Introducing Shadows: Flexible Document Representation and Annotation on the Web

Matheus Silva Mota #1, Claudia Bauzer Medeiros #2

# *Institute of Computing, University of Campinas (UNICAMP)*
Campinas, SP, Brazil
1 mota@ic.unicamp.br
2 cmbm@ic.unicamp.br

***Abstract*—The Web is witnessing an exponential growth of increasingly complex, distributed and heterogeneous documents. This hampers document exchange, as well as their annotation and retrieval. While information retrieval mechanisms concentrate on textual features (corpus analysis), annotation approaches either target specific formats or require that a document follows interoperable standards. This work presents our effort to handle these problems, providing a more flexible solution. Rather than trying to modify or convert the document itself, or to target only textual characteristics, the strategy described in this work is based on an intermediate descriptor – the *document shadow*. A shadow represents domain-relevant aspects and elements of both structure and content of a given document, as defined by a user group. Rather than annotating documents themselves, it is the shadows that are annotated, thereby providing independence between annotations and document formats. Our annotations take advantage of the LOD initiative. Via annotations users can derive correlations across shadows, in a flexible way. Moreover, shadows and annotations are stored in databases, therefore allowing uniform database treatments of heterogeneous documents.**

I. Introduction and Motivation

The Web has become a huge platform for document authoring. The proliferation of document formats is a result of both specific environments and multiplication of authoring tools. In most cases, such tools have not been conceived to produce files with explicit structure and interoperable formats, strongly coupling the content to the file structure and software representation [1], [2].

Document management and retrieval systems use three main strategies to deal with large volumes of complex and heterogeneous documents [3], [4], [5]. The first strategy supports only some specific file format, making it necessary to convert the original document to the supported format. The second strategy requires documents that follow interoperable standards (e.g., XML). The third strategy considers a document to be a general digital artifact, supporting only metadata and requiring user assistance. The first strategy presents problems when preservation of the original file is needed. In strategy two, the main difficulty is to handle format diversity, since interoperable formats and predefined schemes are a prerequisite. On the other hand, approach three deals very well with file format diversity, but provides limited support to indexation, retrieval and annotation.

This work presents *Shadow-driven Representation* (SdR), a novel strategy to represent documents independently of format, preserving the original file and handling large volumes of documents. A *shadow* is a document descriptor that summarizes key aspects and elements of a document, preserving their structural relationships. These elements (e.g., sections, tables, embedded multimedia artifacts) are defined by users (e.g., research groups may have different interests). Unlike other approaches to document summarization and description, our work considers that it is up to the end-user to specify the elements of interest in a set of documents, and thus one document may have many shadows. Once a set of elements of interest is defined, shadows are instantiated automatically, based in this set. Unlike other approaches in the literature that restrict document summaries to text, shadows consider other kinds of elements within a document, such as tables or images, thereby supporting a wide variety of operations and correlations. Though we have implemented shadows as XML documents stored in a database (our shadow base), this is just a possible materialization of the concept, which allows querying documents using DBMS.

This paper is organized as follows. Section II introduces concepts and related work. Section III presents a detailed explanation of SdR. Section IV presents our implementation of SdR. Section V presents a case study where we use shadows to allow semantic annotations of documents in the biodiversity context. Section VI presents conclusions and ongoing work.

II. Concepts and Related Work

A. Resource Descriptors

The representation strategy presented here is inspired by the concept of *resource descriptors*. *Descriptors* are structures that summarize aspects of some digital object in order to help its indexing, comparison and retrieval [6]. More specifically, our representation strategy is inspired on the concept of descriptors borrowed from two research fields: image management and metadata standards.

1) Metadata and Metadata Standards: Metadata can be seen as a high level description of data. Metadata, or meta-information, is a structured information and regulatory tool to explain, locate, identify, describe and provide semantic increment to resources, helping users or management tools [7]. Different domains and needs require different metadata vocabularies[8], [9]. As presented in Section IV, we use a set of metadata standard initiatives to define types of document elements in a shadow.

2) Image Descriptor: An Image Descriptor is a data structure that summarizes the content of an image. According to [6], an image descriptor can be defined as a pair composed of a feature vector and a distance function. The feature vector represents a set of properties (e.g, shape, color, texture) extracted from the images. The distance, or similarity, function is used to compare feature vectors through a specific metric.

