elpub2002 Proceedings — J. A. Carvalho et. al. (Eds.) — VWF Berlin, 2002

Configurable Presentation of Filtered Metadata

Steve Probets

Department of Information Science, Loughborough University
s.g.probets@lboro.ac.uk

Abstract. This paper looks at how metadata, encoded in XML, can be used to support better resource discovery and extensible information presentation, through the application of standard XML related technologies. By applying XSLT transformation scripts to a structured metadata corpus, the metadata can be presented in differing styles. These scripts can be used to filter and present specific resource descriptions within the corpus. Because the metadata that is presented is a function of the scripts applied to the corpus, altering the scripts can alter how the metadata is presented. This paper looks at the advantages of presenting metadata through the use of dynamically generated XSLT scripts that give a simple yet wide-ranging degree of control over the presentation. The possibility of graphical metadata presentation is discussed. The services described in this paper utilize metadata harvested by the OAI's Protocol for Metadata Harvesting.

1 Introduction

'*The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation*' [1]. As part of the underlying framework for the proposed semantic web, metadata descriptions of web resources are becoming more important and ever more common. These descriptions will probably become the underlying information assets that will help develop the world-wide web that we use today into the semantic web of tomorrow. It is likely that to facilitate the semantic web, both humans and intelligent agents will consume metadata to manage e-commerce, information retrieval and information space personalisation.

1.1 Metadata

Metadata, which is often concisely defined as data about data, is becoming widely used to describe electronic resources across many disparate domains of interest.

Each of these domains has a different notion of what metadata elements are required, and many different metadata schemes have been (and are continuing to be) developed by different user communities. These different schemes specify which elements should, or could, be used to describe resources, and sometimes specify allowable options and vocabularies that should be used within the content of these descriptive elements. Just a few of the many metadata schemes that describe electronic resources in different ways for different domains include: MARC and Dublin Core (resource discovery), InDECS and XrML (rights and intellectual property), MPEG-21 (multimedia), OLAC (language), DCMI-ED, GEM and IMS (education).

Metadata instances that are specified in XML can be validated against associated XML schemas to check that they conform to specific criteria. These schemas are formal specifications, outlining which elements may (or must) be present, the order they can appear, and additional information such as data types or additional qualifications or attributes, that may occur. These schemas effectively formalise the 'rules' to which the metadata instances must comply.

There is a degree of flexibility when describing resources using metadata. Resources can be described multiple times using different schemes, or elements from different schemes can be combined together into a single description allowing metadata instances to employ a 'mix and match' approach that uses namespaces to identify which terms belong to which schemes.

1.2 Metadata Dissemination

Whilst XML enables representation of metadata, the semantic web will only become a reality if the metadata can be effectively disseminated. There are a number of different approaches to this problem, but a notable one comes from the Open Archives Initiative (OAI) [2]. The OAI is a metadata initiative aimed at providing a mechanism for metadata dissemination and interoperability. There are three constituents to the OAI model – data providers, service providers and a protocol for metadata harvesting (PMH). In order to support the OAI model, data providers make metadata available. Service providers collect (or harvest) this metadata and use it as the basis on which to build value added services. The PMH specifies the manner in which service providers harvest the metadata from data providers and the manner in which the data providers respond to these requests. The requests are http-based, and consist of verbs specifying the information required from the data provider (thus a verb of `ListMetadataFormats` in a request will receive a response specifying the metadata schemes that are supported by the archive e.g Dublin Core, OLAC etc.). The responses consist of valid XML documents containing information about the metadata repository (such as which metadata schemes are supported) or the metadata instances themselves. There is no specification as to how the metadata is stored either by the data providers, or the service providers,

only the structures of the requests and responses are specified by the PMH. However, the OAI does specify that all OAI compliant data providers must provide a base-level of metadata – this level is the Dublin Core, although other schemes can be made available by the data providers if desired. These schemes must have associated schemas, referenced through a namespace so that the metadata can be validated.

2 A Configurable Presentation Service

Due to the diversity of metadata schemes supported by OAI data providers it is difficult for service providers to develop services which will support querying and presentation of the variety of metadata instances harvested from the different providers. The aim of this research is to investigate the feasibility of providing an OAI service that will support resource discovery through the storage, searching and presentation of metadata instances that conform to a variety of schemas. This consists of three phases: *schema analysis* which determines the markup, structure and datatypes of the metadata instances; *filtration* which enables users to be guided by the schema analysis to extract resources of interest from the metadata corpus; and *presentation* which presents these resources to the user (again, this may be informed by the results of schema analysis).

