Wikicrawl: reusing semantic web data in authoring Wikipedia

Matthew Yau, Alexandra I. Cristea
Department of Computer Science
The University of Warwick, Coventry CV4 7AL
United Kingdom
C.Y.Yau@warwick.ac.uk, acristea@dcs.warwick.ac.uk

Abstract: This paper presents the main part of a project conducted at the University of Warwick regarding a tool for retrieving semantic web data and reusing the retrieved data in authoring Wikipedia pages. The goal of this tool is to enable semantic web crawling with a user friendly interface by applying a semantic web framework API to an existing web archiving system and an easy way for reusing the data in authoring by implementing a retrieved data delivery engine. To demonstrate this new tool, this paper presents an example scenario for retrieving semantic data from a specified domain and utilizing drag and drop method in authoring. The presented paradigm is evaluated and future work is briefly discussed.

Introduction

We are now in a high speed Internet technology developing era. Within recent years, the Internet has gone through stages from the previous simple data communication, static information presentation to a new concept, the Web 2.0 stage, with emphases on interaction, collaboration and involvement of end users [1]. Bart Decrem, the founder and former CEO of Flock, a company for developing a social web browser, calls Web 2.0 the "*participatory Web*"[2] and regards the *Web-as-information-source* as Web 1.0. E-learning applications rely heavily on information from the open web: resources all around the world, placed by practitioners and educational specialists everywhere.

Yet the evolution of the Internet has never stopped, and some new web technologies are now shaping the concept of Web 3.0. The Semantic Web is widely discussed as the main technology group in Web 3.0. Nowadays, more and more web pages are adopting semantic web technology, including the Resource Description Frame work (RDF) [3] and the Web Ontology Language (OWL) [4], in order to present data in a more structured way which is readable by machines, so that the machines can free the users from carrying out the repeated, tedious work of searching and organizing information, and can allow the user to combine and share information more easily.

One very popular information sharing web application is Wikipedia [5], which is the world's largest collaboratively edited source of encyclopedic knowledge. Most people have experience of gathering information from it, but only a few are willing to make some contributions in authoring the Wikipedia. The research [6] shows that 80% of articles in Wikipedia were contributed by 10% of Wikipedia user.

This is due to the fact that authoring in Wikipedia is not easy for the beginners or for people who lack web application related knowledge [7]. For one thing, it takes time to search for the information to insert, and organizing information can be a tremendous task, which can be quite time consuming. For another, the user needs to take extra time to learn the new language syntax ("Wikitext") in authoring wiki pages.

There is one tool to overcome this difficulty, which is called the CKeditor [8], a third party application that enables a WYSIWYG ('*What you see is what you get*') authoring interface for users to author the Wikipedia like in a normal text editor, without learning Wikitext. But this just partly addresses the issue why not many people are willing to contribute to wiki contents. The CKeditor saves users time in not requiring the learning of a new editing syntax, yet it does not make the information searching and combining work easier.

Reflection on these issues, connected as well to personal experiences in encountering difficulties, suggests there is space for some interesting development and research. As the semantic web is developing, it is interesting to explore whether there can be an application which crawls information on the Semantic RDF or on OWL pages, and extract semantic web data for authoring in Wikipedia. In this sense, the question arises of how to reuse the semantic web data in an efficient way, while saving users' time in selecting, combining and organizing data in a usable way. The goal is to eventually attract more users to make contributions in authoring

wiki pages by using semantic web data. These topics will be further discussed in the paper, as well as illustrated by the prototypes and some evaluation work.

Background research

This section introduces background concepts in web crawling and authoring in Wikipedia. This area of research brings together two streams. One is research related to the semantic web technology, and the other one is research on web crawlers for information retrieval [9]. We will briefly discuss the advantages and limitations of the current crawling tools.

Some key concepts are frequently mentioned in semantic web technology. Developed by the World Wide Web Consortium (W3C) [10], the Resource Description Framework (RDF)[3] is a standard to represent information about resources on the web. It enables structured and semi-structured data to be processed by machines rather than by end users. It is formed of a *subject-predicate-object* structure, known as triples in RDF to describe "something". "Something" could be any objects, resources or literal elements. OWL [4] is a web ontology language which is able to store machine interpretable web content in XML format. SPARQL [11], the query language for RDF/OWL, is considered a key semantic web technology. The SPARQL query syntax is working similarly to the SQL query language [12].

