A Superimposed Architecture for Enhanced Metadata

Extended Abstract

Mathew Weaver, Lois Delcambre, David Maier
Computer Science and Engineering Department
School of Science and Engineering
Oregon Health and Science University
Portland, Oregon
{mweaver, lmd, maier}@cse.ogi.edu

1 Introduction

Imagine you're preparing an environmental impact statement for a proposed highway in the Willamette National Forest and you're particularly interested in assessing the impact on Douglas Fir trees, a species native to the area. You're interested in information from similar projects conducted in a similar environment. You may benefit from a wide range of information including: existing environmental impact statements, watershed assessments, scientific studies and surveys, records of decision and so forth. Your task is a typical information gathering task. We propose an enhanced architecture for metadata – Metadata^{++} – where metadata is represented as explicit objects and where explicit relationships among terms and properties are exploited to maximize search capabilities, minimize metadata entry requirements, and support a rich form of similarity search.

One popular search mechanism based on techniques from information retrieval retrieves documents by matching search keywords or phrases with text found in electronic documents. The documents must be electronically accessible and processible. Another approach, from the knowledge acquisition community, focuses on making common knowledge explicit using an *ontology* [1]. By defining concepts (usually called classes) in terms of related concepts, the ontology supports the inference of new knowledge about concepts and related documents and thus serves as the gateway to accessing the documents. The digital library community presents a third popular approach to document management and searching based on metadata fields and values, as shown in Figure 1. The Dublin Core [2] defines standard metadata fields. Traditional metadata supports searches based on field-value queries. Such an interface might allow you to choose the "Location" field and enter a value of "Willamette National Forest." The query would then return those documents explicitly associated with "Willamette National Forest."

2 Metadata^{++}

Metadata^{++} is designed to help you find documents but it places few requirements on the documents themselves because Metadata^{++} attaches metadata to *document proxy* objects, as represented by the document-shaped symbol in Figure 2. The document can be in any format and in any location. Searches return relevant document proxies and each proxy provides information about how to retrieve the document, e.g., using a URL or a phone number of someone to call. Having documents exist outside of the system is an important advantage of the Metadata^{++} architecture. A national forest agency may maintain its own web server with online documents [3], while a research laboratory may have its own database of scientific datasets [4]. By superimposing Metadata^{++} over multiple, existing systems, you get the added value of using Metadata^{++} without little or no disruption.

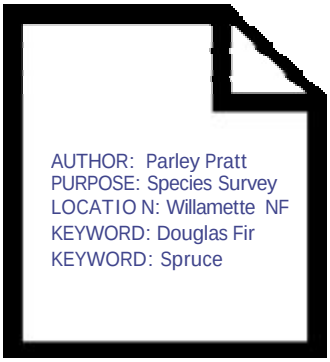

Figure 1: Traditional Metadata

The key difference between Metadata^{++} and other approaches is that values from traditional metadata are represented as explicit objects called *terms*. For example, in Figure 1 'Parley Pratt' is a string value that appears in one or more metadata records (once for each document he authored). But in Metadata^{++}, `Parley Pratt` is a single object associated with one or more document proxies, as shown in Figure 2. Similar to using an index, you simply

