Designing and Creating a Web Site Based on RDF Content

Eero Hyvönen, Markus Holi, and Kim Viljanen
Helsinki Institute for Information Technology (HIIT), University of Helsinki
P.O. Box 26, 00014 UNIV. OF HELSINKI, FINLAND,
FirstName.LastName@cs.Helsinki.FI
http://www.cs.helsinki.fi/group/seco/

Abstract

This paper presents a method and a tool for designing and automatically creating an HTML web site for publishing Semantic Web content represented in RDF(S). The idea is to specify the needed RDF to HTML transformation on two separate levels. On the HTML level, the layout of the pages can be described by an HTML layout designer by using templates and tags. On the RDF level, the semantics of the tags are specified by a system programmer in terms of logical rules based on the RDF(S) repository. The idea is to apply logic for defining the semantic linkage structure and the indices of the page repository. The method has been implemented as a tool called SWeHG *for generating a static, semantically linked site of HTML pages from an RDF repository. As real life case applications, web exhibitions generated from museum collection metadata are presented.*

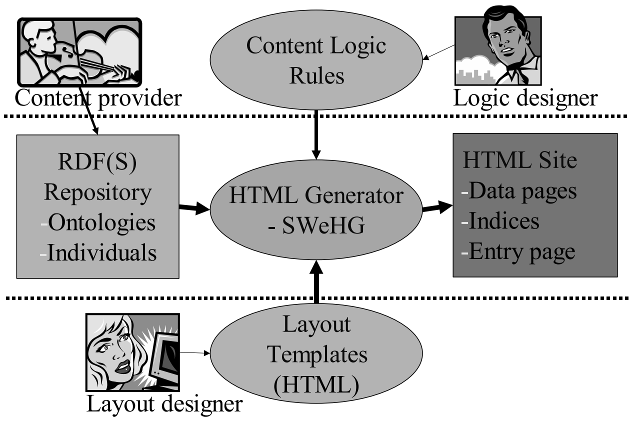


Figure 1. Rendering RDF(S) content as an HTML web site.

1. Two Views of the Semantic Web

The notion of the Semantic Web[1][1, 3] has two interpretations. From the machine's viewpoint, the Semantic Web manifests itself as a distributed source of interpretable metadata concerning resources, such as web pages[2], documents, photos, and real world object. The metadata descriptions are given in terms of ontologies using frameworks and languages such as RDF(S)[3] and OWL[4]. From the human's viewpoint, the Semantic Web looks like the current web, i.e., it is a repository of HTML pages, but empowered with more useful semantics-based links, search engines, and intelligent web services.

A central question in the development of Semantic Web applications is how the content represented for the machine can be transformed for the human to view, i.e., how machine interpretable RDF(S) or OWL content proliferating the web can be rendered to the human end-user as a searchable and browsable HTML web site or space. In this paper we present a new approach and tool named "Semantic Web HTML Generator" (*SWeHG*) [9] to address this problem (cf. figure 1). The idea is to specify the structure and the layout of an HTML web site in terms of a set of HTML templates using a tag language. The templates can be used by a web layout designer who does not know the details of the underlying RDF(S) content or Semantic Web technologies. The semantics of the tags, i.e., the machine's view on the RDF level, is specified by a Semantic Web programmer in terms of logic predicates. A benefit of separating the HTML and RDF levels is that ontological details and variance can be hidden from the HTML designer. By modifying the semantics of the tag, content represented using different ontological structures can be mapped on the same HTML tags that the HTML designer is capable of using. The tag definitions can be re-used directly in applications based on

1 http://www.w3.org/2001/sw/
2 See, e.g., http://dmoz.org.
3 http://www.w3.org/RDF/
4 http://www.w3.org/2001/sw/WebOnt/

similar ontologies and annotation schemas. The templates provide a declarative description of the web site structure, indices, and linkage. By modifying the templates alone in HTML, the same RDF(S) content can more easily be rendered in different ways in different applications to human end-users.

In the following, we first discuss two examples of semantically indexed and linked HTML web sites generated by *SWeHG*. The layout specifications with the corresponding tag definitions needed for the RDF to HTML transformation are then discussed. After this, the transformation process and its implementation are presented. In conclusion, experiences of our research and experimentation are summarized, related work is described, and directions for further research are outlined.

