Structuring and presenting annotated media repositories

Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

INS

Information Systems

INformation Systems

Structuring and Presenting Annotated Media Repositories

Lloyd Rutledge, Jacco van Ossenbruggen,
Lynda Hardman

REPORT INS-E0402 FEBRUARY 12, 2004

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

Structuring and Presenting Annotated Media Repositories

ABSTRACT
The Semantic Web envisions a Web that is both human readable and machine processible. In practice, however, there is still a large conceptual gap between annotated content repositories on the one hand, and coherent, human readable Web pages on the other. To bridge this conceptual gap, one needs to select the appropriate content from the repository, structure and order the material, and design a Web page that effectively conveys the selected content and the chosen structure. In addition to this conceptual gap, there is also a technological gap. On one side of this gap, we find the semantic-oriented technology deployed to build annotated content repositories. This includes RDF, RDF Schema and OWL. On the other side of the gap is the syntax-oriented technology deployed to build Websites. This includes XML, XSLT, CSS, XHTML and SMIL. In this paper, we discuss the conceptual relationships between the world of explicit metadata semantics and the world of Web presentations and their underlying syntactic formats. We also explore to what extent this gap can be bridged automatically, and how current Web technologies can be used to support this process.

1998 ACM Computing Classification System: H.5.4, H.5.1 [Information Interfaces and Presentation (e.g., HCI)]: Hypertext/Hypermedia – architectures, navigation; Multimedia Information Systems – Hypertext navigation and maps, Evaluation/methodology; I.7.2 [Document and Text Processing]: Document Preparation – Hypertext/hypermedia, Markup languages, Multi/mixed media, standards.
Keywords and Phrases: Algorithms, Documentation, Design, Experimentation, Human Factors, Standardization, Languages, Theory.

Structuring and Presenting Annotated Media Repositories

Lloyd Rutledge, Jacco van Ossenbruggen and Lynda Hardman

STATEMENT

We generate hypermedia presentations from annotated media repositories using document structure as an intermediate phase. We discuss the application of Web technologies at each stage.

TABLE OF CONTENTS

Structuring and Presenting Annotated Media Repositories

Lloyd Rutledge, Jacco van Ossenbruggen and Lynda Hardman
CWI (Centrum voor Wiskunde en Informatica)
P.O. Box 94079
NL-1090 GB Amsterdam, The Netherlands
Tel: +31 20 592 40 93, +31 20 592 41 41
E-mail: FirstName.{van.}LastName@cwi.nl

ABSTRACT

The Semantic Web envisions a Web that is both human readable and machine processible. In practice, however, there is still a large conceptual gap between annotated content repositories on the one hand, and coherent, human readable Web pages on the other. To bridge this conceptual gap, one needs to select the appropriate content from the repository, structure and order the material, and design a Web page that effectively conveys the selected content and the chosen structure.

In addition to this conceptual gap, there is also a technological gap. On one side of this gap, we find the semantic-oriented technology deployed to build annotated content repositories. This includes RDF, RDF Schema and OWL. On the other side of the gap is the syntax-oriented technology deployed to build Websites. This includes XML, XSLT, CSS, XHTML and SMIL.

In this paper, we discuss the conceptual relationships between the world of explicit metadata semantics and the world of Web presentations and their underlying syntactic formats. We also explore to what extent this gap can be bridged automatically, and how current Web technologies can be used to support this process.

Categories and Subject Descriptors

H.5.4, H.5.1 [**Information Interfaces and Presentation (e.g., HCI)**]: Hypertext/Hypermedia – *architectures, navigation;* Multimedia Information Systems – *Hypertext navigation and maps*, *Evaluation/methodology;* I.7.2 [**Document and Text Processing**]: Document Preparation – *Hypertext/hypermedia*, *Markup languages*, *Multi/mixed media*, *standards*.

General Terms

Algorithms, Documentation, Design, Experimentation, Human Factors, Standardization, Languages, Theory.

Keywords

Style, Document Structure, Discourse, Semantics, XSLT, RDF, CSS, XHTML, SMIL, content and coding (meta data), hypermedia (link architecture, navigation), languages and standards (hypertext and hypermedia, metadata on the Web)

1. INTRODUCTION

In the early days of the Web, systems presented large amounts of textual information stored in (legacy) databases to end-users by dynamically filling HTML templates with database content. This scenario is already beginning to repeat itself with the desire to generate human-readable Websites from knowledge repositories. In the first case, the mapping is purely syntactic and based on SQL queries, taking advantage of a predefined database structure. The situation we are now facing, however, requires a semantic equivalent of the syntactic mapping, taking into account the much richer knowledge structure of Semantic Web content repositories.

The conceptual step from repository to coherent presentation is traditionally a human task. For example, imagine a creator of a video (or interactive multimedia) documentary going to a physical archive and using the catalog system to gather a collection of media items such as photographs, video clips, sound recordings and paper text documents. Once collected, the creator spreads the items out on a table and rearranges them, looking for patterns, relations and groupings that pull the items together into a structured storyboard. The creator then edits these items together to form a coherent documentary.

While this process is less obvious when using HTML templates and SQL queries as a front-end to a database, similar design decisions have already been taken by the Website designer when constructing the templates and corresponding queries. This approach can also apply to generating pages from Semantic Web repositories by replacing SQL with, for example, an RDF query language [4][17]. This not only makes RDF data available through human-readable HTML pages, but also allows semantic inferencing in order to provide better answers to the request by, for example, using the RDF Schema subsumption hierarchy. The disadvantage, however, is that by predefining the templates and their corresponding queries it is difficult to take advantage of the rich and heterogeneous structure of the knowledge base.

In this paper, we explore to what extent one can generate coherent Web presentations from knowledge repositories. We have modeled this processing of semantics into presentation with the steps shown in Figure 1. General document structure comes from applying a clustering algorithm to the RDF encoding of the underlying semantic network [26]. XSLT then transforms this structure into a presentation in XML-defined formats such as XHTML and SMIL. This transform gets additional information from the RDF store with queries encoded in SeRQL. RDF [6] and other Semantic Web formats have a strong technical distinction from, yet also a strong conceptual relationship with, document structure as defined by XML [23]. Since our system derives XML-defined structure from RDF-defined concept maps, this technical distinction needs bridging by exploiting this strong conceptual relation.

