Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

INS

Information Systems

INformation Systems

Towards Smart Style: Combining RDF Semantics with XML Document Transformations

Jacco van Ossenbruggen, Lynda Hardman,
Lloyd Rutledge

REPORT INS-E0303 OCTOBER 27, 2003

CWI is the National Research Institute for Mathematics and Computer Science. It is sponsored by the Netherlands Organization for Scientific Research (NWO).
CWI is a founding member of ERCIM, the European Research Consortium for Informatics and Mathematics.

CWI's research has a theme-oriented structure and is grouped into four clusters. Listed below are the names of the clusters and in parentheses their acronyms.

Probability, Networks and Algorithms (PNA)

Software Engineering (SEN)

Modelling, Analysis and Simulation (MAS)

Information Systems (INS)

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

ISSN 1386-3681

Towards Smart Style: Combining RDF Semantics with XML Document Transformations

ABSTRACT
The “Document Web" has established itself through the creation of an impressive family of XML and related languages. In addition to this, the “Semantic Web" is developing its own family of languages based primarily on RDF. Although these families were both developed specifically for “the Web", each language family has been developed from different premises with specific goals in mind. The result is that combining both families in a single application is surprisingly difficult. This is unfortunate, since the combination of semantic processing with document processing provides advantages in both directions --- namely using semantic inferencing for more intelligent document processing and using document processing tools for presenting semantic representations to an end-user. In this paper, we investigate this integration problem, focusing on the role of (RDF) semantics in selecting, structuring and styling (XML) content. We analyze the approaches taken by two example architectures and use our analysis to derive a more integrated alternative.

1998 ACM Computing Classification System: H.5.4 Hypertext/Hypermedia
Keywords and Phrases: Semantic Web, Semantic Transformations, XSLT, RDF
Note: The research described here was funded by the Dutch national NWO NASH project, Telematica Instituut's Topia project and the NWO ToKeN2000/I2RP project.

Towards Smart Style: Combining RDF Semantics with XML Document Transformations

Jacco van Ossenbruggen, Lynda Hardman, Lloyd Rutledge

Abstract

The "Document Web" has established itself through the creation of an impressive family of XML and related languages. In addition to this, the "Semantic Web" is developing its own family of languages based primarily on RDF. Although these families were both developed specifically for "the Web", each language family has been developed from different premises with specific goals in mind. The result is that combining both families in a single application is surprisingly difficult. This is unfortunate, since the combination of semantic processing with document processing provides advantages in both directions — namely using semantic inferencing for more intelligent document processing and using document processing tools for presenting semantic representations to an end-user.

In this paper, we investigate this integration problem, focusing on the role of (RDF) semantics in selecting, structuring and styling (XML) content. We analyze the approaches taken by two example architectures and use our analysis to derive a more integrated alternative.

1 Introduction

The Semantic Web aims at making the information on the Web *machine-readable*. It does so by building an ever increasing stack of metadata-related Web languages on top of RDF, including RDF Schema and the variants of OWL. It silently assumes that the initial problem — of making information on the Web *human-readable* — has already been solved by the "Document Web". With the Document Web, we mean the ever increasing stack of languages built on top of XML, including XHTML, CSS, SVG, SMIL, MathML, XSLT and XPath.

The Semantic Web and the XML communities have indeed both built an impressive set of languages, but each set has been developed with particular goals in mind. Even though these languages have been developed within the same organization — the World Wide Web Consortium (W3C) — it turns out to be surprisingly difficult to combine both worlds in a single application.

The two key integration problems reflect two sides of the same coin. On the one hand, we have a large set of XML-based tools that do not support RDF-related languages. From a document engineering perspective, this means that it is extremely hard to build XML-oriented applications that make use of the machine-readable, explicit knowledge available on the Semantic Web. On the other hand, we have a large set of RDF-based tools that are not designed to be interoperable with the XML-related tools. From a knowledge engineering perspective, this means that it is extremely hard to build RDF-oriented applications that make their knowledge available for human consumption by using readily-available XML-oriented technology.

We are of course not the first to point out the gulf between the semantics expressible in the RDF graph and the hierarchical syntactical structures expressible within XML [21]. We share the goal of Patel-Schneider and Siméon of allowing each side of the fence to benefit from the other, that is, how the RDF world can take advantage of XML (in particular XML query languages), and how the XML world can take advantage of the reasoning capabilities available for RDF. Our emphasis, however, is on creating and manipulating documents for (semantically coherent) presentation to end-users, rather than the underlying representation of knowledge or data. This gives a slightly different perspective on the roles of RDF and XML and their relationship to each other. Our goal is to allow the use of semantics to guide how "syntax"

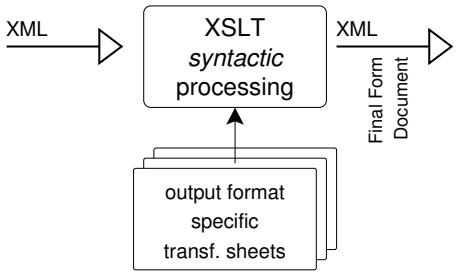

Figure 1: Syntactic processing in XSLT.

transformations need to be carried out, not only in order to improve the quality and flexibility of the results, but also to make sure the results still make sense in the case of more complex transformations. Our focus is on integration of semantics and syntax in the context of document transformation and document style, rather than unification of the representation and querying languages.

