SWAD-Europe Deliverable 9.3: Semantic Web tools to help authoring: A Semantic Web image annotation tool

Project name:
Semantic Web Advanced Development for Europe (SWAD-Europe)
Project Number:
IST-2001-34732
Workpackage name:
9. Visualisation and Accessibility
Workpackage description:
┤ http://www.w3.org/2001/sw/Europe/plan/workpackages/live/esw-wp-9.html
Deliverable title:
9.3: Semantic Web tools to help authoring: A Semantic Web image annotation tool
URI:
┤ http://www.w3.org/2001/sw/Europe/reports/report_semweb_access_tools/
Author:
Libby Miller, ILRT
Charles McCathieNevile, W3C
Abstract:
This document describes the background, usecases, requirements, implementation and use of a Semantic image annotation tool. The tool is written in Javascript and uses RESTful web services to access remote information. It is designed to be a quick and easy means of creating structured information about images, including who or what is depicted in the image; where and when it was created; creator and licensing information. The aim is to create and enable the reuse of alternative formats for both text and images for use in an accessibility context, although the potential application is much wider.
Status:
This is a completed report, last modified 2004-06-17.
This document may be updated during the life of the SWAD-Europe project to reflect or link to further developments in this area.
Comments on this document are welcome and should be sent to the ┤ public-esw@w3.org list, archived at ┤ http://lists.w3.org/Archives/Public/public-esw/. General discussion of semantic web tools and technologies should be sent to www-rdf-interest@w3.org which is archived at ┤ http://www.w3.org/Archives/Public/www-rdf-interest.

Contents

Summary

This report describes a tool created to generate machine-readable descriptions of several aspects of images, using client-side javascript to consult multiple remote database-driven services to offer the user choices of objects to include in the file. The data created with this tool can be aggregated and searched in multiple dimensions - by person, date, event, type of thing, author, license; an example of such an aggregator is the Codepiction search tool ⊣ [CodepictionSearch], developed as part of the FOAF project ⊣ [FOAFProject].

Introduction

The aim of this workpackage (WP9) ⊣ [WP9] was to describe an implementation of a tool to assist in authoring (from WP9's description):

"*Authoring content that is accessible requires the author to provide some information in multiple formats. For example it is important to illustrate content with appropriate multimedia for people with disabilities that affect their ability to read text. It is also important that the content of this multimedia is accessible to people who cannot see it, or cannot hear it, or perhaps both. Rather than creating all information twice, in many cases it is possible to use existing material, and to find or provide a version in a particular medium. For an author who herself has a disability, it is of critical importance to find relevant materials without having to develop them in all formats.*"

The implementation described in this report takes the particular case of finding images to illustrate a particular topic and can be used for the quick and simple creation of machine-processable information about the image. This information can then be aggregated and searched in various ways making it much easier to find appropriate images to illustrate textual content, and conversely find textual descriptions of image content. This tool separates out the textual descriptions of the content of a photo (e.g. "looks like Bob and Alice are having a great time at the seaside") from much more structured information about which Bob and Alice you mean specifically; where the seaside was they were visiting; whether the photo contains a bucket and spade or a glass of sangria. Making it simple for these types of distinctions to be made enables much more complex and meaningful descriptions

of images; and thereby better searching of images.

Related work in SWAD-Europe -

Image-description-related work as part of WP3 involved liaising with various commercial, academic and independent individuals and organisations to develop a common vocabulary for parts of images, and some guidelines for combining vocabularies for a particular application [W3PhotoVocabs] - the W3photo project [W3Photo], a project to annotate photos from the WWW series of conferences. The vocabulary is available [ImageParts]. A description of the liaison activity is on the ESW weblog [W3PhotoWeblog] and in the Dissemination and Use Plan [DUP]. SWAD-Europe has also held two workshops on image annotation [SWADE1], [SWADE2].

Background to the tool

Many tools are available for Semantic Image annotation. This section describes the objectives and requirements of this tool, usecases underpinning the requirements, the functionality of the tool, and the vocabularies used.

