RDFaCE – The RDFa Content Editor

Ali Khalili and Sören Auer

Universität Leipzig, IfI/BIS/AKSW, Johannisgasse 26, 04103 Leipzig, Germany
{lastname}@informatik.uni-leipzig.de, http://aksw.org

Recently practical approaches for managing and supporting the life-cycle of semantic content on the Web of Data made quite some progress. However, the currently least developed aspect of the semantic content life-cycle is the user-friendly manual and semi-automatic creation of rich semantic content. In this paper we present the RDFaCE (RDFa Content Editor) approach for combining WYSIWYG text authoring with the creation of rich semantic annotations. WYSIWYG text authoring is meanwhile ubiquitous on the Web and part of most content creation and management workflows. It is part of Content Management Systems, Weblogs, Wikis, fora, product data management systems and online shops, just to mention a few. Our goal with this work is to integrate the semantic annotation directly into the content creation process and to make the annotation as easy and non-intrusive as possible. The RDFaCE implementation is open-source and available for download together with an explanatory video and online demo at http://aksw.org/Projects/RDFaCE.

The RDFaCE system architecture is depicted in Figure 1 and consists of three layers. The foundation layer on which we ground the RDFaCE plugin includes the *TinyMCE Rich Text Editor*(http://tinymce.moxiecode.com). This open source HTML editor was chosen because it is very flexible to extend and is used in many popular Content Management Systems (CMS), blogs, wikis and discussion forums, etc. Therefore, by focusing efforts on this one particular editor, it is possible to quickly propagate accessible RDFa authoring practices to a number of other tools. RDFaCE includes the following components:

Annotator UI uses the TinyMCE API as well as jQuery UI to create user friendly user interfaces for RDFa content editing. *RDFa DOM Manipulator* is responsible for manipulating the Document Object Model (DOM) according to the desired RDFa annotation. The main goal of the *Inline Semantic Visualizer* is to provide a kind of on-demand visualization which can be included/excluded on the fly within the WYSIWYG content editing. *RDF Triple Browser and Editor* extracts the RDF triples embedded in the text and provides the edit and delete functionality for these triples. *Online Resource Suggester* provides the user with a set of accessible online resources. In order to perform this task, it accesses a number of external Web APIs. *RDFa Proxy for Enricher APIs* acts as a proxy to make the output of enricher APIs (i.e. NLP text annotation services) consumable as RDFa.

To facilitate semantic annotation of content, RDFaCE also uses a number of external Web APIs. Online APIs are invoked to carry out the following functions:

– **RDF Namespace Lookup**: In order to avoid that users have to type complete URIs, common namespace prefixes can be used everywhere in RDFaCE.

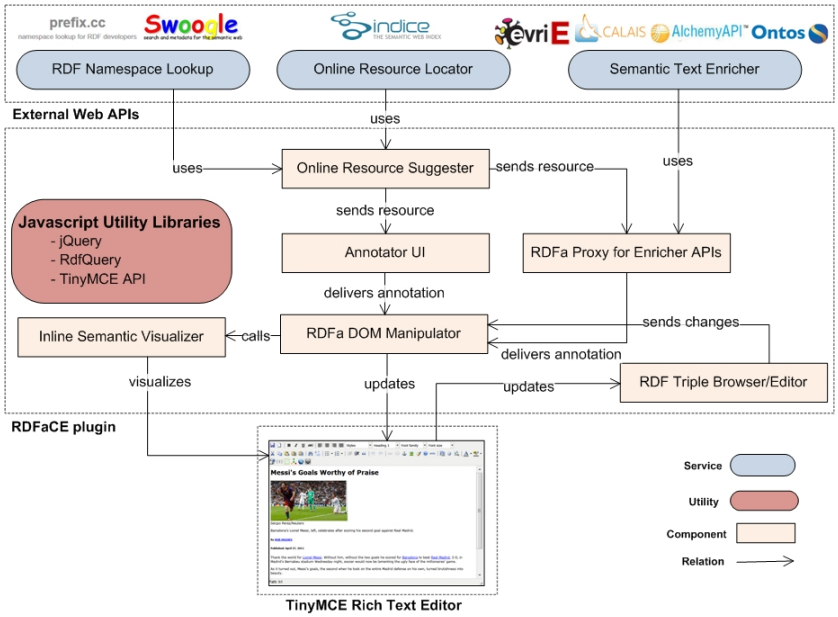

Fig. 1. RDFaCE system architecture.

These are looked-up using the `prefix.cc` service. Furthermore, in case users want to add a new property for which they do not even know an appropriate vocabulary, RDFaCE can look-up an appropriate vocabulary and property resource using the Swoogle (`http://swoogle.umbc.edu/`) service.

– **Online Resource Locating**: Finding an appropriate URI for the resources which are selected by users can facilitate annotation process to a good extend. When users select a part of the text and want to create a statement about the respective entity, the online resource locator will do a Sindice (`http://sindice.com`) search to find suitable resources that match with the users selected item.