2. Example Applications

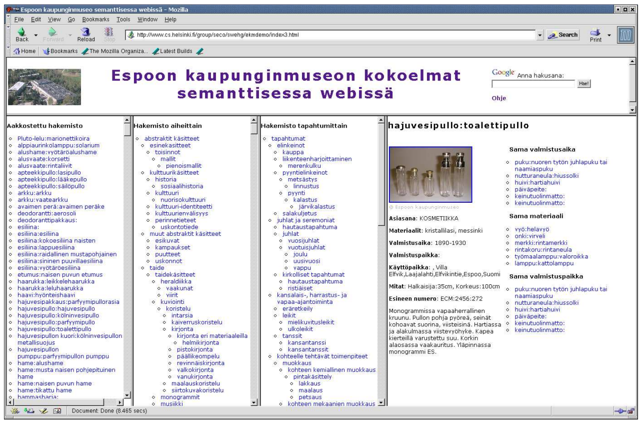
2.1. Helsinki University Museum

The virtual exhibition of a photo archive in the Helsinki University Museum[5] was generated. The archive contained 629 photographs about the promotion ceremonies of the University of Helsinki. The content of the archive was transformed into RDF(S) format in an other application project [7] and was used as it is by *SWeHG*. The domain knowledge consists of six ontologies with 329 promotion-related concept classes, such as "Person" and "Building", 125 properties, and 2890 instances, such as "Linus Torvalds" and the "Entrance of Cathedral of Helsinki".

In the photo annotation schema, the subject of a photograph is represented by a collection of ontology classes and individuals that appear on the image[6]. For example, if Linus Torvalds appears in a photo on a particular street, then the photo record is related *directly* with the corresponding person and street resources with a property corresponding to `dc:subject`. However, the relation between photos and subjects can be *indirect*, as well, involving traversal through several RDF arcs in the underlying knowledge base. For example, Linus Torvalds is present in a photograph as a Honorary Doctor. Then only an instance of such a role is associated with the image. The person instance in not directly linked with the image, but indirectly through the role instance. *SWeHG* predicate definition facility is very handy in hiding such annotation schema specific details from the HTML designer: the persons can be associated with images either directly or indirectly through roles. The criterion for association can be defined freely and conveniently by a declarative predicate.

5 http://www.helsinki.fi /museo/

6 The annotations also include other metadata, such as the photographer, free text descriptions, some technical information of the images, etc.


Espoon kaupunginmuseo semanttisessa webissä - Mozilla
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
Search Print
Home Bookmarks The Mozilla Organiza... Latest Builds
Google Anna hakusana:
Hae!
Ohje
Document: Done (8.465 secs)
© Espoon kaupunginmuseo

http://www.cs.helsinki.fi/group/seco/swehg/ekmdemo/index3.html

Espoon kaupunginmuseon kokoelmat semanttisessa webissä

Aakkostettu hakemisto

- Pluto-lelu:marionettikoira
- alppiaurinkolamppu:solarium
- alushame:vyötäröalushame
- alusvaate:korsetti
- alusvaate:rintaliivit
- apteekkipullo:lasipullo
- apteekkipullo:lääkepullo
- apteekkipullo:säilöpullo
- arkku:arkku
- arkku:vaatearkku
- avaimen perä:avaimen peräke
- deodorantti:aerosoli
- deodoranttipakkaus:
- esiliina:
- esiliina:esiliina
- esiliina:kokoesiliina naisten
- esiliina:lappuesiliina
- esiliina:raidallinen mustapohjainen
- esiliina:sininen puuvillaesiliina
- esiliina:vyötäröesiliina
- etumus:naisen puvun etumus
- haarukka:leikkelehaarukka
- haarukka:leluhaarukka
- haavi:hyönteishaavi
- hajuvesipakkaus:parfyymipullorasia
- hajuvesipullo:hajuvesipullo
- hajuvesipullo:kölninvesipullo
- hajuvesipullo:parfyymipullo
- hajuvesipullo:toalettipullo
- hajuvesipullon kuori:kölninvesipullon metallisuojus
- hajuvesipullon pumppu:parfyymipullon pumppu
- hame:alushame
- hame:musta naisen pohjepituinen hame
- hame:naisen puvun hame
- hame:tikattu hame
- hammasharja:

