One Click Annotation

Ralf Heese, Markus Luczak-Rösch, Radoslaw Oldakowski, Olga Streibel, and Adrian Paschke

Freie Universität Berlin, Institute of Computer Science,
Corporate Semantic Web,
Berlin D-14195, Germany,
{heese,luczak,oldakowski,streibel,paschke}@inf.fu-berlin.de,
http://www.corporate-semantic-web.de

Abstract. The realization of the Semantic Web as a linked machine readable Web of data depends on the availability of authoring tools that enable ordinary Web users (i.e. non-experts w.r.t. Web technologies) to create and publish semantic data. In this paper, we introduce the "One Click Annotator" for enriching texts with RDFa annotations and linking resources through an intuitive user interface, thereby hiding the complexity of creating semantic data. With this easy to use approach we aim at contributing to the change of the Web into a Semantic Web consisting of semantic information objects and linked data. We discuss requirements, challenges, and solutions for the implementation of the annotator.

1 Introduction

Gathering information from the Web on a special topic of interest is a time consuming task for Web users which are required to manually discover different heterogeneous and scattered Web data sources. The Web 2.0 movement, which changes the Web into an information interchange platform for ordinary Web users, helps to ease and partially automates the discovery processes. The information chunks published online become smaller and are part of larger interconnected so-called data silos or information clouds which are accessible via common Web 2.0 portals. This is emphasized, for example, by the recent success of social networks and micro blogging.

Semantic technologies, and especially the linked data principles, promise on the one hand automation by turning the information Web into interconnected and interoperable machine processable data sources. On the other hand, linked data creates a basis for consuming and contributing semantic data by everyone. The technologies have reached an acceptable maturity state, however, their benefits are not broadly adopted in commercial and public Web 2.0 applications, yet. We believe this is mainly because creating Semantic Web metadata without any tailored, easy-to-use Semantic Web tool support is an additional burden.

In this paper we address this issue and introduce a conceptual approach and an implementation of a tool for efficiently and easily authoring semantically enriched contents. We build our approach on top of the principles of linked data.

Embedded into social Web applications, e.g., as replacement for existing text editors, our approach allows everyone to create semantic data. Since everyone can reuse and modify published text fragments, they can also create semantic data on top of existing one, e.g., refer to resources created by other users.

We develop loomp (`http://loomp.org/`), a lightweight semantic content authoring system, which allows *non-expert users* to enrich Web content with detailed semantics, e.g., annotate resources and relate them. The resulting data is afterwards publicly available as linked data. To minimize the effort of content editors, the main goal of loomp is to make semantic annotation of Web contents as easy as to format the style of text contents in word processors. The key component of loomp is the "One Click Annotator" (OCA) allowing users to easily add annotations to their texts by selecting text blocks and annotating them via choosing a concept from a toolbar.

The paper is structured as follows: In Section 2 we give an overview of the related work. Afterwards, in Section 3, we briefly introduce the loomp platform and describe an illustrating example of using loomp in a Web application. In Section 4, we describe the One Click annotator in detail discussing requirements, challenges, and solutions. In Section 5 we conclude and outline our future work.

2 Related Work

The concept of our One Click Annotator is based on the idea of enriching texts with semantics by annotating them manually with one click. Regarding the semantic annotation of texts, there are approaches based on text mining techniques, ontology-based approaches whereby the annotating process uses methods from computational linguistic, and approaches based on the Web 2.0 phenomenon like social tagging and collaborating by wiki-systems.

Social tagging and collaboration is a possible way of creating semantic data, e.g., wiki-based systems such as OntoWiki [1], Ikewiki [2], or Semantic MediaWiki [3]. These wikis extend traditional wikis by functionalities that enable users to add annotations to a wiki page and to specify relationships between pages based on ontologies. In our opinion semantic wikis are far from being usable by non-experts. Besides the effort to learn a special syntax to write and to annotate content, a user has to cope with technical terms such as resource, different kinds of relationships, and namespaces.

