Semantic Web 0 (0) 1
IOS Press

Watson, more than a Semantic Web search engine

Editor(s): Jérôme Euzenat, INRIA Grenoble Rhône-Alpes, France
Solicited review(s): Philipp Cimiano, Universität Bielefeld, Germany; Laura Hollink, Delft University of Technology, The Netherlands; Eero Hyvönen, Aalto University, Finland; anonymous reviewer

Mathieu d'Aquin and Enrico Motta
Knowledge Media Institute, The Open University
Walton Hall, Milton Keynes, UK
{m.daquin, e.motta}@open.ac.uk

Abstract. In this tool report, we present an overview of the Watson system, a Semantic Web search engine providing various functionalities not only to find and locate ontologies and semantic data online, but also to explore the content of these semantic documents. Beyond the simple facade of a search engine for the Semantic Web, we show that the availability of such a component brings new possibilities in terms of developing semantic applications that exploit the content of the Semantic Web. Indeed, Watson provides a set of APIs containing high level functions for finding, exploring and querying semantic data and ontologies that have been published online. Thanks to these APIs, new applications have emerged that connect activities such as ontology construction, matching, sense disambiguation and question answering to the Semantic Web, developed by our group and others. In addition, we also describe Watson as a unprecedented research platform for the study the Semantic Web, and of formalised knowledge in general.

Keywords: Watson, Semantic Web search engine, Semantic Web index, Semantic Web applications

1. Introduction

The work on the Watson system[1] originated from the idea that formalised knowledge and semantic data was to be made available online, for applications to find and exploit. However, for knowledge to be available does not directly imply that it can be discovered, explored and combined easily and efficiently. New mechanisms are required to enable the development of applications exploring large scale, online semantics [12].

Watson collects, analyses and gives access to ontologies and semantic data available online. In principle, it is a search engine dedicated to specific types of 'documents', which rely on standard Semantic Web formats. Its architecture (see next section) therefore includes a crawler, indexes and query mechanisms to these indexes. However, beyond this simple facade of a Semantic Web search engine (see Section 3), the main objective of Watson is to represent a homogeneous and efficient access point to knowledge published online, a *gateway to the Semantic Web*. It therefore provides many advanced functionalities to applications, through a set of APIs (see section 4), not only to find and locate semantic documents, but also to explore them, access their content and query them, including basic level reasoning mechanisms, metrics and links to user evaluation tools. Of course, Watson is not the only tool of its kind (see related work in Section 6), but it can be distinguished from others by its focus on providing a complete infrastructure component for the development of applications of the Semantic Web. It has led to the development of a large variety of applications, both from our group and from others. In this paper, we present a complete, up-to-date overview of the Watson system, as well as of applications which are made possible by the functionalities it provides. We also show through several examples how, as a side effect of providing a gateway to the Semantic Web, Watson is being used as a platform to support research activities related to the Semantic Web (see Section 5).

[1] http://watson.kmi.open.ac.uk

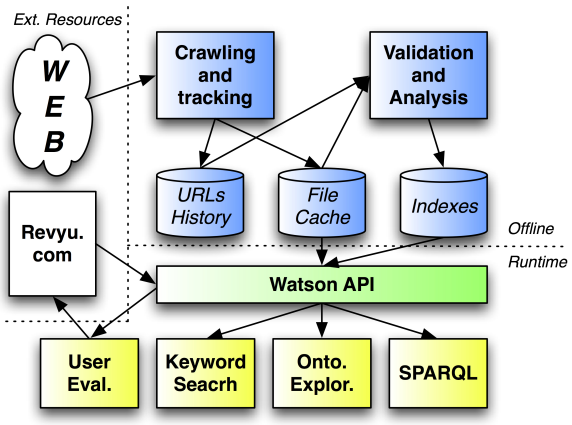

Fig. 1. Overview of the Watson architecture.

2. Anatomy of a Semantic Web search engine

Watson performs three main activities:

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

While these three tasks are generally at the basis of any classical Web search engine, their implementation is rather different when dealing with semantic content as opposed to Web pages.

To carry out these tasks, Watson is based on a number of components depicted in Figure 1, relying on existing, standard and open technologies. Locations of existing semantic documents are first discovered through a *crawling and tracking* component, using Heritrix, the Internet Archive's Crawler[2]. The *Validation and Analysis component* is then used to create a sophisticated system of indexes for the discovered documents, using the Apache Lucene indexing system[3]. Based on these indexes, a core API is deployed that provides all the functionalities to search, explore and exploit the collected semantic documents. This API also links to the *Revyu.com* Semantic Web based reviewing system to allow users to rate and publish reviews on ontologies.