Figure 1 sketches the familiar "syntactic" transformation process (as typically carried out by XSLT). The application of such an XSL transformation is usually seen as a straightforward syntax transformation. The two prototypical applications are the use of XSLT to map one document format to the other (e.g. for interoperability purposes) and its use as a style language in cases where CSS alone does not suffice (e.g. because XSL formatting objects are needed or the presentation needs to be reordered). Both applications are typically regarded as "just syntactic" or "only style".

In this article, we claim that this assumption is, in many cases, a blunt oversimplification of the truth (see also [26, 27]). A well designed XSLT transformation sheet generates a coherent presentation that effectively conveys information to a human end-user. Critical aspects of this task include selection of the content, structuring of this content and styling of the content. These three tasks are in fact tasks many of us need to perform manually on a daily basis. Writing scientific papers, making slide show presentations or designing interactive web pages are all examples that combine these three crucial tasks. We all know how hard they are to do right, and how knowledge intensive each task is. They require in-depth knowledge of the topic, the target audience, and the medium used. So a well-designed XSLT transformation sheet uses these types of knowledge to do what it is supposed to do. Unfortunately, XSLT remains a language designed for syntax transformations. All semantics remains implicit, deeply hidden in the functional transformation rules that make up the XSLT code.

The rest of the paper is structured as follows. After a related work section we describe the Topia and Cuypers projects as motivating our problem. We then discuss the role of semantics in content selection, structuring and styling in the document transformations that characterize these projects. We analyze the pros and cons of each approach and used this analysis as the basis for the more integrated approach proposed in Section 5.

2 Related work

The use of explicit semantics for the automatic generation of user interfaces in general, and multimedia presentations in particular, is far from new. For example, Boll et al. have done relevant work on the semantics of multimedia adaptation [4, 5]. Research on model-driven multimedia presentations in the early nineties by the AI community employed planning, grammars, machine learning and constraint satisfaction techniques to generate multimedia presentations from explicit models of the content of these presentations [1, 2, 3, 12, 29, 31].

While this fruitful line of research culminated in the Standard Reference Model for Intelligent Multimedia Presentation Systems (SRM-IMMPS [6]), the field seems to have faded in importance since the late nineties. The emergence of the Semantic Web, however, has triggered new interest, with work on the use of semantics for hyperlink generation [8] and ontology-based knowledge extraction and biography generation [19].

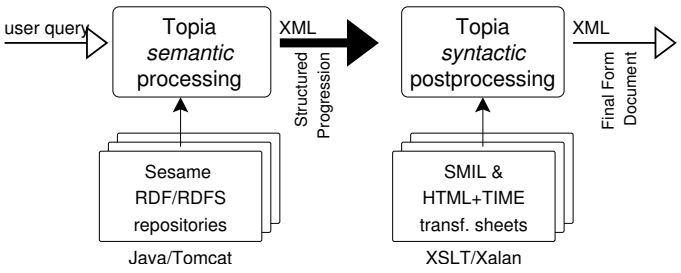

Figure 2: Semantic processing and syntactic post-processing in Topia.

A variant of XSLT that works directly on the RDF Graph has been developed by Ian Davis under the name RDF Templates [10]. This, however, focuses on the use of an XSLT-like mechanism for transforming RDF graphs, rather than incorporating explicit semantics during the transformation process.

Generating documents, websites and visualization directly from RDF data is also a well researched area [23, 18, 17]. The gap between XSLT, working on the XML surface syntax, and the underlying semantics of the RDF graph has been discussed by Cawsey in [9], where she emphasizes the use of deeper reasoning on the XML serialization of the RDF as future work.

3 The Topia project

The key research question in the Topia project is to what extent one can automatically generate a coherent, informative presentation from an annotated multimedia database without relying on application-specific approaches. An architecture has been developed for generating Web presentations from RDF-encoded metadata stores. The project's demonstrator[1] is based on an RDF network of 1250 artifacts in the Rijksmuseum Amsterdam. In this section, we give only a brief summary of the role of semantics in Topia's content selection, structuring and styling process. See [22] for a more complete overview.

The content that is to be conveyed to the user is selected by querying the RDF repository. Note that this requires an extra processing step (implemented in Java) before the actual XSLT transformation process — see Figure 2. The Java code also takes care of structuring the content by clustering the query results. This organizes the information into an explicit *hierarchical*structure in which all content is explicitly *ordered*, and *recurrent* objects that appear at multiple locations in the tree are explicitly linked.

The resulting simple discourse structure is called a *structured progression*, and is encoded in XML. This provides the content sufficiently structured for the XSLT step to convert it directly to the final presentation, for example in HTML or SMIL.

This last step can be done server-side, or client-side if the end-user is using an XSLT-enabled client. The XSLT transform is responsible for all remaining layout and style decisions, while it has no access to the original RDF graph. This means that all information needed for determining an effective layout and style for a particular presentation needs to be explicitly represented in the structured progression.

4 The Cuypers multimedia transformation engine

The Cuypers engine [24] was developed to provide a testbed for our research on the automatic generation of multimedia documents. Cuypers selects multimedia data items from a repository and combines these into a coherent multimedia presentation [25]. In contrast to the Topia project, the initial focus of the engine was not on creating a coherent discourse, but on automatically adapting the spatio-temporal layout of multimedia presentations to a wide variety of situations, including different devices with varying screen sizes and aspect ratios, varying network bandwidths and user preferences.