To extract visual properties, image processing algorithms usually focus on specific characteristics of an image and mainly follow two steps: (i) points of interest are identified and pass through a feature extraction process; and (ii) values are computed based on each point of interest, according to the type of information that needs to be extracted or recognized [10]. Image descriptors have two advantages: (i) the features extracted can be stored for subsequent processing; and (ii) different image descriptors (e.g., based on color, shape etc.) can be combined, implying on scalability [11].

Image descriptors are particularly helpful in understanding the SdR approach. Rather than looking for matches of metadata or annotations, or opening a document to extract specific characteristics – which is the usual approach in document management systems –, the SdR strategy pre-processes and extracts points of interest (key elements) of a document. Then, based on the extracted features, we generate the shadow, an intermediate structure that describes the document. Subsequently, we can annotate parts of a document via its shadow or perform a shadow-based search.

B. Semantic Annotations and Linked Data

The concept of Semantic Annotation is derived from the textual annotation concept. Such annotations can have different objectives [12] and be produced [13], [14] and structured in many forms (e.g., links, free remarks, tags, floating layers etc). Annotations are used, among others, to describe a resource, its relations and what it represents. Informal annotations are usually inserted on documents for human consumption. This hampers computer processing and annotation exchange.

Semantic annotations appeared with the purpose of third-party interpretation, providing explicit and machine interpretable semantics, as supported by Semantic Web standards [15], [13]. As will be seen in Section V, instead of annotating documents, we annotate shadows, taking advantage of the LOD[1], thereby concentrating all processing requirements on the shadows, again ensuring independence from specific document formats.

The use of the Linked Data paradigm has made data sharing on the Web easier and enhanced the possibility of aggregating like concepts, creating semantic clusters (e.g., [16]). We process our shadow base under the Linked Data principles, annotating documents via their shadows with DBPedia concepts. Thus, Shadows are used as a means to immerse documents in the Semantic Web.

[1]http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData

C. Document Management

Shadows describe documents, and thus must be compared with document description and extraction techniques found in the Information Retrieval (IR) and database literature. IR is primarily concerned with textual evidence. There are countless techniques to extract relevant keywords, concepts, sentences from a document in order to represent or describe it. Once representation structures are created, sets of documents can be clustered according to them – e.g., to correlate or summarize them. A set of documents can be summarized by a document, i.e., by the structure that represents a document, e.g., [17].

Another means to represent documents in IR is the use of metadata, such as author or title, often taking advantage of standards like Dublin Core. In particular, if documents are written in XML, then element tags can also be used (and in this case they are sometimes called *facets* [18]). Unlike IR approaches, we follow a database, data-centric, approach, in which shadows are stored and managed by XML database systems. Here, the emphasis is not on (semi)automatic extraction of information, but customizing user requirements and providing a flexible mechanism to immerse documents (and pieces of it) in the Semantic web. While IR techniques concentrate on mining structure and content from text, we are concerned with describing arbitrary documents under a user predefined structure, with semantics as goal.

III. Shadow-driven Representation

This work treats documents as special cases of complex objects [19], i.e., they are self-contained units, defining recursive hierarchical containment structures – e.g., a document contains sections, which contains subsections, which can contain an image or a table etc. From a conceptual point of view, shadows can be seen as document descriptors, in the sense that they describe structure and contents of a document according to the specification of a group of users. We stress that a document can have as many shadows as needed by distinct users.

1) Shadow Schema and Elements of Interest: Different domains may have different needs of document handling [2]. To illustrate that, consider a collection of scientific papers stored in different formats. Users of a domain P_1 may be interested in searching the collection by using parameters like title, authors and keywords. A different domain P_2 may be interested in more specific tasks over the collection, such as processing images inserted in the documents, or even correlate bibliographic references. The standard solution is to design and develop specialized applications.

In the SdR approach, instead of concerning themselves with IR-based solutions, users of both P_1 and P_2 need to define their *elements of interest*. In analogy to databases, we say that these elements define the shadow's *schema*, and that this schema is *instantiated* for the individual documents in the set.

