WEB EXTENSIONS FOR SEMANTIC DATA CREATION

Adam Markovski
Makpetrol A.D., ADP Division
Mito Hadzivasilev Jasmin 4, 1000
Skopje, Macedonia

Milos Jovanovik
Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University in Skopje
Rugjer Boskovik 16, PO Box 393,
1000 Skopje, Macedonia

Dimitar Trajanov
Faculty of Computer Science and Engineering, Ss. Cyril and Methodius University in Skopje
Rugjer Boskovik 16, PO Box 393,
1000 Skopje, Macedonia

ABSTRACT

The existence of the Semantic Web is very dependent on the semantic content which exists on the current web. Thus, the process of creating more semantic content actually means building the Semantic Web itself. This process can be done using various information extraction and annotation mechanisms. However, they usually focus on domain specific approaches manually built and adapted for pre-assigned tasks, and are designed for more advanced users. In this paper we introduce an innovative way for creating semantic data and content by involving the common web user and with the use of an easily operated extension for content bookmarking, supported by a blogging platform. The content gathered by the user is processed by existing service-oriented information extractors where it is semantically annotated, and then published on the web.

I. INTRODUCTION

The average Internet user spends 68 hours per month online [1]. Most of the Internet browsers [2] let the users have their own way for customizing the browser. On the other side, when we talk about the Semantic Web, we usually talk about the data that defines it.

The growth of unstructured data which is available on the World Wide Web, as well as private and enterprise file sharing, demands flexible and generic solutions for extracting the information they hold. The current information extraction solutions focus on domain specific approaches manually built together and adapted for a pre-assigned task [3].

Having that in mind, we built an application which creates semantic data with the help of web browser extensions and with the use of service-oriented information extractors. We also included a Web 2.0 blogging platform in the process, as a backbone of the application.

We use the extensions as content bookmarkers, with the option to send the desired content to the user's blog. When the blog receives content from the extension, it processes it and semantically annotates it, in a process which is transparent for the user. The semantic transformation is done using a few popular semantic generators. The result is a tagged and categorized blog post, ready for editing and sharing [4].

II. RELATED WORK

There has been a lot of work done in the field of semantic data generation, but very little concerning the average Internet user. If we want to popularize the use of the semantic data, we must implicitly explain its meaning to the average user. In our work we are trying to do this with incorporating the semantics in everyday web tools – such as browser extensions and blogs.

This paper describes the implementation of our idea. The details of the implementation and the generated content are described in Section 3. Section 4 presents our conclusions and plans the future work.

III. IMPLEMENTATION

First introduced in Internet Explorer 4, browser extensions (known as Add-ons in Internet Explorer [5] and Mozilla Firefox [6], and Extensions in Google Chrome [7]) allow users to add functionalities to the browser and enhance the user interface in a way that is not directly related to the viewable content of web pages [8]. Nowadays, every popular browser has its own extension system, and the browser extensions can be programmed to do pretty much anything in the browser environment. Almost every web users adopts the bookmarking system of the browser for marking the desired visited page for later use. The user is not always interested in the entire page content, but only a small fraction of data with specific meaning. The later described content bookmarker extension does exactly this - it bookmarks only the selected content. The bookmarker is an add-on called PinPart, the first component of the system depicted in the Fig.1.

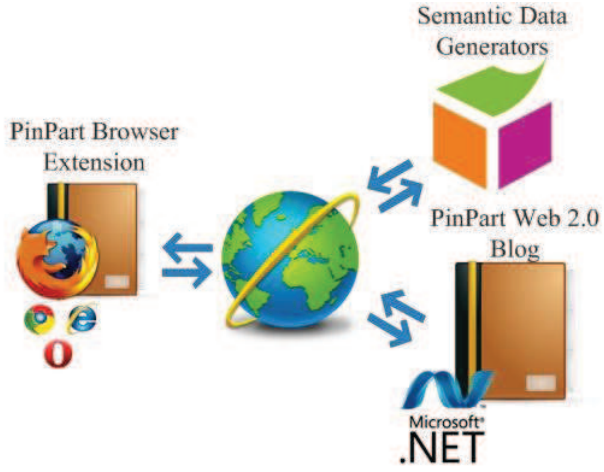

Figure 1: A simple diagram of the PinPart system, composed of the PinPart browser extension and the PinPart Web 2.0 blog.

