External Semantic Annotation of Web-Databases

Benjamin Dönz, Dietmar Bruckner
Institute of Computer Technology, Technical University of Vienna
{doenz, bruckner}@ict.tuwien.ac.at

***Abstract*-This paper presents a new solution for making existing web-databases accessible for semantic agents. Instead of adding semantic annotations to a page itself or publishing the content separately, the elements of a web-page are annotated and published in an external file, maybe even on an external site. A semantic agent can use these annotations like an instruction manual to find out how to interpret the page's content and how to access databases such as product catalogues, price lists or technical specifications for components. Since these annotations can be published separately from the actual page, anyone can annotate any website and make it accessible for semantic agents.**

I. Introduction

In 2000, Tim Berners-Lee published his vision of the future of the Internet [1]. Since then, large research efforts have been made to push the Internet to its next logical evolutional stage – the Semantic Web. Here, information is not only available for humans, but is also interpretable for tools and agents. The World Wide Web Consortium, of which Tim Berners-Lee is currently the Director, has published the Semantic Web stack [2] shown in Fig. 1: a layered stack of technologies and standards that resemble a road-map to realize this vision. Today, standards in the form of W3C Recommendations exist for all the layers up to the Ontology Web Language (OWL), and have also been put to practical use: For example in projects like DBpedia[1] where information is extracted from Wikipedia and published as RDF[2] or GoPubMed[3] where biomedical research articles are published.

But the Semantic Web has not yet left the bounds of academia and only very few early adopters in the industry, e.g. the BBC [3], publish data for the Semantic Web. In fact, a web-crawl done by the authors of this work [4] that crawled the top and second level pages of all 121.000 company homepages published in the Austrian yellow pages in April 2011 (1.300.000 pages total) has shown, that only about 3.2% of the sites contained any RDFa tags at all. For most companies, the cost-benefit calculation of publishing information for the semantic web does not work out as long as there are no services that actually make use of them. And services will only be developed when enough content is available. This results in a chicken-and-egg problem [5] that makes it hard for the Semantic Web to get off the ground.

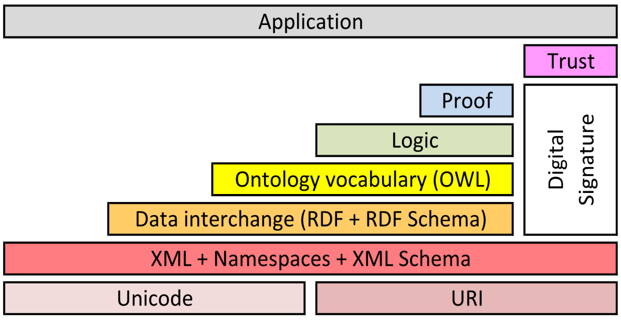

Fig. 1. Semantic Web Stack [2]. *Abbreviations: Ontology Web Language (OWL), Resource Description Framework (RDF), Uniform Resource Identifier (URI), eXtensible Markup Language (XML)*

For human users on the other hand, the Internet offers an immense amount of data [6] and has become a primary information source for many subjects. The usual way to find a desired piece of information is to use a search engine like Google or Bing. But these can only properly index normal document-style webpages. Already in 2001, Ref. [7] have shown that the so called "Deep Web" or "Hidden Web", which is publically available content that is "hidden" in databases behind search forms, is 400 to 550 times larger than the commonly defined WWW [7]. Only in 2008 Google has started to develop strategies to "surface" information from this vast source: in Ref. [8], employees of Google describe their solution to automatically extract information from databases by creating queries with sets of keywords depending on the apparent topic of the site. But this method can only extract parts of the data in the database. According to Ref. [8] the success depends mainly on the size of the database and was as low as 20% in the documented results.

We believe that these web-databases could be a great opportunity to show the strengths of the Semantic Web. If a practical solution can be found to make these databases usable for semantic agents, there would be no need to index all of its content, because it could be accessed on demand. And the prospect of having 400-550 times more information available than current search engines can access would instantly solve the chicken-and-egg problem and make it worthwhile to develop tools and services for the Semantic Web.