We identify the following key components in our document structure: *nodes*, *hierarchy*, *sequence* and *recurrence*, all within a model we call *structured progression* [26]. A node is a unit of information, all of which is typically displayed simultaneously and viewable in one display. Each node in the structured progression is associated with a concept (or, more precisely, an instance of an RDF class) in the repository. Section 2 explains the selection of nodes for inclusion in the structured progression.

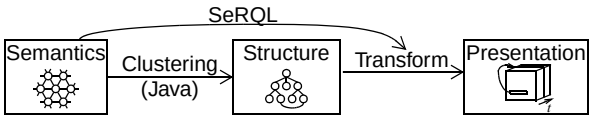

Figure 1. Semantics to Presentation Path

The first step towards a document structure is grouping the nodes into a *hierarchy* of composite and leaf nodes. Sibling nodes in the hierarchy also receive a meaningful *order* wherever this is appropriate for conveying relations between them. Finally, a *recurrence* is a node that appears repeatedly at different locations in the hierarchy, such as a recurring theme. Recurrence constructs place structure around an otherwise unstructured group of concepts to communicate a (potentially domain specific) relation among those concepts. Section 3 discusses how to determine appropriate hierarchical, sequential and recurrence relations.

Based on the structured progression described above, the final, but crucial, step is designing a Web-based presentation that conveys not only the content of the selected nodes and their underlying domain relations, but also their context in the structured progression in terms of hierarchical position, order and recurrences. For text, we have well defined and commonly used layout techniques to convey the structured progression to the user. For example, presentations often convey the hierarchical structure of sections and subsections by section numbering (e.g. section 4.1.2) and by using decreasing font sizes for the section titles. Indentation within nested bulleted list is one means of communicating hierarchical structure.

The sections following Section 3 describe the presentation devices that convey each component of a structured progression and how XHTML+SMIL and CSS encodes them. In this paper, we explicitly make use of these devices to communicate the structured progression to the end user. In addition, we introduce some new presentation devices that target multimedia content explicitly. Presentation devices explore the use of style in a *functional* way, going beyond the use of style for purely aesthetic reasons. Section 4 describes how to present each individual node. Section 5 discusses communicating the hierarchy around the nodes. Section 6 explains how to convey the sequence of the nodes' presentation. Finally, Section 7 describes how to convey the recurrence of individual nodes throughout the presentation.

2. SELECTING CONTENT

Our system starts by processing a user request for a presentation on a particular topic against RDF-encoded annotations over media representing the collection of the Rijksmuseum Amsterdam [26]. This returns matching resources from the RDF store that will become leaf nodes in the structured progression. The clustering process then finds additional resources that will become composite nodes. We call both these types of resources *concept resources* because they represent the conceptual content of the presentation. The transform process will access these resources with SeRQL queries to find media for presenting the concepts they represent. For these SeRQL queries, each concept resource in the RDF has closely-related *media resources* for conveying it. The media items they locate become the media content of the presentation.

Concept Selection. The user request becomes a query on the RDF returning the leaf node content resources. Our system accepts a text string and finds text resources that contain it, but, of course, one could use more intricate media search techniques. The concept resources linked to these become the leaf nodes. For the most part, the clustering treats the RDF encoding like a node-edge graph to derive property tables for applying concept lattice clustering to [26]. However, inferencing encoded by Semantic Web technology can provide additional properties for these tables. Our system, for example, arranges genres of artwork in a subclass hierarchy, allowing subsumption to infer the whole ancestry of genres for each concept node referring directly to any genre. This gives the concept lattice a larger property table to handle, enriching the possibilities for the resulting structured progression.

Inferencing. XSLT transformations can benefit from domain-specific knowledge that does not appear in the XSLT transformation sheet or XML source file. When inference rules derive new information from the repository, the RDF queries embedded in the XSLT sheets automatically benefit from this new information. In addition to providing generation information, an RDF repository can also have inference rules for concluding what media resources fulfill certain template roles, or classes of template roles. Related research applies such rules on Dublin Core metadata [16]. In our system, queries from XSLT could refer to such inferred knowledge to determine, for example, the "main" image for a node display about an artist.

Media Selection. The XSLT encoding our transform process has several types of RDF encoding to exploit in its SeRQL queries for media. The simplest is finding directly related resources that are media objects of a given type. More specific is querying for Dublin Core metadata [11] on the resource. Finally, RDF repositories for specific domains can define their own inference rules for finding media for given roles, thus removing the burden of constructing longer queries from the XSLT.

Media Types. To enable finding images for displaying concept resources, we place RDF resources for images in triples directly associated with concept resources. XSLT-called SeRQL queries can then ask for resources that a share a triple with the concept resource and have URL's with filename suffixes for image formats. This use of format-specific suffixes apply to finding media resources of other types as well, for which the XSLT could query.

Dublin Core. The Dublin Core provides an RDF-encoded set of metadata tags that are relatively universal across domains [11]. These include tags such as `<dc:title>`, `<dc:description>`, `<dc:creator>` and `<dc:date>`. Our system sends queries for Dublin Core tags for retrieving much of its media. Since Dublin Core is a widely accepted standard for metadata, we expect its use within a wide variety of RDF repositories.

Dublin Core Tags. Dublin Core tags apply well in retrieving media for certain parts of the presentation display. The `<dc:title>` tag provides concept titles, which, as we will see throughout the rest of this paper, appear frequently in presentations. The `<dc:description>` provides a textual description to present when a concept is the presentation's current main topic. The `<dc:creator>` and `<dc:date>` tags provide captions to show when presenting museum artifact. Using Dublin Core tags and other cross-domain standardized metadata where applicable facilitates the processing of a semantic network by systems not familiar with the given domain.