The second part (Fig.1) is the core of the application, a fully-featured Web 2.0 blog. Extended to communicate with the browser extension, it gives the user the possibility to edit and share the bookmarked content.

The main purpose of the PinPart system is transformation of the received bookmarked data. Once it receives content, the PinPart blog sends the content to specific online semantic data generators for content analysis and metadata annotation. Afterwards, the processed data is available to the user for editing as a blog post, which is now semantically annotated and enhanced.

Although bookmarker applications are very common, with PinPart we provide the user with a significant improvement – we enrich his content with semantic annotations and assign more machine-readable meaning to it.

A. The PinPart Browser Extension

For our system, we chose to create an add-on for the Mozilla Firefox (8.*) web browser. One reason for this is that more than 2 billion Firefox add-ons have been downloaded worldwide [6].

The Firefox web browser, as well as other Mozilla applications, can be seen as composed of two different parts: a user interface layer which is distinct for each project, and a common platform on top of which the interface layer is built. This platform is known as XULRunner. The XULRunner includes the Gecko rendering engine, the Necko networking library, and several other components which provide OS-independent file management, accessibility and localization, among others. It is this very powerful platform that has allowed such a fast growth of the development community surrounding Mozilla and Firefox [9]. The browser is largely built using four technologies: XUL, CSS, JavaScript, and XPCOM. Extensions are also built using these four technologies [10][11].

The extension we built, PinPart, saves web pages for later reading with just one click. It holds the saved bookmarks in an extension folder, allowing actions such as searching and ordering over them. The most important feature is that the user can bookmark a selected text from a web page, by right clicking the selection. Every newly created bookmark holds the URI of the page from which the content originated, as well as the selected content saved as a description of the bookmark. It also contains the tags extracted from the chosen text. The tags are generated with a simple keyword search algorithm. With a single click after the bookmarking, the content can be sent to the PinPart blog for semantic processing.

For communication with the blog, the extension uses XMLHttpRequest objects [12] and the datajs library [13]. Datajs is a new cross-browser JavaScript library which enables the development of data-centric web applications, by leveraging modern protocols such as JSON and OData [14] and HTML5-enabled browser features. Having this service-oriented type of communication between the PinPart blog and the PinPart add-on, makes the extension development for other web browsers quite easy. With this, we provide the necessary extensibility of the solution.

B. The PinPart Blog

As a blog engine platform in our system, we use BlogEngine.NET. It is an open source .NET blogging platform with less complexity, easy customization, and one that takes advantage of the latest .NET features [15]. The PinPart blog has all the Web 2.0 blog characteristics. The main advantages of the chosen blog platform are: the search capability as one of the most advanced – it also allows the visitors to search within the comments, the standalone pages which allow creation of static pages that exist outside the blog chronology, and the multi-author support. Having the source code, functionalities can be added or changed to the blog. Customization can be achieved with easy development of extensions for the blog, as well.

If the user wants to send its bookmarked content to the PinPart blog, he first must be signed-in as a blog user. Every registered user can send its bookmarks to the blog for semantic processing. When the blog receives the content, before publishing, it sends the content for semantic processing. The content bookmark received after processing is treated as a blog post from the user who initially sent it.

The PinPart blog opens itself to the web through a classic web service for simple communication, and a WCF data service for data operations on the post entities.

C. Semantic Data Generators

As semantic information generators in our application we use the following: AlchemyAPI [16], Zemanta [17] and OpenCalais [18]. In the browser, the user selects certain plain text content and pins it, i.e. creates a bookmark. The PinPart add-on stores the selected text as a bookmark with the page URI, and tags generated from the selection are stored as the bookmark description. The real processing is done when the bookmarked content is sent to the PinPart blog.

