Treating metadata as annotations: separating the content markup from the content

F. Paulsson[1], and J. Engman[2]
[1] Royal Institute of Technology, Nada, Stockholm, Sweden
[2] Center for IT in Northern Sweden, Umeå, Sweden

Abstract— **The use of digital learning resources creates an increasing need for semantic metadata, describing the whole resource, as well as parts of resources. Traditionally, schemas such as Text Encoding Initiative (TEI) have been used to add semantic markup for parts of resources. This is not sufficient for use in a ”metadata ecology”, where metadata is distributed, coherent to different Application Profiles, and added by different actors. A new methodology, where metadata is "pointed in" as annotations, using XPointers, and RDF is proposed. A suggestion for how such infrastructure can be implemented, using existing open standards for metadata, and for the web is presented. We argue that such methodology and infrastructure is necessary to realize the decentralized metadata infrastructure needed for a "metadata ecology".**

Index Terms*—: *metadata, XML, XPointers, content markup, Semantic Web

I. Introduction

Structural markup and metadata is becoming increasingly important as the use of digital learning resources increases. Metadata is needed for discovery of digital learning resources as well as for describing different aspects of a learning resource, such as intended use, technical constraints, annotations, relations to other resources, intended (and previous) context of use etc. Structural markup (e.g. XML) is needed to describe the structure of the content. Structural markup provides data structure that can be used for presentation, transformation of one or more resources or for processing content in various other ways. There is however an increasing need to add metadata descriptions and semantics that addresses the internal structure of a resource in order to describe a part of a digital resource - much in the same way as metadata is used to describe the entire resource. Only a small part of a resource is of interest in many cases, such as an excerpt of a text or a clip from a movie etc. It is sometimes desirable to add markup that add a certain perspective, comments or emphasis to a section, or a part of a resource, as well as to express relations to other resources, or parts of other resources.

Two basic conditions need to be fulfilled in order to describe and use a part of a resource in such way: the part of the resource that is of interest must be identified and marked in order to be "isolated" for use in a new context, and the excerpt needs to be indexed with metadata from the perspective of the new context. We might for example want to add markup and describe a specific concept that is used in a text or, describe relations to other concepts, resources or examples.

Detailed examples of how both types of markup can be used is outlined in [1] where the case of adding markup up the Swedish national curricula (the national steering documents for Swedish schools) is described. The markup of the Swedish national curricula was used as a case for test and evaluation of the markup methodology and the prototype, whose implementation is the subject of this article.

A. Previous and related work

Related work in this area includes research on descriptive metadata for resources is usually managed using generic or domain specific standards such as Dublin Core (DC) [2] or the Learning Object Metadata (LOM) [3] for the Learning Objects, together with application profiles for certain domains or cultures [4]. Examples of Application Profiles based on LOM is the CELEBRATE Application Profile [5] and the well known IMS Metadata [6]. Electronic resources described using DC or LOM is identified using a unique identifier that connects the metadata to the resource. The identifier is usually an URI (or an URL for web resources which is a subtype of URI). Even though there are a number problems associated with this king of "cataloguing" metadata, the basic principles are fairly established and well known as described by Duval et al. in [7].

Traditionally, much of the metadata research has been carried out within the library community. The focus tends to be on physical artefacts such as textbooks, and the needs for markup within resources (internal markup) has been limited. This is however changing as more resources become digitally available. This is also evident from the development and use of standards such as DC, outlined by Nikam in [8] and markup schemas such as TEI [9]. At the same time the need for metadata is becoming increasingly important outside the library sector as well, which is also one of the driving forces behind the development of Semantic Web technology [10]. The main purpose of the Semantic Web is to enable the use of semantically rich, and machine readable – as well as "machine understandable" - metadata in an additional layer that supplements the current web [10]. The main technology for realizing the Semantic web is the Resource Description Framework (RDF) [11]. A basic infrastructure for a distributed metadata ecology is created using RDF. This is an infrastructure that shares many of the advantages of the World Wide Web, while avoiding a few of the most serious disadvantages by adding machine readable data structure (through the use of RDF/XML) and metadata semantics. The characteristics and use of RDF are explained in detail in section 2.2.1.2.