To achieve this, Cuypers supports a Constraint Logic Programming (CLP) based transformation model that exploits the constraint-programming paradigm to provide high-level specifications of multimedia lay-

[1] On-line demo at: `http://topia.demo.telin.nl/`

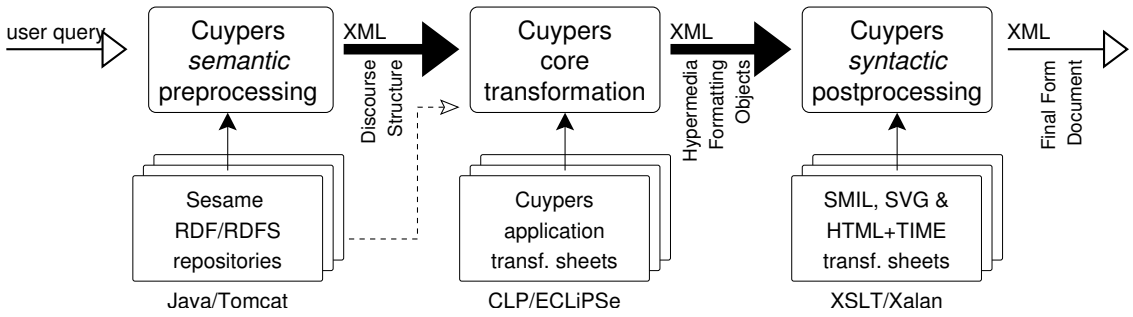

Figure 3: Semantic pre- and syntactic post-processing in Cuypers.

outs, with the unification and backtracking technique of logic programming to provide alternative transformation strategies in cases when the constraints fail (e.g. when attempting to generate a presentation that does not fit on the target device's screen or consumes too much bandwidth).

Despite this focus on multimedia layout, a recurring problem was the large amount of knowledge that was needed to select, structure and style the appropriate multimedia content in an effective way.

Figure 3 sketches an overview of the semantic and syntactic processing in the Cuypers engine. In the middle is the core transformation process. The semantic processing that is not directly part of the core transformation is done in the preprocessing stage, by introducing an (XML-encoded) abstraction layer named the *discourse structure*.

The constraint logic programming paradigm used within Cuypers allows us to encode most of the transformation code at a conveniently high level. The exception to the rule was formed by the transformation's dependence on the low level syntax details of the final delivery format, such as SMIL or HTML+TIME. This problem was addressed by introducing a more abstract intermediate format, defined by the Cuypers formatting vocabulary for time-based hypermedia [25].

The Cuypers hypermedia formatting objects (HFO) can be seen as a first step towards a time-based hypermedia equivalent of the text-based formatting objects (FO) defined by XSL [28]. The final transformation, from the HFO document to the final presentation format is a purely syntactic operation, and can be encoded relatively straightforwardly in XSLT (see [25] for details).

Unlike the final XSLT step of the Topia project, the final XSLT step of the Cuypers project does not require direct access to any RDF semantics. This advantage, however, comes with a price: instead of a standard XSLT transform, all intelligent processing is done by Java and CLP code that depends heavily on the Cuypers server run-time environment and is hard to port to a standard Web server or even to a client-side scenario.

In the two subsections below, we discuss the role of semantics in the content selection, structuring and styling of two Cuypers demonstrators.

4.1 Cuypers/OAI: Using Dublin Core Metadata

The Cuypers/OAI demo[2] exploits the explicit knowledge provided by (Dublin Core [11]) metadata about information provided by the Open Archive Initiative (OAI). The OAI metadata is used for both content selection and structuring the presentation. In [20], the authors discuss the use of Dublin Core metadata to derive relationships between the items returned by a query to a multimedia database, and how these relationships can be used to drive the multimedia generation process in the Cuypers engine.

For the implementation, the metadata used is stored in an SQL database, accessed by a separate module that "mines" the metadata at run-time for relevant relations. It pre-processes the results into an XML-encoded discourse structure using some hardwired rules. This is then further processed by the core constraint-based Cuypers transformations (that take care of all layout and style decisions) and served as SMIL 1.0, SMIL 2.0 or HTML+TIME by a final XSLT step.

[2]On-line demo at `http://homepages.cwi.nl/~media/demo/oai/`

While the prototype demonstrates to what extent common Dublin Core metadata can be used for presentation purposes, it also demonstrates the need for a more generic and reusable module in Cuypers for accessing and processing metadata. In particular, the hardwired mapping of semantic relations to a presentation discourse needs to be replaced by a more flexible approach.

4.2 Cuypers/DISC: Using RDF Schema-encoded Discourse Knowledge

The *DISC* prototype[3], explores the use of explicit discourse knowledge for multimedia generation [13]. It models a set of rules that can map the knowledge about the domain to small, abstract units that declaratively describe fragments of the narrative structure of the presentation that will be generated. Based on these units, and the rules that describe how to combine them, an abstract structure of the presentation is built automatically. This structure then drives the Cuypers engine to create a final multimedia presentation.

DISC demonstrates how discourse knowledge can be made explicit within Cuypers. It also demonstrates how this knowledge can be encoded using standard Semantic Web languages such as RDF and RDF Schema, rather than being stored in a Cuypers-specific data or knowledge base. The DISC implementation defers all functionality for storing and querying the RDF and RDF Schema encoded information to Sesame [7], and can use commonly used RDF editors such as Protege-2000 [14] for editing the discourse knowledge.