Hakemisto aiheittain

- abstraktit käsitteet
  - esinekasitteet
    - toisinnot
      - mallit
        - pienoismallit
  - kulttuurikäsitteet
    - historia
      - sosiaalihistoria
    - kulttuuri
      - nuorisokulttuuri
    - kulttuuri-identiteetti
    - kulttuurienvälisyys
    - perinnetieteet
      - uskontotiede
  - muut abstraktit käsitteet
    - esikuvat
    - kampaukset
    - puutteet
    - uskonnot
  - taide
    - taidekäsitteet
      - heraldiikka
        - vaakunat
        - viirit
      - kuviointi
        - koristelu
          - intarsia
          - kaiverruskoristelu
          - kirjonta
            - kirjonta eri materiaaleilla
              - helmikirjonta
            - pistokirjonta
            - päällikeompelu
            - revinnäiskirjonta
            - valkokirjonta
            - vanukirjonta
          - maalauskoristelu
          - siirtokuvakoristelu
      - monogrammit
      - musiikki

Hakemisto tapahtumittain

- tapahtumat
  - elinkeinot
    - kauppa
    - liikenteenharjoittaminen
      - merenkulku
    - pyyntielinkeinot
      - metsästys
        - linnustus
      - pyynti
        - kalastus
          - järvikalastus
    - salakuljetus
  - juhlat ja seremoniat
    - hautaustapahtuma
    - juhlat
      - vuosijuhlat
      - vuotuisjuhlat
        - joulu
        - uusivuosi
        - vappu
    - kirkolliset tapahtumat
      - hautaustapahtuma
      - ristiäiset
  - kansalais-, harrastus- ja vapaa-ajantoiminta
    - eräretkeily
    - leikit
      - mielikuvitusleikit
      - ulkoleikit
    - tanssit
      - kansantanssi
      - kansantanssit
  - kohteelle tehtävät toimenpiteet
    - muokkaus
      - kohteen kemiallinen muokkaus
        - pintakäsittely
          - lakkaus
          - maalaus
          - petsaus
      - kohteen mekaanien muokkaus

hajuvesipullo:toalettipullo

Asiasana: KOSMETIIKKA

Materiaalit: kristallilasi, messinki

Valmistusaika: 1890-1930

Valmistuspaikka:

Käyttöpaikka: , Villa Elfvik,Laajalahti,Elfvikintie,Espoo,Suomi

Mitat: Halkaisija:35cm, Korkeus:100cm

Esineen numero: ECM:2456:272

Monogrammissa vapaaherrallinen kruunu. Pullon pohja pyöreä, seinät kohoavat suorina, viisteisinä. Hartiassa ja alakulmassa viistevyöhyke. Kapea kierteillä varustettu suu. Korkin alaosassa vaakauritus. Yläpinnassa monogrammi ES.

Sama valmistusaika

- puku:nuoren tytön juhlapuku tai naamiaspuku
- nutturaneula:hiussolki
- huivi:hartiahuivi
- päiväpeite:
- keinutuolinmatto:
- keinutuolinmatto:

Sama materiaali

- vyö:helavyö
- onki:virveli
- merkki:rintamerkki
- rintakoru:rintaneula
- työmaalamppu:valoroikka
- lamppu:kattolamppu

Sama valmistuspaikka

- puku:nuoren tytön juhlapuku tai naamiaspuku
- nutturaneula:hiussolki
- huivi:hartiahuivi
- päiväpeite:
- keinutuolinmatto:
- keinutuolinmatto:

Figure 2. A photo exhibition generated with *SWeHG*.

Using *SWeHG* to publish the archive provides the end-users with two services. First, the photos can be found along the different orthogonal views based on the ontologies. Second, the photos can be browsed by using the links created between semantically related photos. The links are grouped based on the semantics of the link. For example, there is a link group that points to other photos taken of the same person.

2.2. Espoo City Museum

Figure 2 presents the home page of the exhibition "Espoo City Museum on the Semantic Web" that was generated using SWeHG for the museum[7]. Seven RDF(S) ontologies are used with some 10,000 classes and individuals and the metadata is described in terms of 38 properties. The RDF(S) repositories where originally created for the semantic portal MuseumFinland [6]. In this work, we could re-use the semantic recommendation predicates and the inference rule base developed for the original system, and the exhibition could be generated in a day or two.