Beside the "traditional" use of metadata, two additional aspects of markup are usually considered: *structural markup* (describing the data/document structure) and *content markup* (descriptive metadata) that is semantic descriptions *about* the content, addressing specific sections of the content. Traditionally, markup languages such as SGML and XML have been used to describe both structure and semantics usually combined in the same schema. This is the case with the Text Encoding Initiative (TEI [9], which is one of the most used schemas for semantic markup of literary and linguistic texts for research purposes. TEI was tied to SGML in the beginning but has now evolved into using XML as its main carrier. The use of TEI is described by Burnard et al. in [12]. While TEI is used for both structure- and semantic markup, other schemas (also referred to as markup languages) focus mainly on structural markup. This is the case with DocBook [13], which is a schema that is mainly intended for technical documentation. Open Document (OD) [14] is another example of a XML-based document schema, intended for "office" documents. The OD schema is a standard established by OASIS[1]. Open Document addresses not only text resources, but also graphics and audio through the use of related XML- standards, such as SVG for graphics [14].

B. *The metadata problem*

The way in which structural markup and descriptive markup (metadata semantics) are mixed with the content (data) in schemas such as TEI may be sufficient for markup of "*literary and linguistic texts for online research, teaching, and preservation*" [9] – texts that usually doesn't change and can be provided with one set of controlled markup for a specific purpose. This is however not the common case, especially not regarding how digital resources are produced, used, reused and managed for e-learning. In many cases it is eider possible or allowed to alter the original document by adding additional markup that may change the overall semantics of the document. There is often a need to support more than one model or Application Profile for metadata for different contexts, without interfering with the original document or existing markup. It is desirable, as well as necessary in many situations, to allow third party to add the markup of their choice in order to describe a resource or part of a resource in a specific context and according to a specific metadata model or application profile. In addition to the reasons above, TEI documents (and alike) tend to get very messy and unmanageable after a while of heavy markup. It is also hard to manage overlapping markup, where one set of semantic markup overlaps a section that is already marked with other semantic markup. Such markup methodology is mainly suitable for one-time markup in a non-distributed environment under some kind of authoritarian control. In this article we argue for a metadata infrastructure that facilitates a "democratic" and distributed view on metadata: a metadata ecosystem that is controlled by no one and contributed to by anyone – very much like the web! For this purpose it is necessary to use a methodology that allows metadata to be added and stored separated from the resource as well as allowing different metadata records to independently co-exist for the same part of a resource or even overlapping parts of a resource.

Our thesis is that such metadata infrastructure can be implemented using Semantic Web technology together with other, supplementing and existing open standards, such as XML and related technologies.

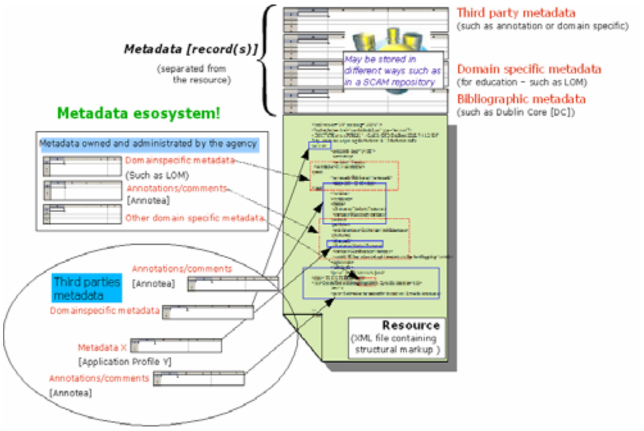

Figure 1. Outline of the basic principle of different "layers" of metadata for a resource as well as for pointing-in metadata "layers" for adding metadata describing parts of the resource.

The problem space above was described in detail in [1], where a case of adding semantic markup to the Swedish national steering documents, NSD (curriculum) was described. In the NSD-case, the markup was used to make semantic connections between the national curricula and digital learning resources in Learning Object Repositories (LOR)[2], as well as for adding new perspectives and for "filtering" concepts used in steering documents. This methodology makes it possible for teachers in one subject to find relevant digital learning resources intended for another subject. A history teacher can for example find relevant resources intended for chemistry teaching. The search for digital learning resources can also be linked to the curriculum through semantic relations determined through reasoning – a mechanism made possible by the semantic markup of the steering documents.

The requirements from the NSD-case gave five criteria that needed to be fulfilled [1]: (1) the content; that is the semantics and meaning of the original document must be preserved. (2) The format for data distribution must be structured, open and application neutral, (3) markup of the whole resource as well as of a portion of a resource must be supported, (4) multiple metadata models and application profiles must be supported for the same digital resource (as well as parts of a resource), (5) possibilities to add markup without interfering with the original document or existing metadata. The five criteria where used as a basis, together with the needs described above, for understanding and isolating the technical challenges that had to be addressed.

