

Improving Human–Semantic Web Interaction: The Rhizomer Experience

Roberto García and Rosa Gil

Abstract—Rhizomer is a generic semantic metadata browser and editor. It is based on “classic” HTML paradigms (table, form, etc.) in order to provide an easy and accessible user experience. For end-users, it looks like the web pages they are used to deal with. However, the semantics that lay behind are used to improve the user experience through an AJAX-enhanced web interface. The user interface is fed by a metadata storage abstraction layer, which offers a SPARQL endpoint and implements the DESCRIBE SPARQL primitive in order to improve the usability of the resulting metadata fragments. The metadata component is combined with a wiki engine that helps managing the content intended for human consumption. Altogether, both components build a simple yet powerful platform for Semantic Web portals.

Index Terms— human factors, Semantic Web, user interfaces, web portal

I. INTRODUCTION

THE SEMANTIC WEB has been around for some time and many people is asking why it has not taken off as quickly as the World Wide Web did [1].

One of the main problems is that it is not reaching the end-users, who can give it the required critical mass for widespread adoption. In this sense, one of the main impediments is that users find Semantic Web applications very hard to use, i.e. they lack usability. This is true even for researchers and practitioners working in the Semantic Web field [2].

Human-Computer Interaction (HCI) is a multidisciplinary effort to improve the human-computer interface. The focus is placed on user, i.e. to take into account user needs from the beginning and through all the development process, and the objective is to get usable and accessible products.

Usability is defined as the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. Accessibility is used to describe the degree to which a system is usable by as many people as possible without modification and it specifically focuses on people with disabilities.

Roberto García is with the GRIHO Human-Computer Interaction Research Group at the Universitat de Lleida, Lleida, E-25001 Spain (phone: +34 973-702-740; fax: +34 973-702-702; e-mail: rgarcia@diei.udl.es).

Rosa Gil is with the GRIHO Human-Computer Interaction Research Group at the Universitat de Lleida, Lleida, E-25001 Spain (e-mail: rgil@diei.udl.es).

The methodology for developing usable and accessible application is called User-Centred Design (UCD). It is based on an iterative development process based on a detailed study of the users’ needs, the tasks they carry on in order to meet them and the context in which they are performed [3].

UCD, as other software development processes, starts with the requirements gathering phase. However, the focus is placed specially on users. First of all, it is important to know who the users are. Then, the following step is to identify the tasks they are going to perform.

The development process continues with the common phases, i.e. design, implementation and deployment. Despite these similarities, the focus continues to be placed on the user.

In order to keep user needs present during the whole development process, the previous phases are complemented with two additional ones that are performed in parallel, for each process iteration, prototyping and evaluation.

Prototypes are created from the beginning, for instance paper prototypes [4], which do not require any implementation, or simple applications with limited functionality.

All of them are used to evaluate the system with users so their requirements are taken into account and contrasted with the developed system just from the beginning and through all the development process iterations.

The evaluation is performed in a controlled environment, usually a usability laboratory, where specialised software applications are used to record and analyse the whole interaction, i.e. screen capture, key strokes, mouse clicks, user video record and voice, etc.

The User-Centred Design methodology has been employed in order to develop a usable and accessible Semantic Web metadata browser and editor. The process is sketched in Section II. Then, in Section III, the solution for metadata browsing is detailed and, in Section IV, it is shown how metadata edition has been faced. Finally, Section V presents the conclusions and the future work directions.

II. USER-CENTRED DESIGN

Following the UCD methodology, the first step has been to identify the target users and their tasks. A generic study of Semantic Web users and tasks by Lisa Battle has been used [5]. Three broad user groups are proposed, together with the main categories of tasks they would perform in order to use the Semantic Web, as it is shown in Table 1.

Table 1. Some proposed user groups and task types for the Semantic Web [5]

User Group	Task Types
1. End users	Information seeking tasks Information synthesis tasks Action-oriented tasks Information sharing tasks
2. Content curators	Content update tasks Content distribution tasks
3. Ontologists	Ontology update tasks Ontology creation & mapping

For UCD it is important to clearly identify the target user. In our case, the objective is to develop a Semantic Web instance metadata application so the target is end users.

The end users profile characterises common users with no knowledge about the Semantic Web. They are used to web applications so the resulting application must look like the kind of appliances they are used to.