In the RDF(S) repository, each ontological property of the collection objects in the exhibition, such as "material" is associated with a *domain ontology* of its own. For example, artifact, material, and technique ontologies have been defined based on the Finnish MASA Thesaurus [10] of keywords used in several museums for indexing data. The ontology MAO [8] created based on MASA contains some 6600 classes organized in a taxonomy. There is also a location ontology that defines geographical concepts such as "country" and "town". Their instances are individual areas

7 The exhibition is on the web at http://www.cs.helsinki.fi /group/seco/swehg/ekmdemo/

and places. The places are related with each other by a part-of meronymy. In the same way, an agent ontology defines concepts such as “person” and “company”, whose instances are active individuals. There is also an ontology for time periods and an ontology of collections in different museums. Still another ontology of “activities and processes” contains a taxonomy of concepts such as “wedding” and “fishing”. It is used to provide the end-user with an event-based view to cultural artifacts by associating them with corresponding events through annotations and logical rules. Each object’s metadata and annotations are given in an RDF card, that points to different classes and instances of the ontologies by the respective URIs through RDF properties. Some of the properties in an RDF card have literal values, and some point to resources by using URIs.

The created HTML site consists of some 1200 *resource web pages* (RPage) describing objects in the museum’s collection database, pages indexing the contents along different classifications, and a short user’s guide. On the left in figure 2, three frames containing indices for the underlying content are seen. The alphabetical index (“Aakkostettu hakemisto”) contains links to the RPages in alphabetical order. By selecting a link, the respective RPage is shown on the right. In figure 2, the user has selected a link to an RPage depicting perfume bottles. Before making a selection, the user’s guide was shown in the same frame. The classified index (“Hakemisto aiheittain”) is based on the RDFS taxonomy of the underlying cultural MAO ontology [8] that was used when creating the collection metadata. When selecting a concept, the rightmost frame shows links to its subconcepts together with links to RPages whose objects are directly related to the concept. By selecting a subconcept link there, the taxonomy can be browsed further downward; by selecting a link to an RPage, the corresponding collection object with its metadata can be viewed in the frame. The third index “Hakemisto tapahtumittain” classifies the collection objects by associating them with the different events, processes or activities in which the objects are used or otherwise related to.

By using the indices, the user can find collection objects of interest. An alternative way is to use a conventional search engine. In the upper right corner of figure 2 a form for using Google to search for the pages in the repository is seen. The hit list will be shown in the rightmost frame.

After finding an PRage of interest, the collection can be browsed by using the semantic links generated between related collection items. For example, in figure 2 links to objects manufactured at the same location, objects of similar material etc. can be clicked. The semantic links are generated based on the underlying ontologies, metadata, and logical recommendation rules.

The museum can publish the content by just copying the pages into a public HTML directory. This is of practical importance, since museums typically do not have competent IT personnel, servers, and resources to create and maintain semantic portals of their own.

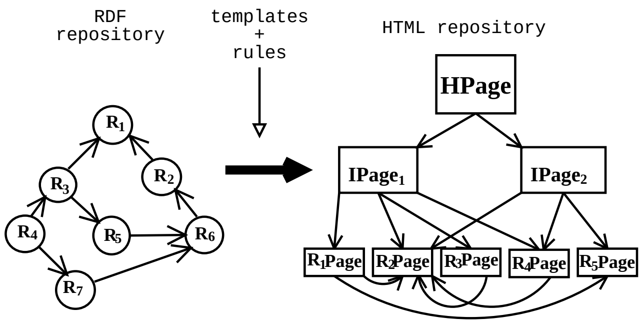


Figure 3. Transforming an RDF repository into HTML pages.

To sum up, the output of SWeHG is a semantically linked space of HTML pages of the following kind: 1) *Resource pages (RPage)* depict selected resources with their metadata. 2) *Index pages (IPage)* classify RPages along conceptual hierarchical classifications, that will be called *facets* or *views* [11]. By using IPages, RPages can be found along different facets. 3) A *home page (HPage)* defines the entrance page to the HTML repository.

3. Specifying the Transformation

Figure 3 depicts the RDF to HTML transformation. The RDF graph is on the left. Each R_n corresponds to a resource corresponding to a data entry in the RDF repository. In our example, the data entries are collection objects with their metadata. On the right, the HPage has links to various IPages classifying the underlying RPages that are related with each other by semantic links.