Different sources are used by the crawler of Watson to discover ontologies and semantic data: Google, Swoogle[4], PingTheSemanticWeb[5], manually submitted URLs. Specialised crawlers were designed for these repositories, extracting potential locations by sending queries that are intended to be covered by a large number of ontologies. For example, the keyword search facility provided by search engines such as Swoogle and Google is exploited with queries containing terms from the top most common words in the english language. Another crawler heuristically explores Web pages to discover new repositories and to locate documents written in certain ontology languages, by including "filetype:owl" in a query to Google. Finally, already collected semantic documents are frequently re-crawled, to discover evolutions of known semantic content or new elements at the same location.

Once located and retrieved, these documents are filtered to keep only the elements that characterise the Semantic Web. In particular, to keep only the documents that contain semantic data or ontologies, the crawler eliminates any document that cannot be parsed by Jena[6]. In that way, only valid RDF-based documents are considered. Furthermore, a restriction exists which imposes that all RDF based semantic documents be collected with the exception of RSS. The reason to exclude these elements is that, even if they are described in RDF, RSS feeds represent semantically weak documents, relying on RDF Schema more as a way to describe a syntax than as an ontology language.

Many different elements of information are extracted from the collected semantic documents: information about the entities and literals they contain, about the employed languages, about the relations with other documents, etc. This requires analysing the content of the retrieved documents in order to extract relevant information (metadata) to be used by the search functionality of Watson.

Besides trivial information, such as the labels and comments of ontologies, some of the metadata that are extracted from the collected ontologies influence the way Watson is designed. For instance, there are several ways to declare the URI of an ontology: as the namespace of the document, using the `xml:base` attribute, as the identifier of the ontology header, or

[2] `http://crawler.archive.org/`
[3] `http://lucene.apache.org/`
[4] `http://swoogle.umbc.edu`
[5] `http://pingthesemanticweb.com/`
[6] `http://jena.sourceforge.net/`

even, if it is not declared, as the URL of the document. URIs are supposed to be identifiers in the scope of the Semantic Web. However, two ontologies that are intended to be different may declare the same URI. For these reasons, Watson uses internal identifiers that may differ from the URIs of the collected semantic documents. When communicating with users and applications, these identifiers are transformed into common, non-ambiguous URIs from the original documents.

Another important step in the analysis of a semantic document is to characterise it in terms of its content. Watson extracts, exploits, and stores a large range of declared metadata or computed measures, such as the employed languages/vocabularies (RDF, RDFS, OWL, DAML+OIL), information about the contained entities (classes, properties, individuals and literals), or measures concerning the richness of the knowledge contained in the document (e.g., the expressiveness of the employed language, the density of the class definitions, etc.) These elements are then stored and exploited to provide quality related filtering, ranking and analysis of the collected semantic content.

3. Watson as a Semantic Web search engine

Even if the first goal of Watson is to support semantic applications, it is important to provide Web interfaces that facilitate access to ontologies for human users. Users may have different requirements and different levels of expertise concerning semantic technologies. For this reason, Watson provides different 'perspectives', from the most simple keyword search, to complex, structured queries using SPARQL (see figure 2).

The keyword search feature of Watson is similar in its use to usual Web or desktop search systems (see figure 2(a)). The set of keywords entered by the user is matched to the local names, labels, comments, or literals of entities occurring in semantic documents. A list of matching ontologies is then displayed with, for each ontology, some information about it (languages, size, expressivity of the underlying description logic) and the list of entities matching each keyword. The search can also be restricted to consider only certain types of entities (classes, properties, individuals) or certain descriptors (labels, comments, local names, literals).

One principle applied to the Watson interface is that every URI is clickable. A URI displayed in the result of the search is a link to a page giving the details of either the corresponding ontology or a particular entity. Since these descriptions also show relations to other elements, this allows the user to navigate among entities and ontologies. It is therefore possible to explore the content of ontologies, navigating through the relations between entities (displayed as a list of relations –Figure 2(b)– or a graph –Figure 2(c)), as well as to inspect ontologies and their metadata.