Currently, the Internet provides for a limited amount of data in RDF/OWL format. The search engine capable of retrieving such data is the semantic web equivalent of Google [13], the Swoogle [14] engine. A user can thus obtain semantic information from Swoogle, but the semantic web documents in RDF/OWL formats are difficult to read or use by humans (e.g. teachers, students) without any specific tools. If this information is needed for authoring, e.g. in Wikipedia, the situation is more complex. It is very likely that the user won't use the retrieved document as it is, but needs to search related information within a specific tag in the RDF/OWL file and then he'd have to copy and paste information repeatedly for Wikipedia authoring. This could possibly cost him a lot of time, especially if information is to come from different sources.

On the topic of information retrieval systems, there are several on-going projects which focus on tools for retrieving information. One project, called SIMILE [15], is led by MIT Libraries and MIT CSAIL. SIMILE aims to boost interoperability among digital assets, schemata/vocabularies/ontologies, metadata, and services, mainly by the application of RDF and semantic web techniques. One key system component in SIMILE is called Piggy Bank[16]. Piggy bank is designed for the Firefox browser, and allows the user to extract information from the web and then to save the information through the online server companion of Piggy Bank-Semantic Bank[17] for later reuse. In addition, the user can share information with other people by providing keywords tags. The main idea of the Piggy Bank API [16] is to allow users to capture information without copying and pasting the pages one by one. The user can use this tool while browsing the Internet and capture the current web page information and store it. All the information can be displayed in a browser, when the user specifies the keyword tags provided earlier. The advantage of the Piggy bank API is to allow the user to create a page with information mash-up from different RDF and HTML information sources. The user is not only able to share the collected web information, but also able to check what the other users have collected with the same keyword tags and combine information that he needs. This is a beneficial feature for collaboratively contributing information for the specified keywords. The disadvantage of the Piggy bank API is that it is only supported by the early version of Firefox 2.0. Furthermore, Piggy bank is not able to extract RDF/XML and RDF/N3 or HTML data from the web, unless the web site supports the Piggy Bank API. For the websites where RDF data is not available, it is requested to provide screen scrapers [18] to extract information from the HTML web. Screen scraper is a program code written by JavaScript [19] and the user needs to write it to define which parts of information he wants to extract from the HTML web. Yet there are only a limited number of screen scraper templates [20] available for extracting information from HTML web site. The user might find it time-consuming and difficult to define the template he wants to use.

Another web crawler project, Heritrix [21], is conducted by a non-profit digital library in the US called "Internet Archive". This is an open-source, extensible, web-scale, archival-quality web crawler project. It could capture information from different HTML web sites and store it into an ARC [22] file, which is a file format used for archiving web data, for which the user needs specific tools for processing. The main reason for this is that the ARC file format is not widely used in computer systems and there are not many programs supporting it. Therefore it is difficult for the user to reuse information retrieved by the crawler. And Heritrix does not support RDF/OWL web crawling, so Heritrix cannot be used directly as a tool for retrieving semantic web data.

As we can see from this short excursion into related work, as well as from further investigated tools [23] there is a large body of work for reusing semantic web data. However, those tools have limitations or different focus on practical applications. Slug [24], for example, is a web crawler system designed for retrieving semantic web content, but it is command-based and it neither provides any graphic user interface for crawling. nor any mechanism for users to reuse the data in authoring. Another system, called EachWiki [23], is mainly focused on suggesting links, categories and relations by semantic methods for authoring wiki pages, yet it hardly address the content authoring issue by releasing users from spending extra time on learning Wikitext syntax. Based on the investigation and analysis, no current tools are integrating the functionalities of crawling semantic open web data, storing them in an easily-accessible database and providing an easy-to-use user interface for authoring in Wikipedia, and thus aiming to eventually address the issue of attracting more users to contribute to building wiki pages.