2.1 Schema Analysis

The presentation of an information corpus with the intention of facilitating resource discovery needs two goals to be satisfied. First the available information (the metadata instances) need to be filtered so that only resources of interest remain, and second, these resources need to be presented in a way which facilitates discovery. But before a corpus can be filtered, the nature of the information contained within the corpus needs to be understood. Many filtering and presentation services have been developed for each specific metadata schema supported, but this research is investigating analysing schemas to determine as much as possible about the likely structure of the metadata corpus. This enables a generic service to be produced which configures its interface depending on the metadata being queried.

A generic XSLT script is being developed that processes XML schemas to determine which elements are likely to be present in the instances. This information is presented as a web (HTML) form (Fig 1.a) that can be used by users to specify their filtration and presentation requirements. Many schemas also specify datatypes, patterns or alternatives that restrict the allowable values of elements and attributes. For example, analysis of the OLAC schema results in knowledge about the various elements, e.g. that the `<date>` element contains a refine attribute, and that this attribute can have a value of created, modified, issued etc. This information can

be used to guide users through the process of filtering the metadata by presenting these alternatives as choices when specifying filtration criteria. Although the options can be presented in the web form, it should be noted that optional elements and flexibility built into schemas mean that this approach can only be used as a rough guide to users. Additionally, although simple schemas can be presented effectively, presentation methods for element hierarchies and complextypes need to be developed [3], however schema analysis shows promise as the first stage of this OAI service.

2.2 Simple Filtering Techniques

The web form mentioned above is used to solicit user requirements for the filtering and presentation service. Although OAI services are built on harvested metadata instances, the services can store the metadata in any format that is appropriate. Services may chose to import the metadata into traditional databases and build services by creating SQL queries to extract the necessary information, alternatively they may keep the harvested metadata as a collection of XML files and attempt to build services from these files using XSLT or XQuery scripts to identify and present extracts of XML instances. However, the method chosen in this research is to use a native XML database [4] to store the instances. The native XML database used in this research is xindice [5]. Input and output to xindice is pure XML, it enables any well-formed documents to be included in the database, though validation via schemas is not (currently) supported. This means that *any* well formed XML document can be imported into xindice and subsequently queried, updated or extracted. In this case, where the harvested metadata may conform to many disparate schema which are unknown in advance, this is beneficial as existing tables, fields and datatypes do not have to be specified in advance. This is another reason that the web form for obtaining user requirements must be based on schema analysis. Xindice enables the harvested instances to be included in the database and supports document query and extraction based on XPath. The XPath based queries are dynamically created based on the user input from the HTML form. The queries enable the requested metadata to be filtered and extracted from the database. This process is analogous to the standard CGI-database interaction scripts of standard web applications and can be summarized as follows:

After harvesting, the metadata instances are stored in Xindice. The OAI service filters the corpus of records based on user specified criteria, and presents the results. The first stage in this process is to employ the schema analysis technique mentioned earlier to present a web form to the user. The content of the form is dependant on the schemas to which the harvested metadata belong. By interacting with this form a user specifies the resources of interest, e.g that he wishes to be presented with resources whose creator is 'Smith', where the subject is 'XML'.

Given these criteria, the database can be queried and the necessary records extracted before a presentation method can be invoked. The method of achieving filtration is dependent on the database used to maintain the XML metadata instances. In the case of an Xindice database, XPath is the query interface – thus a query of //record/metadata/dc/creator[contains(.,'Smith')] would return instances where Smith is contained in the content of the creator element.

Simple queries such as this are created and implemented server side as part of the OAI service, with submission of the HTML form invoking a CGI script which creates scripts that filter the metadata database. Once filtered, the metadata has to be presented to the users. A seemingly logical way to do this would be to use XSLT scripts to transform the instances into a simple tabular format. In her paper on presenting descriptions of web resources using XSLT [6], Cawsey describes how simple schema analysis techniques, similar to those described above, can be used to create a web form which, upon submission, would automatically generate an XSLT script to present tailored descriptions of resources. Cawsey describes how presentation of the results in tabular form is possible, though presentation in a more accessible form, such as natural language, is difficult given the capabilities of XSLT. This is partly due to natural language being dependant on prior content, combined with XSLT's inability to review its output tree. However translation into a tabular or natural language form is not the only kind of presentation possible.