The problem addressed in this article is twofold: addressing the criteria above and doing it in a technology

[1] OASIS is the acronym for: Organization for the Advancement of Structured Information Standards. OASIS is found at http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=office

[2] NoT-navet. http://www.skolutveckling.se/kunskap_bedomning/matematik_naturvetenskap_och_teknik/notnavet/

neutral way, using existing and open standards for metadata and information structure.

C. Delimitation

This article suggests a system-architecture for a distributed metadata infrastructure and methodology for adding to a "metadata ecology". The focus is on technical challenges and exploring implementation strategies through the experimental implementation of a prototype: *the Annofolio*. Details on pedagogical- and usage aspects are out of scope of this article.

II. Method, methodology and implementation

A general-purpose methodology for separating semantic metadata markup of a resource (or part of a resource) from the structural markup and data - e.g. the resource XML - is suggested in this article. The method is general in the sense that it can be applied to all types of XML-based resources in an application neutral fashion. Even though the experiments focus on textual resources, the method is general enough to support non-textual resources, such as SVG graphics, as well. Such general approach would however made the implementation much more complex. This is also the reason to focus on textual resources for this first prototype.

The empirical basis for this article is mainly experimental, based on implementation and prototyping. The basic theories and ideas for a metadata infrastructure originates from previous work, experiences from real life cases such as [1] and from limitations identified in markup languages such as TEI, described in [9], and [15]

A. A general-purpose metadata infrastructure

The infrastructure and architecture underlying the Annofolio is based on existing standards and recommendations, such as XML technologies [16], and Semantic Web Technologies [10]. The Annofolio prototype handles metadata records as Annotea annotations [17]. The beauty of using Annotea is that the only thing that is compulsory is that annotations should be expressed using RDF. This general approach makes it possible to use any schema, and by that optional metadata models expressed using RDF. Annotea was chosen as it is a W3C LEAD project, based on existing W3C recommendations, implemented using the RDF Annotation Schema [18] – a simple and straightforward approach, suitable for the Annofolio. Other, similar tools that where considered was Semtag [19], Melita [20] and S-Cream [21]. All three with a focus on automation or semi-automation that makes them more complex to manage.

The basic idea is to start out with a resource containing structural markup (XML), in this case a textual resource structured using Simplified DocBook [22], and "point-in" semantic metadata that describes a specific part of the resource.

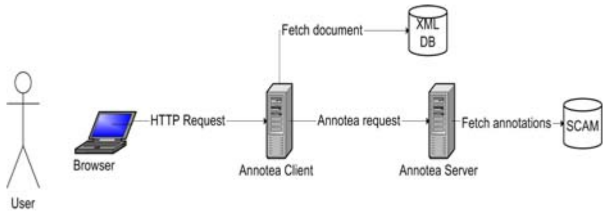

Figure 1. *The principle of querying the system for an XML-document to browse and/or annotate.*

Annotations are pointed into the right section of the XML-document using XPointers [16], which are determined within the browser, using the Fujitsu XLink Processor (XLiP) [23] that processes the documents Document Object Model (DOM) [24]. The fact that metadata is stored externally, separated from the resource, makes it (theoretically) possible for anyone to add their metadata. Metadata can be stored in a distributed fashion and different actors can use their own independent metadata store without interfering with each other, or with the original resource that remains untouched. This flexibility and independence is what creates the metadata ecology.

B. Technical settings and implementation

The technical settings were setup based on a set of third party systems, each system managing its respective part of the data and functionality. This approach was chosen to reduce the complexity of the implementation and to eliminate as many sources of errors as possible from the start, which means more focus on the actual problems to be solved. The architecture and its subsequent design decisions are described "from the bottom up" in the following sections.

1) Data layer

The data layer consists of two parts: the XML store consisting of an Apache Xindice [25] XML database, and the RDF store, consisting of the *Standardized Contextualized Access to Metadata (SCAM)* system described in [26] and [27].

a) XML store

The choice of a native XML database was made based on the need for an XML-store that supports XML-mechanisms such as XPath, XPointer and XML–DOM. Two alternatives where also considered: using a traditional Relational Database Management System (RDBMS) or simply using the web-server file system.

Using a traditional RDBMS as an XML-store is associated with a number of problems, related to the different nature and properties of the unordered, relational model used by the RDBMS and the ordered, three-structure data model used by XML. This means that XML files have to be striped down and forced into a relational model through mapping. Those problems – as well as a suggested solution - are described in detail by Tatarinov et al. in [28]. There are no straightforward ways to access and quire the inner structures of XML within a relational database; even though many RDBMS provide mechanisms for working with XML it is still needlessly complex and not all that easy to access XML specific mechanisms. SQL statements must be used in order to quire the inner XML mechanisms. This creates unwanted overhead, as well as potential fault sources.