Implementation Description

In this research, 2 prototypes were built. Based on the evaluation which is discussed in detail in next chapter, a 2nd prototype was built to reflect users' need and enhanced functionalities. It is named Wikicrawl. This 2nd prototype aims to retrieve any semantic web data needed by user (for example teachers, studenst) from the open web, and store it in a relational database which can be easily accessed and processed. Additionally, a delivering engine with a user friendly interface has been created, for users to select essential information for authoring Wikipedia articles for education.

The diagram below shows the system architecture of the improved prototype.

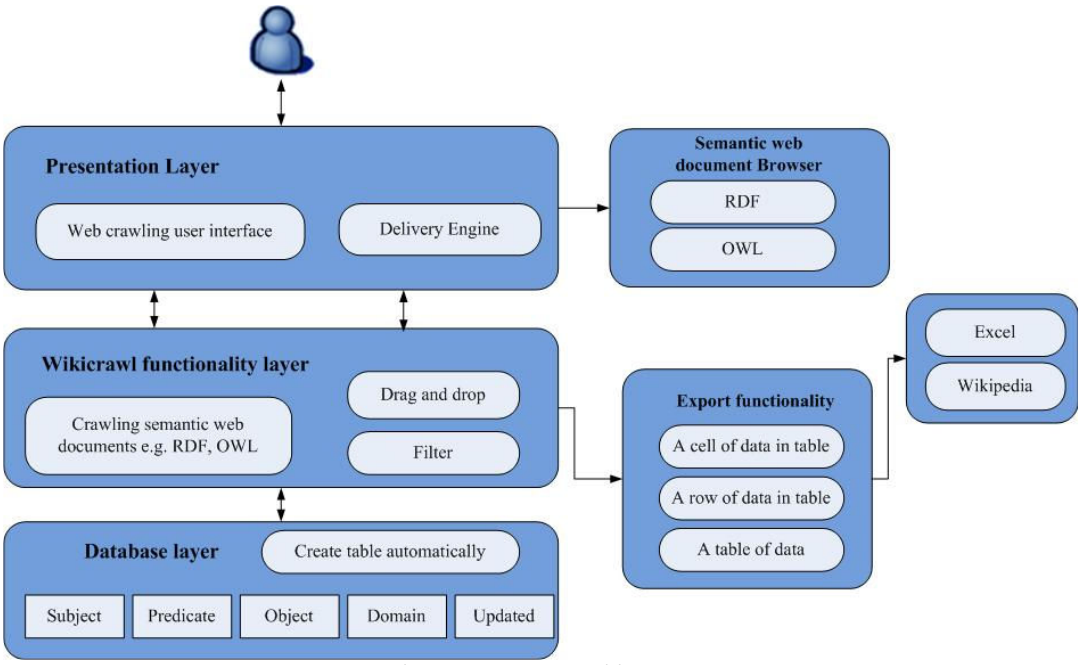


Figure 1: System architecture

There are three main layers in the system. In the *presentation layer*, the user can place the retrieval request via the Heritrix-based user interface and use the delivery engine for browsing the retrieved data. In the *Wikicrawl functionality layer*, semantic web data is retrieved and the user can select the data via drag and drop and filter the data down to the desired list of data. It provides an export data function to Wikipedia or Excel [25] for data archiving. In the *database layer*, the system is able to create tables for storage automatically in the triple syntax (Subject-Predicate-Object, which is also close to natural language and thus readable by humans).

There are two main modules which operate respectively as semantic web data crawler and delivery engine. The crawling system is an enhancement and extension of the current Heritrix web crawler with the JENA API [27] which is an open source API to process semantic web data in Java environment, and which supports SPARQL queries. The delivery engine uses Rico API [26] for browsing and exporting the retrieved information. Thus, the system can extract semantic web data and store it into the MYSQL database in triple format.

In order to understand better the workflow, the following scenario is provided.

Illustrative Scenarios

Scenario 1

Suppose a user (teacher, student) is editing a Wiki page on academic professionals from various universities, which is useful for both students and teachers to find out more about a certain reputed researcher and his novel publications. The user uses Wikicrawl to (semi)automatically extract semantic web data from the website of the University of Southampton where the data about a professor and his list of publications is stored in RDF forma. The user can find the RDF data on the person's home page by clicking the RDF link. For example, this is a link for semantic web data on Professor Hugh C. Davis [29]: http://rdf.ecs.soton.ac.uk/person/46

The following steps can then be performed by the user:

1: Use the Wikicrawl system to enter http://rdf.ecs.soton.ac.uk/person/46 as a seed link (see Figure 2 below). The following screenshot shows how the user specifies the seed link for the web crawler, the domain scope and host scope in crawling.