In order to facilitate the assessment and selection of ontologies by users, it is crucial to provide easy to read and understand overviews of ontologies, both at the level of the automatically extracted metadata about them, as well as at the level of their content. For each collected semantic document, Watson provides a page that summarises essential information such as the size of the document (in bytes, triples, number of classes, properties and individuals), the languages used (OWL, RDF-S and DAML+OIL, as well as the underlying description logic), the links with other documents (through imports) and the reviews from users of Watson (see Figure 2(d)). Watson also generates small graphs (Figure 2(e)), showing the 6 first key-concepts of each ontology and an abstract representation of the existing relations between these concepts, based in the key concept extraction method described in [21].

Finally, a SPARQL endpoint has been deployed on the Watson server and is customisable to address a selected semantic document to be queried. A simple interface allows to enter a SPARQL query and to execute it on the selected semantic document (Figure 2(f)). This feature can be seen as the last step of a chain of selection and access tasks using the Watson Web interface. Indeed, keyword search and ontology exploration allow the user to select the appropriate semantic document to be queried.

4. Building Semantic Web applications with Watson

As explained above, the focus of Watson is on implementing an infrastructure component, a gateway, for applications to find, access and exploit ontologies and semantic data published online. To achieve this, Watson implements a set of APIs giving access to its functionalities through online services (see Figure 3).

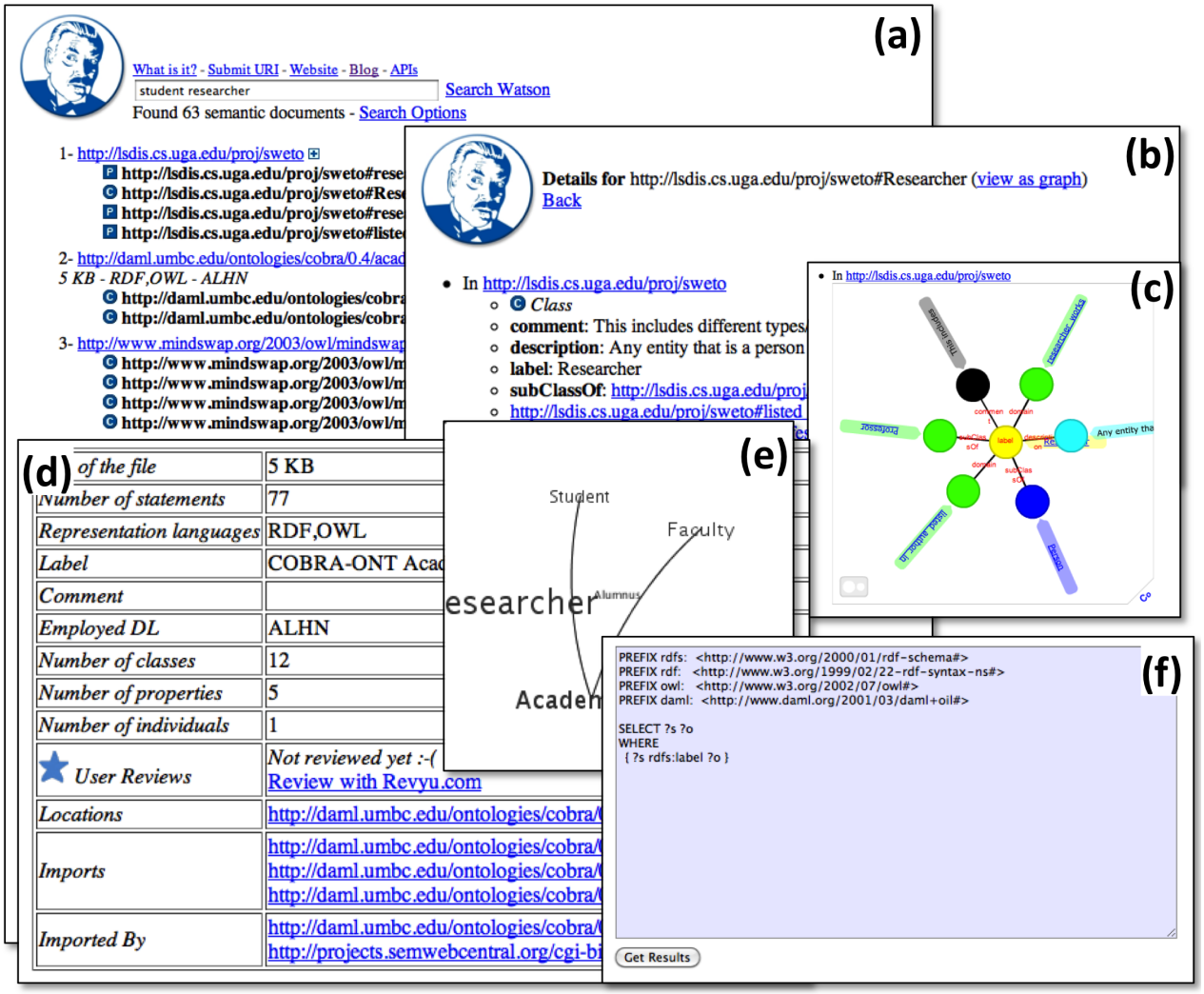

Fig. 2. Overview of the Watson Web interface.