The transformation is based on descriptions on two levels: 1) The layout of the HTML pages is described on the *HTML level* by templates using custom tags. 2) The semantics of the tags is defined on the *RDF level* in terms of logical rules based on the input RDF(S) content. The idea is that an HTML designer can design the layout of the page repository to be generated by using tags without knowing details of the underlying RDF structures, RDFS ontologies, and Prolog programming. RDF(S) related knowledge as well as programming capability in Prolog is needed only for the system programmer when defining the tags. The same tag definitions can be re-used in applications conforming to similar ontological schemas.

SWeHG provides the HTML designer with three major tags: getProperty, getLinks, and getView. The tag <getProperty name=*p*> is used for rendering a label related to the resource underlying an RPage. For example, the metadata property values of the bottles and the photo in figure 2 are rendered in this way. The relation *p* can be specified by the system programmer on the RDF level freely by a binary logical predicate.

The tag <getLinks> is used for rendering links between RPages. For example, the tag

```
<swehg:getLinks name="SameLocation"
   listType="ul" listStyle="text-size: 10;"/>
```

could expand into the following HTML code linking photographs taken at the same location:

```
<ul style="text-size: 10;">
  <li><a href="entry.Mediacard_00071.html">
     View from Eiffel-tower</a></li>
  <li><a href="entry.Mediacard_00143.html">
     Cafe Parisienne</a></li> ...
</ul>
```

On the RDF level, the criterion `SameLocation` for the linkage could be defined by the predicate below[8]. It associates the attribute `SameLocation` with the HTML link label 'Same Place' and the predicate `photosWithSameLocation` defining the link relation.

```
swehg_relation_rule( 'SameLocation',
  'Same Place', photosWithSameLocation).
photosWithSameLocation(Context, Target) :-
  photo(Context), photo(Target),
  rdf(Context, _:place, Location),
  rdf(Target, _:place, Location),
  not(Context == Target).
```

The tag <getView> renders into a hierarchical index-like view of category resources used in IPages. Each category is associated with a set of subcategories and additional individuals of the categories. A view is defined by specifying 1) the root resource selector, 2) a binary subcategory relation predicate, and 3) a binary relation predicate that maps the hierarchy categories with the individuals used as leaves in the view. For example, the tag

```
<swehg:getView
  roots="buildings" branches="subclass"
  leaves="photoOf" listType="ul" />
```

expands recursively into a hierarchical unordered tree (`ul`), where the leaves are links to photo record resources related to different building categories. The predicate definitions defining the meaning of the attribute values can be, for example, the following:

```
buildings(URI) :-
  rdf(URI, rdf:type, 'http://some.org#building').
subclass(SubCategory, SuperCategory) :-
  rdf(SubCategory, rdfs:subClassOf, SuperCategory).
photoOf(Class, Record) :-
  rdf(Instance, rdf:type, Class),
  rdf(Record, dc:subject, Instance).
```

Here `buildings` selects the class `building` as the view root, and the hierarchy is expanded along the `rdfs:subClassOf` property. The `photoOf` predicate relates each building type *c* of this tree with a set of photo record resources which are used as the leaf categories of *c*. These are rendered as HTML links to the corresponding RPages. The tag definitions could also be much more complex than this, depending on the structure of the RDF(S) repository, and the desired output. The view expansion into HTML can be controlled with the help of additional tag attributes for, e.g., ordering the categories.

The following is an example of a complete RPage template. It could be used for rendering the images using the HTML `img`-tag and links to related RPages:

```
<swehg:template selector="photo">
<html>
<body>
  <h2><swehg:getProperty name="Title_Of_Photo"/></h2>
  <p><img src="<swehg:getProperty
               name="PhotoURL"/>" /></p>
  <h3>Photos from the same place:</h3>
  <swehg:getLinks predicate="sameLocation"
                listType="ul"/>
</body>
</html>
</swehg:template>
```

The tag attribute `selector` in the tag <swehg:template> tells the criterion for selecting *context resources* from the RDF repository. Each context resource will have an RPage of their own on the HTML level. The attribute value, here `photo`, is the name of a unary Prolog predicate called *selector* that should evaluate true for context resource URIs.