Using the web server file system would be an easy, straightforward and practicable way. One advantage of such approach is that there is no need to integrate a complex third party system for storing XML-data. Such approach would however not give any specific support for accessing and querying XML, other than what is implemented by the application itself, adding the disadvantage that the system and its code becomes much more complex and with an impending risk of drawing the focus away from the actual problem.

Hence, using an XML database makes it possible to avoid some of the drawbacks with RDBMS and file system storage. Data can be represented as XML through the whole chain as Xindice is optimized for managing XML. The use of an XML database is also likely to enhance performance and native XML search is obtained as a “bonus”. Other reasons for choosing Xindice is the supports for the Document Object Model (DOM) as well as the XPath query language. Xindice is schema independent, which means that it is not sensible to a specific XML schema. A drawback is however that there is a small risk to cause unwanted dependencies to third party software in a way that makes the system less general. This is however considered to be acceptable in an experimental study, as it doesn’t affect the results of what is the objective of the study. On the contrary the choice of a native XML database can be seen as a way to eliminate unnecessary complexity, making the principles to be proven stand out clearer.

It is however important to point out that the basic principles and methodology applied in this study are not depending on the use of an XML-database and that it is probably wise to make the application independent of the type of XML storage in the long run.

b) RDF metadata store

THE CHOICE OF RDF

There are several reasons (besides being a consequence of Annotea) for choosing RDF for metadata representation: there is a need for a uniform way to represent metadata, that can cover different metadata models and Application Profiles; RDF adds machine readable (and “understandable”) metadata semantics, which is also why RDF is a core component of the Semantic Web. Today’s web is only suitable for human understanding and the information makes no “meaning” to computers. The meaning for humans is tied to the presentation layer, which is mixed with the data to be presented. The relations between different pieces of information are expressed using context and hyperlinks that offers no semantic meaning for computers. Machine-readable (and “understandable”) semantics creates the possibility to use metadata from different sources for advanced reasoning and filtering of information, making it possible to associate metadata and extract new (semantic) relations in otherwise unrelated information, as described in [1].

The basic principles of RDF are quite simple and straightforward: *Someone* makes a *statement* about *something* (a *resource*), using different *properties*. RDF expressions can be made in several ways. Example 1 illustrates how “*Fredrik Paulsson is the creator of http://www.frepa.org/*” is expressed using a RDF graph.

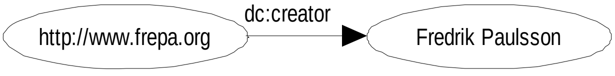

Figure 2. “Fredrik Paulsson is the creator of http://www.frepa.org/” expressed using RDF

The Dublin Core metadata model (the dc:creator element) [2] is used to specify the property, expressed as a literal in the statement of the property-value pair. Example 2 shows that statements can be made using a resource (and its properties) as well. This makes it possible to add more complex semantics as well as reusing statements. If the information at the URI (in this case http://www.frepa.org/creator) changes, this will be reflected in the semantics of the expression that uses the original expression. In example 2, the vCard standard [29] is used to describe the creator of a website. This makes it possible to add more useful information in an easy and standardized way - information that can be used and reused in other contexts and by other systems.

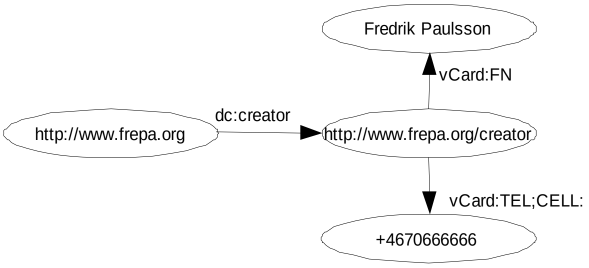

Figure 3. The same expression as in example 1, using a vCard instead of a literal.

Sophisticated semantics can be expressed by combining several, relatively simple statements, into “semantic nets” as shown in example 3.

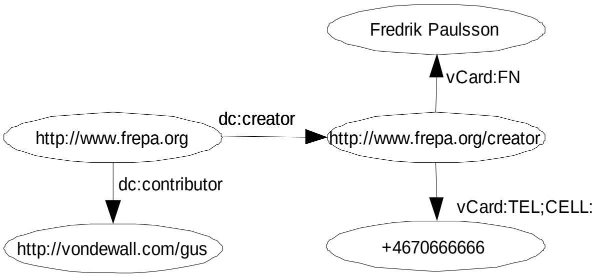

Figure 4. An example of how two statements: “Fredrik Paulsson is the creator of http://www.frepa.org/” and “Gustaf von Dewall is a contributor to http://www.frepa.org/”, are combined into a semantically richer statement.