Intuitively, a Shadow Schema specifies which elements should be recognized and represented in the corresponding shadow. The possibility of defining different subsets of elements and element levels makes the shadow representation scalable.

The generation of a shadow is divided in two steps: (a) Definition of elements of interest (the schema); and (b) instantiation of the shadow (for each document in a collection) based on the schema defined by users.

Shadows are XML documents. The shadow schema is thus a tree of labeled types of elements, corresponding to the elements of interest. Elements are associated with document content (e.g., table, paragraph, list of references) and can be related hierarchically as regards a document's structure (e.g., users may be only interested in captions of images, but not on captions of tables).

As presented in Section IV, for implementation purposes, element types are specified via metadata standards and/or namespaces in a pre-defined shadow schema vocabulary, and can preserve an element's hierarchical structure within a document d. The use of such standards in the schema definition allows processing documents according to distinct domain needs and vocabularies.

A *shadow* base is a data repository where shadows are stored. A set of documents D gives origin to one (or multiple) shadow base(s) B, in which shadows can be queried and processed independent of the original document format.

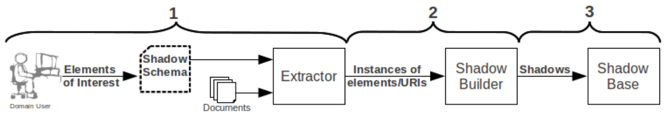

Figure 1. Overview of the shadow generation

Figure 1 illustrates the main steps for constructing a shadow base. First, users define the elements of interest and their hierarchical relationships, thereby specifying the schema. Next, the set of documents is processed by an $Extractor$ module that analyzes each document to recognize the elements of interest, forwarding their instances (or their URI's) to the $ShadowBuilder$ module. Finally, the $ShadowBuilder$ constructs the shadows and stores them in the shadow base.

The $Extractor$ module is a key component in this pipeline. Conceptually, it must be able to process any document format and identify arbitrary elements, themselves defined using arbitrary standards and namespaces. Obviously, it is impossible to construct a universal extractor to satisfy such requirements.

The implementable idea behind the *Extractor* is that it comprises an extensible set of functions that recognizes elements within specific document formats, always obeying the hierarchical definition. In other words, an $Extractor$ can "extract" any kind of element within any type of document, as long as code is developed to perform this task. Again using the image descriptor simile, image descriptors are defined mathematically, but the actual implementation varies with image format.

Our extractor implementation already supports[2] documents in three formats – .pdf, .doc and .odt. In other words, our shadows can uniformly describe documents in these three formats, and recognize a multitude of document element types.

[2]It is an initial prototype

IV. Implementation Overview

We implementend shadows as well formed XML files, and stored them in a database with native support for XML queries. With this approach, we were able to use already established technologies and solutions for XML. Figure 2 shows our implementation to generate shadows, divided in two main steps: (Step A) schema creation; (Step B) shadow generation – based on the schema defined by domain users. *Step B* is organized in three parts: The first part (item 2. of Figure 2) concerns scanning the document in order to recognize the types of elements of interest; the second part (item 3. of Figure 2) concerns the production of shadows, based on the recognized types, according to the schema.

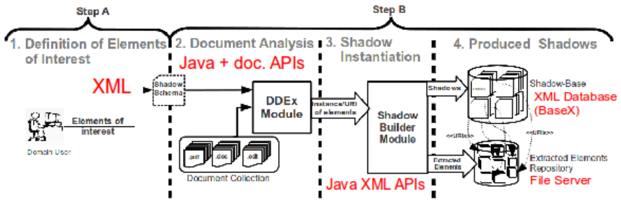

Figure 2. Technologies adopted

The third part creates the shadow and also extends repositories with elements extracted from the documents(e.g., an image repository or a repository with tables that appear in documents). These repositories are a means to support the manipulation of parts of documents without interfering with the documents themselves. For instance, in one experiment we built one repository with images extracted from documents – thereby allowing indexing of these images. In our implementation, users can specify the schema via an XML file, according to an XML-Schema specification. Basically, this file acts like a template, which includes all element types, defined via terms of ontologies or metadata standards. For instance, users who want the author(s) of a document to appear in a Shadow can define the term *author*, from the Dublin Core Standard [20], to represent the author field in a shadow. Alternatively, a user can define his/her own definition of author.