However, following the current tendency that the Web is not just a single way communication medium like wikis and blogs demonstrate, we do not keep a separate group of tasks for the content curators' user profile.

Consequently, although we acknowledge the existence of users specialised in content update and distribution tasks, we consider that our target end users should have the possibility to edit metadata. Therefore, we consider the following kinds of tasks:

- Information-seeking tasks, e.g. to look for a restaurant near the theater that will still be open when the movie is over.
- Information-synthesis tasks, e.g. to organise the agenda of a conference attendant.
- Action-Oriented Tasks, e.g. to build a personalized portal to manage research tasks.
- Information-Sharing Tasks, e.g. to share pictures with friends and family.
- Content update tasks, e.g. to add new books to a catalog of published books and edit the metadata of previously added ones.

Following this user profile and tasks selection, the design of a metadata browsing and edition component called Rhizomer continued. The Rhizomer project constitutes a technological framework that can be used to build up semantic web portals.

Rhizomer manages RDF metadata in a user-friendly way. It facilitates not just the common Semantic Web to end-user interaction provided by semantic web browsers, it also provides the reverse interaction path: end-users can create, edit and remove semantic metadata.

The whole interaction is performed through a "classical" HTML interface in a usable way, which minimises user efforts and maximises the benefits they obtain from their Semantic Web experiences. The overall intention is to minimize the gap among computers and human beings in the context of Semantic Web.

The semantic metadata management part provided by

Rhizomer is combined with an easy to use and simple content management system based on a wiki engine [6]. The wiki provides the means to create and maintain information objects intended for human consumption. In this case, HTML is better suited so it is used instead of semantic metadata.

Therefore, Rhizomer is used just for semantic metadata management while the wiki is used for HTML content. Semantic metadata might be used to describe resources, among them the wiki documents but also other documents and resources.

The Rhizomik web site¹ is based on Rhizomer and a wiki engine and constitutes an example of how they can be used to produce a semantic portal for research purposes.

III. SEMANTIC METADATA BROWSING

Usability guides how end-user interaction with the Semantic Web is faced by Rhizomer. This interaction is commonly viewed, when talking about the Semantic Web, just from the Semantic Web application towards the user, i.e. what can be called Semantic Web browsing, which is detailed in this section.

This area is being extensively explored in the Semantic Web area. Many approaches are based on a graph paradigm, i.e. the user interacts with a nodes and links graphical representation. As it has been pointed out, this is not the best choice as it is not natural to force the user to interact with semantic data through the same model that is used for its representation [7].

The graph model might be useful for the user in some very specific scenario, e.g. in order to get a quick view of how some data is distributed, but even this is not usually the case for Semantic Web data.

Consequently, the latest developments for semantic metadata browsing are based on different paradigms. Almost in all cases, they are based on a browsing paradigm that can be called the "Subject-centric Approach". This approach is described in Section B.

The "Subject-centric Approach" has some usability problems that have motivated us a slightly modified approach. Our proposal is detailed in section C and it is employed in Rhizomer in order to construct a more user-friendly experience when browsing the Semantic Web. The Rhizomer browsing approach is based on a simple algorithm, detailed in sections D and E, and it is compared to the subject-centric one in section F.

There are other browsing approaches like the table paradigm used in mSpace [8]. However, this table-based system must be configured in order to operate with a concrete set of data. Consequently, it does not constitute a generic paradigm for semantic metadata browsing.

Finally, the unidirectional experience provided by Semantic Web browsers is not enough. The other direction must be considered and the user should be able to create, edit and

¹ Rhizomik initiative, <http://rhizomik.net>

remove semantic metadata in a usable way. This part is explained in Section IV.

A. World Wide Web browsing paradigm

In the Web, the browsing paradigm is based on navigating web pages and links, which constitute its basic building blocks. Web pages have content intended for human consumption and links relate web pages. Web pages content and links are based on HTML and derived languages. Their design should have accessibility and usability principles in mind.

This browsing paradigm cannot be directly applied to the Semantic Web because it is based on a different ground model. The building block of the Semantic Web is the triple $\langle \text{subject}, \text{predicate}, \text{object} \rangle$, which combined builds-up graphs.