3. STRUCTURING

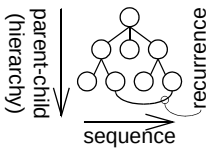

Figure 2. Structured Progression

Since the stage before presentation style processing in our architecture is structured progression, the structure encoding must provide this information. However, because structure processing cannot know all the information that style processing will need from the phases before it, structure encoding must provide access to appropriate data in these phases so that the final style phase can makes its own conclusions. Thus, the XML for structured progression should contain only the information needed by the XSLT code about the general structure, as determined by clustering. Since the XSLT should be able to make its own conclusions from the RDF about other aspects of the presentation, it should provide only the information needed for the XSLT to access the RDF and make these conclusions. Figure 2 illustrates the structured progression model.

After the clustering of the RDF graph into a structured progression, XSLT [8] transforms its XML encoding into the final presentation. Earlier work presents our architecture and implementation of XSLT processing of XML-encoding structured progression

derived from RDF-encoded media annotations into an output XML-encoded presentation [26]. More recently, we describe the technical underpinnings of new potential for accessed RDF-encoded information from XSLT style sheets, making RDF-encode information accessible during presentation generation along with XML-encoded information [22]. We extend our previous architecture and implementation here by applying these Web technologies to extend our XSLT processing to query semantic networks.

3.1 Nodes

The XSLT processes the XML to get information on the structured progression as a whole and where each node sits within it. The XSLT also queries the RDF directly to get media information about each individual resource. Thus, the XML should only encode the hierarchy. Furthermore, each node in the XML should have only a single reference to that node's representation in the original RDF, enabling the XSLT to choose the media conveying the node's topic.

The XSLT code chooses media to fulfill each template role to help ensure good quality and combination of all media selected for each node. For example, our current primary template has a space for displaying for the concept a *title* for the node, one main *image*, a *descriptive text* and several optional small text *captions*. The captions vary based on the type of the node. For museum artifacts, we include captions for the creator's name and the date of creation. For artists, we have captions showing his or her dates of birth and death. The XSLT code must recognize the role and what type of node it falls into call a SeRQL query for it. The next subsection discusses the natures of these queries.

3.2 Hierarchy

XML directly offers hierarchical structure, with elements containing other elements as well as direct content. In our architecture, hierarchy of the XML-encoding of the structured progression directly represents the hierarchy of the document. XML is the primary source for XSLT to derive structured progression, with no need for the XSLT to query the RDF for additional information. As with generating the main displays, the XSLT acquires from the XML a reference to the RDF for each node, which it uses to acquire media items for presentation. It requests Dublin Core title tags to put in the outline display and in overviews. The XSLT also uses the same queries for images described in Section 2 to get images for the summary thumbnail displays.

While semantic markup has enhanced search for single media items, little work has been done on how semantics help build presentational structure around multiple returned items. This subsection discusses three aspects of RDF processing for hierarchy in presentation: the use of Dublin Core titles, the processing of RDF for clustering from which original hierarchy derives and the further querying of the RDF for additional hierarchical information beyond what the original XML provides.

Clustering. In earlier work, we describe how to group the retrieved RDF instances into clusters that serve as precursors to the subsequent derivation of a hierarchy and recurrences [26]. By applying the concept lattice algorithm for clustering, common properties of the instances returned by the retrieval process determine these directed acyclic graphs of clusters and sub-clusters. For example, the screen dump of Figure 3 shows a top-level cluster of paintings that share an RDF property `material` with the value "Oil on canvas". Within this cluster, the paintings situated in The Hague form a sub-cluster. We are currently exploring additional clustering techniques to apply here [2]. While the hierarchies inherent in the RDF encoding do not always provide the best hierarchies for final presentation, they do sometimes provide this clustering technique with properties that apply to the resulting hierarchy.

Semantic Processing for Clustering. Inferencing on the `rdfs:subClass` and `rdfs:subProperty` hierarchies in the repository provides additional properties for clustering, resulting in more possibilities for deriving document structure. For example, if one painting has a `createdby` property while the other has a `paintedby` property and the RDF Schema defines `paintedby` as an `rdfs:subProperty` of `createdby`, then the system infers the subsumption relation and returns a `createdby` property for both paintings. We implemented the clustering algorithm as a Java servlet accessing the RDF repository using the Sesame SAIL API [7]. This API supports a number of RDF Schema-aware query languages, including RQL (RDF Query Language) [13] and SeRQL (Serialized RDF Query Language) [1]. Sesame's built-in subsumption inferencing on RDFS hierarchies provides the servlet with the additional properties for the concept lattice processing [7].

Tangential Stories. Research on the DISC system explores guiding the automatic creation of coherent presentations based on discourse structures, including hierarchy [12]. This work typically builds its presentations *top-down*, starting with domain-specific discourse-based general structure and then determining the lower-level details. Our presentation construction, on the other hand, works *bottom-up*, starting with selected content and then generating higher-level, broader presentation structure such as hierarchy around it. The two systems' hierarchies differ in nature because the DISC system uses human-crafted structural templates, which can thus have richer inherent discourse. Our computer-generated hierarchies, on the other hand, are simpler in discourse, but their simplicity and derivation from general relation structure within semantic networks apply more readily to a wider variety of domains

3.3 Sequence

As we have mentioned, RDF does not define sequence over its resources. RDF explicitly assumes no significance to the order in which it encode items and, in the absence of the controversial exception of the `<rdfs:seq>` construct, has no concept of significant order. However, it must provide the information necessary for other processes to determine sequence over the resources RDF returns. Our demo uses the date of creation as the RDF-encoded basis for sorting.

Like hierarchy, sequence is a primary component of XML syntax. XML code as a text stream is sequential. However, while this XML unavoidably applies a syntactic sequence to the tuples, there is neither significance nor determinism to this sequence. Our XSLT must generate presentations with devices for conveying sequence and, of course, recognize the sequence to convey. In addition to the HTML and SMIL constructs in the previous section, our XSLT also generates text for describing the progression of a sequence that transitions out of one node and then into the next. Since our XML inputs communicate no significant sequence, our XSLT must derive the document's sequence from the RDF, either by knowing the order in which to request RDF resources, or by retrieving multiple resources at once and sorting them afterward.