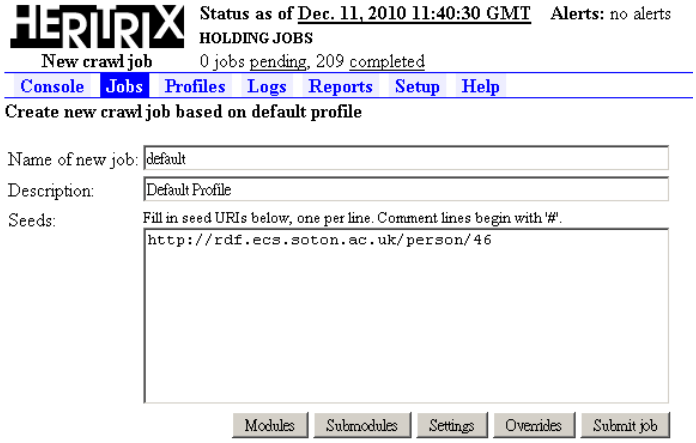


Figure 2: The crawler interface for entering a seed link

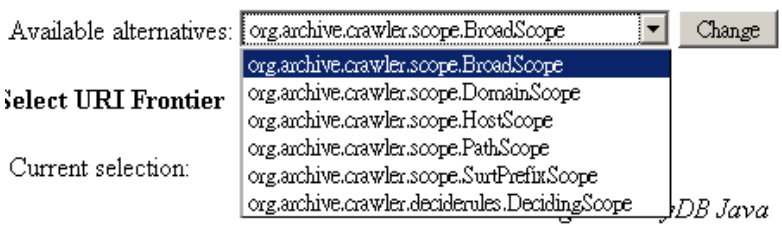
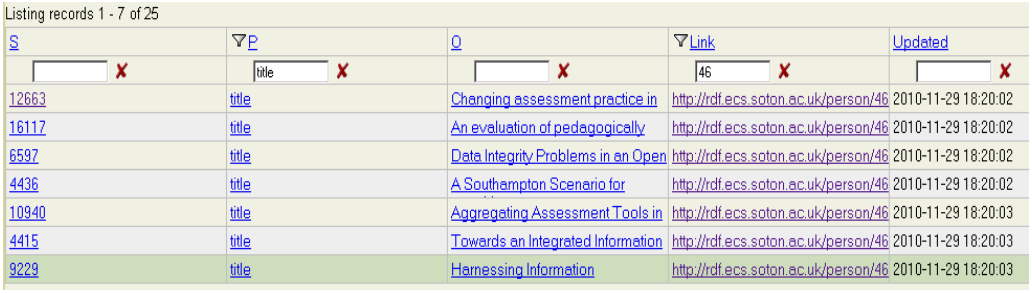


Figure 3: The setting modules menu for selecting Crawl Scope

Both of the screenshots above show the basic Heritrix interface for entering a seed link. This basic crawling functionality is enhanced however by enabling users to retrieve semantic web data and store the data in a MYSQL database instead of an ARC file.

2. Open the link in the Wikicrawl system as an RDF browser tool and thus retrieve a list of data from the RDF file in a simple table format, which displays all data as *Subject-Predicate-Object*. Additionally, the *Link* information is provided, and the last time of *Update* (see Figure 1 below).



Listing records 1 - 7 of 25

S	P	O	Link	Updated
	title		46	
12663	title	Changing assessment practice in	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:02
16117	title	An evaluation of pedagogically	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:02
6597	title	Data Integrity Problems in an Open	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:02
4436	title	A Southampton Scenario for	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:02
10940	title	Aggregating Assessment Tools in	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:03
4415	title	Towards an Integrated Information	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:03
9229	title	Harnessing Information	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:03

Figure 4: The delivery engine user interface

For adaptability in the display, there is a filter function under each header, so that the user can specify search keywords within the RDF document. In the screenshot above, one can see the interface of the delivery engine showing the list of retrieved information regarding "titles" of papers written by the particular person"46",by entering these keywords in the filter.