– **Semantic Text Enrichment**: Starting to annotate a document from the scratch is very tedious and time consuming. There are already some Natural Language Processing (NLP) APIs available on the Web which extract specific entities and relations from the text. By using these APIs, we can provide a good starting point for further user annotations. Users then can modify and extend these automatically pre-annotated content. RDFaCE currently uses the *OpenCalais*, *Ontos*, *Alchemy*, *Extractiv* and *Evri*[1] APIs to enrich the text. Besides annotation by each individual API, RDFaCE sup-

[1] OpenCalais - `http://www.opencalais.com`, Ontos - `http://www.ontos.com`, Alchemy - `http://www.alchemyapi.com`, Extractiv - `http://extractiv.com` and Evri - `http://www.evri.com`

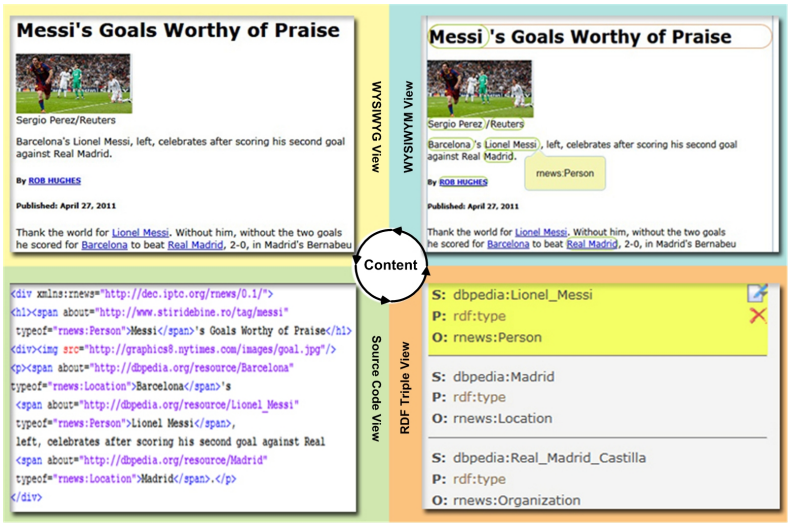

Fig. 2. The four views for semantic text authoring.

ports combining the results of multiple NLP APIs which yields in superior performance compared to each individual.

As shown in Figure 2, RDFaCE supports four different views for semantic text authoring. The user can easily switch between these views and even use them in parallel. The views are syncronized so that applying changes in one of the views automatically updates other views.

WYSIWYG View. The What-You-See-Is-What-You-Get view is the classical interface for rich-text authoring and used by authors, journalists etc. WYSIWYG text authoring is meanwhile ubiquitous on the Web and part of Content Management Systems, Weblogs, Wikis, fora, product data management systems and online shops, just to mention a few.. WYSIWYG implies a user interface that allows the user to view something very similar to the end result while the document is being created.

WYSIWYM View. The What-You-See-Is-What-You-Mean view is an extension of the WYSIWYG view, which highlights named entities and other semantic information. The highlighting is realized with special CSS3 selectors for the RDFa annotations. They are thus easily configurable in terms of color borders, backgrounds etc. When pointing with the mouse on a highlighted annotation RDFaCE shows additional information concerning the particular annotation as a tooltip. RDFaCE also supports editing in the WYSIWYM view by letting a user select entities she wants to annotate and provisioning of respective annotation functionality either via the context menu or a specific form, which opens as an overly.

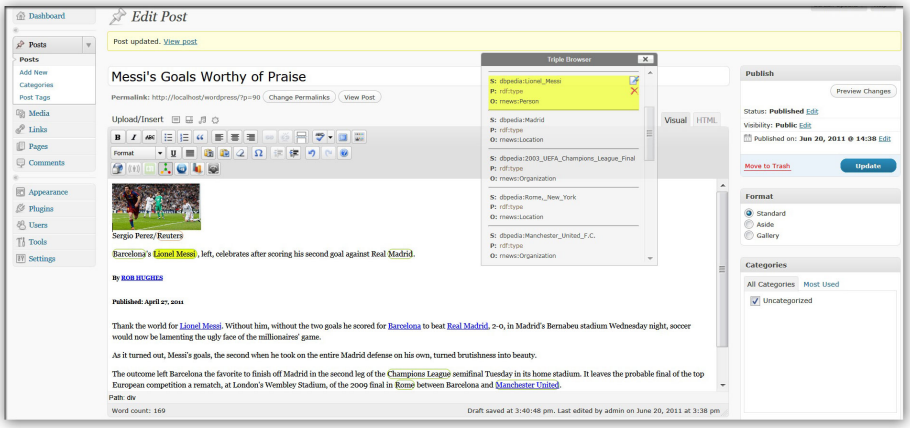

Fig. 3. Integration of RDFaCE into Wordpress.

RDF Triple View. This view summarizes all the facts, which can be extracted from the annotated text. It provides a deeper semantic view when compared to WYSIWYM view. There might be some triples not visible in the WYSIWYM view (e.g. annotations hidden using the CSS `display:none` style) but visible in this view. The triple view is (as all other views) updateable, i.e. facts can be directly deleted, which results in the removal of the corresponding RDFa annotations. The triple view is useful for curators and to a lesser extend for the authors for quickly verifying that entities and facts were correctly annotated.

Source Code View. Finally, the source code view shows the HTML source of the article including the RDFa annotations. This view is primarily intended for software engineers supervising the publication workflow as well as knowledge engineers.

The RDFaCE approach is very versatile and can be applied in a vast number of use cases. *rNews* (`http://dev.iptc.org/rNews`) as a proposed standard for using RDFa to annotate HTML documents with news-specific metadata, *OntoWiki* (`http://ontowiki.net`) as a tool for collaborative semantic content authoring and *Wordpress* (`http://wordpress.org`) as the most popular blogging platform are some of the RDFaCE use cases.

Related Work : There are already a few RDFa editors available. Loomp (`http://loomp.org`), RDFAuthor (`http://aksw.org/Projects/RDFauthor`) and RADiFy (`http://duncangrant.co.uk/radify/`) are some of the related works in this area.