A Java servlet processes the user query and builds this semantic description of the presentation. For this task, the servlet can deploy a number of RDF query languages to have direct, efficient and high level access to a variety of RDF(S) metadata and ontology repositories managed by Sesame (see [13] for details).

Again, the core transformation takes the discourse structure as its input and transforms this to a multimedia presentation, which is converted to the final form format by an XSLT transformation.

The DISC demo has, however, the same limitation as the OAI demo described above, in that it only uses the RDF metadata in a first pre-processing step, prior to the second transformation step that is the key process of the Cuypers engine. While the results of this first step are used as input to the second step, when this transformation step requires extra knowledge, it has no access to the information available in the Sesame RDF repository.

5 Towards an integrated transformation model for the Semantic Web

The discussion so far can be summarized by stating that the main task of style and transformation sheets is the effective communication of a message to an end user. This task can be divided into three subtasks: content selection, structuring and styling. All three subtasks are inherently knowledge intensive since they require understanding of the message that is to be conveyed, the target audience to which the message needs to be conveyed and the medium that is used to convey the message.

The current, purely syntactic application of CSS and XSL style sheets and XSLT document transformations takes no explicit semantics into account. It assumes, either, that all design choices can be made independently of the content of the documents being styled or transformed, or, that the required knowledge has been incorporated into the style sheet (and has thus become implicit). To make the appropriate design choices at the appropriate time, easy access to knowledge that drives these design choices is a requirement for almost every non-trivial transformation process.

The example architectures discussed above explore the use of explicit semantics during the transformation, but at a cost. First, they subdivide the transformation into a number of steps that are implemented using different environments: Java and XSLT in Topia; Java, CLP and XSLT in Cuypers. While this approach has the advantage that at each step the most suitable environment can be used, the disadvantage is that it is more difficult to access information across the various steps (e.g. the lack of access to RDF during Topia's XSLT step and Cuypers' CLP step). Second, by introducing non-standard (read non-XSLT and non-CSS) style and transformation steps, they lose the advantages of standardization (including portability across server implementations and flexibility in server or client-side execution models).

To address these issues, we explored to what extent it is feasible to access RDF semantics directly from within XSLT transformation sheets. This approach, sketched in Figure 4 on the next page, could

[3] On-line demo at `http://homepages.cwi.nl/~media/demo/disc/`

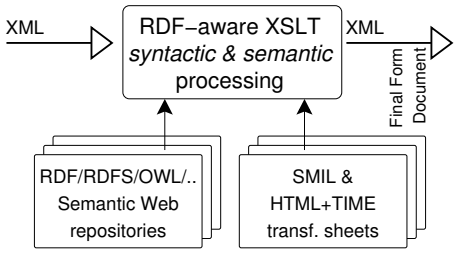

Figure 4: Semantic Web aware XSLT scenario, integrating semantic and syntactic processing

```
<queryResult>
 <tuple>
   <uri>http://www.cwi.nl/~media/publications/ins.rdf#lynda_hardman</uri>
   <literal>Lynda Hardman</literal>
  </tuple>

  <tuple>
    <uri>...</uri>
    <literal>...</literal>
  </tuple>
</queryResult>
```

Figure 5: Query results: an XML-encoded list of tuples returned to XSLT.

potentially reduce the complexity and lack of flexibility, and return to an architecture that is closer to our initial Figure 1 on page 2 than to those of the example architectures in Topia (Figure 2 on page 3) and Cuypers (Figure 3 on page 4).

The approach relies on XSLT's extensibility[4]. Our current proof-of-concept prototype uses a small but powerful XSLT extension to be able to query a Sesame RDF repository from within the XSLT transformation sheet. This extension can be used to semantically augment all three subtasks normally found in an XSLT style sheet: content selection, structuring and styling. It gives potentially every template rule in XSLT access to the RDF repository by any of the query languages supported by Sesame (currently the RDF query language RDQL, and the RDF/RDF Schema query languages RQL and SeRQL).

The extension provides a thin interface between XSLT and Sesame. Only one extension function is available to select the desired RDF repository: `sesame:repository(name)`. In addition, three extension elements are available, one for each of the three query languages currently supported by Sesame: `<sesame:rql>`, `<sesame:rdql>` and `<sesame:serql>` (in the example, we use only SeRQL queries with the `<sesame:serql>` element).

The result of each select query is a list of zero or more tuples, which is returned to XSLT as a simple XML result tree fragment as shown in Figure 5. The structure of this tree follows the internal format used by Sesame: all result trees have `<queryResult>` as the root element, with the resulting tuples as their direct children. Each element in the tuple is denoted by its type, currently only the most common `<literal>` for RDF literals and `<uri>` for RDF resources are supported.

The example below illustrates the potentials of this approach by selecting content from an RDF repository and styling[5] the results by using further ontological knowledge.

Example

The example is based on an RDF repository containing descriptions of the members of our research group and the people we collaborate with. The annotations are mainly based on the vocabulary defined by the

[4]Note that the acronym XSL stands for eXtensible Style Language, not XML Style Language as it is sometimes referred to.