Semantic Web metadata is primarily intended for machine consumption. However, it is clear that it must be also accessible for human users in order to facilitate semantic web applications debugging, results presentation, querying, etc.

However, the WWW browsing paradigm cannot be applied to the Semantic Web. Semantic Web metadata can be packed in web documents, e.g. a web-accessible XML file; however it is not practical to use this approach as the basis for Semantic Web browsing.

First, not all metadata is available as web documents. Moreover, semantically related metadata might be packed in different web documents, which reduces the benefits of semantics-enabled metadata browsing.

Usually, Semantic Web metadata is available from databases. In this case, as it can be also the case for web documents, the amount of metadata is too big for human-user consumption.

Therefore, what is needed is a Semantic Web browsing paradigm and support system that can browse metadata coming from different sources through semantically coherent fragments that facilitate human consumption. Now the question is: how to define fragments?

B. Subject-centric Approach

The simpler approach to fragment semantic web graphs is to define the fragment as the set of all triples with the same subject. This is also semantically coherent because these triples are those describing the subject resource.

This approach is used in almost all Semantic Web browsers. For instance, Piggy Bank [9] or Brownsauce [10] generate HTML views for a given resource, i.e. a node of the graph. Table 2 shows an example of such kind of view.

Table 2. HTML Table view of a metadata fragment

Rosa Gil	
EMAIL	rgil@diei.udl.es
FN	Rosa Gil
N	(anonymous item)
Show Referrers	

The view contains all the triples that have the resource as subject and it is shown as a HTML table. The table header shows the identifier of the described resource, i.e. the subject for all triples in the metadata fragment. The following table rows contain two columns. The first one is for each triple property and the second one for the corresponding objects.

Tabulator [11] also uses an approach similar to the Subject-centric one. However, metadata about resources, anonymous or not, is recursively expanded as a tree. Therefore, it is possible to expand the properties about an anonymous resource in the same place where the anonymous resource is referred. However, this approach leads to deep tree expansion that mix metadata about different resources, which might confuse users and make them loose track of the resource that is being described.

There are also other frameworks that are not explicitly oriented to RDF browsing and provide heavyweight solutions that can be adapted to navigation, e.g. the SEAL framework [12]. Moreover, there are other Semantic Web navigation tools that generate non-HTML views, e.g. Haystack [13] generates text views and [14] graphical representations.

In many cases the shown metadata fragment is augmented with triples that have the described resource as object, i.e. reverse triples that do not have the described resource as source but as destination.

Additionally, more triple levels can be included, i.e. the triples that have the objects of the original triples as subject, and thus recursively. However, it is not common to include additional triple levels because it can make the fragments too big and break semantic coherence, i.e. the user loses the perspective about what is being described.

Finally, the semantic coherence is also lost when this approach is used and anonymous resources are involved. Anonymous nodes get identified by the context in which they appear, i.e. the triples that reference them and the identified resources appearing in these triples.

However, this context is broken when the metadata fragments are built. Anonymous nodes get temporal identifiers, which are needed in order to build the graph, but they are also used in the user interface as if anonymous nodes were like other identified resources.

For example, Table 2 shows a reference to an anonymous resource as the value of the “N” property. An additional browsing step is necessary to get the triples describing the anonymous value and make it all semantically coherent, as it is shown in Table 3.

Table 3. Additional step for anonymous resource metadata

(anonymous item)	
Given	Rosa
Family	Gil
Show Referrers	

Consequently, the same approach is applied to anonymous resources. All the triples that have the anonymous resource as

subject are shown. Therefore, the metadata describing the anonymous resource is shown but the identification context is lost in the presented view. There might be reverse triples to go to the identified resource that defines the identification context for the anonymous resource. In any case, however, the presented view does not include the whole context so the user might not be aware of this fact.

For example, one common examples of this situation is when RDF containers are used. They do not usually get an identifier so the resources for the containers are anonymous. Therefore, when the metadata for the resource described with container values is shown, the set of container values are not shown together with the resource to which they are associated.

In order to solve the inconveniences of the Subject-centric approach we propose the Rhizomer Approach that is detailed in the next section.

C. Rhizomer Approach

The Rhizomer Approach for Semantic Web browsing is also based on fragmenting the metadata graph in a subject-centric way. However, a metadata fragment generated by Rhizomer is more than just the considered resource and all the triples in which it participates as the subject.