Need for XSLT Sequence Construction. XSLT can detect and process sequence from its input XML. However, this feature is irrelevant for the two sources of XML that our system provides its XSLT code. The XML that clustering provides has no significance in its inherent ordering because our clustering does not process any sequence. Furthermore, RDF does not encode sequence, so the query returns feed to XSLT from the RDF has no significant sequence. Therefore, the XSLT program must construct any significant sequence that the processing of either of these source. Fortunately, it is very much in XSLT's nature to create its own sequence to input XML.

Pre-RDF Request Sequencing. Sequence in the structured progression comes from processing inherently numeric properties of the clustered artifacts, such as date of creation. In applying the

Sesame extension over Xalan to the demo, we rely on getting just one tuple back each time. This is a single fact request, which keeps things quite simple. The other option, getting back a bag of tuples, is handy as well. These results are unsorted: they appear in the order that Sesame happens to return them, with neither significance nor determinism to their order.

Post-RDF Request Sequencing. However, XSLT can sort the multiple tuples returned by a single request with a `<xsl:sort>` and a `<xsl:for-each>` around the return of the query over the RDF encoding. This is XSLT syntax-oriented processing on the inherently non-syntactic, non-structured and non-sequential information from RDF. There are limitations on what the sort key can be — that is, it must fit in the `select` attribute as an XPath on the returned item. Furthermore, such XPaths apply only to the simple XML structure returned, not the underlying RDF.

3.4 Recurrence

The clustering technique produces recurrences along with the hierarchy. If an artifact has separate characteristics in common with separate groups of other artifacts, then this artifact falls in separate groupings and thus in separate places in the hierarchy. The XSLT code detects recurrences as multiple nodes with identical RDF resource references.

While no XML construct directly represents recurrence, the inter-element referencing provided by ID and IDREF attributes readily enables recurrence. Our system thus detects recurrences in the XML with straightforward XSLT code for finding nodes with identical URI attribute values. What makes some commonalities recur and others unite into hierarchical groups is determined by the size and weight the clustering technique assigns to them, as we describe in earlier work [26].

4. PRESENTING NODES

Figure 3 shows an example of a presentation generated by our demo. The core of any presentation is its content, regardless of how it is organized. In our architecture, the selected semantic resources passed through as nodes in the structured progression are the presentation's conceptual content. Each resource node appears as the main current topic at some point in the presentation. Our system makes presentations on both the concepts the query returns and the *composite node* concepts that relate them to one another. The core of our presentation model for presenting all of these concepts is the *main display*. The principle technique in building these main displays is *templates*.

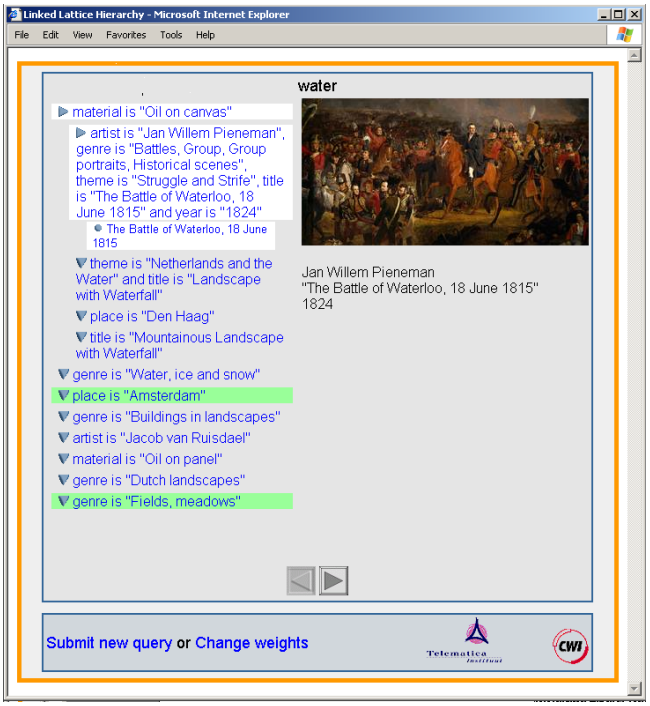

image © Rijksmuseum Amsterdam, used with permission

Figure 3. A Generated Presentation

Formats. The output of XSLT, the final presentation, is traditionally defined for the Web by CSS [5] and HTML [24]. More recently, SMIL [9] has added multimedia behavior to Web presentation. The combination of HTML and SMIL features into XHTML+SMIL [20] provides a single XML-encoded format representation a presentation in a single file, facilitating its XSLT generation. We need these formats to provide presentation behavior that effectively communicates our structured progression. The primary Web presentation format constructs for implementing main displays come from spatial positioning and SMIL timing. Two types of spatial positioning, absolute and relative within text flow, fulfill different requirements for main displays. Timing divides the node displays while positioning defines the consistent form that the templates give different displays.

Displaying Composite Nodes. The previous version of our system only presented leaf nodes of the structured progression hierarchy [26]. The composite nodes never received their own devoted presentation. The leaf nodes represent RDF resources matching the user's information request. The composite nodes represent the properties their descendant nodes have in common. However, while not matching the user's request, composite node concepts are often useful in understanding the directly matching resources, such as by providing insight into what causes these returns to have their matching property. This paper introduces to our architecture the presentation of composite nodes of the structured progression in the main display, making the topics they represent first-class concepts along with the information request returns.

Main Display. For presenting both leaf and composite nodes, our presentation model reserves a core component of the screen space for communicating the current document node, which we call the *main display*. We place requirements on the main display regarding the display *size* and the *quality*, *combination* and *consistent placement* of its media. First, for simplification, we place the constraint that all information for a node must be presentable *simultaneously* within the main display space, preferably without scrolling. Secondly, the system must select media that is *appropriate* for communicating the node's topic. Furthermore, the *combination* of media items must be appropriate, with enough variety of information sources, but not too much so that it overflows the available space or overwhelms the user. Finally, as we describe in earlier work [27], *consistent placement* of components over different main displays is important for facilitating the user's comparison of the different nodes.