[5]Note for the reviewers: we are still looking for an example of content structuring that is sufficiently simple to be included in the article.

```
<xsl:stylesheet version='1.0'
  xmlns:xsl='http://www.w3.org/1999/XSL/Transform'
  extension-element-prefixes='sesame'
  xmlns:sesame='xalan://SesameXML'
  xmlns:xalan='http://xml.apache.org/xalan'
>
```

Figure 6: SeRQL queries in XSLT. Declaring the Sesame XSLT extension

FOAF ("friend of a friend") project. The associated RDF Schema has also been loaded into the repository, so the extended XSLT can exploit the subClass- and subProperty-based subsumption reasoning supported by Sesame. In addition to the FOAF vocabulary, the example uses some other publicly available vocabularies, including those of Wordnet and OntoWeb. The example performs two content selection steps: it first finds all resources of class person, and then all known friends of these persons are retrieved. Simple styling is performed on the results by requesting further type information from the RDF repository.

Figures 6 to 9 on page 10 all show a fragment of the XSLT transformation sheet used for the example. Together they form the complete style sheet. The code implementing the style sheet is available from the Internet[6].

Prerequisites: Declaring the extension The XSLT specification defines how to use XML Namespaces to declare extensions. This is shown in Figure 6, which shows the start of our XSLT example. The first two lines can be found in almost all XSLT sheets, defining the XSLT version (`1.0`) and XSLT namespace. The next line declares all elements and functions starting with the `sesame:` prefix to be extension elements. The next line defines the `sesame:` namespace prefix, the value of the corresponding URI is used by the XSLT engine (in this case, Xalan) to locate the Java class implementing the extension. Finally, the last line defines the `xalan:` namespace prefix.

Content Selection: Using the Sesame XSLT extension to find all `wn:Persons` Given the extension declared above, we can now write an XSLT template rule that queries Sesame in SeRQL. The main template of our example is shown in Figure 7 on the next page. Note that it matches the root node of the source dummy XML document. The first line employs the extension function to select the target RDF repository in Sesame, in this case the one containing the metadata of our research group: `ins2research`. All subsequent queries will be implicitly executed in the context of this repository, until another one is selected explicitly.

The following line contains the first SeRQL query, asking Sesame to retrieve all instances of `rdf:type wn:Person` and the value of their associated `foaf:name` property. Note that this query mixes three different RDF vocabularies: RDF, WordNet and FOAF. Also note the extensive escaping that is necessary to make sure the query is well-formed XML (the usual escaping of the '<' and '>' characters) and correct XSLT (escaping of the curly brackets). For readability, this is the original SeRQL query without escaping:

```
select person, name from
{person} <rdf:type>  {<wn:Person>};
         <foaf:name> {name}
using namespace
  foaf = <!http://xmlns.com/foaf/0.1/>,
  wn   = <!http://xmlns.com/wordnet/1.6/>,
  rdf  = <!http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

The query benefits directly from Sesame's built-in subsumption inferencing. For example, it needs to infer what instances are of type `wn:Person`, since none of the persons in the repository have been explicitly declared to be of that type. Instead, the repository originally only used the Ontoweb ontology,

[6] `http://homepages.cwi.nl/~media/cuypers/java/`

```
<xsl:template match='/'>
  <!-- select ins2research repository in sesame: -->
  <xsl:variable name="oldrep" select='sesame:repository("ins2research")'/>

  <!-- get everything of type wn:Person: -->
  <xsl:variable name="persons">
   <sesame:serql query='
 select person, name from
   {{person}} &lt;rdf:type&gt; {{&lt;wn:Person&gt;}};
              &lt;foaf:name&gt; {{name}}
 using namespace
   foaf = &lt;!http://xmlns.com/foaf/0.1/&gt;,
   wn   = &lt;!http://xmlns.com/wordnet/1.6/&gt;,
   rdf  = &lt;!http://www.w3.org/1999/02/22-rdf-syntax-ns#&gt;
      '/>
  </xsl:variable>
  <html>
    <title>XSLT+SeRQL FOAF Demo</title>
    <h1>XSLT+SeRQL FOAF Demo</h1>
    <ul>
      <xsl:for-each select="xalan:nodeset($persons)/queryResult/*">
          <xsl:call-template name="friends">
            <xsl:with-param name="person" select="uri[1]"/>
            <xsl:with-param name="name"   select="literal[1]"/>
          </xsl:call-template>
      </xsl:for-each>
     </ul>
  </html>
</xsl:template>
```

Figure 7: SeRQL queries in XSLT. Main template selecting the Sesame repository and querying for a list of persons.

which also happens to contain a `ow:Person` class. It is this class that was used to annotate the persons in our repository. To be interoperable with FOAF, our metadata used RDF Schema to declare the Ontoweb person class to be a subclass of the FOAF version, and the FOAF ontology did the same by defining its person class to the WordNet version. Since all this knowledge has been uploaded to the repository, Sesame can use the inference rules defined by the RDF Model theory [15] to infer that all instances of the Ontoweb person class are also instances of the FOAF and Wordnet person classes.

The remaining code of Figure 7 builds the main skeleton of our output HTML page, including the title, a level one header and a bulleted list. This list then needs to be filled with one bullet for each person. This is done by looping over all tuples returned by the SeRQL query, and calling a named template that takes care of the formatting of each bullet. Note that the `friends` template is called with two parameters, one containing the URI of the person, the other containing a literal with the person's name.

Content Selection: Using the Sesame XSLT extension to find the friends The `friends` template is shown in Figure 8 on the following page. It shows how to use the results of the first SeRQL query by a second. After declaring its two parameters it follows a similar pattern as the first template. One noteworthy exception is the use of the value of an XSLT `$person` parameter as a resource within the SeRQL query, which shows the two languages can really be mixed and matched. So for the person passed as a parameter, the query finds all friends and their names, by querying for the `foaf:knows` property. The results are again stored as an XML result tree fragment and stored in the XSLT variable `friends`. The list of friends is then generated by looping over the query results and outputting each friend followed by a comma. The actual formatting of the person's name and the names of the friends is delegated to a third template, named `format-person`.