Requirements and objectives -

The main objective was a tool that made the creation of RDF data about image simple and fast for all ranges of users, including those with high and low technical skills. To expand on this we created a number of brief, high-level usecases:

- **Multi-platform, no-download installation**. A user can use the tool straight away, without having to ask an administrator for help in installing the program or related libraries.
- **Auto-upload of RDF documents**. A user can use the tool without owning their own web space.
- **'Importing' already-identified objects from remote sources.** To avoid duplicating work, the ability to access objects that have already been annotated and to re-use that information within the UI, **without** running one's own RDF harvester and database locally.
- **Ability to view the image being described.** This needs to be at a sufficient size for the user to be able to identify the aspects of the image that she wants to annotate.
- **Ability to annotate several images at once.** For annotation speed, user defaults should be stored for that repetitive duplicate entry is not required.
- **Ability to annotate any image on the web.** When the user finds a suitable image, she should be able to create RDF data about it wherever it is located.
- **Extensibility.** For developers, the tool should be extensible for use with different vocabularies, datasources and uses.

Functionality -

- **Javascript implementation**. Initially the tool was prototyped in Java, but Java requires administrator access for installation on Windows platforms, and is also problematic for Debian users. Javascript was chosen because of the good support available in two major browsers - the multiplatform Mozilla and Internet Explorer (IE) for Windows, and Javascript tools (written by Jim Ley) are available [JSParser]. Javascript is a fully functional programming language running inside browsers. It does not require installation; however each browser runs a different variant of the language, so writing fully interoperable code is difficult. In particular, the capacity to download and parse XML using the browser's native parser is restricted to IE on Windows and Mozilla. This functionality is used by the Javascript RDF parser used to access remote services, and so the application only runs on these browsers.
- **Auto-upload of generated documents.** A test database was made available for people to upload their RDF data directly to the codepiction database from the

application, without access controls as a first pass.

- **Using web services to access remote data sources.** Various remotely-hosted datasources are available describing people, locations, things in RDF. Some of these are harvested data sources (such as the codepiction database); others are services, for example Jim Ley's airports database provides information about airports, their GPS locations and their names; Dan Brickley's Wordnet in RDF datasource provides information about nouns in the Wordnet database. All these datasources have two things in common - they are get-able over http, and they return RDF/XML in response to a query. Because of these features we can use these datasources in a structured way to interface between human-readable names for things and already-used, machine-processable identifiers for them, without the user having to run their own RDF database locally. For example a search for 'Bristol' of the remote airports database returns RDF about Bristol, England and the Tri-City region airport in the US. the user can then choose between these to accurately reflect the annotation they wish to make. More information about this aspect is available in the user section below.
- **Use of iFrames and a proxy to display any image on the web in the context of the application.** The tool uses a proxy to download either a page of links to thumbnails, a page with images in it, or a single image, into an iframe. The images are accessed using the DOM, and displayed in the application. The image can be clicked on to start to create annotations.
- **Open Source download available**. Restrictions on the licenses of component parts mean that the tool is make available under the ⊣ W3C license ⊣ [W3CLicense]. The code may be ⊣ downloaded ⊣ [Download], and there's ⊣ online demo ⊣ [JSPhotoAnno]

User interface considerations -

Applications such as Matt Biddulph's IRC bot-based conversational interface to image annotation ⊣ [ImgBot], and Damian Steer's ⊣ [FOAFFinger] foaffinger rendezvous-based foaf creation tool have successfully used stateful, text-based interfaces for data creation. For a trained cataloguer doing a large batch of annotation work, command-line tools can be faster to use. Hence an early version of the tool used a text-based interface based on JavaScript Shell ⊣ [JSShell].

However, a significant issue for image annotation is that as the user catalogues images they need to be able to see the image. It is also useful to be able to pick from a list of thumbnail images and then annotate several; this limits the usefulness of command line or bot interfaces. In response to user feedback on the first version, a clickable version ⊣ [JSPhotoAnno] was produced. The visual cues this gives makes cataloging images faster, although there are several significant problems with layout of the information.

Vocabularies -

Multiple vocabularies are used in the RDF generated by the tool, and where possible vocabularies were used which were already well-used elsewhere. More vocabularies used for image description are described on the ESW wiki ⊣ [ImageVocabs].

Description of the content of the image

FOAF Vocabulary ⊣ [FOAF] - for describing people.

