A Platform for Object-Action Semantic Web Interaction

Roberto García[1], Juan Manuel Gimeno[1], Ferran Perdrix[1,2], Rosa Gil[1], Marta Oliva[1]

[1] Universitat of Lleida
Jaume II, 69
25001 Lleida, Spain
{rgarcia, jmgimeno, ferranp, oliva, rgil}@diei.udl.cat

[2] Segre Media Group
Del Riu 6
25007 Lleida, Spain
fperdrix@diarisegre.com

Abstract. Semantic Web applications tests show that their usability is seriously compromised. This motivates the exploration of alternative interaction paradigms, different from the "traditional" Web or desktop applications ones. The Rhizomer platform is based on the object-action interaction paradigm, which is better suited for heterogeneous resource spaces such as those common in the Semantic Web. Resources, described by means of RDF metadata, correspond to the objects from the interaction point of view and Rhizomer provides browsing mechanisms for them. Semantic web services, dynamically associated to these objects, correspond to the actions. Rhizomer has been applied in the context of a media house to build an audiovisual content management system. End-users of this system, journalists and archivists, are able to navigate the content repository through semantic metadata describing content pieces and the domain knowledge these pieces are referring to. Those resources constitute the objects to which, when the user selects one of them, semantic web services dynamically associate specialized visualization and interaction views, the actions.

1. Introduction

The success of the Semantic Web depends, in a great measure, on its adoption by a critical mass of end users. Nowadays, this has not happened yet and, as some reports point out [1], this is due in part to the fact that end users find it very difficult to use. Even researches and advanced users of the Semantic Web find it complicated [2].

The Human Computer Interaction (HCI) discipline proposes a methodology specially focused on this purpose: User Centred Design (UCD). The user needs are taken into account from the beginning and throughout the whole development process, with the aim of obtaining usable products. Usability is defined as the degree of effectiveness, efficiency and satisfaction when a product is used by certain users to achieve specific goals within a defined context of use.

One of the main reasons why there are so many usability issues in the Semantic Web is because it represents a radical change in the way interaction is sustained. Traditionally, many interactive systems have been based on the Action-Object [3] paradigm: first, the user selects the action that he wants to carry out from pull-down list that organises the available actions in a hierarchical manner. Then, the user selects the object over which the action should be carried on. For instance, the user selects first the "Open" action from a menu and next the document to which this action should be applied.

This is a quite usable interaction model when there is a quite conceptually homogeneous set of objects to which actions are applied. If this is not the case, it is difficult to maintain a clear arrangement of actions because, firstly, it is difficult to organize it hierarchically and, secondly, because it requires the user to deal simultaneously with a great amount of them in order to find the one he is interested in.

The Semantic Web promotes and facilitates the creation of very heterogeneous object sets due to the fact that one of its greatest strengths is the ability to integrate multiple sources of data. Consequently, a Semantic Web application that tries to take advantage of the new possibilities it offers will be usually based on a set of heterogeneous objects.

To follow an Action-Object interaction paradigm in these cases will frequently result in a less usable Semantic Web application. By contrast, the alternative based on an Object-Action paradigm is the natural way of interaction in environments characterized by a high degree of heterogeneity of the objects being manipulated. Consequently, it is the choice in many Web systems [4].

In this case, the interaction begins when the user selects an object or a set of objects he is interested in. Then, the user selects the action that he wants to apply on this object, which is chosen from the set of available actions for it. This paradigm simplifies the interaction and can improve usability in heterogeneous contexts like the ones we can find in many Semantic Web applications. Users find it easier to identify and organise objects than actions. In fact, ontologies are mainly about objects and, in the case of the Semantic Web, web ontologies can be used to attain this.

On the other hand, the group of available actions for an object can be easily determined from the restrictions defined by these ontologies. Consequently, it is possible to exploit the knowledge captured by the ontologies in order to give support to the users while they interact under an Object-Action paradigm, freeing them from this burden so they can concentrate on more productive tasks. This approach is especially appropriate in very heterogeneous domains, for instance resulting from data integration from different sources.

A platform that seeks to put this approach into practice in the context of the Semantic Web is described in Section 2. Then, a sample scenario where this platform has been applied is introduced in Section 3. Finally, the future plans and the conclusions are presented in Section 4.

2. The Rhizomer Platform

In order to explore the possibilities of the object-action interaction paradigm in the context of the Semantic Web, the Rhizomer platform is being developed[1] [5]. The objective is a generic web portal, not constrained to a particular application domain or data schema, inspired by Web 2.0 concepts but based on a Semantic Web data model.