2.3 Graphical Presentation

Traditional web search engines typically present results as a simple hit list of titles, ordered by criteria such as the location of the search terms, and the techniques mentioned so far describe similar methods of presentation. However this information can be presented in a number of ways. Recently web search engines such as the Kartoo meta search engine (`http://kartoo.com`) have been developed where results are displayed in a graphical manner. Kartoo presents its search results as a map, with the hits being related by a web of common subjects or terms. This is similar to a Visualisation By Example (VIBE) [7] visualisation. Moving the cursor over the graphical display gives a further description of the hits. The underlying format for the Kartoo engine is flash (or HTML with added JavaScript to provide the interactivity).

Presentation of search results in a graphical information space is not new, though it is only recently that it has started to appear in common web search engines. Many information presentation and visualization techniques, such as the one employed by Kartoo, may be appropriate to metadata presentation, therefore a logical progression of the mechanisms described so far in this paper, was to investigate using XML-based technologies to create a graphical description of the filtered metadata. Because the metadata itself is accessed from xindice as XML instances, the possibility of creating an entirely XML-based solution by present-

ing the results of metadata filtering using the Scalable Vector Graphics (SVG) [8] language has been investigated. SVG is an XML-based language, in many ways it is a XML-based version of Flash's SWF [9], it can be thought of as a '*page*' description language for the web which can, if necessary, support user interaction and animation. SVG images are built up of graphic primitives such as moveto, lineto, curveto etc. These primitives are contained within an SVG canvas that is displayed within a standard HTML page. The language specifies exact location of lines, curves, circles, squares etc. within the SVG canvas. These primitives can be grouped together, labelled and reused throughout the graphic.

SVG is a language for creating 2 dimensional vector graphics, and as such is good for presenting comparative or positional information such as whether an entity is located before, above, below etc. another entity. Christel et. al. [10] have described creating VIBE-like representations in SVG to help navigate a large video library.

In order to describe the graphical presentation process, consider metadata instances in the corpus where the format element is qualified with a size attribute. E.g: `<format size="21306">JPEG</format>`

One simple way to display resources in a graphical fashion would be a 2-dimensional graph, with axes of date and size creating a simple space in which to present the resources. It is possible to create a presentation such as this using a XSLT script that extracts the resources, analyses the content of the date and size elements and places a representation of the resource on a 2-D graph. Simple icons are used to represent the resources, and clicking on these icons invokes simple JavaScript to present more details of the resources in the information space.

To explain this, take a simplified example where a user requires information on resources created by 'Probets', displayed in a size and date information space (see Fig 1.b). The following steps need to be taken to create the presentation. First, the resources need to be filtered so that only resources authored by Probets are considered. Then the extent of these resources in the size and date dimensions is obtained and used to determine the scale and location of the axes of the graphical information space. At this stage the axes and legends can be created in SVG.

Once these extents have been determined the necessary data is extracted for each resource, it may be necessary to perform this in a second stage, after the extents have been determined. XSLT code to obtain this information follows the form: (Note: the XPath in this excerpt assumes the script will be applied to a PMH harvested instance that contains dublin-core metadata – schema analysis is used to inform creation of the necessary XPath):

```
<xsl:for-each
 select="GetRecord/record/metadata/dc/
 creator[contains(.,'Probets')]">
```

```
<xsl:variable name="flsize">
<xsl:value-of select="../format/@size"/> </xsl:variable> </xsl:for-each>
```

Similar information would need to be extracted for the date of publication. Once all the information required to generate the graphical presentation has been extracted from the metadata instance, the location of the resource within the SVG canvas can be determined and a representation of the instance is output into the SVG stream. A simple way to do this is to use reusable graphical components. Within SVG it is possible to specify definitions (reusable groupings of graphic primitives) that can be reused and placed at different positions within the graphic. For instance a circular icon can be used in the graphic to represent a resource – this icon could be defined once as:

```
<defs> <g id="circ">
    <circle cx="0" cy="0" r="7"
     style="fill:rgb(0,255,255);stroke:rgb(0,0,0);
     stroke-width:1"/>
</g> </defs>
```

Each resource is then displayed in SVG by creating a copy of this icon (referenced via the 'circ' identifier), and placing it at the appropriate place on the SVG canvas. (Note: In the following example the copy is created with the `<use>` tag with placement of the copy being achieved with the `<g>` tag with the transform attribute. The SVG representation can also be extended by including further information, such as title and author in JavaScript functions.):

```
<g transform="matrix(1 0 0 1 350 411.302)"
onclick="alert('Probets, Steve G et al \n Dynamic Link Inclusion
in Online PDF Journals')"> <use xlink:href="#circ"/></g>
```