The FOAF vocabulary is used to describe that a photo depicts a person. Hashed mailbox addresses, homepages or weblogs can be used to identify the person.

```
<foaf:Image rdf:about='http://w3photo.org/photos/www2003/photos/IMG_4781_2.JPG'>
<foaf:depicts>
 <foaf:Person>
   <foaf:name>Libby Miller</foaf:name>
   <foaf:homepage rdf:resource="http://ilrt.org/people/libby"/>
 </foaf:Person>
</foaf:depicts>
</foaf:Image>
```

Wordnet ⊣ [WN]

The very large Wordnet vocabulary is used to describe objects in the image, for example, dog, beer. Wordnet is a huge lexical dictionary with definitions of thousands of words. The tool uses Dan Brickley's version of Wordnet 1.6 which makes all Wordnet's nouns accessable over http, as RDF/XML, one class at a time. Several other versions of Wordnet are available in RDF/XML, and the W3C's Semantic Web Best Practices Working Group has a taskforce currently investigating how best to make it available as RDF/XML.

```
<foaf:Image rdf:about='http://w3photo.org/photos/www2003/photos/IMG_4781_2.JPG'>
<foaf:depicts>
   <wn:Dog />
</foaf:depicts>
</foaf:Image>
```

RDF iCalendar ⊣ [RDFIcal]

iCalendar is a file format commonly used in Personal Information Management tools (PIMs) and calendaring and scheduling applications. RDF iCalendar is an RDF transliteration of this vocabulary.

```
<foaf:Image rdf:about='http://w3photo.org/photos/www2003/photos/IMG_4781_2.JPG'>
<foaf:depicts>
 <ical:Vevent>
   <ical:url rdf:resource='http://www.www2003.org'/>
 </ical:Vevent>
 </foaf:depicts>
</foaf:Image>
```

Dublin Core - textual description of the image ⊣ [DC]

Dublin Core is a very well-known vocabulary for metadata about documents, such as a the description of an image, and the creator of an image.

```
<foaf:Image rdf:about='http://w3photo.org/photos/www2003/photos/IMG_4781_2.JPG'>
 <dc:description>description of image here</dc:description>

 <dc:creator>
  <foaf:Agent>
    <foaf:mbox_sha1sum>289d4d44325d0b0218edc856c8c3904fa3fd2875</foaf:mbox_sha1sum>
    <foaf:name>Libby Miller</foaf:name>
  </foaf:Agent>
 </dc:creator>
</foaf:Image>
```

Information about the image itself

Creative Commons - licensing the metadata and the image

Creative Commons ⊣ [CC] is a movement "to build a layer of reasonable, flexible copyright in the face of increasingly restrictive default rules". The idea of using it for images is to make sure that the rights holders are clearly defined and identified, and the license for re-use of images and the RDF data describing them is clear.

```
<foaf:Image rdf:about='http://w3photo.org/photos/www2003/photos/IMG_4781_2.JPG'>
 <cc:license rdf:resource="http://creativecommons.org/licenses/by-nc-sa/1.0/" />
  <dc:rights>
   <cc:Agent>
    <foaf:name>Ben Shneiderman</foaf:name>
    <foaf:mbox_sha1sum>49f9ee9630fa1b694161112d29d8e15c30344cc0</foaf:mbox_sha1sum>
   </cc:Agent>
  </dc:rights>
</foaf:Image>
```

Annotea annotations

Annotea is a vocabulary and protocol for describing annotations. Here we use the vocabulary to describe the author of the annotation and the relationship between the

document containing the annotation (the RDF document), and the image.

```
<rdf:Description rdf:about=''>
 <an:annotates rdf:resource='http://w3photo.org/photos/www2003/photos/IMG_4781_2.JPG'/>
 <an:author>
 <foaf:Person>
 <foaf:mbox_sha1sum>49f9ee9630fa1b694161112d29d8e15c30344cc0</foaf:mbox_sha1sum>
<foaf:name>Ben Shneiderman</foaf:name>
 </foaf:Person>
 </an:author>
 <an:created>2004-05-12T16:14:49Z</an:created>
</rdf:Description>
```

Related tools

Related tools are detailed on the ESW weblog ⊣ [RelatedTools] and in the writeup of the two SWAD-Europe workshops ⊣ [SWADE1], ⊣ [SWADE2].