One Node at a Time. Since our presentation model shows all media for a given structured progression node simultaneously, the display of nodes at different times becomes the main component of presentation structure that separates them. The SMIL temporal composition elements `<seq>` (sequence) and `<excl>` (exclusive) both enforce this one-at-a-time behavior on their children, though each in different ways. Thus, placing all sub-presentations for the nodes into one of these elements will keep the main display devoted to one node at a time. We discuss the issues resulting from the use of each element type throughout the rest of this paper.

Templates. To help meet the requirements laid out above, we use *templates* in our system to construct each node's main display. Templates assign areas of the main display to contain media items playing particular roles for communicating the node's concept.

That each role has a typical *size* keeps the display of particular combinations of roles in the main display size within the desired range. The selection of media items based on fulfilled roles helps assure a good *quantity* and combination of media items. Finally, the existence of specific roles enables the assignment of positions for them, providing *consistent placement*.

5. PRESENTING HIERARCHY

In addition to the nodes selected for presentation, there is the presentation structure built around them. Perhaps the most important and most used aspect of document structure is hierarchy, which is the nested containment of document content. Textbooks, for example, usually follow a hierarchical structure as part of communicating their content, with chapters containing sections, which in turn contain subsections, and so on. This section of this paper's hierarchy discusses hierarchy itself. The following main sections discuss presenting the remaining aspects of structured progression this paper covers: sequence and recurrence.

The main display introduced in Section 4 conveys some information about the document hierarchy. This main display accompanies an *outline display*, whose main purpose is to convey hierarchical structure. The presentation features conveying hierarchy typically either in the main displays or changing the outline display in coordination with the changing current node. One must encode when each main display occurs and what presentation effects bring it gracefully into, and out of, the presentation. This main display activity has corresponding outline display activity, which must show the whole hierarchy and the current node's place in it, all in a visually graceful manner.

5.1 Main Display

As the main display changes between nodes while traversing the hierarchy, its content should show more than just the current node. It should also communicate the traversals occurring. Landow established the importance of explaining the traversals between presentation components as a hypermedia presentation progresses [14]. To address this need, we add additional main displays to our system that occur between nodes to explain the reason for traversing from one to the other. Introductory overviews and summaries are such between-node main displays for hierarchical traversal. Other presentational aspects of the main display also help communicate hierarchy.

Introductory Overviews. The introductory overview is a well-established discourse technique in human-crafted presentations. It helps orient the user in the document's hierarchical structure by communicating that the progression through the tree is about to turn downward into more detail. Overviews also help clarify the conceptual relations between the items in the section. One simple technique we use is a text statement introducing each child, if any, of the current node. The very simplest of such canned text would be "For the topic of ___, we will see ___, ___, ... and ___". Here, the first underline holds the title of the current node, while the remaining underlines hold the titles for the child nodes, in order.

Summaries. Summarizing is a well-established discourse technique in human-crafted presentations for wrapping up sections. Like introductory overviews, summaries also help orient the user in the hierarchy. They inform the user that a transition in the hierarchy it to occur, but now it is out of a section rather than into it. While the just-presented subtopics are fresh in the user's mind, the user receives a reminder of what they are and, again, what their significant commonality is. In our system, a summary is a single screen display containing abbreviated version of multiple other screen displays. It occurs after this group of other displays to provide closure to the user. Specifically, in our system, these summaries appear at the end of hierarchical groups of artifacts as a table of thumbnail images of those artifacts.

Timing. A subtler temporal presentation device for conveying hierarchy is altering the tempo at which subsequent nodes appear. Longer pauses between components communicate a traversal between higher-level composites. Consistent rhythm, on the other hand, emphasizes that components are on the same level.

Fade Transitions. The timing, and even the nature, of fades can help convey hierarchy. For example, the distinction of a cross fade from a complete fade out could communicate the current depth in the hierarchy. A cross fade is a less dramatic transition, which could be used to communicate the transition between leaf node siblings. A complete fade out to a blank screen, on the other hand, communicates a more significant transition: one between sections in the hierarchy. The durations of fades can vary in the same manner described for pauses in the previous subsection.

5.2 Outline Interaction

Our system provides, in addition to the main display, an *outline display* that communicates the hierarchical organization of the document. Each line of this outline refers to one node in the document. By presenting many such lines at once, the outline display provides a larger-scale perspective of the hierarchy.

Dublin Core Titles. The outline display entries and the introductory overview references to child nodes each need a text statement that is small enough for a single line but is also a concise statement of the referenced node. To get such a text, we use the Dublin Core `<dc:title>` metadata construct. Just as it announces a node during its main display, a title can also refer to a node from other points in the presentation, such as from the outline display and overviews as described here, and from other points described in the rest of this paper.

Outline Folding. Since outlines can be very large, by default, only the portion relevant to the current node appears. That is, the outline entries for the node and its siblings, and its ancestors and their sibling, show, but the presentation hides the rest. We call this technical *outline folding*. Outline folding is a well-established technique in many different interfaces. Folding and unfolding the outline display allows focusing on the current node while efficiently communicating its hierarchical context. With this device, only the children of direct ancestors of the current node appear in the outline display.

Animation. SMIL also provides animation, which allows details of hypermedia presentation to change during the course of the presentation itself. We present here three types of targets for animation in conveying hierarchy: color, position and CSS classes. Color and position are direct effects, whereas the animation of classes helps manages animation effects.

Outline-based Navigation. Our outline display also provides *navigation* to any point in the document hierarchy. This navigation occurs when the user clicks on a node's outline entry, which results in the main display presenting that node. Section outline entries also fold and unfold to user clicks on special icons, allowing the user to find points in the hierarchy that otherwise would not show. This enables the users to depart from the default progression of the document and find their own paths. It thus enhances the user's understanding of the structured progression because he or she can choose the best navigation path for perceiving it. However, whatever path a user chooses, the original hierarchy should remain clearly communicated to help orient the user.