```
   <xsl:template name="friends">
    <xsl:param name="person" select="unknown-person"/>
    <xsl:param name="name" select="unknown-name"/>
    <xsl:variable name="friends"><!-- get all friends  -->
    <sesame:serql query='
  select friend, friendname from
  {{&lt;!{$person}&gt;}} &lt;foaf:knows&gt; {{friend}}
                        &lt;foaf:name&gt; {{friendname}}
  using namespace foaf = &lt;!http://xmlns.com/foaf/0.1/&gt;
     '/>
    </xsl:variable>
    <li>
      <xsl:call-template name="format-person">
        <xsl:with-param name="person" select="$person"/>
        <xsl:with-param name="name" select="$name"/>
      </xsl:call-template>
      knows:
      <xsl:for-each select="xalan:nodeset($friends)/queryResult/*">
        <xsl:call-template name="format-person">
          <xsl:with-param name="person" select="uri[1]"/>
          <xsl:with-param name="name" select="literal[1]"/>
        </xsl:call-template>,
      </xsl:for-each>
    </li>
  </xsl:template>
</xsl:stylesheet>
```

Figure 8: SeRQL queries in XSLT. Second template finding friends.

Content Styling: Using the Sesame XSLT extension to find class information The `format-person` template is shown in Figure 9 on the next page. While the other two templates retrieved their *content* from the RDF repository, this template shows one can also make the *style* dependent on (ontological) information in the RDF repository. To keep the example simple, the template only checks whether the person is of type PhDStudent, a concept defined by the Ontoweb ontology. If so, the person's name is typeset in italics, if not, in normal font.

Discussion

In the example information stored in an RDF repository is accessed during the content selection and content styling stages of a document transformation process. This is achieved by introducing a single function allowing "native" access to the RDF layer(s), thus avoiding querying the RDF semantics via the XML syntax layer. This illustrates that integration between the XML and RDF worlds can be carried out with the introduction of a minimum of extra infrastructure.

There are of course disadvantages to this approach. XSLT syntax is already complex, and the interface provided, with its requirement for two levels of escaping, is even more unreadable.

The approach has in its favor, however, that an existing language is used, rather than inventing a complete new one. In addition, because of the native access to the RDF semantics, any improvements at this level are available "for free".

6 Conclusion

We explained the implicit, but significant, knowledge-intensive nature of many XML document transformations. The quality and flexibility of the content selection, structuring and styling decisions can be significantly improved when transformations have access to the knowledge resources available on the Semantic Web. However, the current XML tool set, including XSLT, XPath, XQuery and CSS, provides no direct integration with Semantic Web technology.