Using the tool - end users

The tool is designed to annotate images that are already on the web somewhere. It allows the creation of annotations which describe the content of the image:

- This image depicts a person with a particular name
- This image depicts a thing (a dog, a beer)
- This image depicts an event (a workshop, a meeting, a conference)

and also metadata about the image:

- This image was created near somewhere
- This image has a textual description and a date
- This image was created by someone under a particular license

The tool builds an RDF file describing these characteristics of the image on the fly, using namespaces that are already well-used, where possible. The RDF file can then be uploaded to the web and harvested by aggregators. Here is an example of such a file, with tags from different namespaces in different colours:

```
<rdf:RDF
xmlns='http://xmlns.com/foaf/0.1/'
xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
xmlns:rdfs='http://www.w3.org/2000/01/rdf-schema#'
xmlns:dc='http://purl.org/dc/elements/1.1/'
xmlns:rss='http://purl.org/rss/1.0/'
xmlns:wn='http://xmlns.com/wordnet/1.6/'
xmlns:contact='http://www.w3.org/2000/10/swap/pim/contact#'
xmlns:vcard='http://www.w3.org/vcard-rdf/3.0#'
xmlns:pos='http://www.w3.org/2003/01/geo/wgs84_pos#'
xmlns:doa='http://www.daml.org/2001/10/html/airport-ont#'
xmlns:ical='http://www.w3.org/2002/12/cal/ical#'
>
<rdf:Description rdf:about="">
 <annotates rdf:resource="http://chewbacca.ilrt.bris.ac.uk/~plmlp/paris2003/concorde1.jpg"/>
</rdf:Description>
<Image rdf:about="http://chewbacca.ilrt.bris.ac.uk/~plmlp/paris2003/concorde1.jpg">
 <thumbnail rdf:resource="http://chewbacca.ilrt.bris.ac.uk/~plmlp/paris2003/concorde1.jpg"/>
 <dc:description>Facing north in the Place de la Concorde</dc:description>
 <creationEvent rdf:parseType="Resource">
 <contact:nearestAirport>
 <doa:Airport rdf:about='http://www.daml.org/cgi-bin/airport?CDG'>
 <pos:lat>49.0166666666667</pos:lat>
 <pos:long>2.55</pos:long>
 <doa:location>Paris [Charles de Gaulle], France</doa:location>
 <vcard:country>FR</vcard:country>
 </doa:Airport>
 </contact:nearestAirport>
 <ical:date>2003-09-18</ical:date>
 </creationEvent>
 <depicts>
 <wn:Fountain/>
 </depicts>
</Image>
</rdf:RDF>
```

Choosing a photo -

The tool uses a proxy to download 1) a page of links to thumbnails, 2) a page with images in it, *or* 3) a single image, into an iframe. The images are accessed using the DOM, and displayed. Clicking on an image triggers a download of the image or html page linked to in the initial thumbnails page, and then the tool uses heuristics to determine if the link is to an image or an html page. If the latter, it makes a guess about which is the correct image, and makes that the main item to be catalogued. At this stage the display shows something like this:

image 4 picked

Current photos:

next batch...

and the RDF generated looks like this:

```
<rdf:RDF
xmlns='http://xmlns.com/foaf/0.1/'
xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
xmlns:rdfs='http://www.w3.org/2000/01/rdf-schema#'
>
<rdf:Description rdf:about="">
<annotates rdf:resource="http://swordfish.rdfweb.org/photos/2003/06/12/2003-06-12-Images/4.jpg"/>
</rdf:Description>

<Image rdf:about="http://swordfish.rdfweb.org/photos/2003/06/12/2003-06-12-Images/4.jpg">
<thumbnail rdf:resource="http://swordfish.rdfweb.org/photos/2003/06/12/2003-06-12-Thumbnails/4.jpg"/>
</Image>
</rdf:RDF>
```

Adding information about a person -

For images containing people, it is useful to be able to say that the image depicts a particular, identified person. See the codepiction experiment [CodepictionSearch] for more information about this approach.

One issue is a convenient way of finding people's sha1-encoded email addresses (or their actual email addresses and converting them using a tool). This is where a remote service from a database which already contains this information is useful. This could be, for example, a private address book with a remote interface which produces RDF. In this case, we use an interface to a harvested RDF database.

People:

Aaron Dancygier
Alan Dansey

Brendan Quinn
Dan
Dan Armstrong
Dan Brickley

/>

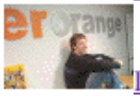
4.jpg

Dan Brickley
Dan Brickley
Dan Brickley

Dan Brickley
Dan Connolly

Dan Connolly

Dan Dickinson



Dan James
Dan James

Sha1-encoded mailboxes and images are shown in response to a query on a substring of a name. Clicking on the image or the name produced adds the person to the RDF. If the person is not in the database, they can be added manually using the forms. At no time is an email address made public.