Once the content is sent to the PinPart blog, it is categorized it with the AlchemyAPI, which uses sophisticated statistical algorithms and natural language processing to analyze it [19]. This process assigns the most likely topic category to the content, e.g. news, sports, business, etc. The new categories, if any, are then added to the blog category system. For the article shown below, which is about the change of the Greek prime minister, the category we get is as expected – politics.

```
"Economist Lucas Papademos was formally sworn in
Friday as the head of Greece's new unity
government, as the nation seeks to regain
political and financial stability after weeks of
uncertainty.
Papademos, a former banker and European Central
Bank vice president, becomes the country's interim
prime minister after several days of political
wrangling.
His ministers were also sworn in at a ceremony
attended by the president and the head of the
Greek Orthodox Church.
Finance Minister Evangelos Venizelos has retained
his post in the new government, the prime
minister's office said.
Stavros Dimas, a former EU environment
commissioner, is the new foreign minister. He
belongs to the New Democracy party, the main
opposition to the previous government…"
```

Named entities are also extracted from the text. The AlchemyAPI in the text shown above has detected three persons, one country, one city, two organizations and a television station. Finally, the sentiment of the article is determined: positive, negative or neutral. Sentiment analysis [20] aims to determine the attitude of a speaker or a writer on

a given topic, or the overall contextual polarity of a document. For the text shown above, the sentiment is positive. Part of the result set in XML format is given below.

```
<?xml version="1.0" encoding="UTF-8"?>
<results>
  <status>OK</status>

<usage>http://www.alchemyapi.com/company/terms.html</usag
e>
  <url></url>
  <language>english</language>
  <category>culture_politics</category>
  <score>0.85</score>
</results>

<?xml version="1.0" encoding="UTF-8"?>
<results>
  <status>OK</status>

<usage>http://www.alchemyapi.com/company/terms.html</usag
e>
  <url></url>
  <language>english</language>
  <docSentiment>
    <type>positive</type>
    <score>0.0094041</score>
  </docSentiment>
</results>
```

Exhaustive semantic metadata in RDF format is also extracted from the content with the use of the OpenCalais service, and is written as a description of the blog post. The OpenCalais service supports a rich set of semantic metadata, including entities, events and facts. However, here we use the OpenCalais' capability to recognize all subject-predicate-object relationships, without predefining their types. An example of semantic RDF metadata extracted with the OpenCalais service is shown below:

```
<rdf:Description
rdf:about="http://d.opencalais.com/genericHasher-1/...">
    <rdf:type
rdf:resource="http://s.opencalais.com/1/..."/>
    <!--Hugh Laurie-->
    <c:relationsubject
rdf:resource="http://d.opencalais.com/pershash-1/..."/>
    <c:relationobject>various awards</c:relationobject>
    <c:verb>win</c:verb>
  </rdf:Description>
  <rdf:Description
rdf:about="http://d.opencalais.com/dochash-1/...">
    <rdf:type
rdf:resource="http://s.opencalais.com/1/type/sys/Instance
Info"/>
    <c:docId
rdf:resource="http://d.opencalais.com/dochash-1/..."/>
    <c:subject
rdf:resource="http://d.opencalais.com/genericHasher-
1/..."/>
    <!--GenericRelations: relationsubject: Hugh Laurie;
relationobject: various awards; verb: win; -->
    <c:detection>[from ER.[9] For his portrayal of
Gregory House, ]Hugh Laurie has won various awards[,
including two Golden Globe Awards for Best]</c:detection>
    <c:prefix>from ER.[9] For his portrayal of Gregory
House, </c:prefix>
    <c:exact>Hugh Laurie has won various awards</c:exact>
    <c:suffix>, including two Golden Globe Awards for
Best</c:suffix>
    <c:offset>1449</c:offset>
    <c:length>34</c:length>
</rdf:Description>
```

The text shown above is a small part of the metadata generated for a selected content from a Wikipedia article about a famous television show (Fig.2). As we can see, a generic relation has been made for the subject "Hugh Laurie" with the object "various awards" and the relation verb "win".

As a final step, we query Zemanta for generation of links to similar articles on the web. Zemanta aggregates the suggested articles from many different web sources. Those include both mainstream news sites, as well as the wider blogosphere, with emphasis on highly-rated blogs. Each of those suggestions can provide a qualitative improvement of the submitted content. With Zemanta we also generate in-text links to help the users when they want to know more about the people, places or phrases from the content.