Semantic tagging engines such as faviki [4] exploit the well-known user interaction procedure of tagging to annotate content. Faviki is based on Zemanta [5] and therefore retrieves suggestions for tags, e.g., Wikipedia terms. Since they do not directly create semantic data and do not interlink resources, they are useful for complementing semantic data. In the one-click annotation concept we use a similar approach for proposing annotations.

Similarly, OpenCalais [6] is a service for semantic text annotation. Using OpenCalais on texts, i.e., on RSS feeds, it allows for adding information about categories like IndustryTerms, NaturalFeature, and Company or linking to concepts of DBpedia like France and Country. Our approach of the One Click An-

notator assumes that a user can manually annotate his content, although he can also choose to get support of automatic annotation services. However, even in the case of automatic annotations the final decision of setting an annotation remains to the user.

The Tabulator linked data browser [7] allows users to edit data directly on the Web of data. However, since it requires a Firefox plug-in in its current stage of development we see it as a proprietary tool. In the context of linked data, OpenLink Data Spaces [8] provide a complete platform for creating a presence on the Web of data, e.g., calendar, weblog, or bookmark manager. While, the data is manipulated directly, we focus on the creation of semantic data on the basis of text content.

3 loomp – Semantic Content Authoring

The One Click Annotator (OCA) is currently developed in the context of loomp which is a lightweight platform to create, manage, and publish/sharing semantically enriched content. In our context enriching means to assign a concept to a word/phrase or to link a concept with another one. Since we address non-experts we think that other constructs such as OWL restrictions will not be understood.

In loomp we distinguish between two basic types of resources: fragments and mashups. A fragment is a closed notional information entity containing annotated text or a SPARQL query. Mashups are composed of an arbitrary number of fragments with a fixed order. For example, Figure 1 shows a mashup with the title Berlin which consists of two fragments. Both fragments and mashups are assigned unique URIs which are used for attaching metadata and to retrieve them. To comply with the linked data principles a user can link from fragments and mashups to other resources on the Web, e.g., concepts of DBpedia. The user interface for modifying mashups exploits modern web technologies to allow drag-and-drop and in-place editing of its fragments. A user can extend a mashup by creating a new fragment or by including an existing fragment (also from other users) at a desired place.

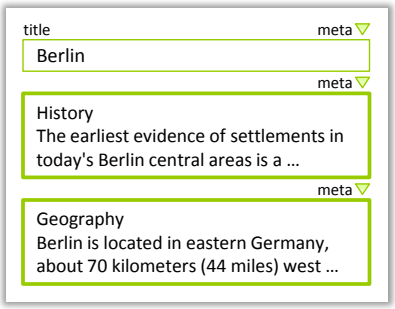

Fig. 1. Mashup with fragments

When a user annotates a fragment, the One Click Annotator extends the XHTML representation of the fragment with RDFa. When the editing is finished the RDFa of the mashup (including all fragments) is sent to the RDF backend on the loomp server, which performs some consistency checks and stores an RDF representation of the mashup, together with metadata (author, date, privacy status, etc.) as well as all annotated resources extracted from RDFa.

Featuring an integrated linked data server, loomp also provides direct access to the RDF data to a wide variety of existing Semantic Web applications like

RDF browsers, RDF crawlers, and query agents. The linked data server also provides a template-based HTML interface presenting loomp content in a human-readable form. Figure 2 illustrates that we focus on separating pure content (syntax) from meaning (semantics), and from presentation usage (pragmatics). As a result a user can view contents according to his information needs.