For easy readability, the interface shown above only displays extracted resources as short keywords in the table. This is a reduced version of the information initially available in the source RDF document, which also contains URI (Unique Resource Identifier) [30] information, which establishes the namespace of the resources. For example, the predicate link for "title" in the RDF document should be *"http://purl.org/dc/terms/title"*. However, it is difficult for a human (teacher, student) to browse the full length of links, as readability would be poor, and the display space limited. Moreover, name spaces tend to be shared within the document, and thus a lot of repetitive information would have appeared, which makes human reading difficult if not impossible. Thus, the Wikicrawl system cuts off the first part of the link and just keeps the keywords after the last forward slash, in order to save space and increase readability when browsing data. However, the original name space is stored, and the user is still able to browse it, by clicking on the resource keyword in the table, which works as a link to the original RDF document.

As shown on top of the interface, there is a counter for calculating the number of links matched the keyword and thus the user can know how many publications were released by Professor Hugh C. Davis, which would be difficult to know by reading RDF document directly. Also, one can apply a matrix query as well, by entering keywords in a few filters at a time to get more accurate results. The above screenshot shows that the query is regarding publications of a specific person 46, which is the URI for Professor Hugh C. Davis.

In this example, we have demonstrated a more user friendly way to browse RDF document. Concluding Scenario 1, we can say that when retrieving information via usual popular tools such as Google, or even via the Swoogle engine, if the user wants to collect information from a specific domain he/she can only browse the links one by one. Moreover, if document-internal links are relevant, these would need to be separately explored, manually. In Wikicrawl, the user can instead benefit from the crawler's automatic information retrieval. In the following example, we show that how the user can get more information by retrieving RDF documents from the same domain.

Scenario 2

There is another important functionality offered by the Wikicrawl delivery engine. The user is able to drag the whole row of data as extracted in the previous scenario from one table to another table, for pre-processing or extra filtering. Moreover, he can drag the desired Subject-Predicate-Object triples, or indeed any data into the text area for authoring the Wikipedia page. The following screenshots show how to drag and drop the whole row of data or a single cell of data into the text area. The data highlighted by underlining with a thick (yellow) line is the data dragged to the second template of the table (see Figure 5).

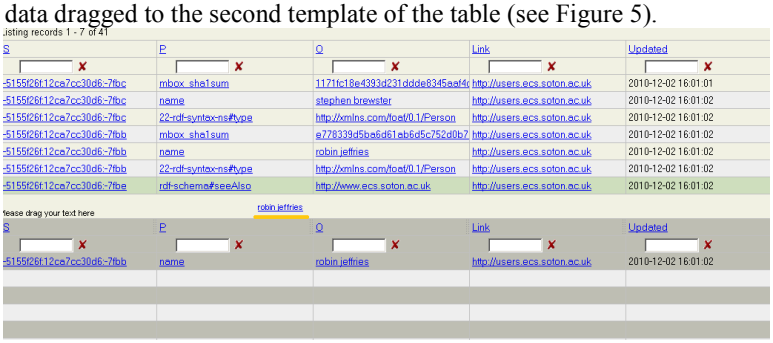


Figure 5: Dragging the whole row of data to the other template table

The screenshot below shows how to drag and drop a cell of data into a Wikipedia page. The data is inserted into the Wikipedia page directly after entering "OK".

Figure 6: Dragging data item into text area

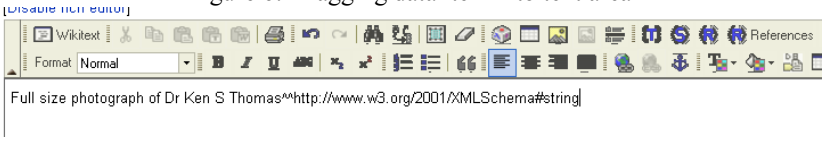


Figure 7: The data imported from the drop area into the Wikipedia authoring area

The user could thus save time in authoring Wikipedia pages via this method, when adding additional data from the open (semantic) web. Importantly, he/she does not need to copy and paste the data into Wikipedia one by one, he/she now can drop all the data into the template of the text area and transfer to the Wikipedia page authoring interface offered by the CKEditor [8].