Figure 2: Using the PinPart extension within the Firefox web browser.

The result is an editable blog post enriched with semantic metadata, categorized and tagged (Fig.3).

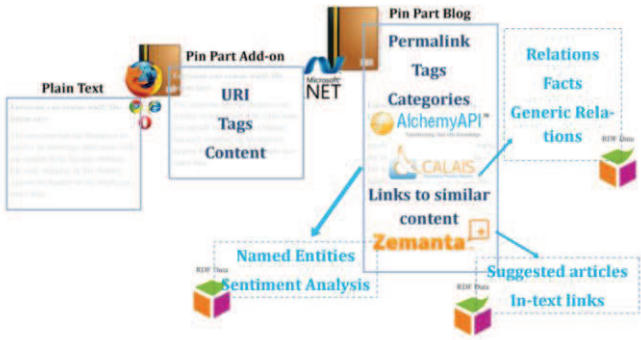

Figure 3: Transforming a simple text into a semantically annotated blog post.

IV. Conclusion and Future Work

This paper presents the idea of using the browser extensions approach for generating semantic content on the web, in order to aid the endeavour of transforming it into the Semantic Web. By using more user friendly applications, it is easier to bring the power of the Semantic Web closer to the common web user, and involve him in the creation of a semantically annotated web. We believe that promoting the use of semantic data can be done only by building applications from which the end users can have some benefits. Our system provides the user with automatic categorization, tagging, and semantic

annotation of his content, as well as other additional information and related articles.

Future directions in the development of the approach include looking for new general semantic web tasks which could be implemented using the web extensions approach. Much can be done in the development of user friendly applications which exploit the advantages of the semantic web technologies. The presented application could also be developed further. As an example, we can extend our system with add-ons for the other popular browsers. With this, the user base could grow, as well as the creation of semantic web content on the current web.

REFERENCES

[1] Nielsen Company – Top U.S. Web Brands and Parent Companies for September 2009: http://blog.nielsen.com/nielsenwire/online_mobile/top-u-s-web-brands-and-parent-companies-for-september-2009/

[2] Under World Magazines – The Popular Web Browsers: http://www.underworldmagazines.com/the-popular-web-browsers/

[3] Seidler, K., "Service-oriented Information Extraction", *Proceedings of the 2011 Joint EDBT/ICDT Ph.D. Workshop*, New York, 2011.

[4] Anderson, P., "What is Web 2.0? Ideas, technologies and implications for education", *JISC Technology and Standards Watch*, 2007.

[5] Internet Explorer Gallery: http://www.iegallery.com

[6] Mozilla Add-ons: https://addons.mozilla.org/firefox/

[7] Chrome Web Store: https://chrome.google.com/webstore/category/extensions

[8] Microsoft Developer Network – Browser Extensions: http://msdn.microsoft.com/en-us/library/aa753587(v=VS.85).aspx

[9] Mozilla Developer Network: https://developer.mozilla.org/

[10] Kenneth C. Feldt: *Programming Firefox*. O'Reilly Media, Inc., Sebastopol, 2007.

[11] Stoyan Stefanov: *JavaScript Patterns*. O'Reilly Media, Inc., Sebastopol, 2010.

[12] W3C - XMLHttpRequest object: http://www.w3.org/TR/XMLHttpRequest/

[13] JavaScript Library for data-centric web applications: http://datajs.codeplex.com/

[14] Open Data Protocol: http://www.odata.org/home

[15] BlogEngine.NET features: http://www.dotnetblogengine.net/page/BlogEngineNET-20-Features-Notes.aspx

[16] AlchemyAPI: http://www.alchemyapi.com/

[17] Zemanta: http://developer.zemanta.com/

[18] OpenCalais: http://www.opencalais.com/

[19] AlchemyAPI – Topic Categorization: http://www.alchemyapi.com/api/categ/

[20] Michelle de Haaff, "Sentiment Analysis, Hard But Worth It!": http://www.customerthink.com/blog/sentiment_analysis_hard_but_worth_it