The set of triples for a subject-centric is enlarged with all the metadata that depends on the selected subject for its identification. Therefore, the graph is traversed starting from the resource acting as the subject through all possible paths until identified resources or literals, which include all the intermediate anonymous resources. All the traversed triples constitute a Rhizomer metadata fragment and it is what is shown at a browsing step. Figure 1 shows a simple RDF graph where four fragments can be identified.

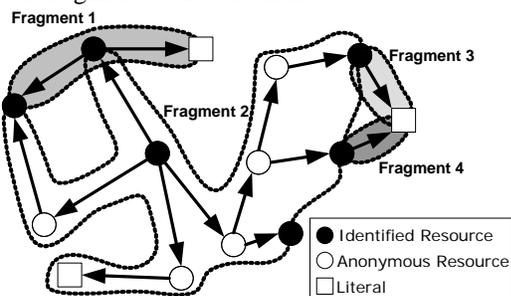


Figure 1. RDF graph and four Rhizomer fragments

As in the case of the subject-centric approach, the resources appearing as the object of the terminal triples can be browsed through new navigation steps that generate new metadata fragments describing the resources asked for detail. This has been already shown in Table 3, which shows a browsing step initiated from the Table 2 fragment.

However, Rhizomer shows all the metadata in the same identification context together. This way, a greater level of semantic coherence is maintained and the user experience is improved. Anonymous resources do not break down the metadata describing a resource and they are shown together with the resources that contextualise them, as it is shown in Table 4.

Table 4. HTML Table view of a metadata fragment

Rosa Gil		
EMAIL	rgil@diei.udl.es	
FN	Rosa Gil	
N	Given	Rosa
	Family	Gil
Show Referrers		

To conclude the presentation of the Rhizomer approach, it is important to take into consideration the presence of cycles in the metadata graph and to avoid considering triples already added to a fragment. This can be easily implemented as it is shown in the algorithm in the next section.

D. Algorithm

The browsing fragments are built from a set of selected resources. For each resource, all the triples where the resource plays the subject role are selected. Then, for each object of the selected triples, if it is an anonymous resource, all the triples where the anonymous resource is the subject are also selected. Then, and recursively, the same procedure is applied to the new object anonymous resources. The algorithm for building the fragment for a given resource is detailed in Table 5.

Table 5. Rhizomer algorithm for fragments browsing

```

TripleSet buildFragment(Resource r, Triples f)
{
  TripleSet rTriples = TriplesWithSubject(r)
  fragment += rTriples
  for each Triple t in rTriples such that NotInSet(t, fragment)
  {
    fragment.add(t)
    if anonymous(t.object)
    {
      TripleSet anonTriples = buildFragment(o, fragment)
      fragment.add(anonTriples)
    }
  }
  return fragment
}

```

E. Multi-language Support

When a metadata fragment is rendered as HTML or other format, the triple URIs are replaced with their corresponding labels if available. If there is not a defined label for the URI, it is shortened to its fragment identifier, the string after the '#', or to the substring after the last '/'.

This is done in order to improve readability. An additional improvement, which is not common in existent Semantic Web browsers, is multi-language support. The Rhizomer algorithm, in addition to the triples that compose the fragment, selects all the triples that define labels for all the involved resources, i.e. subjects, predicates and objects.

These labels use to be annotated with a language attribute. When the fragment is rendered and a preferred language specified, the preferred language labels are selected if they are available. If not, the default language has been set to English

so the labels with the ‘en’ value for their language attribute are selected when available. If there is not any label with the preferred or default language attribute, a label without language attribute is used.

F. Comparative Study

As it has been previously shown, the subject-centric approach is the common method for building fragments for metadata browsing. It has been also shown how the Rhizomer approach works and how it builds more coherent fragments by incorporating all the metadata identified by the described resource, i.e. the anonymous resources related to it.

It might seem that the Rhizomer approach makes the fragments too big for human user consumption. A statistical study has been performed with different sets of Semantic Web metadata. The results of this analysis are shown in Table 6.

As it can be seen from the statistical analysis, there is a slight increase in the size of the metadata fragments when the Rhizomer approach is used. It is greater when there is an intensive use of anonymous resources, for instance in ontologies because class restrictions do not use to be identified or in common real world metadata because we do not use to give identifiers to every resource.