Adding wordnet keywords -

Dan Brickley has produced a service whereby appending a noun to the namespace http://xmlns.com/wordnet/1.6/ gives you the wordnet hierarchy for that noun, if it exists. The image annotating tool uses this trick, so if you type 'parrot' into the 'keyword' box, the tool uses Jim Ley's RDF parser to fetch the RDF associated with http://xmlns.com/wordnet/1.6/Parrot, and display it in a useful way so that the tool user can check that it displays the term they are interested in, and also see if a sublcass of the main term might be more appropriate.

choose the wordnet class by clicking on it, or see subclasses by clicking on 'more detail'

Wordnet keywords:
* Parrot [1] usually brightly colored zygodactyl tropical birds with short hooked beaks and the ability to mimic sounds
more detail...

The wordnet term is then added to the generated RDF by clicking on it, for example:

Wordnet keywords:

* Parrot [1] usually brightly colored zygodactyl tropical birds with short hooked beaks and the ability to mimic sounds
* Popinjay archaic
* Poll a tame parrot
* Poll_parrot a tame parrot
* African_gray commonly domesticated gray parrot with red-and-black tail and white face; native to equatorial Africa
* Psittacus_erithacus commonly domesticated gray parrot with red-and-black tail and white face; native

The RDF generated looks something like this:

```
<rdf:RDF
xmlns='http://xmlns.com/foaf/0.1/'
xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'
xmlns:rdfs='http://www.w3.org/2000/01/rdf-schema#'
>
<rdf:Description rdf:about="">
<annotates rdf:resource="http://swordfish.rdfweb.org/photos/2003/06/12/2003-06-12-Images/4.jpg"/>
</rdf:Description>
<Image rdf:about="http://swordfish.rdfweb.org/photos/2003/06/12/2003-06-12-Images/4.jpg">
<thumbnail rdf:resource="http://swordfish.rdfweb.org/photos/2003/06/12/2003-06-12-Thumbnails/4.jpg"/>
   <depicts>
      <wn:Parrot/>
   </depicts>
</Image>
</rdf:RDF>
```

Describing the location the image was created -

Ideally the location of a photo would come from the EXIF (EXchangeable Image Format) data the camera produces: the same goes for date. The first of these is not yet available so we have had to come up with something else. The latter is readily available, but we have not come across EXIF parsers in javascript as yet.

We have chosen to use the nearestAirport property to associate an image with location data at this time. This is because information linking airports with latitude and longitude is freely available. As an added bonus, this method preserves privacy.

The key issue in terms of accessing geo data is human-readable to lat/long mappings. As an approximation the airports data works well because there is a human-readable name for the airport which includes the nearest town or city. This means we can search on the airports data using user-inputted names of places and get out the lat/longs. A similar (and more finegrained) approach would be to use the spacenamespace data; at the moment this is UK-only however. Where GPS data is available, a good modeling idiom is that used by Morten Frederiksen [ImageVocabs].

Modeling the nearestAirport information was difficult. It is not the nearestAirport to the picture as an artifact (the picture may be held on one or more servers, well away from the location). Nor is it necessarily a picture of a location. Instead, it's the location the camera was in when the picture was taken. Similar arguments apply to the date the picture was taken. An experimental new property, creationEvent, was created to test this out. The use of creationEvent masks a hidden resource - an object representing the event, to which nearestAirport and date can be attached.

```
<foaf:Image rdf:about='http://w3photo.org/photos/www2003/photos/IMG_4781_2.JPG'>
 <test:creationEvent rdf:parseType="Resource">
 <contact:nearestAirport>
  <doa:Airport rdf:about='http://www.daml.org/cgi-bin/airport?JFK'>
   <pos:lat>40.633333333333</pos:lat>
   <pos:long>-73.783333333333</pos:long>
   <doa:location>New York, NY [John F. Kennedy International Airport],
USA</doa:location>
   <vcard:country>US</vcard:country>
  </doa:Airport>
 </contact:nearestAirport>
</test:creationEvent>
</foaf:Image>
```