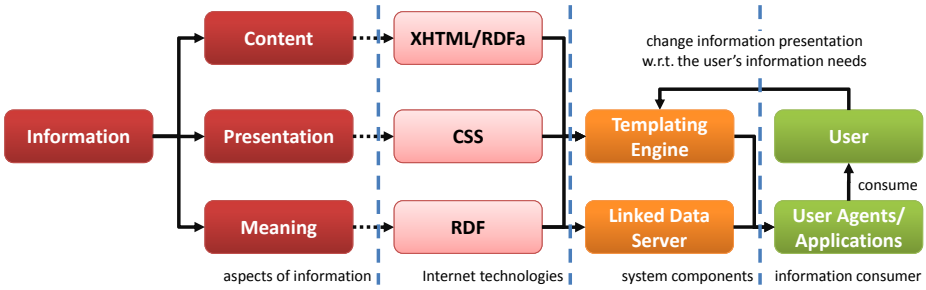

Fig. 2. The loomp architecture of information sharing

4 One Click Annotator

With the One Click Annotator (OCA) we aim at providing tool support for annotating content semantically and, thus, creating RDF data. This content and the resulting RDF data is intended to be shared with other users. For example, they could add further annotations, add links between resources, or access the RDF with a linked data client. To facilitate the dissemination of semantic data throughout the Web we consider non-experts who have little or no knowledge of semantic Web technologies as the primary target group of the OCA. Thus, the annotator has to hide the complexity of creating semantic data by providing an intuitive user interface. As a consequence, it follows common mindset and uses well-known components of existing user interfaces.

Although the OCA is developed in the context of loomp, it is implemented as a component that can be used independently from the loomp platform, e.g., it could be integrated into wiki systems. Thus, we describe the design of the OCA as a client communicating with some server.

In the following, we first describe the requirements on the OCA that are – from our viewpoint – important to enable non-expert users to annotate their content. As a result of these requirements we made some design decisions and had to solve some problems that we will discuss afterwards. Finally, we present deeper details on the implementation of the One Click Annotator.

4.1 Requirements

The main driver for our requirement analysis is to enable *non-expert users* to create semantic data. "Non-expert" means that a user is able to use a computer,

e.g., select some text, click a button, or tag content, but he has no knowledge about semantic Web technologies such as RDF, ontologies, rules, and linked data. We want to transfer the compelling simplicity of Web 2.0 applications to the task of creating semantically rich content and RDF data: light-weight, easy-to-use, and easy-to-understand. In the following list, we focus on design requirements which in our opinion are necessary to enable non-expert users to annotate their content and to benefit from semantic technologies.

Intuitive user interface. The annotator hides the complexity of creating semantic annotations by providing an intuitive user interface. It follows common mindset and uses procedures of system interaction known from word processors to produce the annotations.

Simple vocabularies. Although Web users know the term "URL" or "Internet address", most of them are not aware of namespaces. Thus, the annotator provides access to vocabularies without going into technical details and does not support complex constructs such as OWL restrictions. Each concept of a vocabulary has a meaningful label and is explained in simple terms and with examples of usage.

Focus on the user's task. Usually, a user wants to perform the task of writing some text and not to annotate content. Thus, the toolbar for creating semantic annotations is seamlessly integrated into a text editor, so that the user can add annotations without distracting himself from his primary task. Furthermore, the annotations do not blemish the editor pane.

Interoperability. To minimize the problems of interoperability the annotator is build on top of standards, e.g. to represent and to access annotated content. Regarding vocabularies, the annotator provides access to widely accepted vocabularies and is able to map concepts of equal or similar meaning.

Finally, before we started to develop the OCA we made the following two key design decisions on the implementation: The OCA has 1) to work with modern Web browsers, and 2) to store the annotated content and the annotations itself in a format that corresponds to widely accepted Web standards. As a consequence the OCA is independent of a certain backend implementation and can easily be integrated into other Web applications.

Due to the requirements we were confronted with the challenges as compiled in Table 1. The table also shows a short description of our solutions to meet them. In the following sections, we explain those challenges and solutions in detail.

4.2 User interface

With the objective of developing a tool for creating semantic annotations by non-experts fulfilling the above requirements, we design and implement the *One Click Annotator* (OCA). In the following, we describe the functionalities and the user interface of the One Click Annotator and discuss challenges we were confronted and propose solutions for them.

Challenge	Solution
Clear and intuitive UI	Transfer the look&feel of the toolbar for style sheets known from word processors.
Represent annotations in RDF(a)	Selected text is the value of a property assigned to a resource.
Reuse existing resources	Use the selected text to guess the resource to link to, but the user makes the decisions.
Create links between resources	Allow different ways of creating relationships and provide inverse properties.
"Don't repeat yourself."	Support the user with annotation services, but under control of him. Treat tables specially.
Compatibility and ease of integration	Implementation is based on XHTML+RDFa (client) and RDF (server)
Independence of domain ontologies	An RDF schema defines a set of annotations which can refer to well-known ontologies.