In this paper we present a practical solution to semantically annotate existing websites and databases so semantic agents can use and access them to extract the information they are seeking. The method can be applied for multiple applications from purely private use to B2B solutions - contributing to the Digital Ecosystem and allowing businesses to interact in an open loosely coupled, agent driven environment [9]

II. Related Work

The W3C also made the task of making existing databases available for the Semantic Web one of its goals and initiated

[1] DBpedia: http://dbpedia.org
[2] Resource Description Framework: http://www.w3.org/RDF
[3] GoPubMed: http://www.gopubmed.org

978-1-4673-0157-2/12/$31.00 ©2012 IEEE

the RDB2RDF Incubator Group. In the course of their work in 2009, they collected and evaluated the state of the art in this field and published their final report in Ref. [10]. This survey showed several approaches – from a complete transformation of an existing relational database to an RDF database on one side, to on-demand mapping and query translations from SPARQL to SQL on the other side.

When publishing data that has to be maintained and updated over time, it is impractical to have to publish the same information twice – once for humans and a second time for tools, especially if content is also provided by users of the site. In this case, it is not just a single effort when initially publishing the data, but results in consequently having to update and maintain two separate pieces of work. A practical and easy way to integrate semantic information into an existing document is to use annotations. A first solution for this was proposed by [10] who described the concept of miroformats. These are small sets of semantic data that can be embedded in a webpage, invisible to the user but visible for tools and search engines. By using this data, structured information about things like authorship or even cooking recipes can be given that can be extracted from the page. But these formats have to be agreed on by the community in order to understand the structure and the content. A list of microformats can be found online on the portal microformats.org.

The W3C adopted the idea of annotations quickly and published the annotation standard RDFa (Resource Description Framework - in - Annotations) which became a W3C recommendation in 2008[4]. In contrast to microformats, statements given in RDFa do not need a pre-defined vocabulary everyone has to agree upon to make them useful. In RDF (and consequently in RDFa), any vocabulary can be used and defined in an ontology.

But if everyone can use a proprietary vocabulary, the obvious problem is, that semantic agents have to deal with a plethora of different ontologies. In order to combine information from two sources it is necessary to align these ontologies. This problem is still an active research area and a survey of current solutions can be found in Ref. [12] and [13] which include using word similarities or thesauruses to find possible alignments, e.g. WordNet[5]. But there are still many challenges to be solved: Ref. [14] published a list of what they have identified as the ten most important questions in ontology matching and there is even an annual contest held since 2004 and organized by the Ontology Alignment Evaluation Initiative[6] to evaluate progress.

However, if the ontologies are not isolated, but part of a network of information like the Internet, another approach becomes feasible that renders the process of "guessing" a possible alignment unnecessary: By referencing other ontologies with constructs like "same as" or "part of", the author of the ontology can link his classes to other published ontologies. By following these references, a semantic web agent can find an alignment to an ontology it "understands" resulting in a semantic bridge [15] which makes the proprietary vocabulary usable by the agent.

The Linked Data project[7] is currently the largest project connecting different ontologies and semantic knowledgebases on the Internet. There are ontologies ranging from common knowledge databases, e.g. the World Fact Book[8] or DBpedia[9] to scientific databases, e.g. PubChem[10] or CiteSeer[11]. According to [16], the complete graph of interlinked ontologies contained 4.7 billion RDF triples in 2008 and has surly risen since. When annotating a site with a proprietary vocabulary, linking to one of these already interconnected ontologies will therefore help an agent to find a semantic bridge.

Another problem for agents when analyzing a webpage made for humans, besides the vocabulary and content of a page, is the structure. In order to fill out forms or extract information, an agent must figure out which elements of the page belong together, which elements constitute an index or menu and which elements are just advertisements. Several solutions have been published exploiting the visual structure and the DOM (document object model) tree of a webpage [17]. In Ref. [18], the authors have recently proposed an integrated statistical model for both understanding the webpage structure and processing the natural language sentences in the HTML element for the purpose of information retrieval.

III. SCENARIO AND USE CASES

There are many types of databases available on the Internet. Even though we believe the concept can be applied to other areas, we defined the following scenario that can be compared to existing solutions as a means to evaluate the feasibility of the concept: a product search and price comparison. In our evaluation scenario, a user is looking for an item he or she wants to buy online, vaguely described by the type of product and some properties. For example: A digital video camera for less than 300€ with good optical zoom and HD recording capability.