Table 6. Average fragment size in triples

Metadata Set	Set Size	Subject Centric	Rhizomer
Musicbrainz ²	8121	6.6	7.7
CIA World Factbook ³	876	4.3	14.4
Wine Ontology ⁴	1839	2.6	7.3

In the case of the Musicbrainz metadata, the difference between the subject-centric and Rhizomer approach is smaller because there are few anonymous resources. Musicbrainz describes artists, albums, tracks, etc. and all of them must have an identifier in order to be easily referred. The only anonymous resources are containers, which are used, for instance, to group all the album tracks.

This slight increase in size of the fragments does not put them out of the human user reach. On the contrary, it facilitates browsing because fewer steps are required. Moreover, all the metadata related to the same identifier is shown together and this increases its coherence and the usability of the whole system.

It is also important to consider maximum fragment sizes for these metadata sets. For the Wine Ontology, the maximum fragment size is 42 triples, which corresponds to the “CavernetSauvignon” class definition. This is a usable size as it can be rendered as HTML using the method detailed in Section G in a 370x540 pixels area of a web browser. For the CIA Factbook metadata about Spain, the maximum fragment size is 549. This is a really big fragment, which requires a lot

of browsing from the user. In any case, all browsing approaches will suffer this problem because it corresponds to the description of the resource describing the whole country and all triples in this fragment are directly attached to it. Finally, the maximum fragment size for the Music Brainz U2 discography is 170 triples. As in the previous case, it corresponds to the U2 concept that directly links to all its discography. Consequently, other browsing approaches will also require a lot of browsing from the user.

As it has been pointed out, the inclusion of the anonymous resources together with the resource that identifies them is not common in the existing Semantic Web browsers. However, we have found a similar approach in the DBin project [15]. This project defines the term called RDFN (RDF Neighbourhood) that is based on a similar approach. The main difference is that this approach is specially tailored for producing digital signatures of pieces of a RDF graph and that it does not consider triples directionality.

Therefore, it is not suited for Semantic Web browsing but it shows that the best way of segmenting a graph is using an approach based on a subject-centric approach plus the anonymous resources identified by the subject.

Another much more similar approach is Concise Bounded Descriptions (CBD) [16] enriched with back links, i.e. links pointing back to the subject being described.

G. HTML Metadata Rendering

Once the metadata fragments have been generated, they are shown to the user as an HTML rendering in the web browser. This rendering allows getting a user interface users are comfortable with. It looks like the kind of webs they are used to interact with.

In order to produce this rendering, a generic XSL transformation from RDF/XML to HTML has been developed. This approach produces consistent results as long as the metadata fragments to render are generated with the Rhizomer approach and an abbreviated RDF/XML serialisation of them is produced.

The abbreviated serialisation produces an XML stream that keeps all the related triples grouped. Consequently, it is possible to render them as a set of HTML tables, one for each resource being described, with nested tables corresponding to the descriptions of the anonymous resources they refer to.

Resource and property names are rendered as text using the appropriate label for the preferred language if available, as it has been detailed in Section E. Literals are also rendered as text and, if different language versions are available, the preferred one is selected. Finally, there are HTML links for all resource and property names that allow browsing the metadata describing them. These links correspond to calls to a SPARQL endpoint, concretely they are DESCRIBE queries. This kind of queries is resolved in a proprietary way in order to generate Rhizomer metadata fragments and to enrich them with the RDF labels that make the multilingual rendering possible.

The resulting HTML pages look like the simple example previously shown in Table 4, or the screen capture from the

² <http://musicbrainz.org>, U2 discography

³ Factbook for Spain, <http://www.daml.org/2003/09/factbook/sp>

⁴ <http://www.w3.org/TR/owl-guide/wine.rdf>

Rhizomik site shown in Figure 2. Moreover, it can be tested from the ReDeFer web page [5] for arbitrary metadata sets.

A related approach to render RDF as HTML is Fresnel [6]. Fresnel lenses are specifications about how to render some resources, classes and properties as HTML or other presentation languages. They allow a great level of personalisation but they require that the corresponding lens has been specified in order to generate a rendering. However, it is also possible to create a generic lens for unforeseen kinds of metadata.