Table 1. Realizing the One Click Annotator: Challenges and solutions

Clear and intuitive UI. The first challenge addresses the following question: What user interface is suitable to create semantic annotations but is also clear and intuitive? We transferred the well-known concept of style sheets from modern word processors to the OCA (cf. dashed box in Figure 3). In a word processor, a user can choose between different sets of style sheets. After having chosen a set he can select some text and assign a style sheet to it, e.g., *heading 1*. In the OCA, a user can similarly choose between different vocabularies, e.g., personal information, geography, or events, and assign an annotation to some selected text, e.g., `geo:cityName`. Please note, that the toolbar only displays human understandable labels (stored as `rdfs:label`) to the user but and not the namespace prefixed property names as it can usually be seen in other applications.

As a consequence of the limited space in the UI, the annotation sets may only contain a small number of annotations. Large annotations sets would increase the effort of annotating due the need of scrolling. Thus, the vocabularies are especially designed for the use with the OCA. However, we believe that a user will usually work in a certain domain and will only use a limited set of annotations.

Represent annotations in RDF. The procedure of adding annotations to text leads to another challenge. In our opinion, a non-expert user is hardly aware of the difference between a concept and the label of a concept. For example, if a user reads the word *Berlin* in some text, he probably wants to state that *Berlin* is a city but not that it is the name of a city. We consider this behavior in the OCA by using the labels of the properties in the toolbar (in the following also referred to as annotation) and creating a resource that has a property with the selected text as value. In many cases it will also be possible to assign a type to the generated resource automatically, because the domain of the used property

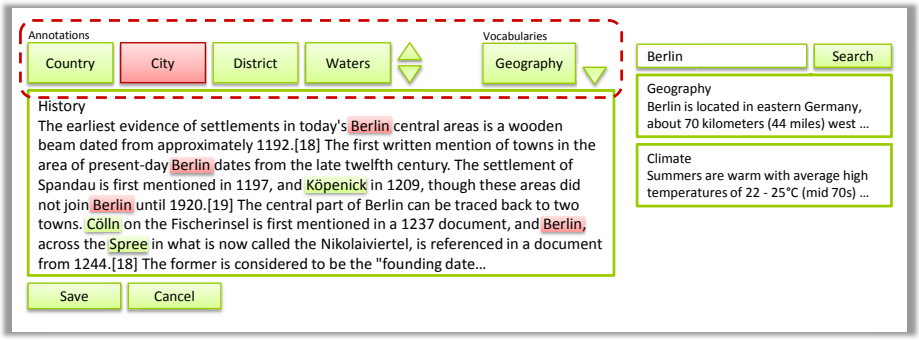

Fig. 3. The annotation toolbar of the One Click Annotator

is defined in the schema or in the annotation set, e.g., `geo:City` is the domain of `geo:cityName`.

Reuse existing resources and create links between them. If a user annotates the same resource in different texts, it is important to reference the same resource in the generated RDF statements. Otherwise, we get many resources that are not interlinked and, thus, the statements are not very meaningful. After the user has selected some text and clicked on an annotation in the toolbar, the system presents a list of existing resources that are likely to be chosen (see Figure 4). The resources may originate from a local or an external repository, e.g., DBpedia.

To retrieve the appropriate resources from the repositories we use three sources of information: 1) the selected text, 2) the schema information stored in the vocabulary definition, and 3) annotation services such as OpenCalais. To enable the user to choose an appropriate resource, we display other properties of the resources in the list and the user can view the text fragments where these resources occur. At any time we try to avoid to display the URI to the user, because they are generated and do not provide any meaning to the user. Instead, we use the values of properties of a resource, i.e., the annotated text. If he does not find the resource in the list then he can manually search for a resource or create a new one.