Timing. The variety of rendered navigation paths provided along the hierarchical structure (and, as discussed ahead, sequential and recurrence structures) means that the nodes of the presentation could appear in any of the many different orders that the user can select. This behavior of one-at-a-time presentation with many alternative paths corresponds most directly with SMIL's `<excl>` (exclusion) element, which we use in our implementation. Its

behavior resembles that of radio buttons, which lets the user make new selections at any time, with each new request stopping anything current playing. Our code has the displays for all nodes in one <excl> element, with user activation of any outline entry triggering the start of its corresponding node's main display, and thus the stopping of the current node. While the navigational effective behavior suggests the use of linking constructs, technically this use of <excl> is temporal synchronization with user interface events. The core of this synchronization is having a user click on an outline entry trigger the start of the corresponding node's main display.

5.3 Outline Visual Style

Our system also uses the visual style of a presentation's components to reflect the underlying structured progression. Here, visual style is the collection of presentation properties defined for the Web by CSS. These include text typeface and the color of all visual media.

Typeface. Variations in text font can communicate different meanings of different text displays. One example, having the current node in the outline display be of a large font size makes it stand out. Having this text also be in boldface also draws attention to it and its topic.

Color. Color often communicates certain aspects of a document. We apply it here as varying background colors of the textual components of the outline display. Different background colors communicate the following three possible states of an outline component: *not yet shown*, *currently showing* and *already shown*. As the user progresses through the presentation, the node title just seen will turn from the "current" color to the "already seen" color. The following node's background will then switch from "not yet seen" to "current". The different background colors for visited and unvisited nodes allow the user to keep track of sections that have been skipped over, and can thus serve as reminders to the user of components that can be navigated back to later. Distinguishing visited from non-visited links in particular enjoys standardization in CSS and wide implementation and adoption.

Contrast. Color contrast can also convey structured progression. For example, a text color that has relative little contrast with its background appears "ghosted". In future versions of our demo, we plan to give already seen nodes in the outline display a lower contrast distinction than the others. The current node will have the highest contrast. The future nodes will have a level of contrast in between these two. This contrast technique can work in combination with color selection.

Animating Visual Style. An animation-based presentation device in our demonstration is the gradual changing of an outline display component's background color. Animation of the background color of these text segments can occur simultaneously with the fade transitions of the screen display sequences they outline, emphasizing the correlation between them. Furthermore, the animation of "current" background color to "already seen" simultaneously with animation from "not yet seen" to "current" color gives the user time to see that both are happening, and on what outline nodes. Animation can also target typeface changes in the same manner.

Animating CSS classes. By assigning CSS classes to the end states of SMIL animations on document presentation properties, we allow the specification of how these properties change to be separated from the document itself into any number of cooperative or alternative associated CSS style sheets. However, SMIL animations of `class` attribute values cannot be continuous because this attributes values can only be discreet, even though the CSS properties associated with them may be. Thus, while class animation provides useful encapsulation of desired change, it does not apply to properties whose values should animate gradually.

6. PRESENTING SEQUENCE

Sequence means that sibling nodes with a composite node in the hierarchy have a clear and significant order. Textbooks, and text in general, have a meaningful sequence in which their contents appear. This paper applies the structured progression tree to the generation of a single linear thread. The basis of this procedure is using depth-first traversal to flatten a tree into a sequence of nodes. Some Web presentation format constructs, such as those providing text flow space and time, are inherently sequential, thus conveying sequence readily. Though not as directly sequential as space and time, interaction choices form a chain, making them also sequential. Visual transitions apply to sequence not by conveying them but by smoothing the impact of the traversal between them on the user.

Outline Order. The outline display communicates the sequence of the document as a whole. The spatial arrangement of node titles in the outline display, either from left to right, or from top to bottom, follows the underlying sequence of the nodes they reference. Highlighting the current node in this sorted outline display lets users recall what they just saw and what is coming next.

Space. The obvious spatial display of sequence is following the text flow of the presentation space, which is typically left-to-right then top-to-bottom of Western text. We apply this to the generation of thumbnail summaries, which follow text flow. This also applies to the outlines display, for which layout can follow text flow or go strictly top-to-bottom. CSS offers direct constructs for all of these layout patterns, so the XML can transform readily into XHTML+SMIL elements in the same order with references to appropriately defined CSS classes, as we have done in this paper's demo.

Fade Transitions. Presentations often use visual transition effects, especially fades, to remove the jarring effect of abrupt switches between displays. SMIL provides constructs for fade transitions, which XHTML+SMIL has. We use these in temporally sequential main displays to smooth the visual sensation of progressing from one node to the next.

Text Transitions. When the presentation of a node and its contents is complete, the presentation can help *transition out* by generating and presenting the text describing the progression. Such text could be "Now that we have seen", followed by the title of the node being departed from. The transition out of the previous node could lead to a *transition in* into the current node. This could be the concatenation of the text "we will now discuss", followed by the title of the current node. Such a combination of out and in transition text makes a transition between sections, though in and out transition text could appear independently. The transitions described in these generated texts always appear between siblings, communicating sequence in the local context.

Directional Buttons. *Directional buttons*, specifically *next* and *previous* buttons, are widely used presentation devices, providing navigation along the depth-first traversal of outline trees. They convey local sequence by providing access to the nodes directly before or after the current one. Hierarchy-based navigation devices, such as *left*, *right*, *up* and *down* buttons, follow the tree structure directly rather than the depth-first traversal through it.

Temporal Composition. The obvious temporal presentation of underlying sequence is one after the other, with the temporal order in the same sequence, as provided by the SMIL <seq> element. However, as Section 4 describes, the multiple possible paths through the nodes makes using <excl> a better choice. The consequence is that the sequential temporal relations implicit in <seq> require the same explicit synchronization archs in <excl> that hierarchical (and, as we will see, recurrence) navigation does.