Expressions must however be made using a machine-readable syntax in order for systems to actually make use of the semantics. RDF/XML is the XML representation of RDF, and the standard way to represent RDF in a system. The use of RDF/XML makes adds some important XML features to RDF, such as namespace awareness and schema awareness. RDF/XML is the standard way for SCAM to manage RDF.

c) SCAM RDF repository

There are only a few RDF repositories to choose from and to keep the consistency and flexibility of using RDF it is advantageous to choose one that uses RDF native, without mapping it to a proprietary data model. SCAM uses RDF as its native data format [26] and the main purpose of SCAM is to serve as a framework and development platform for RDF-based repositories [27]. SCAM uses a standard RDBMS and Jena [30], which is a Java API for managing RDF/XML in relational databases. As described in [30] Jena is based on the concept of RDF models and takes care of the complex task of writing RDF in RDF/XML as well as parsing XML using DOM or SAX. An RDF model is viewed as a set of statements that forms a RDF graph, where very statement belongs to a specific model. Jena provides APIs for manipulating RDF models by implementing the most common operations. One useful concept introduced by Jena is selectors. Selectors can be used to find all properties of a specific type in a RDF model, such as finding all vCard properties. Selectors are potentially useful when working with several metadata models, which is often the case with the methodology presented in this article. Jena can be used for persistent storage in databases as well as for in memory storage - that is when the model is stored in RAM. As a complement to the Jena API, and in order to facilitate the development of web based applications, SCAM implements a set of utility classes called “Drutten”.

SCAM uses the WebWork[3] MVC-framework and a presentation layer based on Apache Velocity[4]. The use of established third party frameworks leads to enhanced stability as well as a cleaner, layered architecture. The later is a consequence of third party frameworks need to be better separated from other layers as it has its own code-base.

The RDF Data Query Languages (RDQL) is used as the primary query mechanism in SCAM. RDQL is (like Jena) based on the idea that a RDF model is a set of triplets. A RDQL expression consists of a list of triple patterns, where each triple pattern is comprised of named variables and RDF values (URIs and literals). RDQL expressions may contain a list of constraints on these triple patters, such as a (partial) string match on a literal, an URI match on a resource, or a property. Listing 2 gives an example of a simple RDQL query.

```
SELECT ?sub
WHERE (?sub,
<http://www.w3.org/1999/02/22-rdf-
syntax-ns#type>,
       <http://www.example.com/exampleTy
pe>)
```

Figure 5. Listing: RDQL query that selects all subjects that have a http://www.example.com/exampleType as RDF type. The variable “?sub” is bound to the subject and the resulting model will only contain a list of subjects (without any properties at all).

RDQL expressions are relatively powerful when querying complex graphs, but there are some drawbacks as well: RDQL has no support for querying literals typed in a specific language (it is for example impossible to match a Swedish literal in an Alt of different translations). RDQL has no support for ordering the resulting set of triplets and in certain complex models it is complicated to do a full-text query since RDQL cannot return anonymous nodes as subjects in a result set. The choice of RDQL is basically a consequence of using Jena.

d) Application layer

The “application layer” is actually several layers of application logics and much of it has been described in previous sections. Functionality for querying XML-documents is provided by Xindice and SCAM/Jena provides the functionality for querying and managing RDF-metadata. Besides serving the complete XML-file, Xindice is used to filter out parts of an XML-file, such as for retrieving a fragment for a given XPath-pointer.

The functionality implemented using SCAM and Xindice are restricted to search and filtering of data and metadata, and writing of data and metadata the repositories. Still it handles some of the most complex and important functionality of the system. Session management, as well as user and security management is handled by SCAM through integration with the SCAM ePortfolio[5] system. The SCAM ePortfolio is an ePortfolio system built on top of the SCAM framework [26]. The ePortfolio provides a user-friendly environment for storing, organizing, managing and sharing digital resources and associated metadata - including annotations.

The SCAM ePortfolio act as an Annotea client (see figure 2) that retrieves the document from the XML store and then issues a request for all annotations associated to the document. The request is a standard Annotea compliant request [31], translated into a SCAM query. The resulting metadata record is sent back to the Annotea client and the annotations are merged into the XML document. The XML document is inserted into an XHTML “skeleton” that enables the use of ECMA-script on the resulting page.