Assigning an annotation to a selected text is only one facet of creating semantic annotations. The other facet is the creation of relationships between resources. To simplify the creation of links between resources we offer different approaches: 1) Drag an annotated resource, drop it onto another one, and select a property or 2) use a context menu associated with an annotated text to choose a property and a resource to link to.

"Don't repeat yourself." Annotating text is a time-consuming task that, in most cases, users do not like to perform. According to the principle "don't repeat yourself" the user can invoke annotation services which automatically analyze the text and propose annotations. Besides external annotation services such as OpenCalais we develop a client-side component that creates annotation

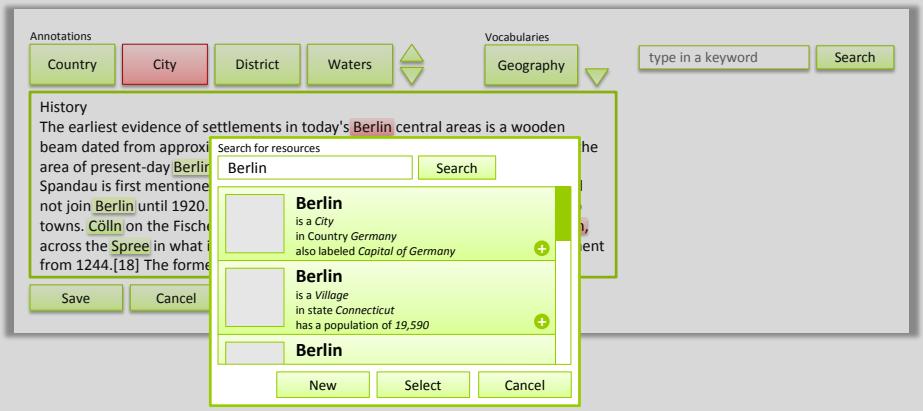

Fig. 4. Mock-up of the user interface for selecting an existing resource

while the user is typing the text. This component keeps track of the annotations created by the user and adds the same annotations to other parts of the text. Independent of the used annotation service, an icon is placed behind an automatically generated annotation which opens a select box when the user clicks on it (see Figure 5). He can then decide to accept, to reject, to choose an alternative annotation, or to disable automatic annotation for a phrase.

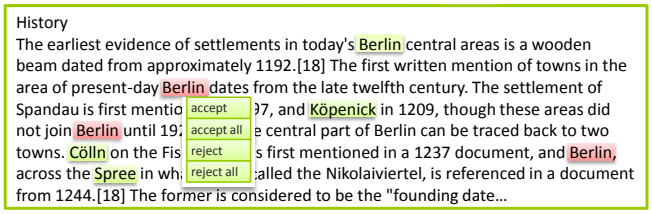

Fig. 5. Mock-up of the user interface for accepting or rejecting an annotation

4.3 Technologies

The One Click Annotator is the client part of a client server architecture for semantic content authoring. It displays the content and allows to modify and to enrich it semantically. The server is only contacted if the OCA needs information about resources or has to invoke external annotation services. In this section we describe details about the processing of the user's content and the creation of semantic data.

Compatibility and ease of integration. The OCA is implemented as a plug-in of the tinyMCE [9] which is a WYSIWYG editor for Web browsers. The

tinyMCE and, thus, the OCA plug-in is only able to handle HTML content. As a consequence of this and due to our requirements of a high compatibility with and an easy integration into existing Web applications, the OCA consumes and produces XHTML+RDFa [10]. The server can extract the RDF from the XHTML+RDFa content (e.g., RDFa Distiller [11] or ARC [12]), link it to RDF data of other sources, and serve it as linked data.

Convert annotations to RDF(a). The procedure of annotating some text is as follows: After a user has selected some text and clicked on the button of an annotation, the OCA queries the server for resources that the user may link the text to. If the user does not find an appropriate resource, he may manually search for it or create a new one. Finally, the OCA inserts a *span* tag containing the semantic data as its attributes into the XHTML content.For example, Figure 6 shows a snippet of a XHTML+RDFa document stating that there is a resource with the name of a city *Berlin*. When the user saves the content then it is sent to the server for further processing.