On the other hand, the XSL-based approach used in Rhizomer does not allow this level of personalisation but it is capable of dealing, in a very generic way, with any piece of metadata it encounters. In any case, if a greater level of personalisation is required, the Fresnel lenses rendering engine can be integrated in the Rhizomer platform.

Consequently, the Rhizomer rendering approach added value is that it is very easy to implement, it just requires an XSL processor so this work load can be put away from the web server and passed to the user web browser. Moreover, it can be managed with AJAX [17] so a greater level of interaction through the browser can be achieved.

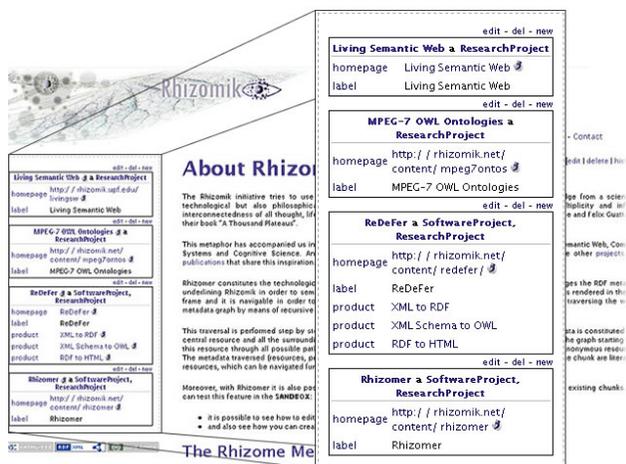


Figure 2. Rhizomik interface with Rhizomer zoom

IV. METADATA EDITION

The browsing capabilities already shown provide a simple Semantic Web user interface. It is simple because the interaction is limited to selecting the piece of metadata to show next. However, the user cannot take this interaction further.

In order to improve user experience, the browsing capabilities have been enriched with editing and querying functionalities based on semantics-enabled forms. These are common HTML forms that take advantage of some simple conventions to make them semantics-enabled.

Moreover, they are automatically generated from RDF

metadata using a XSL transformation. Therefore, it is possible to generate a query form from an example piece of metadata. Table 7 shows an example of a form to build semantic queries.

Table 7. Semantics-enabled query form

```
<form method="QUERY" onSubmit="buildSPARQL">
  <input type="text" name=
    "http://purl.org/dc/elements/.../title"/>
  <select name="http://purl.org/dc/.../subject">
    <option value=
      "http://www.iptc.org/subjects#01011000">
      ...
    </select>
    ...
  </form>
```

As it can be seen in the form, the names of the form fields are URIs coming from RDF schemas and web ontologies. The intention is to make the form fields the building blocks to generate triples from forms. The fields' names are the predicates and their values the objects. The subject, in the case of a query, is not known and so it does not need to be specified. In fact, the resource URI will be the response we will get from the query.

The filled form fields are interpreted as the known terms we use to perform the search. For instance, the form in Table 7 can be filled with the "Corporate news" value for the input field, i.e. Dublin Core title, and the first option of the select field, i.e. music is the subject. In the case of a query form, the fields are interpreted as the triples that build up the query pattern of a SPARQL⁷ query. A query like the one shown in Table 8 will be generated by a Javascript from the form values as a result of an "on submit" event.

Table 8. SPARQL semantic query for Table 7 form

```
DESCRIBE ?s
WHERE {
?s http://purl.org/dc/.../title ?y0;
  http://purl.org/dc/.../subject ?y1.
FILTER regex(?y0, "Corporate news") &&
?y1=<http://www.iptc.org/subjects#01011000>
}
```

If the objective is to edit metadata, then the form fields are interpreted as the building blocks for a set of triples for the new metadata. In other words, edit forms are used to edit new or existing metadata. Now, it is necessary to specify the subject for the triples. This is done using a field named "rdf:ID" or "rdf:about". Moreover, there might be also other subjects for the anonymous resources included in the edited metadata fragment.

The anonymous subjects are defined using hidden form fields named "rdf:ID" and valued with a temporal identifier just to make possible to build the graph. When all the triples for the anonymous subject have been specified, and another anonymous subject or the main identified subject is to be described, a new hidden field with the "rdf:ID" of the subject of the following triples is introduced. Table 9 shows an example of a metadata edition form.