When the XML-file and its corresponding style-sheet is sent to the browser, its DOM is determined using the XLink Processor (XLiP) [23]. The XLiP processing includes the generation of an XLink data model, a data model containing the DOM and the XLink data set that is determined from the DOM. The XLink data model is used for identifying the nodes for the link start-resource and the link end-resource that together makes up the locator information needed for “pointing out” the range of the XPointer. The range can either consist of an XPath node sub-tree, or a range with arbitrary start- and end points. An XLink data model may consist of more than one DOM, which is useful for managing data from more than one XML-file at a time. Only one DOM at a time is however used in the implementation described in this article. XLiP has several built in methods for generating and manipulating XPointer location information. The interaction within the browser is handled using a simple ECMA-script that determines which section is marked in order to pass the information on to the XPointer processor classes that returns the XPointer information needed to manage the metadata markup in the browser. When a user add or update an annotation, an ECMA-script in the browser issues a HTTP POST to the Annotea client containing the XPointer that points at the specified location. The Annotea client responds with a metadata form. The form is used to create or edit the underlying RDF structure. When the form is posted to the Annotea client, an add-operation is forwarded to the Annotea server that adds or updates the RDF graph in SCAM. The resulting metadata record, and the corresponding URI is constructed in accordance with the Annotea Schema [31]

[3] http://www.opensymphony.com/webwork/

[4] http://jakarta.apache.org/velocity/index.html

[5] http://scam.sourceforge.net/

where every annotation has its own XPointer. The metadata record is stored in the SCAM repository.

The annotated resources can be used for advanced metadata services; such as the service, which semantically connects the Swedish national curriculum to digital learning resources described in [1].

2) Presentation layer

The presentation layer is obviously very important for the usability of the system as well as for the over all user experience. Usability was however not a primary target for this study and the methodology and principles that were implemented and tested works independently of the presentation layer. For those reasons, the presentation layer is only described briefly.

The presentation layer is built on top of the SCAM ePortfolio. The main reasons for this are that the SCAM platform is already used as the RDF-store and that the ePortfolio component adds a user interface and a user context that is intuitive and easy to use. Taken together, this results in a familiar user context and a well-known metaphor (the ePortfolio) for storing and managing digital resources. The SCAM ePortfolio uses the Apache Velocity templating engine[6] for laying out the user interface and this makes the integration straightforward. The integration with SCAM ePortfolio is not necessary for the methodology to work, but it provides an easy opportunity for non-expert users to upload and manage XML-documents in an on-line, web-based environment for document- and metadata management.

III. Results

The implementation described in this paper clearly shows that it is possible to accomplish a separation between content and structural markup, using the suggested methodology, and based on existing standards and recommendations. The methodology of “pointing in” metadata as annotations proves to be a fairly stable and general way to accomplish the wanted separation.

Even though the implementation is still on an experimental stage it points in the direction of a working methodology and architectural solution for separating data and metadata as well as adding to a “metadata ecology”. Some of the third party frameworks and applications that where used in order to avoid some of the most complex part of the implementation should preferably be exchanged to avoid dependencies, such as the dependency of an XML database and the binding to the XLiP, which is free, but not Open Source and by that somewhat hazardous to rely on. Those frameworks do however not affect the results presented in this article.

IV. Discussion

The methodology for distributed metadata markup that is presented in this article differs from the traditional view of metadata and textual markup. We argue that structural markup of content (data) and metadata (semantic) markup benefits from being separated and that such separation is the only feasible way to leave the original data untouched and by that preserving the original semantics.

It is also our view that such separation is necessary in order to establish reasonable conditions for a metadata ecology where a large number of contributors can contribute with their descriptions, markup, and models in a way that carries, not only machine readable, but has also machine “understandable” semantics. The methodology presented in this article challenges the traditional view of metadata in the respect that it provides a much more un-authoritarian and open (possibly more democratic) model where metadata may reflect many different views in many different contexts and therefore cannot be regarded as always being “objective”. Metadata can be completely distributed and there is no means of controlling all metadata from a central point. As long as the content is open, it will be possible for anyone to contribute with metadata. This opens up a range of possibilities as well as resulting in a lot of questions. Issues such as trust and weighting of metadata are such questions. Whose metadata can be trusted and do we need mechanisms for weighting metadata in different ways so that metadata some types of metadata or metadata provided by some actors are “worth” more that metadata from other actors? The trust issue must be addressed in a way that preserves the openness and democratic aspects of the metadata ecology. There are emerging technologies for social tagging and social networking [32] [33], such as the OWL based Friend Of a Friend format (FOAF), that could be explored as way of addressing this issue.