The extractor module is based on a Java Framework we implemented, called DDEx[3]. It scans documents based on schema definition and identifies the elements of interest. Each element is encapsulated in a standard representation and forwarded to the shadow builder. In order to process distinct document formats and recognize specific elements types, DDEx encapsulates special purpose modules – e.g., to recognize an image element in a .docx document.

DDEx is implemented according to a specific software design pattern – the *Pattern Builder* [21]. DDEx adopts

[3]Open Source Project available at http://code.google.com/p/ddex

several APIs for document handling, such as *iText*[4], *PDFBox*[5], *PDFClown*[6] and *PDF Renderer*[7] for PDF documents. In case of documents produced in *Microsoft Word*, DDEx adopts the *Apache POI*[8] framework. Furthermore, DDEx adopts the *ODF Tool Kit*[9] and JOpendocument[10] for files following the *Open Document Format*. For more details on DDEx, see [22].

Document Analysis and Content Extraction

Figure 3 abstracts an important concept in this work: composition. The *Document Analyst* within DDEx is responsible for the task of analysing the composition of elements according to the schema. The left part of Figure 3 shows an abstraction of a specification of an element. This element, whose composite type is *Image*, is defined as follows. An image contains a picture and a caption. A Picture contains a byte stream[11] of the image file. A Caption contains a Paragraph. A Paragraph contains a string and a newline command, and a Caption should appear below a Picture. The right side of Figure 3 shows a specific part of a document that fits this specification of many levels of composition. Since this part of the document "matches" this specification (defined in the shadow schema), it should be forwarded to the shadow builder.

Notice that this approach differs from that of IR-based techniques, since the emphasis in on implementing modules that will extract arbitrary user-defined information from documents, rather than general algorithms that will look for unifying descriptors.

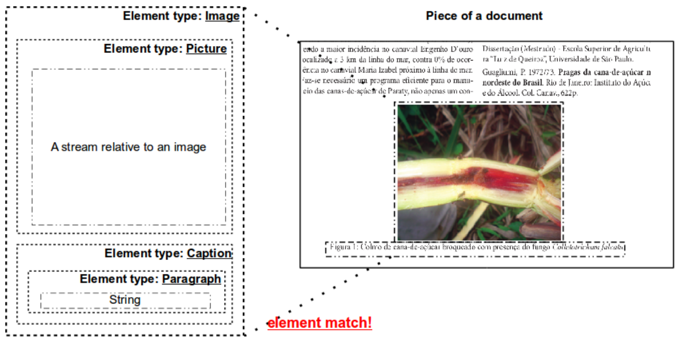

Figure 3. Example of how the document analyst recognizes an image with a caption via type matching of composite elements

Our decision to build DDEx based on the *Pattern Builder* ensures extensibility of the supported document formats. Since the Pattern Builder allows the separation between the extraction and the production process, we were able to construct three specialized document processors – for *.doc*, *.pdf* and *.odt* files. Hence, the Shadow Builder does not need to change, even if DDEx provides support to new formats of documents.

[4]http://itextpdf.com
[5]http://pdfbox.apache.org
[6]http://sourceforge.net/projects/clown/
[7]http://java.net/projects/pdf-renderer/
[8]http://poi.apache.org
[9]http://odftoolkit.org/projects/odfdom
[10]http://www.jopendocument.org/
[11]The raw set of bytes that correspond to the image

Shadows are implemented as well formed XML files (Item 3 of Step B of Figure). In order to do this, we adopted the JDOM[12], DOM4J[13] and Xerces[14]Java APIs to parse, manipulate and serialize XML files. The Shadow Builder constructs the shadow according to a schema. For each element instance forwarded by the extractor (see Figure 2), it builds an XML expression that represents an *instance* or URI of the instance (pointing at the instance in a repository).

Dublin Core, *ORE* and *Docbook* and other metadata standards and ontologies are already supported by our implementation. For more details on the implementation of the shadow Builder, the reader is referred to [23], [24]. Here, we just give a general idea, since the emphasis is on semantic issues, and not the internals of the shadow construction.