What is worth noticing is that the Wikicrawl system is able to export the data from the template table to a traditional HTML web table format or other formats. The following screenshot shows how to export the data into Excel [25] as well as into the Wikipedia page. It uses the previous example for demonstration.

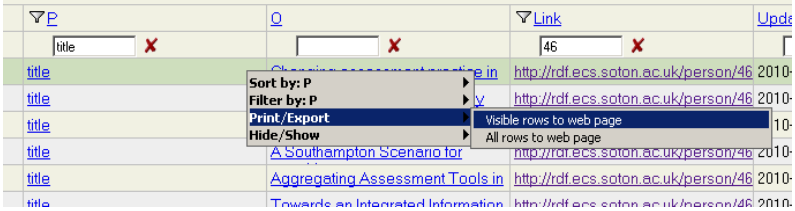
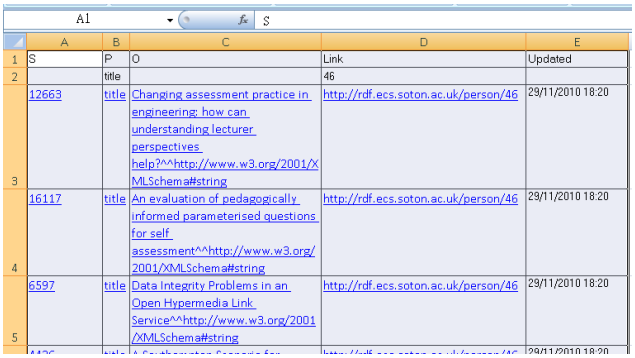


Figure 8: The menu of the export function

S	P	O	Link	Updated
	title		46	
12663	title	Changing assessment practice in engineering: how can understanding lecturer perspectives help?^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:02
16117	title	An evaluation of pedagogically informed parameterised questions for self assessment^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:02
6597	title	Data Integrity Problems in an Open Hypermedia Link Service^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	2010-11-29 18:20:02

Figure 9: The output table into a new HTML web page

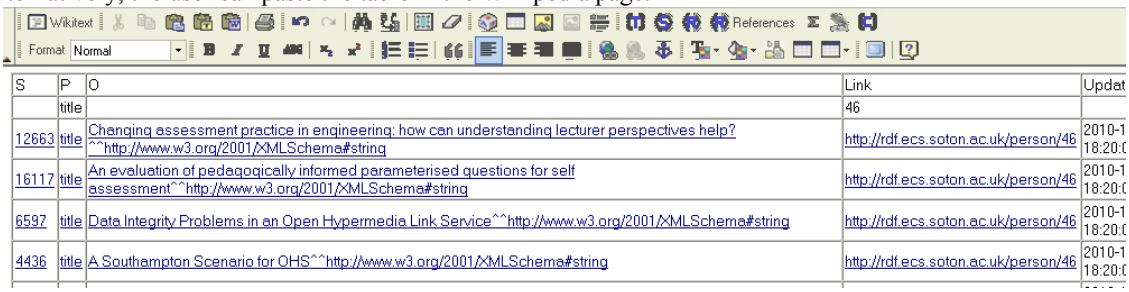
The use can use the table above in Excel worksheets by copying and pasting the table into an Excel file for future reusing (see Figure 10).



	A	B	C	D	E
1	S	P	O	Link	Updated
2		title		46	
3	12663	title	Changing assessment practice in engineering: how can understanding lecturer perspectives help?^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	29/11/2010 18:20
4	16117	title	An evaluation of pedagogically informed parameterised questions for self assessment^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	29/11/2010 18:20
5	6597	title	Data Integrity Problems in an Open Hypermedia Link Service^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	29/11/2010 18:20

Figure 10: Pasting the export table into Excel

Alternatively, the user can paste the table in the Wikipedia page:



S	P	O	Link	Updat
	title		46	
12663	title	Changing assessment practice in engineering: how can understanding lecturer perspectives help? ^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	2010-1 18:20:(
16117	title	An evaluation of pedagogically informed parameterised questions for self assessment^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	2010-1 18:20:(
6597	title	Data Integrity Problems in an Open Hypermedia Link Service^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	2010-1 18:20:(
4436	title	A Southampton Scenario for OHS^^http://www.w3.org/2001/XMLSchema#string	http://rdf.ecs.soton.ac.uk/person/46	2010-1 18:20:(

Figure 11: Pasting the export table into a Wikipedia page

The user (teacher, student) may find that some of the columns are not necessary for his final result. This point is well addressed in this system, as it allows hiding columns in the export menu. The following screenshot shows a menu interface for selecting which columns to hide and which to show.

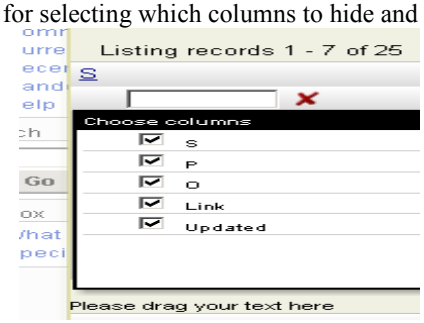


Figure 12: Selection menus for hidden columns

It is noted that regular editing in Wikipedia is not disturbed. So the page can be authored with other contents, besides the paper list of professor Hugh Davis, also information about his research topics, etc.

Evaluation description

The Wikicrawl system was evaluated in a first evaluation round with around 40 subjects who are have had previous experience with authoring of Wikipages and searches via web browsers. They were selected from a wide range of people, including university researchers and university graduates, over 50% with Computer Science knowledge and another 50% having a mixture of engineering, mathematics and other backgrounds. Over 70% were educated at the level of a Master Degree or above. Over 60% are male, and their age range is between 27 and 65. The subjects were mainly from the United Kingdom and Asia, but also from beyond. In the first evaluation, we have evaluated the first version of the Wikicrawl system.

Slightly differently to our scenario example, this version focussed on retrieving information from social web sites, so that a user can find with its help a list of friends (students, teachers, etc.) with common interests (for instance, for learning together, or working together on authored material). In the evaluation

questionnaire, we aimed at getting some feedback in using the Swoogle semantic web search engine. The results show that most of the users found that it is difficult to use the semantic web documents without any tools. More than 70% of the people have agreed that they can use Wikicrawl for capturing information from semantic web documents easier and faster. More than 75% of the people think that they will use Wikicrawl for crawling semantic web documents frequently. However, there are only 41.5% of the people think that the various functions in Wikicrawl were well integrated. They also felt that there is lack of information provided in the explanation of extracted data, which pointed to some obvious improvements that could be done. The evaluation also compared Wikipedia with WikiText authoring, to Wikipedia authoring with Wikicrawl, and the results showed that more than 65% of the subjects didn't like the WikiText authoring method but preferred authoring a Wikipedia page by the dragging and dropping method offered by Wikicrawl. More than 80% of the subjects thought that some of the social web site query interface they used was not able to get the information that they expected, but they were able to find that information with the help of the Wikicrawl system. The users would also like to see the integration of a ranking system and tagging system for Wikicrawl, in order to share their favourite retrieved data and in order to improve the data reuse and the authoring accuracy.

Future developments

The evaluation and feedback of the first prototype from the participants raised the question whether this system could be used to retrieve semantic web data from any domains rather than only one specified social web site. Moreover, it raised the issue that users do not have use SPARQL query languages to complete the crawling task, as was done in the first evaluation.

Based on the feedback, the second prototype which has been discussed in the current paper's implementation part addressed these issues and attempts to make the crawling process simpler. The evaluation hasn't been started on a large scale, yet from the feedback of a few colleagues who tried it and further reading, there are two major functionalities that can be useful and need to be investigated in future work. One is to be able to share easily the retrieved data with others for reuse by tagging them with keywords, the other is a ranking function which shows in the interface the mostly cited entries. By implementing these functions, the accuracy of authoring can be increased. For further work, the new version of the system is now currently undergoing another evaluation, which aims to gather feedback based on usability in comparing existing authoring tools, on one hand, and functionality, on the other. The results of the second prototype will be gathered and analyzed in the following weeks.