As a drawback of using an XML based format, the user cannot annotate partially overlapping phrases. However it is possible to annotate phrases where one includes the other.

```
<span about="http://www.loomp.org/resource/fdff8477..."
      property="http://.../geography#cityName">Berlin</span>
```

Fig. 6. Example of an RDFa annotation

Independence of domain ontologies. The One Click Annotator is extensible with respect to the usable ontologies. A domain is usually described by several ontologies. Since we also want to hide the complexity of choosing an ontology and of mapping different ontologies, we decided to develop a simple RDF schema for describing the elements of the annotation toolbar. Using this schema an annotation is described by the referred property of a standard ontology (e.g., Dublin Core, FOAF, etc.), a label for displaying in the toolbar, a description of the annotation, and examples of its usage. Furthermore, the engineer of an annotation toolbar can define the domain of a property which is assigned to a new resource. It is useful to assign a more specific type to the resource than defined in the original ontology. For example, the domain of the property `foaf:name` is `owl:Thing` [13]. In the context of a toolbar about personal information it is more meaningful to create a resource of type `foaf:Person` or `foaf:Agent`.

5 Conclusion and Outlook

A key prerequisite in the vision of a Semantic Web providing machine readable linked data is to have enough semantically enriched data on the Web. Currently, the burden for ordinary Web users to create such semantic data is much too high since there are no easy-to-use Semantic Web authoring tools.

In this paper, we introduced a simple-to-use "One Click Annotator" (OCA) for enriching texts with RDFa annotations and linking resources. The resulting contribution to the Semantic Web aims at a quantitative increase of semantic data on the Web. The OCA has been implemented as a part of loomp – a lightweight linked data authoring platform for non-expert Web users. However, the OCA has been design independently of loomp and may easily be integrated into other Web applications.

In the future we focus on automatic generation of links to external resources (e.g., to the Linking Open Data dataset cloud), improved integration of external annotation services such as OpenCalais, and privacy issues. The latest version of loomp can be tested at `http://loomp.org/`.

References

1. Auer, S., Dietzold, S., Riechert, T.: OntoWiki - A Tool for Social, Semantic Collaboration. In: Proceedings of the 5th ISWC. Volume 4273 of Lecture Notes in Computer Science., Springer (2006) 736–749
2. Schaffert, S.: Ikewiki: A semantic wiki for collaborative knowledge management. In: 1st International Workshop on Semantic Technologies in Collaborative Applications (STICA'06), Manchester, UK (June 2006)
3. Krötzsch, M., Vrandecić, D., Völkel, M.: Semantic MediaWiki. Lecture Notes in Computer Science. In: The Semantic Web - ISWC 2006. Springer Verlag (2006) 935–942
4. faviki: faviki - tags that make sense. http://www.faviki.com Visited on June 4th, 2009.
5. Zemanta Ltd.: Zemanta. http://www.zemanta.com/ Visited on June 4th, 2009.
6. Reuters, T.: Opencalais. http://www.opencalais.com Visited on June 4th, 2009.
7. Berners-Lee, T., Hollenbach, J., Lu, K., Presbrey, J., D'ommeaux, P.E., Schraefel, M.: Tabulator redux: Writing into the semantic web. Technical report, School of Electronics and Computer Science, University of Southampton (2007)
8. OpenLink Software: Openlink data spaces (ods). http://virtuoso.openlinksw.com/wiki/main/Main/Ods Visited on June 4th, 2009.
9. Moxiecode Systems AB: Tinymce. http://tinymce.moxiecode.com/ Visited on June 4th, 2009.
10. Adida, B., Birbeck, M.: RDFa primer – bridging the human and data webs. W3C Working Group Note (October 2008)
11. Herman, I.: Rdfa distiller. http://www.w3.org/2007/08/pyRdfa/ Visited on June 4th, 2009.
12. Nowack, B.: Easy rdf and sparql for lamp systems. http://arc.semsol.org/ Visited on June 4th, 2009.
13. Brickley, D., Miller, L.: Foaf vocabulary specification 0.91. http://xmlns.com/foaf/spec/20070524.html (May 2007)