V. Case Study

A. *Constructing a Shadow Base*

Our case study concerns biodiversity studies, for papers in Portuguese and English. To build a collection, we implemented a Python crawler to automatically get documents from Google Scholar, constructing a collection of 3300 documents. We constructed a schema based on our experience of working with biologist in biological large biodiversity projects.

From the document collection (in *.doc*, *.pdf*, *.odt* formats), we generated 3104 shadows[15]. The documents had 2053 images, which were stored in a repository. The collection occupies 1.9 GBytes and the shadow base 202 MB. Shadows were created in 26 minutes in a computer with 16 processors and 32 GB of RAM.

We point out that the goal of this case study was to validate shadows as a basis for semantic annotation. Thus, data volume *per se* is not a major factor; the issue is flexibility in handling semantics across documents in many formats.

B. *Querying the Shadow Base*

The shadows were stored in an XML database, the BaseX[16] database system. We formulated queries in Xquery against the BaseX shadow base, following end-user (biologist) requirements. We also executed queries that were not relevant to biologists, but which show the potential of our proposal (as compared to IR approaches, or document management proposals).

Figure 5 is an example of a query result that can be used to analyze the document collection via the shadow base, though it is not a query requested by our end-users. It shows the shadow base (as displayed by BaseX) in terms of documents that contain a number X of images (for $X > 0, X > 10$ and $X > 100$).

Figure 4 presents a XQuery code that produced the result presented in Figure 5. Both lines 1 and 2 concern the definition of the namespaces that will be used on the query. Line 4 is

[12]http://www.jdom.org
[13]http://dom4j.sourceforge.net/
[14]http://xerces.apache.org/xerces-j/
[15]Some documents were not valid
[16]http://basex.org

```
XQuery example
declare namespace docshadow  ="http://purl.org/docshadow";
declare namespace textshadow ="http://purl.org/textshadow";

/docshadow:shadow/docshadow:metadata[textshadow:imgcount>X]
```

Figure 4. Simple XQuery query that can be posed against shadows

the query itself, a XPath expression that points to an element of the shadow that contains the number of images within a document.

Figure 5A shows that a large number of shadows (and therefore documents) contain images, whereas 5C shows that very few documents contain more than 100 images.

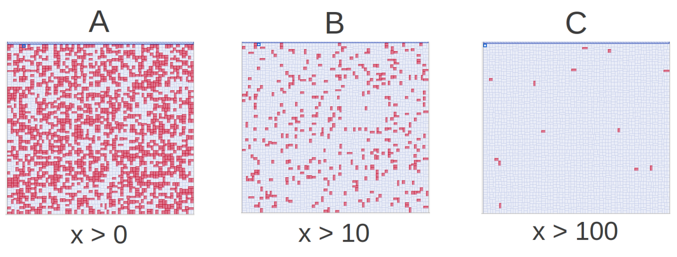

Figure 5. Fetched shadows from a query "documents with *x* images"

Examples of the latter include a technical report on analysis of vegetation cover in agricultural regions in Brazil, using satellite images, a book chapter on Computational Biology and a paper discussing parasites that compete for host species. This is an example that shows that the shadow base can be queried to partition the document collection according to several criteria - and this criterion (number of images), in particular, is not viable in other approaches. Examples of queries that can be processed by IR techniques as well include:

Q1 Documents whose authors include researcher called ”A” and whose title contains the word ”biodiversity”.

Q2 Documents containing keywords ”C” and ”D”.

For instance, query Q1 returned 17 shadows (corresponding to *.pdf* and *.doc* files) for “A”= “Marcelo” and 8 shadows for “A” = “Vera”. Query Q2 returned 33 shadows for “C” = “monitoring” and “D” = “report”.

Examples of queries that cannot be processed by other approaches include:

Q3 Documents with more than 10 pages, at least 5 images, and one section called ”Case Study”.

Q4 Documents without images or tables.

Queries Q3 and Q4 combine structure and content. They are not useful in terms of biodiversity research, but are included to show the versatility of shadow management.

Queries Q5 and Q6 are posed against captions of images and tables, and captions are part of the schema. While Q5 can be solved using IR techniques, Q6 requires structure knowledge (because the table must follow the image within the document).