To answer this request, an agent has to perform following tasks: search for internet databases containing product information, e.g. online stores, magazines with product tests or newsgroups with user reports, and use these to extract possible products including additional information such as recording and zoom capability, user ratings and prices. The combined information from several sources is then filtered by removing items that do not meet the user's requirements, ranked by the evaluation criteria (user or product test ratings, optical zoom capability and price) and presented to the user.

[4] RDFa: http://www.w3.org/TR/rdfa-syntax/
[5] Wordnet: http://wordnet.princeton.edu
[6] OAEI: http://oaei.ontologymatching.org
[7] Linked Data: http://linkeddata.org
[8] See http://www.cia.gov/library/publications/the-world-factbook/
[9] DBpedia: http://dbpedia.org
[10] Pubchem: http://pubchem.ncbi.nlm.nih.gov
[11] CiteSeer: http://citeseer.ist.psu.edu

The actual ranking is not the primary focus of this work, so a simple algorithm is used that calculates the mean rank of the evaluation criteria, i.e. rank 1 in price, rank 5 in user test rating and rank 3 in optical zoom would result in a mean rank of 3. Pure performance is also not the focus of this work, so the agents will not utilize any pre-processed databases, but always start with an empty result database and perform all search operations online.

Several systems for searching and comparing products and prices exist that use conventional web-scrapers or proprietary webservices to update their databases. Prominent examples include PriceGrabber[12], PriceRunner[13] or Nextag[14]. The example scenario evaluates the capability of the system to search for information sources, extract data from web databases intended for human use and combine information from several heterogenic sources. We believe that these capabilities could be used for many other applications such as looking up technical specifications for components, exchanging price lists or catalogues and many other types of information. But the proposed evaluation scenario has the additional advantage of allowing a comparison to existing systems.

IV. Annotation and Information Extraction

The main focus of this work is to present a concept to make existing web databases accessible and usable for software agents by external semantic annotations. The whole setup therefore includes a semantic agent for information extraction, the concept for the annotation of web pages and ontologies containing the vocabulary.

A. Semantic Web Agent

The agent is initialized with a query that is resembled by a data structure containing filters, merging rules and ranking rules. With reference to the evaluation example, the user defines an object, e.g. "digital video camera", with the structure shown in Table I. Different websites will give different values for a property. The user must therefore also define how records should be merged: "none" states that two records with differing values will not be merged, e.g. two cameras with different names are treated as different cameras. Setting "minimum" or "maximum" will result in the agent using only the lowest or highest value found. And "most stated" or "average" will result in merging records and using appropriately calculated values.

The agent's main task is to populate this data structure by searching for relevant information sources and harvesting appropriate records. These are then filtered, merged and ranked according to the query and presented to the user. Fig. 2 shows a flow chart of the agent's functionality: the first task is to find possible information sources. This is done by using a predefined list of search engines the agent may use. The search engines are annotated like the actual web-databases and can therefore be treated equally. The results of this initial search are then used to find the actually desired information. Each site is accessed one by one and if the agent can find an annotation for the site, it attempts to call a query and harvest the results. The annotations are looked up in the following order: annotations contained in the page itself, annotations linked to by the page using a "link" tag in the pages header section and finally annotations contained in annotation databases. These databases use the page's URL as index and return an annotation file for the page if available. The agent can use several databases but they have to be known by the agent. A discovery process could be developed to render this predefined knowledge unnecessary.