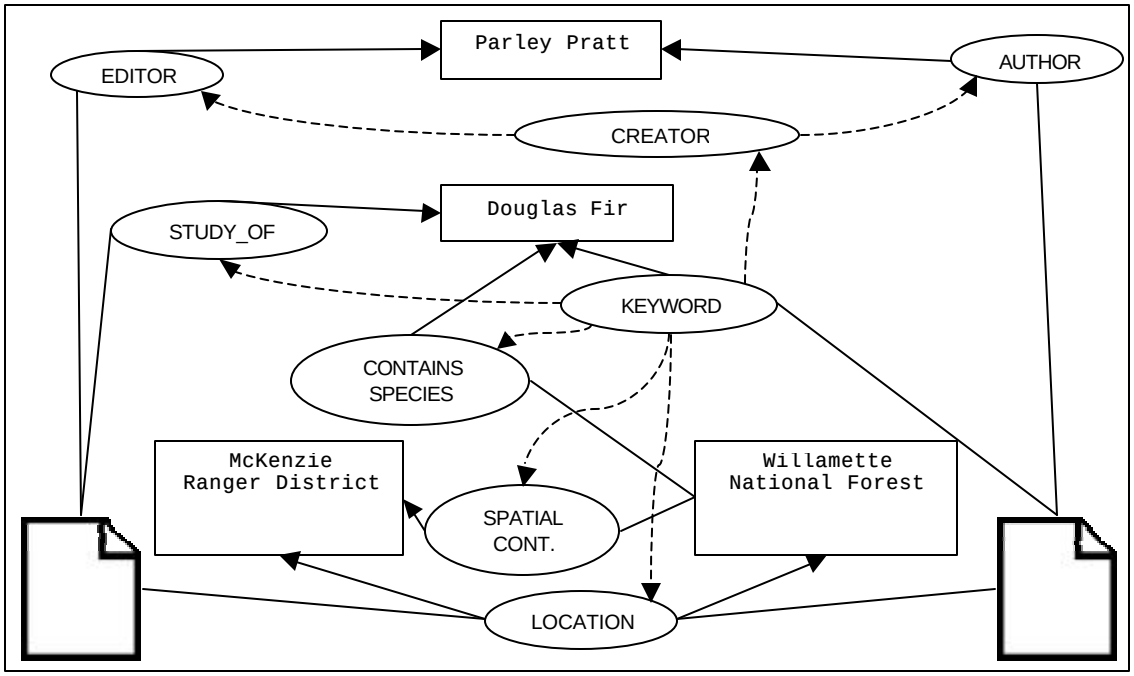

Figure 3: Terms, properties, document proxies, and the hierarchy of properties in Metadata^{++}
[Term = rectangle; Document proxy = document icon; Property = solid line; Hierarchy of properties = dashed line]

ask the `Parley Pratt` object for all of its associations – which will take you to the relevant documents.

Besides document proxies and terms, Metadata^{++} also represents *properties* explictly as objects. Properties are used to associate documents with terms, as with the LOCATION property in Figure 2, and to associate documents with other documents, as with the CONTAINS SPECIES property in Figure 2.

Metadata^{++} also allows terms to be associated with terms via properties, e.g., using the SPATIAL_CONTAINMENT property to associate `Willamette National Forest` with `McKenzie Ranger District`. As another example, `Willamette National Forest` is associated with `Douglas Fir` using the CONTAINS_SPECIES property, indicating that Douglas Fir trees grow in the Willamette National Forest.

The most distinctive aspect of Metadata^{++} is the ability to explicitly relate properties using properties. Metadata^{++} hierarchically relates properties. In Figure 2, CREATOR is a more general property than the EDITOR property and the AUTHOR property. Thus CREATOR is placed higher in the hierarchy than EDITOR and AUTHOR.

Metadata^{++} separates and relates properties so that you can be precise in describing the content of the document.

3 Searching

Metadata^{++} exploits the connections among metadata terms and properties to perform an extensive search based on a simple query. For example, a query that mentions a single term can automatically find documents for associated terms. Suppose your task is to find all

information pertaining to Willamette National Forest. In addition to finding all documents that are explicitly related to Willamette National Forest – which could be done with a traditional metadata search – you also need to find documents associated with places within the forest, such as McKenzie Ranger District. Metadata^{++} will find all of the documents explicitly associated with `Willamette National Forest` and then use the associated SPATIAL_CONTAINMENT property to find the `McKenzie Ranger District` term – and then find all of the documents explicitly associated with `McKenzie Ranger District`.

$$M = (D, T, P, \Delta, \Phi)$$
$$D = \{\text{a finite set of document proxies}\}$$
$$T = \{\text{a finite set of terms}\}$$
$$P = \{\text{a finite set of properties}\}$$
$$\Delta(n, p) = G \text{ where } n \in (D \cup T), p \in P, G \subseteq (D \cup T)$$
$$\Phi(p) = Q \text{ where } p \in P, Q \subseteq P$$

Figure 3: Metadata^{++} Formalization

Finding the same set of documents with traditional metadata would be more complex. One approach would be to issue a more complicated query that uses boolean operators to combine results from simpler subqueries but this approach does not scale well; the complexity of the query increases linearly with the number of relevant terms. A second traditional approach is to relate each document with all relevant terms. Associating each document with all relevant terms increases the effort required to create metadata and new terms would require that new metadata be associated with existing documents.