Synchronization. The primary navigation along sequential structure comes from the next and previous buttons. Because we use the `<excl>` element instead of `<seq>`, sequential navigation uses the same SMIL constructs as hierarchical and any other type of navigation. These multiple types of navigation result in each node having more than one trigger for its begin time. This is defined by SMIL's MultiArc timing, in which `begin` attributes to be semi-colon-delimited lists of timestamps, any one of which begin active at any one time will start the element.

7. PRESENTING RECURRENCE

A recurrence is the appearance of the same node at multiple locations in the tree. Recurrences find their place in traditional discourse as recurring themes in stories or as topics that different parts of a textbook discuss. Because our structured progression is a directed acyclic graph instead of a tree, artifacts and entire sections can occur repeatedly throughout a generated presentation. Proper communication of recurrences conveys an additional dimension beyond hierarchy along which document components unite. That is, it is another way to convey relationships between concepts, thus giving the user access to more relationships. Communicating how each node recurs through the presentation helps users understand better how all the document weaves its topics together into a coherent whole. However, conveying recurrence is challenging since it crosses the grain of the presentation hierarchy.

Abbreviated Displays. An abbreviated display in our demonstrator is simply a smaller presentation than the original. Its purpose is to remind the user of the original recurrence instance and refer to it rather than repeating a potentially long sub-presentation. We change our selection threshold for main displays for repeated recurrence instances by only showing *single glance media*, which we define as media that can be recognized in a few moments. This keeps a single image and the various text captions, but excludes text descriptions. Since presentation sub-trees can also be recurrences, we also give them abbreviated displays. For each repeated recurring sub-tree instance, we show only the abbreviated display for its main display.

Space. Information about and access to other recurrences of the current node requires space on the display. To enable consistent placement of other components of the main display, recurrence communication items should appear on extreme side of the display. We display these indications at the bottom, below the next button. Text clips describing the recurrence and buttons to other instances of this recurrence appear here as well. Corresponding components of recurrence communication should appear the same place.

Text Cues. The user should know when the current node has recurrences elsewhere in the presentation. One way to convey this is to simply provide text saying so. Such textual clues can say that the topic occurs again, has already occurred, or both. In addition to stating *that* the current node recurs, text cues can also say *where* by listing the titles of the parent nodes of the recurrences. Each title displays can also be linked to the display of its recurrence.

Outline Highlighting. The system also says where the current node recurs by highlighting the entries in the outline display for the recurrences. If the outline folding does not show a particular recurrence, then the highlighting applies to its closest ancestor that the outline display shows. The user can thus click on highlighted outline display items to unfold the outline to the actual recurrences and then navigate to them.

Inter-instance Navigation. In addition to the text cue and outline navigation to recurrences described above, our demonstrator provides sequential navigation joining all iterations of each recurrence, including the original. This is a bi-directional sequence of next and previous bi-directional links put in each display of a recurrence. In the repeated, abbreviated instances, these appear with the link to the original instance. This enables users to step back to the original display to review information that they may have forgotten. It also allows users who have skipped ahead over the original display to step back in the default presentation progression to see it for the first time. A button for each recurrence unfolds the outline display to its iteration and highlights it.

Timing. Time enables more effective use of outline display space. Unfolding all outline entries for all instances of a recurrence can take a lot of space. Unfolding them one at a time, while refolding the rest, would use less space, though, of course, more time.

Interaction. Code for recurrence navigation uses the same timing and user event constructs as sequential next buttons and outline navigation. The highlighting of outline entries as recurrences uses the same constructs for SMIL animation on CSS properties that highlighting for visited nodes does. The use of a specific typeface variant, such as italic, can draw attention to an outline item and communicate that its node is another instance of the recurrence.

8. CONCLUSION

This paper discusses the generation of coherent hypermedia presentations from RDF knowledge bases. The structure of these presentations derives from semantic information encoded in the underlying RDF Schema. Consequently, modifications on the content and schema level automatically affect the resulting Web presentations. While the final transformation from the structured progression to the end-user presentation derives mainly from the structured progression, the (client-side) transformation process can now access additional information with direct RDF queries. Presentation devices guide this transformation process by relating structured progression components to means for conveying them.

Our approach applies current Web technology wherever possible. The underlying repository uses RDF and RDF Schema stored and processed by Sesame. An XML Schema-defined format encodes structured progression through this RDF-encoded information. Finally, XSLT encodes the transformation from the document to the end-user presentation in XHTML+SMIL. This paper's primary technical contribution is the integrated use of semantic queries (SeRQL) in a syntactic transformation (XSLT). In addition, this paper applies style sheets (CSS and XSLT) to convey discourse structure using presentation devices in Web-based presentations (XHTML+SMIL).

Despite using W3C recommendations wherever applicable, we still found some key components of a future Semantic Web that requires standardization. These components include a commonly agreed upon Semantic Web query language (currently we use both RQL and SeRQL) and a Semantic Web rule language for defining domain specific inferencing (currently these are embedded in the Java code). At the time of writing, the first steps to set up W3C working groups for addressing these issues are currently being taken. In addition, there are many formats supporting access to relational database content, including JSP, XSP, ASP, and PHP. We believe providing access to semantic content will prove to become more and more important. No such steps, however, are being taken to provide a common means of deploying the results of RDF rules and queries in generating Web pages.

Retrospectively, the paper investigates a bridge spanning the gap between the semantics of the content and the syntax of the final presentation. We plan to use the current testbed to further explore the gap between the stored semantic representations and the coherent Web-based presentations that are required to effectively communicate the content and underlying document structure to the user. We hope to inspire the Web community to address the current technological gap between the Semantic and Document Webs and strive towards a Semantic Web that is truly an extension of the current Web instead of separate Web for machines only.

The demo and other resources for this paper are accessible at http://homepages.cwi.nl/~media/demo/topia/ .

ACKNOWLEDGMENTS

This work was funded by the Topia project of the Telematica Instituut, sponsored in part by IBM and by the NWO NASH project. We thank the Rijksmuseum Amsterdam for their permission to use their Website's database and media content [25]. Ivan Herman provided valuable insights into the comparison of XML, RDF and XSLT. Geert-Jan Houben gave useful suggestions in motivating and structuring this work. We followed Patrick Schmitz's expert tips in the use of XHTML+SMIL.