4.1. The Watson APIs

The most commonly used and complete API to Watson is a Java library, giving remote access to the many functions of Watson through a set of SOAP services[7]. The basic design requirements for these APIs is that they should allow applications to exploit ontologies online, which they might have to identify at runtime, while not having to download these ontologies and the corresponding data, or to implement their own infrastructure for handling, accessing and exploring them. More precisely the Watson Java/SOAP API gives access, through three different services and in a lightweight (for the application) way to functions related to:

Searching ontologies and semantic documents: Using keywords and restrictions, related for example to the type of entities (classes, properties, individuals) the keywords should match to, or the place where they can match (name, label, comment or other literals in the entity), these functions allow to locate ontologies that relate to a particular domain and contain particular concepts.

[7] http://watson.kmi.open.ac.uk/WS_and_API-v2.html

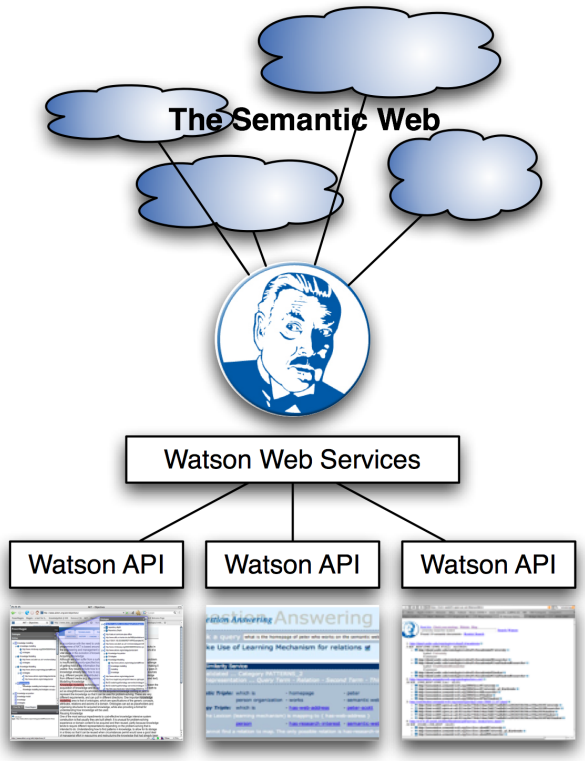

Fig. 3. Using the Watson API to build applications that exploit the Semantic Web as source of knowledge.

Searching in ontologies and semantic documents: Similarly, using keywords and restrictions, functions are provided to identify, within an given ontology or semantic document, the entities that match the given keywords.

Retrieving metadata about an ontology: Many functions are implemented that allow to characterise a particular ontology through automatically generated metadata (such as the languages used, the size, labels and comments, imported ontologies, etc.), as well as evaluations from users of Watson.

Retrieving metrics on ontologies and entities: Measures are provided in a dedicated service regarding ontologies and entities (e.g., depth of the hierarchy of an ontology or 'density' of an entity in terms of connections). This allows applications to define filters and selection criteria ensuring certain characteristics from the elements they exploit.

Exploring the content of ontologies: Functions are provided that allow an application to access the content of an ontology, through exploring its entities and their connections. These functions include the possibility to ask for the subclasses of an RDF, DAML or OWL class in any of the collected ontologies, the labels of a given entity or any relation 'pointing to' a given individual. Some of these functions are also available in a variant providing basic level reasoning, in order to, for example, obtain all the subclasses of a class, i.e., both the directly declared ones and the ones inferred from the transitivity of the subclass relation.

SPARQL Querying: In case the functions provided are not sufficient, and complex queries are required, a SPARQL query can be executed directly from the API, or alternatively by using the deployed SPARQL endpoint.

[6] provides an example of a simple, lightweight application using the Watson API overviewed above to achieve some basic ontology-based query expansion mechanisms for a search engine. This application simply suggests to the user keywords that are either more general or more specific than the ones used, by querying Watson for the subclasses and superclasses of corresponding classes in online ontologies. While being only a basic demonstrator, this application shows one of the key contributions of Watson: giving applications the ability to efficiently make use of large scale semantics in an open domain. More advanced examples are described in the next section.