Conclusions

In this paper, we have proposed a novel application, exploring the possibility of reusing semantic web data for contributing to the Wikipedia system and experimenting with the integration of semantic web technology into the Wikipedia authoring process. The actual system is based on the Heritrix infrastructure for information retrieval over the Semantic Web, utilizing the JENA API. MYSQL is used as a database for storing the data for authoring Wikipedia content for educational purposes and beyond. The easy to use interface of the delivery engine, based on the CKEditor, makes the Wikipedia authoring process simple for users. The hope is that this tool, or tools like this, will encourage more users to contribute to Wikipedia authoring in not only education, be they teachers or students but also in diverse fields.

References

[1] Peter, M. & Tom, E. (2009). Interaction possibilities on Web 2.0 websites as a framework for cluster analysis. http://miha2.ef.uni-lj.si/cost298/gbc2009-proceedings/papers/P013.pdf.

[2] Bart, D. (2006). Introducing Flock Beta 1. Flock official blog. http://www.flock.com/node/4500.

[3] http://www.w3.org/RDF/. Retrieved 2010-12-13

[4] http://www.w3.org/TR/owl-features/. Retrieved 2010-12-13

[5] http://www.wikipedia.org/. Retrieved 2010-12-13

[6] Zlatic, V., Bozicevic, M., Stefancic, H., Domazet, M.(2006). Wikipedias: Collaborative webbased encyclopedias as complex networks. In: arXiv. http://arxiv.org/pdf/physics/0602149

[7] Linyun, F., Haofen W., Haiping Z., Huajie Z., Yang W. & Yong Y. (2007). Making More Wikipedians: Facilitating Semantics Reuse for Wikipedia Authoring. http://www.springerlink.com/content/vt2673h302747256/.

[8] http://ckeditor.com/. Retrieved 2010-12-13

[9] Amit, S. (2001). Modern Information Retrieval: A Brief Overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering* 24 (4): 35–43. http://singhal.info/ieee2001.pdf.

[10] http://www.w3.org/. Retrieved 2010-12-13

[11] http://www.w3.org/TR/rdf-sparql-query/. Retrieved 2010-12-13

[12] http://www.w3schools.com/sql/default.asp. Retrieved 2010-12-13

[13] http://www.google.co.uk/. Retrieved 2010-12-13

[14] http://swoogle.umbc.edu/. Retrieved 2010-12-13

[15] http://simile.mit.edu/wiki/SIMILE:About Retrieved 2010-12-13

[16] http://simile.mit.edu/wiki/Piggy_Bank Retrieved 2010-12-13

[17] http://simile.mit.edu/semantic-bank/index.html Retrieved 2010-12-13

[18] David, H., Stefano, M. & David, K. (2005) Piggy Bank: experience the semantic web inside your web browser, MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA.

[19] http://www.w3schools.com/js/default.asp . Retrieved 2010-12-13

[20] http://simile.mit.edu/wiki/Category:Javascript_screen_scraper. Retrieved 2010-12-13

[21] http://crawler.archive.org/. Retrieved 2010-12-13

[22] http://www.archive.org/web/researcher/ArcFileFormat.php. Retrieved 2010-12-13

[23] Haofen W., Huajie Z., Linyun F. & Yong Y. (2007). EachWiki: Facilitating Semantics Reuse for Wikipedia Authoring

[24] Leigh, D. (2006). Slug: A Semantic Web Crawler. http://www.ldodds.com/projects/slug/slug-a-semantic-web-crawler.pdf Retrieved 2010-12-13

[25] http://office.microsoft.com/en-us/excel/. Retrieved 2010-12-13

[26] http://openrico.org/ Retrieved 2010-12-13

[27] http://openjena.org/ Retrieved 2010-12-13

[28] Christian, B., Tom, H. & Tim, B. (2009). Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems (IJSWIS), Vol. 5(3), Pages 1-22. DOI: 10.4018/jswis.2009081901

[29] http://www.ecs.soton.ac.uk/people/hcd Retrieved 2010-12-13

[30] http://www.w3.org/Addressing/ Retrieved 2010-12-13