There are still a couple of unsolved problems to be addressed in our future work. One such problem is the implementation of the application logics in the browser. Currently there is a dependency to a JavaScript that handles the interaction with the XML-file and the connection to the XLiP classes. This does however not affect the methodology as such, but makes it impossible to use the Annofolio to manage metadata outside our system and is not a sufficient long-term solution. There are however some advantages to work in a controlled environment for the purpose of the study to gain better control of the implementation of the methodology. An alternative path could be to implement the client functionality through a plug-in, an approach that could also be used to solve another problem: namely the varying support for XML-technologies in different browsers. This would however include adding more of the XML specific functionality into the Annofolio prototype, and by that making the implementation much more complex and browser specific. An other solution could be to develop an AJAX-based (Asynchronous JavaScript and XML) [34] client for the Annofolio system. The AJAX technology would eliminate most platform or browser dependencies, as well as providing the same flexibility and several of the advantages of a plug-in. An AJAX-based client would also be suitable for implementation in the current Annofolio architecture.

Another important issue that needs further attention is the reliability and stability to changes of XML-files. There is an impending risk that XPointers changes, or that an existing XPointer points at the wrong section of the content if there are many and extensive changes to the XML-file. Some tests where carried out in [1] that showed that minor changes is normally not a problem, depending on where they occur in the file and where the changes occur in relation to existing semantic markup. The reason for those dependencies is that the XPointers are determined from the closest XML-tag. This is clearly a weakness in the methodology, but only minor tests where carried out in [1], as this was not a real issue as the

[6] http://jakarta.apache.org/velocity/index.html

national steering documents are version managed and quite stable. The results from the NCM tests where a bit ambiguous and this issue must be thoroughly examined in future research.

The methodology presented in this article opens up interesting opportunities for non-textual resources as well. It is (at least in theory) possible to use this methodology for all kinds of resources that are expressed using XML. This means that other resources, such as graphics or video could very well be targeted and subject to the same type semantic markup. Non-textual resources are one of our main objectives for the future.

V. References

[1] F. Paulsson, and J. Engman, "Marking the National Curriculum - a new model for semantic mark-up," (2005, 19-21 October). In the proceedings of *eChallenges 2005*, pp. 1731-1738, Ljubljana, Slovenia.

[2] *Dublin Core Metadata Element Set, Version 1.1: Reference Description*. (Reference Description): Dublin Core Metadata Initiative, 2004.

[3] (2002). Final 1484.12.1-2002 LOM Draft Standard. IEEE: IEEE-Standards Association.

[4] P. M. Heery R, "Application profiles: mixing and matching metadata schemas - introduce the 'application profile' as a type of metadata schema," *Arriadne, vol.* (25), pp., September 2000 2000.

[5] *CELEBRATE Application Profile*. Retrieved August 10, 2006, from http://www.eun.org/ww/en/pub/celebrate_help/application_profile.htm

[6] *IMS Learning Resource Meta-Data Best Practice and Implementation Guide. Version 1.2.1 Final Specification.* (No. Version 1.2.1): IMS Global Learning Consortium, Inc., 2001.

[7] E. Duval, W. Hodgins, S. Sutton, and S. L. Weibel, "Metadata Principles and Practicalities," *D-Lib Magazine, vol. 8*(4), pp., April, 2002 2002.

[8] K. Nikam, "Metadata for Managing Internet Resources," (2002, 13-15 March). In the proceedings of *Workshop on Information Resource Management*, DRTC, Bangalore.

[9] N. Ide, and J. Véronis. (1995). *Text encoding initiative : background and context*. Dordrecht: Kluwer Academic.

[10] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic Web - A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities," *Scientific American, vol.*, pp., May 2001 2001.

[11] P. Hayes, *RDF Model Theory*. Retrieved February 14, 2003, from http://www.w3c.org/TR/rdf-mt/

[12] L. Burnard, and C. M. Sperberg-Mcqueen. (1990). *Guidelines for the encoding and interchange of machine-readable texts* (Draft ed.).

[13] *What Is DocBook?* Retrieved May 9, 2006, from http://www.docbook.org/oasis/intro.html

[14] D. Alberg, S. Davis, P. Grosso, P. Boutros, J. Chelsom, J. Harrop, et al., *Open Document Format for Office Applications (OpenDocument) v1.0*. 1.0, Retrieved May 10, 2006, from http://www.oasis-open.org/committees/download.php/12572/OpenDocument-v1.0-os.pdf

[15] Text Encoding Initiative (Projekt), L. Burnard, and C. M. Sperberg-McQueen. (1994). Guidelines for electronic text encoding and interchange (pp. xxvi, (1290)). Chicago: TEI.