Similarly to the Java/SOAP API, Watson also provides a set of REST services/APIs[8] including a subset of Watson's functionality. This API is more convenient (and more often used) in dynamic Web applications using scripting languages such as JavaScript (an example of such an application is described in [18]).

4.2. Derived tools and example applications

The Watson APIs described above as been developed for, and following the requirements of, new Semantic Web applications exploiting online knowledge. Many of such applications have been developed, with varying degrees of complexity and sophistication, mostly by research groups involved in the Semantic Web area. Details of several of them are available in

[8] http://watson.kmi.open.ac.uk/REST_API.html

the two papers [12] and [15].

Probably one of the most obvious applications of an automated ontology search mechanism is for ontology engineering itself. Related to this task, the Watson plugin [14] is an extension of an ontology editor (namely, the NeOn toolkit[9]) that allows the ontology engineer to check for knowledge included in online ontologies, and reuse part of them while building a new ontology. It can integrate statements from ontologies discovered by Watson and keep links between the created ontology and the elements reused by generating mappings connecting entities on the local ontologies with the ones of identified, online ontologies.

Staying in the Semantic Web domain, Scarlet[10] follows the paradigm of automatically selecting and exploring online ontologies to discover relations between two given terms, as a way of realising ontology matching [23]. It achieves this by finding, through Watson, ontologies that contain relations between the two given terms, possibly exploring multiple ontologies and deriving a relation from complete paths across different ontologies. When evaluated on two large scale agriculture thesauri, Scarlet demonstrated good precision, while using hundreds of external ontologies identified at run-time.

Using PowerAqua[11], a user can simply ask a question, such as "Who are the members of the rock band Nirvana?" and obtain an answer, in this case in the form of a list of musicians (Kurt Cobain, Dave Grohl, Krist Novoselic and other former members of the group). The main strength of PowerAqua resides in the fact that this answer is derived dynamically from the relevant data available on the Semantic Web, as discovered and explored through Watson.

Garcia et al. in [17] exploit Watson to tackle the task of word sense disambiguation (WSD). Specifically, they propose a novel, unsupervised, multi-ontology method which 1) relies on dynamically identified online ontologies as sources for candidate word senses and 2) employs algorithms that combine information available both on the Semantic Web and the Web in order to integrate and select the right senses. They have shown in particular that the Semantic Web provides a good source of word senses that can complement traditional resources such as WordNet[12].

Also, in [22], the authors use Watson in a sophisticated process to gather information about people and include this information in an integrated way into a learning mechanism for the purpose of identifying Web citations. In [20], Watson is used as a main source of knowledge for annotating Web services from their documentation published online.

Finally, the Watson engine itself is at the basis of the Cupboard[13] ontology publishing tool, which provides functionalities to expose, promote, evaluate and reuse ontologies for the Semantic Web community [10].

These are only brief descriptions of a few of the applications developed so far that rely on the Watson platform. Moreover, as described in the next section, Watson is also used within research processes, as a way to obtain corpora of real-life, online formalised knowledge for analysis and tests.

5. Using Watson as a research platform

A Semantic Web search engine such as Watson is not only a service supporting the development of Semantic Web applications. It also represents a unprecedented resource for researchers to study the Semantic Web, and more specifically, how formalised knowledge and data are produced, shared and consumed online [8].

For example, to give an account of the way semantic technologies are used to publish knowledge on the Web, of the characteristics of the published knowledge, and of the networked aspects of the Semantic Web, [9] presented an analysis of a sample of 25,500 semantic documents collected by Watson. This analysis looked in particular into the use of Semantic Web languages and of their primitives. One noticeable fact that was derived from analysing both the OWL (version 1) species and the description logics used in ontologies is that, while a large majority of the ontologies in the set were in OWL Full (the most complex variant of OWL 1, which is undecidable), most of them were in reality very simple, only using a small subset of the

[9] http://neon-toolkit.org
[10] http://scarlet.open.ac.uk/
[11] http://kmi.open.ac.uk/technologies/poweraqua/
[12] http://wordnet.princeton.edu
[13] http://cupboard.open.ac.uk

primitives offered by the language (95% of the ontologies where based on the ALH(D) description logic).