TABLE I
QUERY DATA STRUCTURE

Property	Merge	Filter	Rank
Category	None	Digital Video Camera	
Name	None		
Price	Minimum	<300€	ascending
Optical zoom	Most Stated		descending
Rating	Average		descending

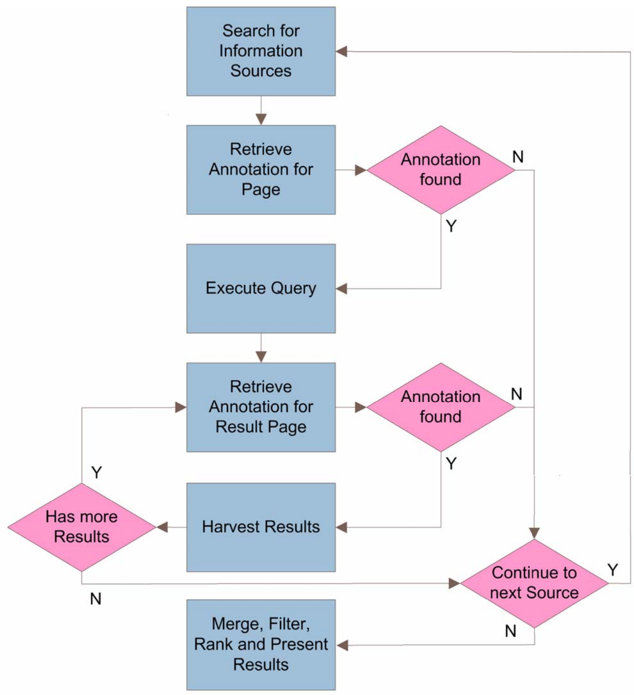

Fig. 2. Search Process Flow-Chart

If an annotation exists and the necessary content is available, i.e. a definition for query submission or usable data, the agent attempts to harvest the results by executing a query and extracting the data. The annotation of a query page must contain a definition of the search function including the fields where the necessary parameters should be entered and how to execute the function. Pages with relevant data must contain a description of the content, possibly referencing individual records or tables including the semantic meaning of columns or rows. Using the initial query, the agent will try to create a semantic bridge between a required property and a result column by using the chain of references from the

[12] PriceGrabber: http://www.pricegrabber.com
[13] PriceRunner: http://www.pricerunner.com
[14] Nextag: http://www.nextag.com

column annotation to the property definition. If relevant properties are found the records will be extracted and added to the result table.

After reaching a predefined termination condition, which may be a timeout or a maximum number of records, the agent will stop accessing new sites and evaluate the collected results by applying merging, filtering and ranking rules. The final table is then presented to the user.

B. *Annotation of Webpages*

One of the key principles in modern software engineering is the separation of concerns. This concept, described in Ref. [19], states that by separating the basic algorithm from special purpose concerns makes each of the parts easier to write, maintain and test. For example, the separation of data and layout of a webpage into a style-sheet[15] and the html content page has become an engineering practice: It not only allows changing the layout and content independently, but it also allows different people with different skill-sets and training to work on the webpage independently, e.g. a developer and a designer. This paradigm is also at the core of other state of the art and emerging programming paradigms like Service-Oriented Architecture [20] or Aspect-Oriented programming [21].

In the context auf semantic annotations, we propose to follow this scheme and separate the page and the annotations. This would hold many aspired properties: The original page would not be cluttered with additional tags, a separate tool could be used for the annotation process, and a page could also be annotated by a third party.

The external annotations can basically contain the same information as inline annotations, but have to include a reference to an html element in the original file. If the page itself cannot be changed (e.g. when it belongs to someone else), this makes it difficult to annotate individual words in a paragraph of text, but still allows to annotate structural elements like tables, fields, links, buttons and others. In contrast to most existing applications for the Semantic Web, the task of extracting information from web databases intended for human use not only involves interpreting data, but also calling functions on the page such as submitting a query to start the search or calling a function to retrieve the next 10 results on a result page. The first main distinction is therefore a separation of functions and data. For functions, the page description must contain the intent of the function, the element to use for executing the function (button or link) and a list of associated parameters including the fields where they should be entered.