```
  <xsl:template name="format-person">
    <xsl:param name="person" select="unknown-person"/>
    <xsl:param name="name" select="unknown-name"/>
     <xsl:variable name="isPhdStudent"><!-- check if PhD -->
      <sesame:serql query='
select p1 from {{p1}} &lt;rdf:type&gt; {{&lt;ow:PhDStudent&gt;}}
where p1 = &lt;!{$person}&gt;
using namespace
   ow   = &lt;!http://www.ontoweb.org/ontology/1#&gt;,
   rdf  = &lt;!http://www.w3.org/1999/02/22-rdf-syntax-ns#&gt;
                  '/>
    </xsl:variable>
    <span>
      <xsl:attribute name="style">
        <xsl:choose>
          <xsl:when test="string($isPhdStudent)">font-style: italic;</xsl:when>
          <xsl:otherwise>font-style: normal;</xsl:otherwise>
        </xsl:choose>
      </xsl:attribute>
      <xsl:value-of select="$name"/>
    </span>
  </xsl:template>
```

Figure 9: SeRQL queries in XSLT. Third template: ontology based formatting.

We analyzed the role of semantics in the content selection, structuring and styling decisions made in the document transformation processes within the Topia and Cuypers architectures. Based on this analysis, we explained the need for a more integrated approach that combines the extensive syntax-level support of the standard XML tools with the semantic expressivity and inferencing capabilities of the Semantic Web languages and their implementations.

We explored the use of a simple XSLT-extension that gives access to a number of RDF repositories managed by Sesame. Advantages of this approach include the fact that it uses XSLT as its host language and does not invent a new language (unlike RDF Templates). It relieves style and transformation developers from requiring to address RDF on the level of its XML serialization syntax using XPath. It avoids the creation of (yet) another new RDF implementation, leveraging on the work already done by the Sesame community. By doing so, it gets three query languages, RDF Schema-level inferencing and persistent storage for free. In addition, we expect to benefit from the current Sesame work on OWL support.

Disadvantages, however, include the fact that XSLT syntax is inherently complex. Mixing in Semantic Web query languages makes the problem even worse (as shown by the XML/XSLT escaping of the SeRQL syntax in the examples). It should also be noted that the solution requires an XSLT extension that is currently only available for the Xalan XSLT engine.

While our Semantic Web-enabled XSLT experiment shows the potential of the approach, it is still a far cry from a commonly agreed-upon and standardized solution to mix the Document Web and the Semantic Web. There is, for example, still no W3C recommended Semantic Web query language. The SeRQL language used and the two other languages supported are all languages proposed by individual W3C members. And even when such a language becomes standardized, the examples show that a clean integration in the document transformation chain is far from trivial.

Still, we think that the role semantics play in the effective communication of Web content is too important to continue to accept the current “syntax only” approach of XML document transformations. The Semantic Web should rather become an extension of the current “Document” Web, allowing information to pass across the “semantic” gap. This would allow document processing tools to access relevant semantics during their processing tasks and allow inferencing tools to display their results using standard document processing technologies.

Acknowledgments

The research described here was funded by the Dutch national NWO NASH project, Telematica Instituut's Topia project and the NWO ToKeN2000/I^{2}RP project.

References

[1] E. André, W. Finkler, W. Graf, T. Rist, A. Schauder, and W.Wahlster. WIP: The Automatic Synthesis of Multimodal Presentations. In *Intelligent Multimedia Interfaces* [12], pages 75–93.

[2] E. André and T. Rist. The Design of Illustrated Documents as a Planning Task. In *Intelligent Multimedia Interfaces* [12], pages 94–116.

[3] Elisabeth André, Jochen Müller, and Thomas Rist. *WIP/PPP: Knowledge-Based Methods for Fully Automated Multimedia Authoring*. London, UK, 1996.

[4] Susanne Boll and Wolfgang Klas. ZYX — A Semantic Model for Multimedia Documents and Presentations. In *Proceedings of the 8th IFIP Conference on Data Semantics (DS-8) — Semantic Issues in Multimedia Systems*, Rotorua, New Zealand, January 1999. Available from World Wide Web: `http://mmit.informatik.uni-oldenburg.de/pubs/bollK_DS8_1999.pdf`.

[5] Susanne Boll, Wolfgang Klas, and Jochen Wandel. A Cross-Media Adaptation Strategy for Multimedia Presentations. In *ACM Multimedia '99 Proceedings*, pages 37–46, Orlando, Florida, October 30 - November 5, 1999. ACM, Addison Wesley Longman. Available from World Wide Web: `http://www.acm.org/pubs/articles/proceedings/multimedia/319463/p37-boll/p37-boll.pdf`.

[6] M. Bordegoni, G. Faconti, M.T. Maybury, T. Rist, S. Ruggieri, P. Trahanias, and M. Wilson. A Standard Reference Model for Intelligent Multimedia Presentation Systems. *Computer Standards & Interfaces*, 18(6-7):477–496, December 1997. Available from World Wide Web: `http://www.dfki.uni-sb.de/imedia/lidos/papers/csi97/`.

[7] Jeen Broekstra, Arjohn Kampman, and Frank van Harmelen. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In Ian Horrocks and Jim Hendler, editors, *The Semantic Web - ISWC 2002*, number 2342 in Lecture Notes in Computer Science, pages 54–68, Berlin Heidelberg, 2002. Springer. Available from World Wide Web: `http://link.springer.de/link/service/series/0558/papers/2342/23420054.pdf`.

[8] Leslie Carr, Sean Bechhofer, Carole Goble, and Wendy Hall. Conceptual Linking: Ontology-based Open Hypermedia. In *The Tenth International World Wide Web Conference* [16], pages 334–342. Available from World Wide Web: `http://www10.org/cdrom/papers/246/`.

[9] Alison Cawsey. Presenting tailored resource descriptions: Will XSLT do the job? In WWW2000 [30]. Available from World Wide Web: `http://www.www9.org/w9cdrom/119/119.html`.