In order to obtain a browser based solution, while maintaining a great range of interaction possibilities, AJAX is the client side choice. The server counterpart looks for simplicity and provides a set of really simple services on top of a RDF metadata store for query, insertion, update and deletion operations implemented through REST [6] commands.

Queries, based on a SPARQL endpoint, are sent to the server using a GET command. However, in order to add support for the other operations, a new range of functionalities have been added to a common SPARQL endpoint: the HTTP PUT and POST commands are used for insertions and updates; the DEL command is used for deletions.

The whole user experience is built on top of these operations. In order to increase usability, RDF is completely hidden. End-users are used to interact through their browsers with HTML web pages. Consequently, Rhizomer incorporates a generic transformation from RDF to HTML, which is not tied to any particular ontology or scheme as most template-based approaches. This HTML rendering is used to build a transparent browsing experience on top of the SPARQL endpoint.

The browsing steps are based on a fragmentation of the underlying RDF graph, which is detailed in Section 2.1. The same fragments are used in order to constraint the range of the update and deletion operations, as it is detailed in Section 2.2. Updates, and new metadata generation, are carried out through semantics-enabled HTML forms that also hide the burdens of RDF metadata from users.

The previous metadata management operations and HTML rendering facilities provide a very generic way to deal with the object part of the Object-Action Interaction Paradigm. RDF metadata is the way to describe objects and this metadata is structured using ontologies. Moreover, none of these operations or rendering facilities is specialised in a particular kind of metadata, schema or ontology.

All of this constitutes the object part of the paradigm. In order to deal with the action part in a highly dynamic way, Rhizomer incorporates Semantic Web services. Each action corresponds to a Semantic Web service that incorporates in its description the constraints an object must satisfy in order to be a valid input for the service. Consequently, the semantic description of the objects, the RDF metadata describing them, is considered in order to determine which actions can be applied to them.

For instance, we consider a scenario where the platform is used in order to retrieve and browse a set of objects that are described as objects of type event. These descriptions include date and time information and, in some cases, the geographical localization of these events. At first, these descriptions are visualised as generic HTML pages based on the RDF to HTML rendering. They allow the user to visualise the descriptions of the corresponding objects, the metadata, and to browse it interactively by browsing the underlying graph.

[1] http://rhizomik.net/rhizomer

This is the generic approach that can be applied to any kind of object. However, in this scenario, it would surely be more appropriate to have more specific views and more appropriate ways to interact with events. Calendars or timelines are good choices for time stamped resources, while maps should be helpful for geographically located ones.

There are some tools that already provide these specialised views on different kinds of semantically described resources, such as Tabulator [7] or Exhibit[2]. However, the range of alternative views is fixed a priori and new views are incorporated in an ad-hoc way to the underlying RDF metadata browsing facilities.

The objective of the Rhizomer platform, and the reason why Semantic Web services have been chosen as the way to implement actions, is to build a generic and dynamic system, which can directly deal with RDF metadata describing different kinds of objects while being easily extensible in order to incorporate specialised ways to view and interact with particular kinds of them.

The underlying interaction paradigm that guides the whole process is the Object-Action one. Consequently, the user gets an object and then, the system offers the available actions that can be performed on it. This set of actions is not fixed a priori but it is dynamically determined based on the semantic descriptions of the objects, RDF resources, and the actions, Web services. Therefore, deploying a new action only needs to load its description into the platform.

The description specifies the restrictions that determine the types of objects to which the service is applicable. Restrictions are combined with the ones defined by the ontologies that structure the metadata describing the objects. Therefore, it is possible to build a very flexible objects-to-actions matching mechanism based on Semantic Web reasoning tools.

From the end-user point of view, the platform is in charge of determining, once the particular object the user is interested in has been selected, what are the available actions. The users are freed from this task, which can become very complex when the domain is very heterogeneous and the range of actions broadens. The users do not need to memorize these associations because they are captured by the underlying ontologies. More details about the implementation of actions as Semantic Web services are available from Section 2.3.

2.1. Metadata Browsing

Browsing is the basic interaction paradigm in the Web. It is based on the successive visualisation of Web pages following the links that connect them. Pages and links are the main building blocks upon which the interaction is built. Web pages are intended for human users' consumption and well established methodologies to make them usable and accessible exist.

However, both the browsing paradigm and the principles to make the whole thing usable and accessible cannot be directly applied to the Semantic Web. That is so because it is based on a model not built upon pages and links but on triples (subject-predicate-object), which makes the browsing approach quite different from the Web.

[2] http://simile.mit.edu/exhibit

The combination of many triples builds up a graph and, though the resulting model is easier to process by computers, the consumers of Semantic Web metadata are, at the end, human users so usable and accessible interaction mechanisms are also required.