For the annotation of the page we suggest a vocabulary of classes and properties described in Table II: beginning with a *pagedesc:Range* object referencing a section of the page, e.g. a div-tag or the whole body-tag, the content of this section is described by adding *pagedesc:Element* individuals. These can be used to reference buttons, fields, images and other elements of the page. The element is enriched with statements

TABLE II
RELEVANT PARTS OF THE PAGE DESCRIPTION META VOCABULARY

Class/Property	Description
Class: pagedesc:Element	Represents an html tag in the page and can be anything from a div-section to a table
Property: pagedesc:Referencepath	Represents a link to the element in the html file (in XPath syntax)
Class: pagedesc:Range	Subclass to pagedesc:Element representing a whole page or part of it and can contain descriptions and elements such as ranges, tables and functions contained in the page
Class: pagedesc:Table	Subclass to pagedesc:Element representing a table and containing a template of the table
Class: pagedesc:Tablecell	Represents a table cell
Class: pagedesc:Function	Represents a function on the page
Property: pagedesc:Functioncall	Range object to use for calling the function
Property: pagedesc:Functionparam	Range object containing a parameter of the function.

using other vocabularies in the usual manner that describe the actual semantic meaning of the content. The reference to the tag in the html page is done using the property *pagedesc:Referencepath* which uses the XPath[16] syntax to reference a tag in the html file and link it to the element.

A special subclasses of *pagedesc:Element* is *pagedesc:Table*: in modern programming languages such as Microsoft Silverlight®[17] and others, layout and data of the page are separated and the layout is defined by using a template which is then filled in during rendering with data from a data source. The *pagedesc:Table* class follows the same concept, but with the opposite intent of extracting records from a table to retrieve the original content. The *pagedesc:Table* therefore contains a layout template referencing rows and columns using the basic *pagedesc:Element* classes and using *pagedesc:Tablecell* individuals as placeholder for the actual content. These individuals are again enriched with other statements defining the semantic meaning of the cell. Using this template, the agent interprets every recurrence of the template in the *pagedesc:Table* as a separate record and extracts the content interpreting it by using the *pagedesc:Tablecells*.

The last of the basic classes used for the page description is the *pagedesc:Function* class. This class defines a function that can be executed on the page. The fields containing related parameters are referenced by *pagedesc:Functionparam* properties which are individuals of the class *pagedesc:Element*. The element that should be used for executing the function is defined using the *pagedesc:Functioncall* property which again references a *pagedesc:Element* individual which may be a link or a button. To define a search function for a database the *pagedesc:Functionparam* class is used to define fields where certain parameters should be entered and the button or link to execute the search is set as the *pagedesc:Functioncall* property of the function. An agent can then fill out the parameters referenced in this way and call the appropriate events to execute the query.

[15] Cascading style sheets: www.w3.org/Style/CSS

[16] XML Path Language: www.w3.org/TR/xpath

[17] Microsoft Silverlight: www.silverlight.net

C. Vocabulary

All of the terms used in the query, the agent functions and the annotations are defined in ontologies using the W3C standards RDF and OWL. The agent itself has a knowledge-base containing separate ontologies for functions and data, where the function ontology can reference the data ontology to define parameters. All the terms should be linked to one or more publically available ontologies published on the internet as suggested by the Linked Data project. This allows the agent to search for a semantic bridge between a term mentioned in an annotation file and its internal knowledgebase. The use of ontologies also allows creating taxonomies to differentiate between more abstract terms such as "rating" and more detailed terms such as "user rating" or "test result".

D. Tool Support

Storing the annotations in a separate file allows anyone to annotate any page and publish this information separately. The manual work, especially creating reference paths linking the annotation to the element in the page, is error prone when done manually. The annotation process can therefore be assisted by a tool that allows a user to click on an element in the page or select a region in the page and then define its meaning. Such a tool can also assist the user when creating the vocabulary for describing the page and its functions and help link this vocabulary to other ontologies published on the Internet. As already stated, these interrelations between ontologies are the key to allow an agent to find a semantic bridge between the statements in the query and the available information and functionality on a page.

V. CONCLUSION AND FUTURE WORK

In this paper we presented our concept for making existing web databases accessible for semantic agents and tapping these information sources currently not fully available for conventional search engines. The use of external semantic annotations allows a third party to create "instruction manuals" for semantic agents for any page and to publish this information for others to use. By using standardized Semantic Web languages and the Linked Data approach for defining all terms in the annotation, agents can find semantic bridges to their internal knowledgebase. This allows including heterogenic sources without the need for a predefined and coordinated vocabulary. The external annotation is also an application of the principle of separation of concerns and holds many benefits even if the annotation is done by the page's author and could be included in the source of the page itself.