In addition to simplifying traditional searches, Metadata^{++} enables elaborate similarity searches. For the forester introduced above, how do you find similar forests? Or how do you find studies that measure the impact of highways on forests? One approach would be to perform complex data mining algorithms to find co-occurrences of values. Metadata^{++} allows you to explicitly associate terms – making it easier to compute similarity. As shown in Figure 2, the `Douglas Fir` term is explicitly associated with `Willamette National Forest` using the CONTAINS_SPECIES property (because Douglas Fir trees are native to the forest). By navigating the explicit associations between terms, properties, and documents, Metadata^{++} will help you to quickly and easily find similar forests – and documents about those forests.

4 Formalization

The formal representation of Metadata^{++} is a five-tuple as illustrated in Figure 3. The first three elements are finite sets of objects. The set D is a set of document proxies. The sets T and P contain terms and properties respectively. The function ? represents the associations between document proxies, properties, and terms. The argument n may be either a document proxy or a term and the argument p is a property. The result is a set containing all document proxies and terms associated with n via the property p. The function F represents the hierarchical relationship between properties. The argument p is a property. The result is a set of properties that are children of p in the hierarchy.

A formal Metadata^{++} query includes the initial property to use when finding documents and how many levels of the hierarchy to traverse to find related properties. Additionally, the query specifies the initial term, as well as the property to use to find associated terms. The sample query in Figure 4 will find all documents associated with `Willamette National Forest` using the LOCATION property – as well as documents associated with places within the forest using the SPATIAL_CONTAINMENT property. The query evaluation shown in Figure 4 uses two additional functions, F^ and ?^, that are derived from the formal model definition. These functions are defined and explained in [5].

5 Related Work

Staab et al. [6] present semantic community web portals based on the Ontobroker [7] system. Their approach focuses on a single ontology that represents the shared knowledge of the community. Concepts (*terms* in Metadata^{++} are explicitly represented in the ontology and documents are related to concepts. The ontology is defined in F-Logic [8]. The query capabilities of the semantic portal include predefined queries, an ontology browser, and explicit F-Logic queries. Metadata^{++} explicitly represents terms, documents, and properties – and supports any number of user-defined relationships between these objects. Specifically, the relationships between properties do not seem to be supported in Ontobroker. Additionally, Metadata^{++} does not require that all terms fit into a single ontology.

$P_Q = F \wedge (\ LOCATION, 0\ )$
$T_Q = ? \wedge (\ Willamette\_NF, spatial\_cont, *\ )$
$D_Q = \{\ \}$
$\forall p \in P_Q$
  $\forall t \in T_Q$
    $D_t = ?\ (\ t, p\ )$
    $D_Q = D_Q \cup (\ D_t \cap D\ )$

Figure 4: Sample Query Evaluation

Ambite et al. [9] use an ontology-based approach where multiple domains are accommodated by mapping each domain to an existing reference. Because Metadata^{++} uses interrelated objects, multiple domains are easily represented and terms can be related to other terms within the same domain as well as to relevant terms from other domains. The Ambite project does not explicitly represent properties – it contains a set of predefined relations used between terms. Metadata^{++} allows new properties to be easily created and related to existing properties.

Weinstein [10] uses an ontology focused on bibliographic concepts to generate and search metadata from Machine Readable Cataloging (MARC) records. Weinstein's approach uses a predefined ontology designed specifically for bibliographic data. The concepts are related with a predefined set of relationships. Metadata^{++} generically represents any domain and allows user-defined properties and relationships.

Motta et al. [11] focus on carefully defining the ontology to meet the needs of the users. Instead of annotating documents with metadata, they populate the ontology with documents. While it is important to intelligently choose terms and properties, Metadata^{++} gives you more flexibility. Instead of focusing on designing the ontology completely and correctly the first time, Metadata^{++} allows terms and properties to be created and related as you go along. When a new term is created, it can be related to existing terms – eliminating the need to re-create metadata for existing documents in reference to the new term.

The Simple HTML Ontological Extensions (SHOE) project [12] allows users to annotate web pages with metadata based on one or more ontologies. SHOE uses metadata that is stored within web pages. The metadata is read by a crawling agent and used to answer queries. Because it is an extension to HTML, it is focused primarily on HTML documents. Metadata^{++} makes no stipulations about what type of documents can be used in the system.