An example of a complete IPage template is given below using the view definitions above:

```
<swehg:template>
<html>
<body>
<h1>Building index</h1>
  <swehg:getView
    roots="buildings"
    branches="subclass"
    leaves="photoOf"
    orderby="order_alphabetically"
    listType="ul"/>
</body>
</html>
</swehg:template>
```

4. Web Site Generation

The process for transforming an RDF(S) repository into HTML pages is defined by the algorithms 1 and 2. The input of the procedure is a set of HTML templates, and an RDF(S) repository. The output is an HTML page repository conforming to the templates. The transformation is based

8 The examples are presented in SWI-Prolog (http://www.swi-prolog.org) syntax. Here RDF triples are presented as rdf(Subject, Predicate, Object). Underscore "_" is an unnamed variable.

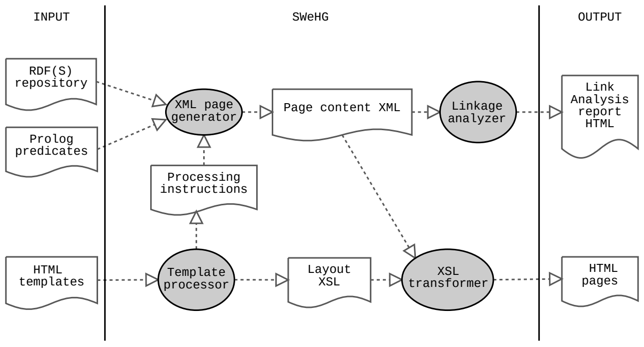


Figure 4. Internal architecture of *SWeHG*.

on a set of logical rules for selectors, properties, links, and views.

The pages are generated using the HTML templates one after another. If a template is associated with a selector, then it is expanded into a set of RPages corresponding to the selected context resources, else it is expanded once without a reference to a context resource. In the latter case, the HPage and IPages are created. When generating an HTML page, the tags are expanded into HTML in the ways described in the previous section.

```
Algorithm: RDF2HTML
Data: Templates T, RDF(S) repository R
HTMLPageRepository H = empty;
foreach Template t in T do
    if t has a selector rule S then
        foreach RDF Resource r in R do
            if S(r) == true then
                h = createHTMLpage(r, t);
                add h to H;
            end
        end
    end
    else
        h = createHTMLpage(T);
        add h to H;
    end
end
```

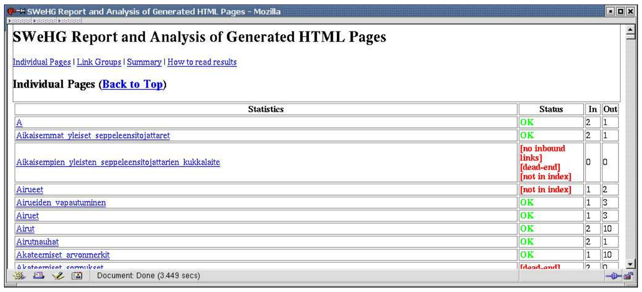
Algorithm 1: Main procedure for the RDF to HTML transformation

Figure 4 depicts the architecture of our implementation. The main program is a Perl script which first builds an XSLT [9] template out of the HTML templates using the module "Template processor". This module also writes out a set

9 http://w3.org/TR/xslt

```
Algorithm: createHTMLpage
Data: Template t, Context Resource r
Result: HTML page
String H = t;
foreach Tag in H do
    h = executeRule(Tag.rulename, r);
    replace Tag in H with h;
end
return H;
```

Algorithm 2: Algorithm createHTMLpage for rendering an HTML template. *Tag.rulename* returns the name of the rule, e.g., "getProperty".


SWeHG Report and Analysis of Generated HTML Pages - Mozilla

SWeHG Report and Analysis of Generated HTML Pages

Individual Pages | Link Groups | Summary | How to read results

Individual Pages (Back to Top)

Statistics	Status	In	Out
A	OK	2	1
Aikaisemmat_yleiset_seppeleensitojattaret	OK	2	1
Aikaisempien_yleisten_seppeleensitojattarien_kukkalaite	[no inbound links] [dead-end] [not in index]	0	0
Airueet	[not in index]	1	2
Airueiden_vapautuminen	OK	1	3
Airuet	OK	1	3
Airut	OK	2	10
Airutnauhat	OK	2	1
Akateemiset_arvonmerkit	OK	1	10
Akateemiset_sormukset	[dead-end]	2	0

Document: Done (3.449 secs)

Figure 5. An analysis page created by *SWeHG*.

of "Processing instructions" into a separate Prolog source code file. These instructions link template tags with the Prolog predicates used in them as attribute values. The module "XML page generator" is a Prolog program that applies the predicates used in the HTML tags with respect to the RDF repository according to the Processing instructions. The result is a set of XML files describing the page contents. These XML files are then transformed using Apache Xalan[10] and with the help of the XSLT templates generated earlier into the final HTML pages.