Our next step will be to develop an annotation tool that helps to create the annotations and publish them in a database using the page description vocabulary presented in this paper. This will allow us to create annotations for several websites and evaluate the concept and compare it to existing recommendation and price comparison systems. Further applications targeting different types of databases can also be explored when this initial evaluation is completed.

REFERENCES

[1] T. Berners-Lee, *Weaving the Web*. HarperCollins Publishers, pp. 177-198, 1999

[2] T. Berners-Lee, Talk on Semantic Web Architecture. Retrieved October 26, 2011 from http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html, 2000

[3] G. Kobilarov et al, "Media Meets Semantic Web-How the BBC Uses DBpedia and Linked Data to Make Connections." *Lecture Notes in Computer Science*, Volume 554, pp. 723-737, 2009

[4] B.Dönz, A.Wendt, D.Bruckner, "Correlation of Link Counts Between Company Homepages and Official Economic Statistics", accepted and awaiting publication in the Proceedings of the IADIS International Conference e-Society 2012

[5] J. Hendler, "Web 3.0: Chicken Farms on the Semantic Web." *Computer*, Vol. 41, Issue 1, pp. 106-108, 2008

[6] "Internet data heads for 600bn gigabytes". Retrieved October 26, 2011 from www.guardian.co.uk/business/2009/may/18/digital-content-expansion, 2009

[7] M. Bergmann, "The Deep Web: Surfacing HiddenValue." *Journal of Electronic Publishing*, Vol. 7, Issue 1. Retrieved October 26, 2011 from http://hdl.handle.net/2027/spo.3336451.0007.104, 2001

[8] J. Madhayan et al, "Google's Deep-Web Crawl." *Proceedings of the VLDB Endowment*, Vol. 1, Issue 2, pp. 1241-1252, 2008

[9] H. Dong, F. K. Hussain, "Digital Ecosystem Ontology." *IEEE Conference on Emerging Technologies and Factory Automation*, pp. 814-817, 2007

[10] A Survey of Current Approaches for Mapping of Relational Databases to RDF. Retrieved October 28, 2011 from www.w3.org/2005/Incubator/ rdb2rdf/RDB2RDF_SurveyReport.pdf, 2005

[11] R. Khare, "Microformats: The Next (Small) Thing on the Semantic Web?" *IEEE Internet Computing*, Vol. 10, No. 1, pp. 68-75, 2006

[12] C. Namyoun et al, "A survey on ontology mapping." *ACM SIGMOD Record*, Vol. 35, Issue 3, pp. 34-41, 2006

[13] F. Lin and K. Sandkuhl, "A Survey Exploiting WordNet in Ontology Matching." *International Federation for Information Processing*, Vol. 276, pp. 341-350, 2008

[14] P. Shvaiko and J. Euzenat, "Ten Challenges for Ontology Matching." *Lecture Notes in Computer Science*, Vol. 5332, pp. 1164-1182, 2008

[15] A. Maedche et al, "MAFRA – A Mapping FRAmework for Distributed Ontologies." *Lecture Notes in Computer Science*, Vol. 2473, pp. 69-75, 2002

[16] C. Bizer et al, "Linked Data – The Story So Far". *International Journal on Semantic Web and Information Systems*, Vol. 5, Issue 3, pp. 1-22, 2009

[17] Q. Zhang, "Web Mining: A Survey of Current Research, Techniques and Software." *International Journal of Information Technology & Decision Making*, Vol. 7, Issue 4, pp. 683-720, 2008

[18] C. Yang et al, 2010, "Closing the Loop in Webpage Understanding." *IEEE Transactions on Knowledge Engineering*, Vol. 22, No. 5, pp. 639-650, 2010

[19] W.L. Hürsch and C.V. Lopes, "Separation of Concerns". *Technical Report NU-CCS-95-03*, Northeastern University, 1995

[20] A. Arsanjani et al, "S3: A Service-Oriented Reference Architecture". IT Professional, Vol. 9, Issue 3, pp. 10-17, 2007

[21] T. Elrad, R.E. Filman and A. Bader, "Aspect-oriented programming: Introduction", Communications of the ACM, Vol. 44, Issue 10, pp. 28-32, 2001