Chung et al. [13] apply sophisticated statistical algorithms to infer relationships between terms automatically extracted from an existing domain. Their focus is implementing the algorithms to efficiently process very large domains. Metadata^{++} is not designed to automatically infer relationships between terms. Some relationships between terms (i.e. Douglas Fir trees grow in Willamette National Forest) are unlikely to be inferred by statistical algorithms.

The semantic networks model [14] developed several years ago is similar to Metadata^{++}. This model used nodes and links to define natural languages by linking words and phrases to capture semantic meaning. Metadata^{++} supports terms and documents associated by properties – as opposed to linked nodes. Additionally, Metadata^{++} is designed to capture semantic metadata – as opposed to capturing the semantics of natural language.

6 Conclusion

By using explicit objects, Metadata^{++} builds a framework within which meaningful queries can be quickly and effectively executed. Our preliminary prototype is based on forest information – as part of a Digital Government project [15] funded by the NSF – but the architecture is applicable to any domain. Our feedback from potential end users includes is very positive and they look forward to additional prototypes – and a deployable system. Future work includes extending the query language and designing an intuitive user interface that exploits the Metadata^{++} architecture. Instead of designing an algorithm to compute a relevance score for a retrieved document, we intended to explicitly display to the user which terms and relationships where considered when retrieving the document.

7 References

1) I. Horrocks, D. Fensel, C. Goble, F. Van Harmelen, J. Broekstra, M. Klein, and S. Staab. *The ontology inference layer OIL.* Technical report, Free University of Amsterdam, 2000. www.ontoknowledge.org/oil/.
2) Dublin Core Metadata Initiative, http://dublincore.org
3) USDA Forest Service, Online Library, http://www.fs.fed.us/library/index.html
4) H.J. Andrews Experimental Forest Publications, http://www.fsl.orst.edu/lter/pubsfr.htm
5) Mathew Weaver. *Metadata^{++}: A Superimposed Architecture for Semantic Searching.* Oregon Graduate Institute Research Proficiency Exam. www.cse.ogi.edu/~mweaver/rpe/metadata.pdf
6) S. Staab, J. Angele, S. Decker, M. Erdmann, A. Hotho, A. Maedche, H.-P. Schnurr, R. Studer, and Y. Sure. *Semantic community web portals*. Computer Networks (Special Issue: WWW9 – Proc. of the 9th International WWW Conference, Amsterdam, The Netherlands, May, 15-19, 2000), pages 473-491.
7) Stefan Decker, Michael Erdmann, Dieter Fensel, Rudi Studer. *Ontobroker: Ontology based access to distributed and semi-structured information*. In R. Meersman et al., editor, DS-8: Semantic Issues in Multimedia Systems. Kluwer Academic Publisher, 1999.
8) M. Kifer, G. Lausen, and J. Wu. *Logical Foundations of Object-Oriented and Frame-Based Languages*. Journal of the ACM 42. pages 741-843. 1995.
9) Jose Luis Ambite, Yigal Arens, Eduard Hovy, Andrew Philpot, Luis Gravano, Vasileios Hatzivassiloglou, Judith Klavans. *Simplifying Data Access: The Energy Data Collection Project*. IEEE Computer (February 2001, pages 47-54).
10) Peter C. Weinstein. *Ontology-Based Metadata: Transforming the MARC Legacy*. Proceedings of the Third International ACM Conference on Digital Libraries, Pittsburgh, Pennsylvania, 1998.
11) Enrico Motta, Simon Buckingham Shum, John Domingue. *Ontology-Driven Document Enrichment: Principles and Case Studies*. Workshop on Knowledge Acquisition, Modeling and Management, 1999.
12) Heflin, J., Hendler, J., and Luke, S. Reading Between the Lines: Using SHOE to Discover Implicit Knowledge from the Web. In *AI and Information Integration. Papers from the 1998 Workshop*. WS-98-14. AAAI Press, 1998. Pages 51-57.
13) Yi-Ming Chung, Qin He, Kevin Powell, Bruce Schatz. *Semantic indexing for a complete subject discipline.* Proceedings of the Fourth ACM Conference on Digital Libraries, Berkeley, CA, August 1999.
14) Nicholas V. Findler, ed. *Associatve Networks: Representation and Use of Knowledge By Computers*. Academic Press, Inc. New York. 1979.
15) Harvesting Information to Sustain our Forests. Digital Government NSF Grant, July 2000. http://www.diggov.org/about/GrantRecipients/granteeDetails.cfm?id=14