The intermediate XML files in figure 4 are also used as a basis for the "Linkage analyzer" module that tries to identify the following potential problems: *Self loops* (a link that points to the page itself), *Bad links* (link pointing to a non existing page), *Dead ends* (an RPage with no outbound links), *No way in* (an RPage with no inbound links from any RPages or IPages), *Not in index* (an RPage with no inbound links from any IPage), and *Unused rules* (rules that are newer referred to when generating the HTML repository). The analysis results are represented as HTML pages. This helps the designer in debugging the specifications.

Figure 5 depicts a portion of the result from the analyzer.

10 xml.apache.org/xalan-j/

On this page the number of in-coming and out-going links can be seen for each RPage together with a status explanation. The analyzer has found out that the page with label "Aikaisempien yleisten ..." is not connected with any other page or index. Furthermore, the page "Airueet" has one incoming and two outgoing links but was not included in any index. This kind of connectivity information is vital when debugging the logical rules that produce the HTML pages.

5. Discussion

5.1. Benefits and Limitations

Our initial experiences indicate that the presented RDF to HTML transformation method is feasible. HTML templates can be created fairly easily and can be adapted to different RDF repositories. Moreover, changes in ontology versions do not affect the usage of the templates on the HTML level in any way. The idea of using logic and Prolog for defining the semantics of the tags seems powerful. Complicated semantic link relations and views can be defined and modified easily thanks to the declarative nature of logic programming. By using generic rules it is possible, in principle, to create tag definitions that will apply to any RDF repository. In contrast to view-based search systems, such as [11, 5], the views are projected from the RDF(S) ontologies. The main benefit is that arbitrary mappings between view categories and data resources can be flexibly defined. The system infers the mapping between views and resources which gives it an "intelligent" flavor. Furthermore, the HTML pages are linked semantically with each other according to the ontologies, metadata, and rule base used. To the end-user, the underlying hidden associations between collection objects is a most interesting aspect of cultural collections. The nature of the associations can be explained to the user by the labels of the links.

The tag definitions are not application specific, and can be used also in different applications that use the same RDF(S) content. For example, we could use the linkage rules, the selector rules, and the rules generating the views of the indices of the "Espoo City Museum on the Semantic Web" developed originally for a semantic portal [6]. On the other hand, the tag language of SWeHG is limited, and it can not be extended easily. Also, the set of different HTML outputs that the tags produce is limited at the moment. The output varies from simple strings to lists of links. In addition, SWeHG does not offer sufficient tools for testing the tag definitions before the actual transformation. A preview or debug function would be useful, because when the RDF(S) database is large, then the transformation process is long.

SWeHG generates static pages in a batch process before publishing them on the web. This approach has the following benefits when compared with dynamic semantic portals: The page repository can be published easily by just copying it into a public HTML directory. SWeHG can be adapted to different contents conforming to different ontologies. The publication process is independent from semantic portal providers—no special server software is needed. The pages need no special maintenance. The static pages are indexed and searched for by general search engines. The pages can be viewed efficiently. Data security problems are minimal. The properties of the resulting HTML page set can be analyzed efficiently.

On the other hand, the static approach taken in SWeHG also has, of course, its limitations. First, static pages can not adapt their content dynamically to different user or patterns of usage. Second, dynamic systems can be connected more easily with other services providing additional functionality. Third, if the RDF repository, the rules, or the HTML templates change, the site has to be regenerated usually from scratch. Dynamic systems can adapt better to such changes. Fourth, if the RDF repository is large and many templates are used, then the number and size of generated pages can be large.

Clearly, both the dynamic and static approaches have their own virtues and application possibilities.