[10] Ian Davis. RDF Template Language 1.0. Early draft, September 2003. Available from World Wide Web: `http://www.semanticplanet.com/2003/08/rdft/spec`.

[11] Dublin Core Community. Dublin Core Element Set, Version 1.1, 2003. Available from World Wide Web: `http://www.dublincore.org/documents/dces/`. ISO Standard 15836-2003 (February 2003), http://www.niso.org/international/SC4/n515.pdf; NISO Standard Z39.85-2001 (September 2001), http://www.niso.org/standards/resources/Z39-85.pdf; CEN Workshop Agreement CWA 13874 (March 2000), http://www.cenorm.be/isss/cwa_download_area/cwa13874.pdf.

[12] Mark T. Maybury (Editor). *Intelligent Multimedia Interfaces*. AAAI Press, 1993.

[13] Joost Geurts, Stefano Bocconi, Jacco van Ossenbruggen, and Lynda Hardman. Towards Ontology-driven Discourse: From Semantic Graphs to Multimedia Presentations. In *Second International Semantic Web Conference (ISWC2003)*, Sanibel Island, Florida, USA, October 20-23, 2003. To be published.

[14] W.E. Grosso, H. Eriksson, R.W. Fergerson, J.H. Gennari, S.W. Tu, and M.A. Musen. Knowledge Modeling at the Millennium (The Design and Evolution of Protege-2000). Technical Report SMI Report Number: SMI-1999-0801, Stanford Medical Informatics (SMI), 1999.

[15] Patrick Hayes. RDF Semantics. Work in progress. W3C Working Drafts are available at http://www.w3.org/TR, 23 Janaury 2003. Available from World Wide Web: `http://www.w3.org/TR/rdf-mt/`.

[16] IW3C2. *The Tenth International World Wide Web Conference*, Hong Kong, May 1-5, 2001. ACM Press. Available from World Wide Web: `http://www10.org/`.

[17] Yuhui Jin, Stefan Decker, and Gio Wiederhold. OntoWebber: Model-Driven Ontology-Based Web Site Management. In *Semantic Web Working Symposium (SWWS)*, Stanford University, California, USA, July 30 – August 1, 2001. Available from World Wide Web: `http://www.semanticweb.org/SWWS/program/full/paper55.pdf`.

[18] Gregory Karvounarakis, Vassilis Christophides, Dimitris Plexousakis, and Sofia Alexaki. Querying Community Web Portals. http://www.ics.forth.gr/proj/isst/RDF/RQL/rql.html. Available from World Wide Web: `http://www.ics.forth.gr/proj/isst/RDF/RQL/rql.html`.

[19] S. Kim, H. Alani, W. Hall, P.H. Lewis, D.E. Millard, N.R. Shadbolt, and M.J. Weal. Artequakt: Generating Tailored Biographies with Automatically Annotated Fragments from the Web. Presented at the Semantic Authoring, Annotation and Knowledge Markup (SAAKM) 2002 Workshop at the 15th European Conference on Artificial Intelligence (ECAI 2002), Lyon, France.

[20] Suzanne Little, Joost Geurts, and Jane Hunter. Dynamic Generation of Intelligent Multimedia Presentations through Semantic Inferencing. In *6th European Conference on Research and Advanced Technology for Digital Libraries*, pages 158–189. Springer, September 2002. Available from World Wide Web: `http://www.cwi.nl/~media/publications/ecdl2002.pdf`.

[21] Peter Patel-Schneider and Jérôme Siméon. The Yin/Yang Web: XML Syntax and RDF Semantics. In *The Eleventh International World Wide Web Conference*, Honolulu, Hawaii, May 7-11, 2002. IW3C2, ACM Press. Available from World Wide Web: `http://www2002.org/CDROM/refereed/231/`.

[22] Lloyd Rutledge, Martin Alberink, Rogier Brussee, Stanislav Pokraev, William van Dieten, and Mettina Veenstra. Finding the Story — Broader Applicability of Semantics and Discourse for Hypermedia Generation. In *Proceedings of the 14th ACM conference on Hypertext and Hypermedia*, pages 67–76, Nottingham, UK, August 23-27, 2003. ACM. Available from World Wide Web: `http://www.cwi.nl/~media/publications/ht03.pdf`.

[23] Steffen Staab, Jürgen Angele, Stefan Decker, Michael Erdmann, Andreas Hotho, Alexander Maedche, Hans-Peter Schnurr, and Rudi Studer. Semantic Community Web Portals. In WWW2000 [30]. Available from World Wide Web: `http://www9.org/w9cdrom/134/134.html`.

[24] Jacco van Ossenbruggen, Joost Geurts, Frank Cornelissen, Lloyd Rutledge, and Lynda Hardman. Towards Second and Third Generation Web-Based Multimedia. In *The Tenth International World Wide Web Conference* [16], pages 479–488. Available from World Wide Web: `http://www10.org/cdrom/papers/423/index.html`.

[25] Jacco van Ossenbruggen, Joost Geurts, Lynda Hardman, and Lloyd Rutledge. Towards a Formatting Vocabulary for Time-based Hypermedia. In *The Twelfth International World Wide Web Conference*, pages 384–393, Budapest, Hungary, May 20-24, 2003. IW3C2, ACM Press. Available from World

Wide Web: http://www2003.org/cdrom/papers/refereed/p383/p383-ossenbruggen.html.

[26] Jacco van Ossenbruggen and Lynda Hardman. Smart Style on the Semantic Web. In *Semantic Web Workshop, WWW2002*, May 2002. Available from World Wide Web: http://www.cwi.nl/~media/publications/www2002-semwebworkshop.pdf.

[27] Jacco van Ossenbruggen and Lynda Hardman. Multimedia on the Semantic Web. In Herre van Oostendorp, Andrew Dillon, and Leen Breure, editors, *Creation, Use and Deployment of Digital Information*. Erlbaum, Publication planned late 2003.

[28] W3C. Extensible Stylesheet Language (XSL) Version 1.0. W3C Recommendation, 15 October 2001, 2001. Available from World Wide Web: http://www.w3.org/TR/xsl/.

[29] L. Weitzman and Kent Wittenburg. Automatic presentation of multimedia documents using relational grammars. In *Proceedings of the second ACM international conference on Multimedia '94*, pages 443–451, San Francisco, October 15 - 20, 1994. Available from World Wide Web: http://www.acm.org/pubs/articles/proceedings/multimedia/192593/p443-weitzman/p443-weitzman.pdf.

[30] *The Ninth International World Wide Web Conference*, Amsterdam, The Netherlands, May 15-19, 2000. Available from World Wide Web: http://www9.org/.

[31] Michelle X. Zhou and Steven K. Feiner. Top-Down Hierarchical Planning of Coherent Visual Discourse. In *Proceedings of the International Conference on Intelligent User Interfaces*, pages 129–136, Orlando, FL. USA, January 6-9, 1997. Available from World Wide Web: http://www1.cs.columbia.edu/~zhou/project/IUI97.ps.gz.