First of all, the basic browsing paradigm should change because the Semantic Web makes it very difficult to base the browsing steps on documents. In other words, it does not seem appropriate, for each step, to show all the triples in the corresponding document to the user as it is done in the Web. The amount of information in a single document can be too large, more than thousands of triplets. Moreover, the frontiers among documents are very fuzzy in the Semantic Web: usually, many documents are combined in order to get a coherent graph.

Semantic Web browsers such as Tabulator [7] follow this approach and show all the triples from a Semantic Web document as an unfoldable tree. As preliminary user tests show, this approach causes many usability problems because, as the tree grows, it rapidly becomes difficult to manage. As it has been said, documents contain many triples and, additionally, each navigation step adds more triples from the new document to the current set.

Another approach is faceted browsing, as in /facet [8]. However, our objective is a simpler and more polyvalent browsing mechanism that, though it might lack the guidance provided by facets, it can deal better with heterogeneous information spaces. Moreover, it is not clear how systems such as /facet can deal with metadata structures that feature many anonymous resources, as it is the case for the semantic metadata managed in the S5T project, which is described in Section 3.

Thus, the problem is where to put the limits of each browsing step when presenting semantic metadata. In other words, how each browsing piece is built and how new pieces are created and presented following user needs in order to compose a browsing experience through the whole graph.

In order to facilitate browsing, the proposed approach is based on the construction of graph fragments. Following this approach, it is possible to construct fragments for any graph starting from any non-anonymous node. For instance, for the metadata that describes a piece of content, the starting point is the node that represents it and that is the subject for all the triples that describe it. This node has an ID and consequently is not anonymous.

All the triples that start from this node are part of the fragment. Next, all the triples that describe objects that are anonymous are also added to this set. This happens for all nodes that are only identifiable in the context of the starting node. For instance, Fig. 1 shows how an example graph would be fragmented following this approach. As it can be seen, there are two fragments, each one corresponding to one identified resource that is described by at least one triple, for which it is the subject. The first fragment describes http://rhizomik.net/~rosa and includes an anonymous resource for the address. The second one, for http://www.udl.cat, can be reached from the first one through a browsing step. On the contrary to the address, it is shown independently because it is not anonymous.

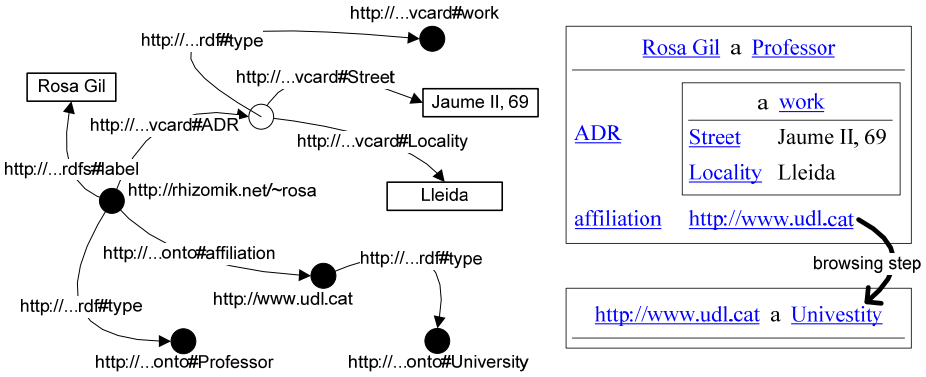

Fig. 1. Fragmentation of an example RDF graph

The proposed approach makes constructing fragments that become usable browsing steps possible. Their size tends to be user-friendly, around 8 triples for MusicBrainz[3] or 14 for the CIA World Factbook[4]. From each fragment, it is possible to follow the next browsing step starting from the triple objects that are not anonymous, if the user wants more information about that resource. The process continues iteratively and interactively, the metadata describing the selected identified node is retrieved and the fragment is built.

The resulting fragments are similar to the ones obtained by the Minimum Self Contained Graph (MSG) approach [9]. The main difference is that, in order to make the results more usable, when they are rendered to the user, all the URIs are replaced by labels if they are available. Consequently, the fragments are augmented with all the available labels and titles for all the affected URIs, even object ones.

In order to show fragments to users, they are rendered using HTML that can be viewed using a web browser, a tool users feel comfortable with. In order to generate HTML from RDF, fragments are serialised as RDF/XML that is transformed using an XSL. The XSL transformation, which is part of the Rhizomer platform, guarantees consistent results whenever the input RDF/XML has been generated from fragments based on the Rhizomer approach.