More recently, research work has been conducted using Watson as a corpus to detect and study various implicit relationships between ontologies and semantic documents on the Web [2]. For example, in [7], we introduced fine-grained measures of agreement and disagreement between ontologies, which were tested on real-life ontologies collected by Watson. We also derived from agreement and disagreement, measures of consensus and controversy regarding particular statements, within a large collection of ontologies, such as the one of Watson. Indeed, an implementation of such measures allowed us to build a tool indicating the level of consensus and controversy that exist on a given statement with respect to online ontologies. We recently integrated this tool in the NeOn toolkit ontology editor, as a way to provide an overview of the developed ontology with respect to its agreement with other, online ontologies [11].

Many other aspects of online ontologies can also be considered for study, including for example how ontologies evolve online [1], as well as for testing new techniques and approaches applicable to ontologies at large. In [5] for example, Watson is used to provide ontologies where 'anti-patterns' can be identified. In a more systematic manner, in [13], the Watson API is used to constitute groups of ontologies with varying characteristics to be used to benchmark semantic tools on resource-limited devices. In such a case, the large number and variety of ontologies online represent an advantage for testing systems and technologies.

6. Related Work

There are a number of systems similar to Watson, falling into the category of Semantic Web search engines. However, Watson differs from these systems in a number of ways, the main one being that Watson is the only tool to provide the necessary level of services for applications to dynamically exploit Semantic Web data. Indeed, we can mention the following systems:

Swoogle has been one of the first and most popular Semantic Web search engine [16]. It runs an automated hybrid crawl to harvest Semantic Web data from the Web, and then provide search services for finding relevant ontologies, documents and terms using keywords and additional semantic constraints. In addition to search, Swoogle also provides aggregated statistical metadata about the indexed Semantic Web documents and Semantic Web terms.

Sindice[14] is a *Semantic Web index* or *entity look-up service* that focuses on scaling to very large quantities of data. It provides keyword and URI based search, structured-query and rely on some simple reasoning mechanisms for inverse-functional properties [25].

Falcons[15] is a keyword-based semantic entity search engine. It provides a sophisticated Web interface that allows to restrict the search according to recommended concepts or vocabularies [4].

SWSE[16] is also a keyword-based entity search engine, but that focuses on providing semantic information about the resulting entities rather than only links to the corresponding data sources [19]. Its collection is automatically gathered by crawlers. SWSE also provides a SPARQL endpoint enabling structured query on the entire collection.

Semantic Web Search[17] is also a semantic entity search engine based on keywords, but that allows to restrict the search to particular types of entities (e.g. DOAP Projects) and provides structured queries.

OntoSelect[18] provides a browsable collection of ontologies that can be searched by looking at keywords in the title of the ontology or by providing a topic [3].

OntoSearch2[19] is a Semantic Web Search engine that allows for keyword search, formal queries and fuzzy queries on a collection of manually submitted OWL ontologies. It relies on scalable reasoning capabilities based on a reduction of OWL ontologies into DL-Lite ontologies [24].

Sqore[20] is a prototype search engine that allows for structured queries in the form of OWL descriptions [26]. Desired properties of entities to be found in ontologies are described as OWL entities and the engine searches for similar descriptions in its collection.

Among these, Sindice for example, is one of the most popular. However, while Sindice indexes a very large amount of semantic data, it only provides a simple look-up service allowing applications/users to 'locate' semantic documents. Therefore, it is still necessary to download and process these documents locally to exploit them, which in many cases, is not feasi-

ble. The Swoogle system is closer to Watson, but does not provide some of the advanced search and exploration functions that are present in the Watson APIs (including the SPARQL querying facility). The Falcons Semantic Web search engine has been focusing more on the user interface aspects, but now provides an initial API including a sub-set of the functions provided by Watson. The other systems focus on a restricted set of scenarios or functionalities (e.g., annotation and language information in OntoSelect), and have not been developed and used further than as research prototypes.

Another important aspect to consider is how open Semantic Web Search engines are. Indeed, Watson is the only Semantic Web search engine to provide unlimited access to its functionalities. Sindice, Swoogle and Falcons are, on the contrary, restricting the possibility they offer by limiting the number of queries executable in a day or the number of results for a given query.

Finally, it is worth noticing that the issue of collecting semantic data from the Web has recently reached a broader scope, with the appearance of features within mainstream Web search engine exploiting structured data to improve the search experience and presentation. Indeed, Yahoo! SearchMonkey[21] crawls and indexes semantic information embedded in webpages as RDFa[22] or microformats[23], in order to provide enriched snippets describing the webpages in the search results. Similarly, Google Rich Snippets[24] makes use of collected semantic data using specific schemas in webpages to add information to the presentation of results. Watson currently focuses on individual RDF documents and does not index embedded formats such as RDFa. Such an extension if planned to be realised in a near future.