The XSLT script responsible for creating the above SVG representation needs to be able to generate the information for the transformation matrix (such as the x coordinate of 350 and the y coordinate of 411) from information (such as the date or file size) that it has extracted from the instances. It only requires limited mathematical support to calculate the SVG coordinates on which to place the icon representing the resource. To illustrate this, the following XSLT code calculates the x coordinate of a resource whose date of publication has been identified and stored in a variable named $RESDATE. (In the following example the horizontal axis is physically located between x coordinates `xmin` and `xmax` on the SVG canvas and represents a date range of `yrmin` to `yrmax`).

```
<!--Map date to a coordinate in the SVG canvas-->
<xsl:variable name="xcoord"> <xsl:value-of select=" (\$RES_DATE -
yrmin) * ((xmax - xmin) div (yrmax - yrmin)) + xmin  "/>
</xsl:variable>
```

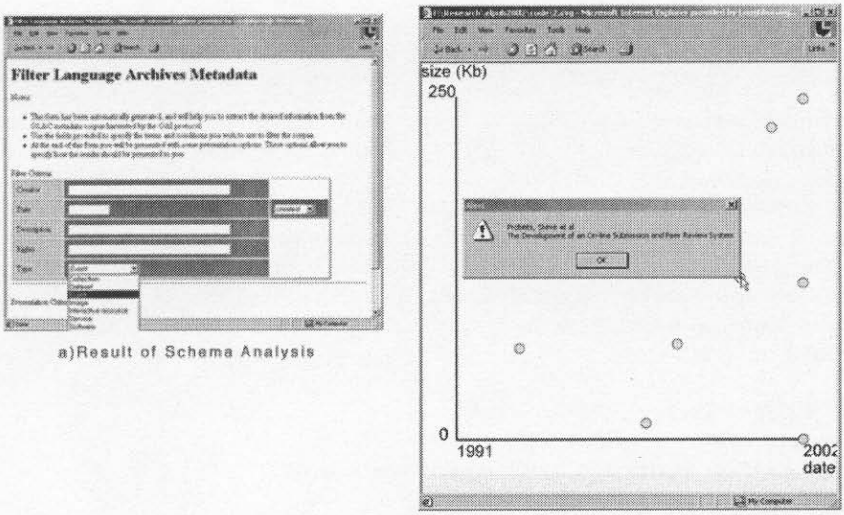

Fig. 1. *Stages in the Process*

A variety of presentations are possible by dynamically creating translation scripts based on the user's presentation requirements (which themselves are guided by the schema analysis). The possibilities take into account the fact that numeric values can be represented in different ways to enumerated values or constrained options, and that free-text elements also need to be handled differently.

3 Conclusions

The approach described in this paper is at an early stage of development. By generating the presentation scripts automatically, users can specify not only what, but also how the information is to be presented. The goal is to analyse the schemas to which the metadata instances belong, and create an input mechanism that specifies what and how the information should be presented. Scripts can then be dynamically created, giving the user a personalised, filtered representation of the metadata they require.

The work has shown that relatively simple schemas can be analysed to produce a web form. The form results in simple graphical presentations by dynamically guiding the creation of XSLT scripts that process metadata instances. However the ease with which more complicated presentations could be generically produced from a wider range of more complicated metadata descriptions needs further investigation.

References

1. T. Berners-Lee, J. Hendler, O. Lassila: The Semantic Web. Scientific American, 284(5). 2001
2. Open Archives Initiative. Available: `http://www.openarchives.org/`
3. S. G. Probets: Structured Metadata Analysis. To be published in Proceedings of the International Conference on Advances in Infrastructure for e-Business, e-Education, e-Science, and e-Medicine on the Internet. SSGRRs, L'Aquila, Italy. 2002.
4. R. Bourret: XML Database Products – Native XML Databases. 2002. Available: `http://www.rpbourret.com/xml/ProdsNative.htm`
5. Apache Software Foundation: Apache Xindice. 2002. Available: `http://xml.apache.org/xindice/`
6. A. Cawsey: Presenting tailored resource descriptions: Will XSLT do the job? In Proceedings of the 9^{th} International World Wide Web Conference. 2000.
7. R. L. Korfhage. To see or not to see – is that the query. In proc of the 14^{th} annual ACM/SIGIR conference. Chicago USA. 1991. pp. 134-141.
8. World-wide web consortium: Scalable vector Graphics. 2002. Available: `http://www.w3.org/Graphics/SVG/Overview.htm8`
9. S. Probets, J. Mong, D. Evans, D. Brailsford: Vector Graphics, From PostScript and Flash to SVG. In Proceedings of the ACM Symposium on Document Engineering. Atlanta, Georgia, USA. 2001.
10. M. G. Chistel, C. Huang: SVG for Navigating Digital News Video. ACM Multimedia '01. Canada. 2001.