The fragmentation makes it possible that the resulting RDF/XML maintains all related triples together, even those for the anonymous resources included in each fragment. Consequently, it is possible to show them like a series of HTML tables, one for each fragment corresponding to the description of an identified resource, that contain nested tables for the descriptions for the anonymous resources contained in the fragment.

Table 1 shows example metadata from the S5T project describing a piece of audio content and one of its audio segments. The segment does not have an identifier and thus its description will be included in the fragment describing the content item. If this fragment is rendered, the user will see the HTML shown in Fig. 2. There is a table for the audio content that shows its identifier, its types and all its properties and the

[3] http://musicbrainz.org, U2 discography
[4] Factbook for Spain, http://www.daml.org/2003/09/factbook/sp

corresponding values. Nested tables are used for the anonymous resources described in the fragment.

Table 1. Metadata fragment describing an audio content item and a segment

```
<?xml version="1.0"?>
<rdf:RDF xmlns:mpeg7=
   "http://rhizomik.net/ontologies/2006/03/Mpeg7-2001.owl#"...>
   <mpeg7:AudioType rdf:about=
      "http://www.segre.com/audio/20070113">
      <dc:title>Butlletí Migdia</dc:title>
      <dc:date>2007-03-23</dc:date>
      <tva:Genre rdf:resource="&srs;11000000"/>
      <mpeg7:Audio>
         <mpeg7:AudioSegmentType>
            <mpeg7:MediaTime>
               <mpeg7:MediaTimePoint>01:27.0
               </mpeg7:MediaTimePoint>
               <mpeg7:MediaDuration>P5S
               </mpeg7:MediaDuration>
            </mpeg7:MediaTime>
         </mpeg7:AudioSegmentType>
      </mpeg7:Audio>
   </mpeg7:AudioType>
</rdf:RDF>
```

Moreover, it can be observer that all URIs have been replaced with labels or the fragment part of the URI if no label is available. This makes the resulting HTML easier to render and more usable. For instance, the URI for the genre value has been replaced with the corresponding label. The RDF to HTML transformation can be tested at the ReDeFer[5] project web site.

edit – new - del

http://www.segre.com/audio/20070323 a AudioType	
title	Butlletí Migdia
date	2007-03-23
genre	politics
audio	a AudioSegmentType
MediaTime	a MediaTimeType
MediaDuration	P5S
MediaTimePoint	01:27.0

Referrers

Fig. 2. HTML rendering for the metadata fragment in Table 1

Finally, the identified resources and properties, for which just the available labels have been included in the fragment, are shown as HTML links that allow continuing

[5] http://rhizomik.net/redefer

the browsing experience. If the user is interested in any of them, by clicking on them the fragment for the corresponding identified resource is retrieved and rendered as HTML.

This mechanism has been implemented as successive calls to the SPARQL endpoint based on a DESCRIBE query for the identified resource URI. The DESCRIBE operation of the SPARQL endpoint has been reimplemented in order to build the proposed fragments, which also include all the available labels. Then, the XSL transformation from RDF/XML to HTML is invoked at the client using AJAX, which is also responsible for sending the SPARQL queries and makes the whole process go smoothly begin the scene in order to make the user experience even more comfortable.

2.2. Editing Metadata

The previous fragment-based approach, besides being the foundation for browsing, allows constraining, to a limited set of triples, the metadata editing and deletion actions that are also available from the Rhizomer interface and shown as links at the top of Fig. 2.

This way, it is possible to implement editing actions as the replacement of a given fragment, the one being browsed when the user clicks the *edit* link, with the one resulting from the editing process. The same applies for the deletion action. In this case, all the triples for the fragment being browsed are removed from the metadata store.

On the other hand, there is also a *new* link that facilitates metadata creation based on a “create from example” approach. It makes possible to create a new description based on the one being browsed. The user should provide a new URI for the resource being described and edit the values generated automatically from the example in order to adjust them to the resource being described.

All these operations (editing, deletion and creation) are also carried out through an HTML interface. In addition to the RDF to HTML transformation, the Rhizomer platform also includes an XSL transformation from RDF to HTML forms. These forms are generated automatically from the RDF/XML corresponding to a fragment. The same approach as in the RDF to HTML transformation is followed but, instead of generating text values and links for literals and resource, this transformation generates input fields for each triple. The field is named using the corresponding property URI its value corresponds to the triple value. The fields can be used in order to edit the property value, either a resource URIs or a literal.