5.2. Related Work

Logic and dynamic link creation on the semantic web have been discussed, e.g., in [4, 2]. Our approach is different in it's use of HTML templates and Prolog for describing the static HTML output. In the *RDF Twig* tool[11] the RDF to HTML transformation is based on XSLT. A problem here is that an RDF graph can be serialized in many ways in XML. Different applications may produce different XML serializations of the same RDF graph, and thus a number of XSLT templates would have to be written for a single graph. In our approach only actual changes in the graph structures are relevant, because in SWI-Prolog, by which we define the logical rules, the RDF graph is processed purely as triplets. In *Spectacle*[12] the RDF to HTML transformation is based on APIs. Then the user must write programs that use the API, and also an application server is needed. In contrast, our approach is based on tags, is declarative, and the result is a set of static pages whose linkage structure is inferred by logical linking predicates.

5.3. Directions for Future Work

SWeHG is a research prototype. More work and testing is still needed in order to evaluate and enhance the usability and extendability of system in different applications. More

11 http:/rdftwig.sourceforge.net/
12 http://www.aidministrator.nl/spectacle/

work is also needed in optimizing the efficiency of the code and in providing better development tools for the HTML designer and system programmer using the system.

Acknowledgments

A. Valo contributed significantly to the implementation of SWeHG. Also M. Kiesilä, V. Komulainen, R. Köppä-Laitinen, and J. Muhonen participated in the implementation project. Our work was mainly funded by the National Technology Agency Tekes, Nokia Corp., TietoEnator Corp., the Espoo City Museum, the Foundation of the Helsinki University Museum, the National Board of Antiquities, and the Antikvaria-group of some 20 Finnish museums.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 284(5):34–43, May 2001.

[2] P. Dolong, N. Henze, and W. Neijdl. Logic-based open hypermedia for the semantic web. In *Proceedings of the Int. Workshop on Hypermedia and the Semantic Web, Hypertext 2003 Conference, Nottinghan, UK*, 2003.

[3] D. Fensel, J. Hendler, H. Lieberman, and W. Wahlster, editors. *Weaving the Semantic Web*. The MIT Press, 2002.

[4] C. Goble, S. Bechhofer, L. Carr, D. De Roure, and W. Hall. Conceptual open hypermedia = the semantic web? In *Proceedings of the WWW2001, Semantic Web Workshop, Hongkong*, 2001.

[5] M. Hearst, A. Elliott, J. English, R. Sinha, K. Swearingen, and K.-P. Lee. Finding the flow in web site search. *CACM*, 45(9):42–49, 2002.

[6] E. Hyvönen, M. Junnila, S. Kettula, E. Mäkelä, S. Saarela, M. Salminen, A. Syreeni, A. Valo, and K. Viljanen. Finnish Museums on the Semantic Web. User's perspective on museumfinland. In *Proceedings of Museums and the Web 2004 (MW2004), Arlington, Virginia, USA*, 2004. http://www.archimuse.com/mw2004/papers/hyvonen/ hyvonen.html.

[7] E. Hyvönen, S. Saarela, and K. Viljanen. Ontogator: combining view- and ontology-based search with semantic browsing. In *Proceedings of the XML Finland 2003 conference. Kuopio, Finland*, 2003. http://www.cs.helsinki.fi/u/eahyvone/publications/ xmlfinland2003/yomXMLFinland2003.pdf.

[8] E. Hyvönen, M. Salminen, S. Kettula, and M. Junnila. A content creation process for the Semantic Web, 2004. Proceeding of OntoLex 2004: Ontologies and Lexical Resources in Distributed Environments, May 29, Lisbon, Portugal (forthcoming).

[9] E. Hyvönen, A. Valo, K. Viljanen, and M. Holi. Publishing semantic web content as semantically linked HTML pages. In *Proceedings of XML Finland 2003, Kuopio, Finland*, 2003. http://www.cs.helsinki.fi/u/eahyvone/publications/ xmlfinland2003/swehg_article_xmlfi2003.pdf.

[10] R. L. Leskinen, editor. *Museoalan asiasanasto*. Museovirasto, Helsinki, Finland, 1997.

[11] A. S. Pollitt. The key role of classification and indexing in view-based searching. Technical report, University of Huddersfield, UK, 1998. http://www.ifla.org/IV/ifla63/63polst.pdf.