Describing the depiction of an event -

Sometimes it is very useful to be able to search for all the images taken at a

particular event, for example a conference or meeting. A simple way of doing this is to say that the image depicts the event, and then give the event a url. There are some difficulties with this approach: it effectively uses ical:url as an identifier for the event, and many events do not have a url, or do not have a single identifying url. Moreover, the semantics of ical:url are not quite appropriate for it to be used as an identifier - foaf:homepage might be better to use as an identifier.

```
<foaf:Image rdf:about='http://w3photo.org/photos/www2003/photos/IMG_4781_2.JPG'>
 <foaf:depicts>
  <ical:Vevent>
    <ical:url rdf:resource='http://www.www2003.org'/>
  </ical:Vevent>
 </foaf:depicts>
</foaf:Image>
```

Textual description and date -

Users can also add a freetext description. This is coded as the Dublin Core description of the image. Similarly, Dublin Core date is created using a form, although it would be better excerpted from EXIF data if available.

Rights for the image and the metadata -

It is important to assign rights where possible to the image and the RDF information about it created, and also important to say who the creator of the image and the author of the annotation are. This is very useful for re-using images in an authoring environment.

The tool uses Dublin Core, Annotea and Creative Commons to assign rights creation information and licensing information. Explanations of the Creative Commons licenses are available on the Creative Commons site ⊣ [CC].

Uploading the file -

When complete, you can upload the RDF data to a test server, or you can copy and paste the information to your own server. If you use the test server, you cannot then edit the data. If you use the automatic uploading feature it notifies the database. If on your own server you need to notify the RDFWeb database ⊣ [RDFWebDatabase] of the location of that file yourself; then that file will be harvested and its data available for querying through the codepiction interface ⊣ [CodepictionSearch].

Using the tool - developers

License -

The tool is licensed using the W3C license ⊣ [W3Clicense]. The UI tab component was written by Derek Anderson ⊣ [JSTab] and he has allowed me to reuse his code under this license. Please read the ⊣ license before downloading.

Download -

A zip of the Javascript code is ⊣ available ⊣ [Download].

Proxying data -

To make the code up on your own server, you will need to create a server-side proxy, which passes html pages and images through to your server intact. This is to bypass the security restrictions in Javascript, in order to download images form any source and use them directly in the application. It means that any html or image page could appear to come from your site, so be cautious if you use this approach. A safer approach might be to use the annotator only for photos on your site.

Saving data -

The experimental online service just posts the RDF data to a server-side application that saves the data locally and then loads it into a database. There is no security at present, although there are checks on the validity of the RDF. If you want to implement this yourself it would be best to add password restrictions on uploads. You could create the files and then automatically add to the RDFWeb database if you like ⊣ [RDFWebDatabase].

RDF data sources -

When accessing remote data, I use Jim Ley's Javascript RDF tools ⊣ [JSRDFTools] to access certain remote datasources accessible over http. There is currently no way for these to advertise themselves or route queries automatically between them - they are hardcoded within the application. Here are the interfaces:

⊣ Airports using location descriptions - query for 'Bristol'
⊣ People by name - query for Libby
⊣ General RDF database interface - access using Squish Queries ⊣ [Squish] - ⊣ example for events with title like www2004.
⊣ Wordnet database - query for 'Dog'.

Further work

Validation, loading and deleting data -

More useful features would be:

- prompt the user when parts of the RDf are missing
- delete and recreate data form a particular tab
- editing existing files when errors are detected

Generalization -

The tool requires that you select an image before you can create the RDF data describing it. Theoretically though, there's no reason why this tool could not be used to create RDF data about any document, any person, or any event.

Extensibility -

Using Javascript has meant that adding new functionality cleanly and consistently has been difficult. I have separated the functionality of each tab into a different file as a start.

Annotating multiple images -

Morten Frederiksen has a tool allowing the user to annotate multiple photos simultaneously - e.g. all these are pictures near Bristol. However it is not clear how to add this to the user interface.

Usability

As we have seen, several different kinds of information are presented in separate boxes on the page, and these can easily overflow a single screen, and some familiarity with the tools are required before users know where the result of a query will appear. With more work on the javascript and style sheets, these prototypes could be developed into visually appealing and unintimidating services.

Trust and Privacy

RDF can be used to say anything about anything, and coupled with the ability to annotate any image on the web, this could lead to both

- people annotating images that were not intended to be annotated - that were perhaps hidden by obscurity, and
- people annotating images incorrectly or maliciously.

Retaining the source of these annotations within the application and the software is therefore essential, in order to be able to remove annotations where there are privacy implications.