REFERENCES

[1] Aidministrator Nederland B.V. SeRQL user manual. April 4, 2003. Available at http://sesame.aidministrator.nl/publications/SeRQL%20user%20manual.pdf

[2] Alberink, M., Rutledge, L. and Veenstra, M., Clustering Semantics for Discourse Generation, In: The First International Workshop on Hypermedia and the Semantic Web, (Nottingham, UK, August 30, 2003).

[3] The Apache XML Project, Xalan-Java version 2.5.1.

[4] Benjamins, V.R. and Fensel, D., The Ontological Engineering Initiative (KA)2, In: Proceedings of the International Conference on Formal Ontologies in Information Systems (FOIS-98) (Trento, Italy, June 1998), IOS-Press, 287-301.

[5] Bos, B., Lie, H.W., Lilley, C., and Jacobs, I. (eds). Cascading Style Sheets, Level 2. World Wide Web Consortium (W3C) Recommendation, May 1998.

[6] Brickley, D., and Guha, R.V. RDF Vocabulary Description Language 1.0: RDF Schema. World Wide Web Consortium (W3C) Working Draft, January 2003. (work in progress)

[7] Broekstra, J., Kampman, A, and van Harmelen, F., Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In: Proceedings of the First International Semantic Web Conference (ISWC 2002) (Sardinia, Italy, June 9-12, 2002), Springer, 54-68.

[8] Clark, J. (ed). XSL Transformations (XSLT) Version 1.0. World Wide Web Consortium (W3C) Recommendation, November 1999.

[9] Cohen, A. (editing chair), Synchronized Multimedia Integration Language (SMIL 2.0) Specification, World Wide Web Consortium (W3C) Recommendation, 7 August, 2001.

[10] Czajka, K. Design of interactive and adaptive interfaces to exploit large media-based knowledge spaces in the domain of museums for the fine arts. Masters Thesis, University of Applied Science Darmstadt, June 2002.

[11] Dublin Core Community. Dublin Core Element Set, Version 1.1, 1999.

[12] Geurts, J., Bocconi, S., van Ossenbruggen, J. and Hardman, L., Towards Ontology-driven Discourse: From Semantic Graphs to Multimedia Presentations, In: Proceedings of the Second International Semantic Web Conference (ISWC 2003) (Sanibel Island, Florida, USA, October 20-22, 2003), 597-612.

[13] Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D. and Scholl, M., RQL: A Declarative Query Language for RDF. In: Proceedings of the Eleventh International World Wide Web Conference (WWW2002) (Honolulu, Hawaii, USA, May 7-11, 2002), ACM Press, 592-603.

[14] Landow, G. P., Relationally encoded links and the rhetoric of hypertext, In: Proceedings of the ACM conference on Hypertext (HT87) (Chapel Hill, North Carolina, November 13-15, 1987), ACM Press, 331-343.

[15] Lassila, O. and Swick, R.R. (eds), Resource Description Framework (RDF) Model and Syntax Specification. World Wide Web Consortium (W3C) Recommendation, February 22nd, 1999.

[16] Little, S., Geurts, J. and Hunter, J., Dynamic Generation of Intelligent Multimedia Presentations through Semantic Inferencing, In: Proceedings of the Sixth European Conference on Research and Advanced Technology for Digital Libraries (Rome, Italy, September 16-18, 2002), Springer, 158-189.

[17] Maedche, A., Staab, S., Stojanovic, N., Studer, R., and Sure, Y. SEmantic portAL - The SEAL approach. In Creating the Semantic Web. D. Fensel, J. Hendler, H. Lieberman, W. Wahlster (eds.) MIT Press, Cambridge, MA. 2002.

[18] Microsoft, Inc. Internet Explorer 6.1. 2003.

[19] Hunter, J., Adding Multimedia to the Semantic Web — Building an MPEG-7 Ontology, In: Proceedings of the International Semantic Web Working Symposium (SWWS) (Stanford University, California, USA, July 30 - August 1, 2001).

[20] Newman, D., Schmitz, P., and Patterson, A. (eds). XHTML+SMIL Profile. World Wide Web Consortium (W3C) Note, 31 January 2001.

[21] van Ossenbruggen, J., Geurts, J., Cornelissen, F, Rutledge, L. and Hardman., L Towards Second and Third Generation Web-Based Multimedia, In: Proceedings of the Tenth International World Wide Web Conference (WWW10) (Hong Kong, China, May 1-5, 2001), ACM Press, 479-488.

[22] van Ossenbruggen, J., Hardman, L., Rutledge, L., Towards Smart Style: Combining RDF Semantics with XML Document Transformations, CWI Technical Report INS-E0303, ISSN 1386-3681, October 2003.

[23] Patel-Schneider, P. and Siméon, J., The Yin/Yang Web: XML Syntax and RDF Semantics, In: Proceedings of the Eleventh International World Wide Web Conference (WWW 2002) (Honolulu, Hawaii, USA, May 7-11, 2002), ACM Press, 443-453.

[24] Raggett, D., Le Hor, A. and Jacobs, I. (eds), HTML 4.01 Specification, World Wide Web Consortium (W3C) Recommendation, December 24, 1999.

[25] Rijksmuseum Amsterdam, Rijksmuseum Amsterdam Website. http://www.rijksmuseum.nl/.

[26] L. Rutledge, M. Alberink, R. Brussee, S. Pokraev, W. van Dieten, and M. Veenstra. Finding the Story — Broader Applicability of Semantics and Discourse for Hypermedia Generation. In Proceedings of the 14th ACM conference on Hypertext and Hypermedia (HT03) (Nottingham, UK, August 26-30, 2003), 67-76.

[27] Rutledge, L., Davis, J., van Ossenbruggen, J., and Hardman, L. Inter-dimensional Hypermedia Communicative Devices for Rhetorical Structure. In: Proceedings of the International Conference on Multimedia Modeling 2000 (MMM00) (Nagano, Japan, November 13-15, 2000), World Scientific, 89-105.