7. Future work and planned functionalities

With many users, both humans and applications[25], and several years of development (the very first version was released in 2007), Watson is now a mature system that provides constant services, with very rare down times. It has evolved based on the requirements, requests and feedbacks from the community of developers using it in many different applications.

Of course, Watson is still being developed, including an ever growing index of semantic documents from the Web. New specialised indexes have been created recently, with manually initiated crawls of large linked data nodes such as DBPedia[26]. Also, the Cupboard system mentioned earlier is an ontology publication platform based on the engine of Watson, which means that any document submitted to it is being indexed using the same process as the one of Watson, and is made available through compatible APIs. A 'federated service' where different instances of Watson, including the current system and Cupboard, can be connected in order to return aggregated results has been developed and is currently being tested. This means in particular that user will be soon able to contribute ontologies to the Watson collection in real time, shortcutting the crawler, by simply submitting them the Cupboard.

New functionalities are also being considered. In particular, Watson includes a minimalistic evaluation mechanism through the connection with *Revyu.com*. Many refinements could be imagined, from simple integrations with social platforms (e.g., a Facebook 'like' button for ontologies) to monitoring and keeping the connections between communities behind ontologies, and the communities using particular ontologies. This requires extensive research on the social aspects of ontologies and how to keep track of them, but would ultimately improve the ability of Watson to support users in selecting ontologies.

An important characteristic of ontologies is that they are not isolated artefacts. They are related to each other in a network of semantic relations. However, apart from exceptions (noticeably, import), these relations are mostly kept implicit. Extensive research work is

[21] http://developer.yahoo.com/searchmonkey/
[22] http://www.w3.org/TR/xhtml-rdfa-primer/
[23] http://microformats.org/
[24] http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html

[25] There is currently about 2,500 queries and 8,000 pages viewed per month on the Watson user interface. The activities of applications using the Watson APIs cannot be traced, but is believed to generate a significantly greater number of requests than the user interface, as some of the applications described earlier in this paper can make several thousands of calls to the APIs in a very short time.
[26] http://dbpedia/org

being carried out currently on formalising such relations (e.g., inclusion, versioning, similarity, see [2]) and deploying efficient methods to detect them in a large scale collection such as the one of Watson, as well as to evaluate the benefit of structuring search results on the basis of ontology relations, for a more efficient ontology selection approach.

References

[1] Carlo Allocca, Mathieu d'Aquin, and Enrico Motta. Detecting different versions of ontologies in large ontology repositories. In *International Workshop on Ontology Dynamic, IWOD 2009 at ISWC*, 2009.

[2] Carlo Allocca, Mathieu d'Aquin, and Enrico Motta. DOOR: Towards a Formalization of Ontology Relations. In *Proc.of International Conference on Knowledge Engineering and Ontology Development (KEOD)*, 2009.

[3] Paul Buitelaar, Thomas Eigner, and Thierry Declerck. Ontoselect: A dynamic ontology library with support for ontology selection. In *Proc. of the Demo Session at the International Semantic Web Conference*, 2004.

[4] Gong Cheng, Weiyi Ge, and Yuzhong Qu. Falcons: searching and browsing entities on the semantic web. In *WWW conference*, pages 1101–1102. ACM, 2008.

[5] Oscar Corcho, Catherine Roussey, Francois Scharffe, and Vojtech Svatek. SPARQL-based detection of anti-patterns in OWL ontologies. In *EKAW 2010 Conference - Knowledge Engineering and Knowledge Management by the Masses, Poster session*, 2010.

[6] Mathieu d'Aquin. Building Semantic Web Based Applications with Watson. In *WWW2008 - The 17th International World Wide Web Conference - Developers' Track*, 2008.

[7] Mathieu d'Aquin. Formally Measuring Agreement and Disagreement in Ontologies. In *International Conference on Knowledge Capture - K-CAP 2009*, 2009.

[8] Mathieu d'Aquin, Carlo Allocca, and Enrico Motta. A platform for semantic web studies. In *Web Science Conference, poster session*, 2010.

[9] Mathieu d'Aquin, Claudio Baldassarre, Laurian Gridinoc, Sofia Angeletou, Marta Sabou, and Enrico Motta. Characterizing Knowledge on the Semantic Web with Watson. In *Evaluation of Ontologies and Ontology-based tools, 5th International EON Workshop at ISWC 2007*, 2007.