Extensions

Further kinds of data that we might want to include in annotations include:

- EXIF data
- Geographical information

References

[ImageVocabs]
Survey of types of RDF used with image annotation
→ http://esw.w3.org/topic/ImageDescriptionRdfExamples

[W3PhotoWeblog]
w3photo image annotation work
→ http://esw.w3.org/mt/esw/archives/000038.html

[W3PhotoVocabs]
Vocabularies for w3photo project
→ http://esw.w3.org/topic/W3PhotoVocabs

[DUP]
SWAD-Europe Deliverable 3.5: Dissemination and Use Plan
→ http://www.w3.org/2001/sw/Europe/reports/dissemination_use_plan/

[SWADE1]
SWAD-Europe Deliverable 3.6: Developer Workshop Report 1 - EARL and Image annotation
→ http://www.w3.org/2001/sw/Europe/reports/dev_workshop_report_1/

[SWADE2]
SWAD-Europe Deliverable 3.12: Developer Workshop Report 5 - Image annotation
→ http://www.w3.org/2001/sw/Europe/reports/dev_workshop_report_5/

[ImageParts]
Image Region Vocabulary: OWL DL Summary Benjamin Nowack et al.
→ http://www.bnowack.de/w3photo/pages/image_vocabs

[JSParser]
Jim Ley's Javascript RDF parser
→ http://jibbering.com/rdf-parser/

[GPL]
GNU General Public License
→ http://www.gnu.org/copyleft/gpl.html

[Download]
Javascript Image Annotation Download (zip)
→
http://www.w3.org/2001/sw/Europe/reports/report_semweb_access_tools/jsimageto
(.zip file)

[Codepiction]
Codepiction (Dan Brickley)
⊣ http://rdfweb.org/2002/01/photo/

[CodepictionSearch]
Codepiction search interface (Libby Miller, Dan Brickley)
⊣ http://swordfish.rdfweb.org/discovery/2001/08/codepict/

[CodepictionPaths]
Codepiction paths interface (Damian Steer, Libby Miller, Dan Brickley)
⊣ http://swordfish.rdfweb.org/discovery/2002/02/paths/

[FOAFFinger]
FOAFFinger (Damian Steer)
⊣ http://rdfweb.org/people/damian/foaffinger/

[ImgBot]
A Semantic Web Shoebox - Annotating Photos with RSS and RDF (Matt Biddulph)
⊣ http://www.hackdiary.com/slides/www2003.pdf

[JSShell]
JavaScript Shell: A command-line interface for JavaScript and DOM (Jesse Ruderman)
⊣ http://www.squarefree.com/shell/

[SVGOutlines]
Adding SVG outlines to co-depiction photo metadata (Jim Ley)
⊣ http://jibbering.com/rdf/foafwho.html

[SpaceNS]
spacenamespace (Jo Walsh)
⊣ http://space.frot.org/

[WP9]
SWAD-Europe Workpackage description: 9: Visualization and Accessibility
⊣
http://www.w3.org/2001/sw/Europe/plan/workpackages/live/esw-wp-9.html

[JSPhotoAnno]
Javascript Photo annotation tool - Libby Miller
⊣ http://swordfish.rdfweb.org/discovery/2004/03/w3photo/annotate.html

[FOAF]
FOAF Namespace
⊣ http://xmlns.com/foaf/0.1/

[WN]
Experimental Wordnet namespace (Dan Brickley)
⊣ http://xmlns.com/wordnet/1.6/

[CC]
Creative Commons
⊣ http://creativecommons.org/

[DC]
The Dublin Core Metadata Initiative
⊣ http://dublincore.org/

[FOAFProject]
FOAF Project

⊣ http://www.foaf-project.org/

[RDFiCal]
RDF iCal Vocabulary
⊣ http://www.w3.org/2002/12/cal/

[RDFWebDatabase]
RDFWeb Database - add a url to the database
⊣ http://swordfish.rdfweb.org/rweb/addurl

[W3Clicense]
W3C Software License
⊣ http://www.w3.org/Consortium/Legal/2002/copyright-software-20021231

[JSTab]
kered.org: HTML/DOM Tabbed Panes
⊣ http://kered.org/article-2003.06.24-html_tabs.html

[Squish]
Squish Query language
⊣ http://swordfish.rdfweb.org/rdfquery/

[RelatedTools]
SWAD-Europe Weblog: Image Annotation Tools
⊣ http://esw.w3.org/mt/esw/archives/cat_image_annotation.html