⁵ ReDeFer project web page, <http://rhizomik.net/redefer>

⁶ Fresnel - Display Vocabulary, <http://www.w3.org/2005/04/fresnel-info>

Table 9. Semantics-enabled edit form

```

<form method="POST" onSubmit="buildTriples"... >
<input name="...rdf-syntax-ns#ID" type="text"/>
<input name="...vcard-rdf/3.0#EMAIL" type="text"/>
<input name="...vcard-rdf/3.0#N" type="text"/>
<input name="...rdf-syntax-ns#ID" type="hidden"
  value="_:anonid1"/>
<input name="...vcard-rdf/3.0#Given" type="text"/>
<input name="...vcard-rdf/3.0#Family" type="text"/>
<input name="...rdf-syntax-ns#ID" type="hidden"
  value=""/>
<input name="...vcard-rdf/3.0#FN" type="text"/>
...
</form>

```

If the fields in the Table 9 form are filled with the appropriate values, e.g. “http://rhizomik.net/~rosa” for the non-hidden “rdf:ID” field, when it is submitted, the metadata shown in Table 10 is generated applying a direct conversion from form fields to triples.

Table 10. Metadata from Table 9 edit form submit

```

<http://rhizomik.net/~rosa>
vcard:EMAIL

<mailto:rosa.gil@diei.udl.es>;
vcard:N [
  vcard:Family "Gil";
  vcard:Given "Rosa" ];
vcard:FN "Rosa Gil".

```

Therefore, semantics-enabled forms facilitate a greater level of interaction with metadata through a Semantic Web browser. In addition to metadata browsing based on Rhizomer-like fragments, it is also possible to edit such fragments or create new ones using the same semantics-enabled forms.

As it has been shown, a direct parallelism can be established from form fields to triples, so the new or edited metadata can be generated from the user interaction with the form when it is submitted. Moreover, the reverse way is also direct, from triples to form fields. Therefore, the edition forms can be generated from existing metadata in order to edit it or generate new metadata based on predefined patterns.

This two-ways mapping has been implemented in the Rhizomik semantic portal (<http://rhizomik.net>). The form fields to triples transformation is implemented when the form is submitted using JavaScript. The RDF triples to form transformation is implemented using an XSL transformation. The implementation details are not given here due to space limitations but they can be obtained from the Rhizomik site. Figure 3 shows a form generated automatically from the RDF metadata shown in Table 10 in order to edit it.

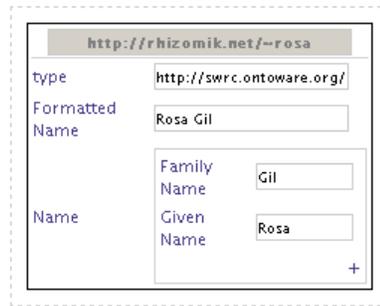


Figure 3. Automatically generated metadata edition form

V. CONCLUSIONS AND FUTURE WORK

As it has been shown, Rhizomer uses a common HTML interface. This approach is usual in Semantic Web browsers but it is enhanced by making it more semantically coherent. The navigation is performed by metadata fragments that are fully contextualised in the frame of at least one resource that identifies it.

Therefore, no additional browsing steps are needed in order to get to the metadata associated to anonymous resources and no metadata is presented without an identification context, as it is the common case when showing the metadata associated to anonymous resources.

In addition, in order to the improved browsing experience, the user has a more interactive experience thanks to an additional set of features. Users can create, edit and remove semantic metadata as it is browsed. This additional functionality is also available through a common web interface based on HTML forms.

The same applies to content, which is also directly editable through the wiki. Therefore, the whole interaction is browser based and the user does not have to install anything else.

The intention is to improve the user experience so all this is currently being tested with real users in the context of a usability and accessibility laboratory. Actually, many of the design decisions during the Rhizomer development have been based on user test results.

For instance, all links in the HTML rendering that allow browsing new metadata fragments are not underlined. Just links that point to new HTML content are underlined. This approach helps user differentiate among metadata and content browsing links. As they are used to underlined links while navigating through html documents, the same behaviour is maintained in the HTML rendering.

Additionally, accessibility is being tested with screen readers. The results are quite user friendly because the generated HTML content is based on the available RDF labels and preferred languages and this makes it easier for text-to-speech applications.