Moreover, properties and values can be removed or added. Currently, the user enjoys little assistance during the editing process. Basically, when the user chooses to add a new property, a SPARQL query is used in order to retrieve all the available properties for the resource being edited. These are the properties that are not constrained to a particular resource type plus all the properties constrained to the types of the resource being edited. The future plan is to improve this support in order to assist users during the whole editing process, as it is detailed in future work presented in Section 4.

Finally, an algorithm has been developed in order to reverse the mapping from RDF to HTML forms. In other words, this algorithm is responsible for generating the RDF that results from the editing process by mapping the form input fields to the corresponding triples. This completes the roundtrip for RDF metadata editing from RDF to HTML forms and back to RDF.

2.3. Actions as Semantic Web Services

The metadata browsing and editing components presented in the previous sections give users access to resources and their descriptions: the static object part of the Object-Action paradigm. The user can pose queries to access the descriptions of the resources managed by the system and browse through the semantic metadata that describe them.

Once the object (or objects) of interest is located, the actions that the user can do upon it are shown to the user following the Object-Action paradigm. In the Rhizomer platform, this part is implemented by means of semantic web services. This allows a completely dynamic integration of the actions because they are not predefined for the different types of objects, i.e. they can be seen as independent entities.

Different semantic web services platforms have been evaluated, mainly OWL-S [10], WSMO [11] and SAWSDL [12]. All of them are too complex for the simple requirements of the platform. The complexity does not lie in the semantic model that these platforms provide, but in the fact that all of them are based on web services standards such as WSDL/SOAP [13]. This kind of semantic web services is more appropriate in business environments but it is over-complex for the Rhizomer platform in which the actions will mainly be used to implement data visualization services.

Besides, many of the publicly available web services, e.g. Google Maps, are not available as WSDL/SOAP. In fact, it seems that services based on WSDL/SOAP are being displaced by REST ones [14]. For instance, the big providers of web services (Google, eBay, Yahoo!, among others) are basing their services on REST and export them by APIs in JavaScript or other languages. This approach is appropriate when strong requirements of security do not exist and a simple development model is an objective. In any case, it is also possible to implement security mechanisms over REST [6].

Therefore, actions in Rhizomer are implemented as web services based on REST. That is to say: simple HTTP requests to the services that get HTTP responses with the result. For instance, Yahoo! Maps provides a REST interface to a service that, given the geographical coordinates to show, returns its location in a map.

REST simplifies the invocation of web services and it is only concerned with this aspect. Therefore, for the localization and automatic invocation of REST-based web services, formal descriptions of these services are needed. We believe that the initiatives of semantic web services are the answer to this problems and that is why we have considered the modelling mechanisms they provide.

It has been considered that the ontologies provided by OWL-S 1.1 are the most appropriate for describing our web services due to their modularity. It has been easier to detect the classes and properties more appropriate to the kind of descriptions we

require and use them in isolation without any concern about the rest of the framework. Only the *Service Profile* provided by OWL-S is used for a high-level description of the service. Neither *Service Grounding* nor *Service Model* are considered because the simplicity of the REST services considered do not make them necessary.

In fact, in the current state of the system, only the class *Process* and the properties *hasInput* and *hasOutput* (defined in OWL-S) are used. *Process* allows identifying the resources that correspond to web services that can be invoked from Rhizomer. Their URI corresponds to the service's access point, so it must be an URL. Input parameters for the service are not used but data is sent in the body of a POST message and corresponds to the RDF/XML serialisation of the description of the resource (or resources) that the service accepts as input.

The *hasInput* property is associated to *Process* resources and identifies the class of things that serves as input for the service. Consequently, for a service to appear as available when a concrete resource is shown, this resource must belong to the class defined as the input of the service. It is not necessary to make an *a priori* classification of the resource. To get the desired dynamism, classes in OWL can be specified to be used in *hasInput* that represent the necessary and sufficient conditions to classify resources automatically. This is possible with a Description Logic (DL) reasoner.

Table 2. Description of a geographical information visualization service

```
<rdf:RDF ...
  xmlns:process="…/services/owl-s/1.1/Process.owl#"
  xmlns:pos="…w3.org/2003/01/geo/wgs84_pos#">
<process:Process
    rdf:about="http://rhizomik.net/rhizomer/services/map">
    <rdfs:label>map</rdfs:label>
    <process:hasInput>
        <owl:Class rdf:ID="GeolocatedEntity">
            <owl:intersectionOf rdf:parseType="Collection">
            <owl:Restriction>
                <owl:onProperty rdf:resource="&pos;lat"/>
                <owl:minCardinality>1</owl:minCardinality>
            </owl:Restriction>
            <owl:Restriction>
                <owl:onProperty rdf:resource="&pos;long"/>
                <owl:minCardinality>1</owl:minCardinality>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </process:hasInput>
    <process:hasOutput>text/html</process:hasOutput>
</process:Process></rdf:RDF>
```

For instance, as it is shown in Table 2, it is possible to define *GeolocatedEntity* as the class of all the resources with properties *lat* and *long* and use it as the *hasInput* class for a service named “map”. There is no need to explicitly classify all the geolocated entities into this class. The reasoner is responsible for classifying into it all the resources that satisfy these restrictions.