[16] J. E. Simpson. (2002). *XPath and XPointer : locating content in XML documents* (1. ed.). Beijing ; Cambridge ; Farnham ; Köln ; Paris ; Sebastopol ; Taipei ; Tokyo: O'Reilly.

[17] M.-R. Koivunen, R. Swick, K. José, and E. Prud'Hommeaux. (2002). Annotea: Metadata based annotation infrastructure. Honolulu,: W3C, presented at WWW2002 Developers Day.

[18] W3C, *Annotea Annotation Schema*. Retrieved August 10, 2006, from http://www.w3.org/2000/10/annotation-ns#

[19] S. Dill, E. Nadav, D. Gibson, G. Daniel Gruhl R, A. Jhingran, T. Kanungo, et al., "SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation," (2003). In the proceedings of *International WWW conference*, Budapest, Hungary.

[20] F. Ciravegna, A. Dingli, D. Petrelli, and Y. Wilks, "User-System Cooperation in Document Annotation based on Information Extraction " (2002, 1-4 October 2002). In the proceedings of *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW 02)*, Sigüenza, Spain.

[21] S. Handschuh, S. Staab, and F. Ciravegna, "S-Cream, Semi-automatic creation of metadata," (2002, 1-4 October 2002). In the proceedings of *European Conference on Knowledge Acquisition and Management*, Sigüenza, Spain.

[22] N. Walsh, (ed.). (2004). The Simplified DocBook Document Type, *Working Draft 1.1CR1*, : The Organization for the Advancement of Structured Information Standards [OASIS].

[23] *XLiP (XLink Processor) User's Guide*. Retrieved May 10, 2006, from http://project.cins.se/docs/xlip/

[24] A. Le Hors, P. Le Hégaret, L. Wood, G. Nicol, J. Robie, and M. Champio, *Document Object Model Level 2 Core*. 1.0, Retrieved 2006-05-10, 2006, from http://www.w3.org/DOM/DOMTR#dom2

[25] K. Staken, and D. Viner. *Xindice 1.1 Developer Guide*. (Developers Guide): The Apache Software Foundation, 2005.

[26] M. Palmér, A. Naeve, and F. Paulsson, "The SCAM Framework: Helping Semantic Web Applications to Store and Access Metadata," (2004). In the proceedings of *the European Semantic Web Symposium 2004*, Heraclion Greece.

[27] F. Paulsson, "Standardized Content Archive Management – SCAM," *IEEE Learning Technology newsletter, vol. 5*(1), pp. 40-42, January, 2003 2003.

[28] I. Tatarinov, S. D. Viglas, K. Beyer, J. Shanmugasundaram, E. Shekita, and C. Zhang, "Storing and querying ordered XML using a relational database system " (2002, June 3-7). In the proceedings of *the 2002 ACM SIGMOD international conference on Management of data*, pp. 204-215, Madison, Wisconsin.

[29] F. Dawson, and T. Howes. *IETF RFC 2426: vCard MIME Directory Profile*. (RFC No. 2426): IETF, 1998.

[30] B. McBride, "Jena: Implementing the RDF Model and Syntax Specification," (2001). In the proceedings of *Semantic Web Workshop, WWW2001*,

[31] R. Swick, E. Prud'hommeaux, M.-R. Koivunen, and J. Kahan, *Annotea Protocols*. from http://www.w3.org/2001/Annotea/User/Protocol.html

[32] G. Avram, "At the Crossroads of Knowledge Management and Social Software " *Electronic Journal of Knowledge Management vol. 4*(1), pp., January 2006 2006.

[33] G. Macgregor, and E. McCulloch, "Collaborative tagging as a knowledge organisation and resource discovery tool," *Library Review, vol. 55*(7), pp. 291-300, 24 February 2006 2006.

[34] L. D. Paulsson, "Building rich web applications with Ajax," *IEEE Computer, vol. 38*(10), pp. 14-17, October 2005 2005.

VI. Authors

F. Paulsson is with the Department for Interactive Media and Learning at Umeå University, SE-901 87 Umeå, Sweden, and associated with the Knowledge Management Research Group at the Royal Institute of Technology (e-mail:Fredrik.paulsson@educ.umu.se).

J. Engman was with the Centre for IT in Northern Sweden. He is now with Sogeti AB, in Umeå. (e-mail: jonas@cins.se).

Manuscript received 05 September 2006. This work was supported by Vinnova.