar to our approach. McGuinness [19] proposes an ontology-enhanced online search for medical documents where users can combine a simple keyword search with content areas described in a

taxonomy. Lei et al. [22] propose a semantic search that intends to make a search simple and effective. Vallet et al. [16] describe an ontology-based IR model where indexes are replaced by a knowledge base. Hyvönen et al. [12] present a semantic portal with an IR system grounded on ontological concepts. Their IR system, based on the multi-facet search paradigm, uses a set of taxonomies. Nevertheless, in most cases an ontology can be more than a taxonomy of concepts, involving in particular constraints and interrelations among concepts [1]. Hermeneus allows domain specialists without skills in IR to build an IR system. The resulting system enables users to interact with the ontology, compose semantic queries, visualize the result set in different perspectives, select retrieved content to refine their searches, and interact with additional information.

7. CONCLUSION AND FUTURE WORK

We described our framework called Hermeneus which uses an ontology to drive the process of building a IR system with interactive components. Hermeneus enables domain specialists without background in IR research to build such a system. Hermeneus contains four modules responsible for indexing and retrieving ontology instances, presenting interactive components, and inferring information. The interactive components were based on information seeking theory that states that users want an environment where they can intervene and interact more actively with information. Our framework enables users to engage in a sequence of interactions not only to facilitate the search process but also to help them understand what they are looking for.

As a future work, we intend to create an environment where domain specialists can compose rules visually without needing to understand a specific rule language. A new graphical way to present the retrieved information to take advantage of semantic content will be developed. We also envisage a mechanism where users can define their own ontologies and configure an IR system according to their notion of reality for a specific domain.

8. REFERENCES

- [1] Guarino, N. Understanding and Building, Using Ontologies. *Int. Journal of Human-Computer Interaction*, 46, 2 (1997), 293-310.
- [2] Sowa, J. F. Architectures for Intelligent Systems. *Systems Journal, Artificial Intelligence*, 41, 2 (2002), 331.
- [3] Chandrasekaran, B., Josephson, J. R. and Benjamins, V. R. What are Ontologies, and Why Do We Need Them? *IEEE Intelligent Systems*, 14, 1 (1999), 20-26.
- [4] Guarino, N. Formal Ontology and Information Systems. *Formal Ontology in Information Systems, Netherlands: IOS Press*1998), 3-15.
- [5] Kuhlthau, C. C. *Seeking Meaning: A Process Approach to Library and Information Services*. Ablex Publishing Corporation, USA, 1993.
- [6] Fonseca, F. The Double Role of Ontologies in Information Science. *Journal of the American Society for Information Science and Technology*, 58, 6 (2007), 786-793.
- [7] Noy, N. F. and McGuinness, D. L. *Ontology Development 101: A Guide to Creating Your First Ontology*. *Knowledge Systems Laboratory*2001).
- [8] Corcho, O., Fernández-López, M. and Gómez-Pérez, A. Methodologies, Tools and Languages for Building Ontologies: Where is their Meeting Point? *Data & Knowledge Engineering*, 46, 1 (2003), 41-64.
- [9] Baeza-Yates, R. A. and Ribeiro-Neto, B. A. *Modern Information Retrieval*. ACM Press, Addison-Wesley, 1999.
- [10] Witten, I. H., Moffat, A. and Bell, T. C. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, San Francisco, CA, 1999.
- [11] Golbreich, C. *Combining Rule and Ontology Reasoners for the Semantic Web*. Springer-Verlag, Hiroshima, Japan, 2004.
- [12] Hyvönen, E., Mäkelä, E., Salminen, M., Valo, A., Viljanen, K., Saarela, S., Junnila, M. and Kettula, S. MuseumFinland: Finnish Museums on the Semantic Web. *Journal of Web Semantics*, 3, 2 (2005), 25.
- [13] Mikheev, A., Moens, M. and Glover, C. *Named Entity Recognition without Gazetteers*. 9th Int. Conf. of the European Chapter of the Association for Computational Linguistics, 1999.
- [14] McBride, B. Jena: A Semantic Web Toolkit. *Internet Computing, IEEE*, 6, 6 (2002), 55-59.
- [15] Hendry, D. G. and Harper, D. J. *An Architecture for Implementing Extensible Information-Seeking Environments*. ACM Press, Switzerland, 1996.
- [16] Vallet, D., Fernández, M. and Castells, P. *An Ontology-Based Information Retrieval Model*. Springer, 2005.
- [17] Capra III, R. G. and Pérez-Quñones, M. A. Using Web Search Engines to Find and Refind Information. *IEEE Computer Society*, 38, 10 (2005), 36-42.
- [18] Fluit, C., Sabou, M. and Harmelen, F. v. *Ontology-Based Information Visualization*. Springer Verlag, 2002.
- [19] McGuinness, D. L. *Ontology-Enhanced Search for Primary Care Medical Literature*. Phoenix, Arizona, 1999.
- [20] Hatala, M., Wakkary, R. and Kalantari, L. Rules and Ontologies in Support of Real-Time Ubiquitous Application. *Journal of Web Semantics*, 3, 1 (2005), 5-22.
- [21] Ingwersen, P. and Järvelin, K. *The Turn: Integration of Information Seeking and Retrieval in Context*. Springer, Netherlands, 2005.
- [22] Lei, Y., Uren, V. and Motta, E. *SemSearch: A Search Engine for the Semantic Web*. Montenegro, 2006.