Then, when the user is browsing resource descriptions, it is checked whether they correspond with the input class of any of the available services. For instance, when a

resource has both latitude and longitude, the reasoner classifies it as an instance of *GeolocatedEntity*, so it is detected as being accepted by the "map" service. Consequently, this service can be invoked passing a description of the corresponding resource as its input. The user can invoke the service using a link, automatically associated to the resource using the mechanism described before, and get a visualization of the position of the resource in a map.

Direct invocation of web services passing them the RDF metadata of the resource that must be used as input is not usually allowed. Therefore, in many cases, the URL associated with a service is actually pointing to a wrapper that receives the RDF, extracts the data needed by the service, and makes the "real" invocation of the service. This additional layer between Rhizomer and the services, though it complicates the implementation, allows using visualisation services such as GoogleMaps or SIMILE Timeline[6] that are only available as JavaScript libraries. In this case the wrapper is implemented as a servlet that generates the web page that uses the JavaScript library and provides the final result.

Finally, the *hasOutput* property specifies the output type of the service. For visualization services a literal representing the MIME type of the output is used. The output is shown in a new HTML layer within the Rhizomer interface and the MIME type is used to correctly interpreting the result. In the next section, a specialised web service for the visualisation of multimedia resources is shown in the context of a business application using Rhizomer.

3. Applying Rhizomer at the Segre Media Group

The Rhizomer platform has been put into practice in the Segre[7] media group in the context of the S5T[8] research project. This project builds on top of the experiences gained during the NEPTUNO[9] project, which developed a set of ontologies and semantic annotation mechanism for digital news [15]. S5T extends semantic annotation to the audio part of audiovisual contents. In order to do that, a transcript of the audio voices is automatically generated and processed in order to detect key terms and produce semantic annotations based on this terms. More details about this process are available in [16], while this paper focuses on the interface that allows users to exploit the resulting annotations.

The interface is based on the Rhizomer platform and allows browsing the audiovisual contents through their transcripts or, in a complementary way, together with the ontologies and metadata in their semantic annotations.

A typical interaction can be started either by building a query to retrieve the pieces of content the user is interested in or, alternatively, by browsing from the main page menus. All the metadata and the ontologies are based on RDF, so they can be browsed in a generic way by means of the RDF to HTML transformation that the platform

[6] http://simile.mit.edu/timeline
[7] http://www.diarisegre.com
[8] http://nets.ii.uam.es/~s5t
[9] http://nets.ii.uam.es/neptuno

provides. It is also possible to perform queries through a SPARQL endpoint and then browse in the same way the results, as detailed in Section 3.1.

On top of this base for user-object interaction, some services have been added in order to carry out actions that are specific to the Segre scenario and which depend on the type of object that is being manipulated. For audiovisual contents with an annotated transcript, a semantic web service has been added that implements one of these specific actions. The rest of the services and features, e.g. metadata rendering and browsing, are directly reused from the Rhizomer platform.

This service provides a specialised interface, detailed in Section 3.2, that allows users to play the content and that shows the corresponding transcript, which is enriched with links to the concepts used for its annotation. These links can be followed in order to retrieve the descriptions for the corresponding concepts, which are also browsed through the RDF to HTML module, as it is detailed in Section 3.3.

The objective of the resulting interface is to exploit, in an integrated and more efficient and effective way, the different types of audiovisual and text contents managed in the Segre media house.

3.1. Content Metadata Browsing

Once a query is executed, metadata associated with the selected resources is shown by means of the HTML interface for metadata browsing, as it is shown in the left part of Fig. 3. In the case of the S5T research project, multimedia metadata is based on the Dublin Core[10] for editorial metadata, i.e. title, date, author, etc. and on an ontology for the standard IPTC News Subjects[11] for genres. For content-based metadata, particularly content decomposition based on audio transcripts, a MPEG-7 Ontology is used [17].

All the resources and properties that appear in the metadata HTML view are links that allow the user to retrieve additional metadata about the clicked resource. For instance, the news items described in Fig. 3 refer to the "politics" genre. If the corresponding link is followed, the metadata for the corresponding resource is retrieved from the IPTC news topics ontology and shown. Consequently, it is possible to browse the descriptions for the news items managed in the S5T project and the descriptions for the terms used in these descriptions.