A part from a more extensive user testing, future work concentrates on incorporating the assisted addition of properties and values depending on the type of the described resource. When the user edits a metadata fragment and tries to add a new property to describe it, a popup window presents a list of all the available properties, as it is shown in Figure 4.

⁷ SPARQL Query Language, <http://www.w3.org/TR/rdf-sparql-query>

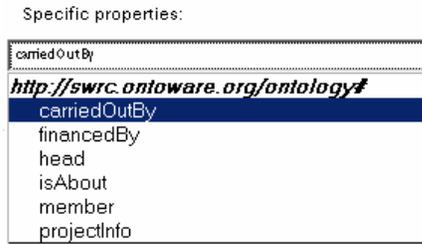


Figure 4. Assisted metadata creation popup for a SWRC Project resource

First, there are all the properties that are specific to the resource at hand, i.e. they are restricted to the resource type or their domain is one of the resource types. Second, all the properties that are generic, i.e. they have no domain defined or it is just restricted to any resource.

The same kind of assisted metadata creation is planned for property values. Altogether may facilitate user interaction with the underlying ontologies that structure the conceptual framework where semantic metadata is generated. In other words, the user will generate metadata following the restrictions defined by the corresponding ontologies without being aware of their existence, at least if the user does not want to.

REFERENCES

- [1] N. Shadbolt, T. Berners-Lee and W. Hall, "The Semantic Web revisited," *IEEE Intelligent Systems*, vol. 21, num. 3, pp. 96-101, 2006.
- [2] T. Heath, J. Domingue and P. Shabajee, "User interaction and uptake challenges to successfully deploying Semantic Web technologies," in *Proc. 3rd International Semantic Web User Interaction Workshop*, Athens, Georgia, USA, 2006.
- [3] D. Norman and S. Draper, *User-centered systems design: new perspectives on human-computer interaction*. Lawrence Erlbaum, 1986.
- [4] C. Snyder, *Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces*. Morgan Kaufmann, 2003.
- [5] L. Battle, "Leveraging work-centered semantics in human-computer applications," in *Proc. 3rd International Semantic Web User Interaction Workshop*, Athens, Georgia, USA, 2006.
- [6] B. Leuf and W. Cunningham, *The wiki way*. Addison-Wesley Longman, 2001.
- [7] D. Karger and m.c. schraefel, "The Pathetic Fallacy of RDF," in *Proc. 3rd International Semantic Web User Interaction Workshop*, Athens, Georgia, USA, 2006.
- [8] m.c. schraefel, M. Karam, S. Zhao, "mSpace: interaction design for user-determined, adaptable domain exploration in hypermedia," in *Proc. Workshop on Adaptive Hypermedia and Adaptive Web Based Systems*, Nottingham, UK, 2003.
- [9] D. Huynh, S. Mazzocchi and D. Karger, "Piggy Bank: experience the Semantic Web inside your web browser," in *Proc. The International Semantic Web Conference*, Springer LNCS 3729, 2005.
- [10] D. Steer, "Brown Sauce: An introduction," Technical Report HPL-2003-10, Hewlett-Packard Laboratories, Bristol, England, 2003.
- [11] T. Berners-Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer and D. Sheets, "Tabulator: Exploring and Analyzing linked data on the Semantic Web," in *Proc. 3rd International Semantic Web User Interaction Workshop*, Athens, Georgia, USA, 2006.
- [12] J. Hartmann and Y. Sure, "An Infrastructure for Scalable, Reliable Semantic Portals," *IEEE Intelligent Systems*, vol. 19, num. 3, pp. 58-65, 2004.
- [13] D. Quan and K.Karger, "How to make a semantic web browser," in *Proc. of the 13th International WWW Conference*, NYC, USA, 2004.
- [14] C. Sayers, "Node-centric RDF Graph Visualization," Technical Report HPL-2004-60, Hewlett-Packard Laboratories, Bristol, England, 2004.
- [15] G. Tummarello, C. Morbidoni, P. Puliti and F. Piazza, "The DBin Semantic Web platform: an overview," in *Proc. Workshop on The Semantic Computing Initiative*, Chiba, Japan, 2005.
- [16] P. Stickler, "CBD - Concise Bounded Description", *W3C Member Submission*, 3 June 2005.
- [17] D. Crane and E.Pascarella, *Ajax in Action*. Manning, 2005.