[10] Mathieu d'Aquin, Jérôme Euzenat, Chan Le Duc, and Holger Lewen. Sharing and reusing aligned ontologies with cupboard. In *Demo, International Conference on Knowledge Capture - K-CAP 2009*, 2009.

[11] Mathieu d'Aquin and Enrico Motta. Visualising Consensus with Online Ontologies to Support Quality in Ontology Development (submitted). In *EKAW 2010 Workshop on Ontology Quality (to appear)*, 2010.

[12] Mathieu d'Aquin, Enrico Motta, Marta Sabou, Sofia Angeletou, Laurian Gridinoc, Vanessa Lopez, and Davide Guidi. Toward a new generation of semantic web applications. *Intelligent Systems*, 23(3):20–28, 2008.

[13] Mathieu d'Aquin, Andriy Nikolov, and Enrico Motta. How much Semantic Data on Small Devices? In *EKAW 2010 Conference - Knowledge Engineering and Knowledge Management by the Masses*, 2010.

[14] Mathieu d'Aquin, Marta Sabou, and Enrico Motta. Reusing knowledge from the semantic web with the watson plugin. In *Demo, International Semantic Web Conference, ISWC 2008*, 2008.

[15] Mathieu d'Aquin, Marta Sabou, Enrico Motta, Sofia Angeletou, Laurian Gridinoc, Vanessa Lopez, and Fouad Zablith. What can be done with the Semantic Web? An Overview of Watson-based Applications. In *5th Workshop on Semantic Web Applications and Perspectives, SWAP 2008*, 2008.

[16] Li Ding, Tim Finin, Anupam Joshi, Rong Pan, R. Scott Cost, Yun Peng, Pavan Reddivari, Vishal C Doshi, and Joel Sachs. Swoogle: A search and metadata engine for the semantic web. In *CIKM'04: the Proceedings of ACM Thirteenth Conference on Information and Knowledge Management*, 2004.

[17] Jorge Garcia and Eduardo Mena. Overview of a semantic disambiguation method for unstructured web contexts. In *Proceedings of the fifth international conference on Knowledge capture table of contents,K-CAP 2009, Poster session*, 2009.

[18] Laurian Gridinoc and Mathieu d'Aquin. Moaw - uri's everywhere. In *4th Workshop on Scripting for the Semantic Web (challenge entry), ESWC 2008*, 2008.

[19] Andreas Harth, Aidan Hogan, Renaud Delbru, J?rgen Umbrich, Se?n O'Riain, and Stefan Decker. Swse: Answers before links! In *Semantic Web Challenge*, volume 295 of *CEUR Workshop Proceedings*, 2007.

[20] Maria Maleshkova, Carlo Pedrinaci, and John Domingue. Semantic Annotation of Web APIs with SWEET. In *6th Workshop on Scripting and Development for the Semantic Web at ESWC 2010*, 2010.

[21] Silvio Peroni, Enrico Motta, and Mathieu d'Aquin. Identifying key concepts in an ontology through the integration of cognitive principles with statistical and topological measures. In *Proceedings of the Third Asian Semantic Web Conference, ASWC 2008*, 2009.

[22] Matthew Rowe and Fabio Ciravegna. Disambiguating Identity Web References using Web 2.0 Data and Semantics. *Journal of Web Semantics*, 2010.

[23] Marta Sabou, Mathieu d'Aquin, and Enrico Motta. Exploring the semantic web as background knowledge for ontology matching. *Journal of Data Semantics*, 2008.

[24] Edward Thomas, Jeff Z. Pan, and Derek H. Sleeman. Ontosearch2: Searching ontologies semantically. In *OWLED workshop*, volume 258 of *CEUR Workshop Proceedings*, 2007.

[25] Giovanni Tummarello, Eyal Oren, and Renaud Delbru. Sindice.com: Weaving the open linked data. In *Proceedings of the 6th International Semantic Web Conference and 2nd Asian Semantic Web Conference (ISWC/ASWC2007), Busan, South Korea*, volume 4825 of *LNCS*, pages 547–560, Berlin, Heidelberg, November 2007. Springer Verlag.

[26] Rachanee Ungrangsi, Chutiporn Anutariya, and Vilas Wuwongse. Sqore-based ontology retrieval system. In *DEXA conference*, volume 4653 of *Lecture Notes in Computer Science*, pages 720–729. Springer, 2007.