3.2. Transcript-based Interaction Service

In addition to the metadata browsing facility, which provides a way to interact with the objects by means of the object-action paradigm, there are some web services, such as the ones described in Section 2.3, that provide some customized actions. Additionally, there is a specific action for the Segre scenario that is enabled for audiovisual resources, i.e. resource of type *mpeg7:AudioType*, with an associated *transcript* property. The corresponding web service provides a view, which is shown

[10] http://dublincore.org
[11] http://rhizomik.net/semanticnewspaper

in the right part of Fig. 3, which allows additional interaction possibilities through the transcript semantic annotations automatically generated [16].

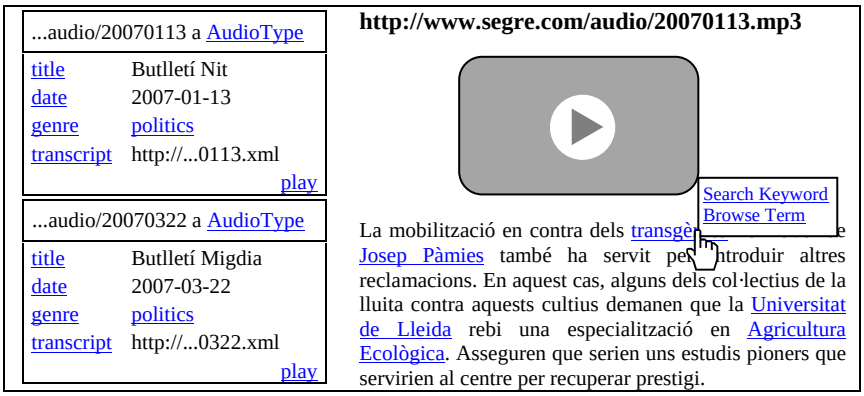

Fig. 3. Metadata view (left) and transcript view (right) available through the "play" service

This view allows rendering audio and video content and interacting with it through a clickable version of the audio transcription. Two kinds of interactions are possible from the transcription. First, it is possible to click on any word in the transcription that has been indexed in order to perform a keyword-based query for all the pieces of content whose transcription contains that keyword.

Second, the transcription is enriched with links to the ontology used for semantic annotation. Each word in the transcription whose meaning is represented by an ontology concept is linked to a description of that concept. Then, that description is presented as it is detailed in the next subsection.

For instance, the transcript includes the name of a politician that has been indexed and modeled in the ontology. Consequently, it can be clicked in order to get all the audiovisual items where his name appears or, alternatively, to browse all the knowledge about that politician encoded in the corresponding domain ontology.

3.3. Domain Knowledge Browsing

When the user chooses to browse concepts in the annotated transcript, the interaction gets back to the generic metadata browsing view. Then, the user can browse the ontologies used to annotate the transcripts. Each browsing step gets the user through these ontologies.

Consequently, continuing with the politician example in the previous subsection, when the user looks for the available knowledge about that person, an interactive view of the RDF data about him is shown. This way, the user can benefit from the modelling effort and, for instance, be aware of the politician party, if he is a member of the parliament, etc. The subsequent browsing steps, e.g. following the links to the politician party or the parliament, will show additional domain knowledge from the annotation ontologies, for instance a list of all the members of the parliament.

In addition to this interactive navigation of all the domain knowledge, at any browsing step, it is also possible to get all the content annotated using the concept currently being browsed. This action might bring the user back to the transcript-based view. Thanks to this dual browsing experience, the user can navigate through audiovisual content and the underlying domain knowledge in a complementary an interwoven way.

4. Conclusions and Future Work

The Rhizomer platform provides an interaction environment based on the object-action paradigm that is better suited for heterogeneous information spaces than the traditional action-object paradigm. The platform is based on Web 2.0 technologies on top of Semantic Web metadata and ontologies.

The platform offers a generic RDF to HTML transformation that makes it possible to navigate through semantic metadata and the associate ontologies. Resources and their descriptions constitute the object part of the paradigm while the actions that the users can carry out on these resources are implemented by means of a Semantic Web services based on a REST approach. Actions are associated to objects in a completely dynamic way computed by a Semantic Web reasoner on the basis of the semantic descriptions of resources and services.

This platform has been applied in the context of the Segre media group in order to develop a user interface. This interface is intended for news items management in the media house and takes profit from the semantic annotations of the audio voice transcripts. The objective of this tool is to facilitate news items management and the production of new content.

In this scenario, besides semantic search, metadata browsing and some generic actions such as showing geolocated entities in a map, there is a specialised action for audiovisual content with a transcript for the audio voice. This service allows to reproduce the content and to see the transcript enriched with semantic annotations for keywords. The annotations can be used in order to retrieve other pieces of content featuring the same keyword or in order to browse the semantic annotations metadata.

To conclude, the future work focuses on metadata edition features and user testing. In addition to the current assistance when a user tries to add a new property to the current description, the idea is also to assist users when they add property values. Properties ranges and restrictions on them that apply to the kind of resource being edited will be considered in order to propose resources that constitute a proper value for the property.

Moreover, our objective is to complement the preliminary user tests we have been carrying out in order to quantify the usability improvements that this approach can produce. Currently, we have preliminary qualitative usability results, like the average size in number of triples of the browsing steps or the average action-object versus object-action ratios for many applications scenarios, which are significantly lower for the later thus making it easier to organise and browse.

Acknowledgements

The work described in this paper has been partially supported by Spanish Ministry of Science and Education through the Scalable Semantic personalised Search of Spoken and written contents on the Semantic Web (S5T) research project (TIN2005-06885).

References

1. Shadbolt, N., Hall, W. and Berners-Lee, T.: "The Semantic Web revisited". Intelligent Systems, Vol. 21, No. 3, pp. 96-101, 2006
2. Heath, T., Domingue J. and Shabajee P.: "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. Bruner, J.: "Action, Thought and Language". Alliance Psychology, 1989
4. Raskin, J. : "The Human Interface". Addison Wesley, 2000
5. García, R. and Gil, R.: "Improving Human-Semantic Web Interaction: The Rhizomer Experience". CEUR Workshop Proceedings, Vol. 201, pp. 57-64, 2006
6. Richardson, L. and Ruby, S.: "Restful Web Services". O'Reilly, 2007
7. Berners-Lee et. al: "Exploring and Analyzing linked dates on the Semantic Web". Proc. of the 3rd International Semantic Web User Interaction Workshop, 2006
8. Hildebrand, M., Ossenbruggen, J., Hardman, L.: "/facet: A Browser for Heterogeneous Semantic Web Repositories". In Proc. of the International Semantic Web Conference 2006. Lecture Notices in Computer Science, Vol. 4273, pp. 272-285, 2006
9. Tummarello, G., Morbidoni, C., Puliti, P., Piazza, F. : "Signing individual fragments of an RDF graph". Proceedings of the WWW 2005 Conference, pp. 1020 - 1021, 2005
10. Martin, D. (ed.) OWL-S: "Semantic Markup for Web Services". W3C Member Submission, 2004. http://www.w3.org/Submission/OWL-S
11. Roman, D., Keller, or., Lausen, H., of Bruijn, J., Lara, R., Stollberg, M., et al.: "Web Service Modeling Ontology". Applied Ontology, Vol. 1, No. 1, pp. 77-106, 2005
12. Farrell, J. and Lausen, H. (eds.): "Semantic Annotations for WSDL and XML Schema". W3C Working Draft, 2007. http://www.w3.org/TR/sawsdl
13. Weerawarana, S., Curbera, F., Leymann, F., Storey, T. and Ferguson, D.F.: "Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More". Prentice Hall, 2005
14. Forrest, B.: "Google Deprecates Their SOAP Search API". O'Reilly Radar, December 18, 2006. http: //radar.oreilly.com/archives/2006/12/google_depreciates_SOAP_API.html
15. Castells, P., Perdrix, F., Pulido, E., Rico, M., Benjamins, R., Contreras, J., et al.: "Neptuno: Semantic Web Technologies for a Digital Newspaper Archive". In C. Bussler, J. Davies, D. Fensel and R. Studer (Eds.): "The Semantic Web: Research and Applications: First European Semantic Web Symposium, ESWS 2004". Lecture Notes in Computer Science, Vol. 3053, pp. 445-458, 2004
16. Tejedor, J., García, R., Fernández, M., López, F., Perdrix, F., Macías, J.A., Gil, R., Oliva, M., Moya, D., Colás, J., Castells, P.: "Ontology-Based Retrieval of Human Speech". In Proc. of the 6th International Workshop on Web Semantics, WebS'07. IEEE Computer Society Press, 2007
17. García, R., Tsinaraki, C., Celma, O., and Christodoulakis, S.: "Multimedia Content Description using Semantic Web Languages". In Y. Kompatsiaris & P. Hobson (Eds.), "Semantic Multimedia and Ontologies: Theory and Applications". Springer, in press, 2008