Q5 Documents that contain one or more images of species ”Glomerella tucumanensis”.

Q6 Documents that contain an image of ”Glomerella tucumanensis” followed by a table that concerns the same species.

C. Annotating Shadows

Since a shadow is a document descriptor, it can be easily used with different purposes to indirectly query and rank documents. In our case study, we used shadows to indirectly annotate documents, creating links between elements of a shadow and ontologies. To annotate shadows, we adopted an RDF based schema for describing annotations and an already established XML reference standard – XPointer – to address shadow elements.

Our annotation strategy is based on the Linked Data paradigm. It follows the simple strategy of considering that two entities are the same if they refer to the same ontology term – i.e., we do not look for more sophisticated IR techniques. The goal is to establish a basis for fact finding and linking documents and concepts via shadows, assuming that the two entities – the element instance in a document, and the concept defined in the ontology – are semantically related. This is a strong assumption (e.g., see [25]), but it is the first step towards semantic entity linkage in a context of otherwise heterogeneous unrelated data sources.

Our annotations are RDF triples that link a shadow element to concepts in DBPedia. Annotations were inserted in the Virtuoso[17] database, that supports RDF triples and SPARQL queries. The piece of SPARQL code in Figure 6 shows how to create an annotation that links a shadow of a document containing an image of species ”Glomerella tucumanensis” to associated concepts in DBPedia. More specifically, this link is associated with an element identified within this shadow by ”erg3423” – see last line of the code. We recall that this identifier is artificially generated by DDEx to support the management of shadow elements within the Shadow Base.

```
Example Insertion
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

INSERT INTO GRAPH
 <http://proj.lis.ic.unicamp.br/annotations/> {

 <http://dbpedia.org/resource/Glomerella_tucumanensis>
<foaf:depiction>
<http://proj.lis.ic.unicamp.br/shadows/64.pdf.xml#
 xpointer(dc:identifier(``erg3423''))> }
```

Figure 6. Example of SPARQL query used to insert annotations that links concept and shadow

The piece of code shown in Figure 7 is a query that looks for documents with associations with the DBPedia concept ”Glomerella tucumanensis”. This query is posed against the local Virtuoso annotations and DBPedia. Recall that an analogous query Q5 looked for documents containing images of this species. Here, however, the result is very different. The first difference, of course, is that Q5 retrieved URIs of shadows, and this query retrieves URIs of shadows and of DBPedia concepts.

[17]http://virtuoso.openlinksw.com

```
Example of SPARQL query
SELECT ?property, ?value

FROM <http://proj.lis.ic.unicamp.br/annotations/> WHERE {
<http://dbpedia.org/resource/Glomerella_tucumanensis>
   ?property
   ?value }
```

Figure 7. Example of SPARQL query used retrieve URIs related to a concept

The second difference is semantically more interesting. While the (Xquery) query Q5 on the Shadow Base only returned shadows of documents that had images of this species, resulting in 3 shadows, the (SPARQL) query on annotations plus DBPedia returned 46 URIs, of which 42 are from DBPedia, 3 point to the same shadows, and one concerns *another* shadow, not identified by Q5, where the only species mentioned is "Colletotrichum falcatum". The reason for this discrepancy is the following. "Glomerella tucumanensis" has several scientific synonyms - one of them being "Colletotrichum falcatum". Thus, the SPARQL query on annotations not only returned the shadows retrieved in Q5; it also returned references to a shadow (and thus a document) that had no mention of "Glomerella", but described it under another name. This is an example of how annotations using LOD can significantly enhance query possibilities.

In particular, in biodiversity studies, experts need to correlate papers in several scientific domains (e.g., climatology, phenology, pedology). The use of Linked Data supports this kind of correlation. Continuing the example, "Colletotrichum falcatum" is a fungus that attacks sugar cane and is widespread in subtropical regions, being also called "red rot disease of sugar cane". Via annotations, experts are able to pose queries such as "*documents that refer to plant diseases in subtropical regions*" or "*documents that contain images of red rot of sugar cane*", even though there is no mention of these terms in the original document.

VI. Conclusions and Ongoing Work

This work proposes the Shadow-driven Representation approach to document management and annotation. It adopts the notion of "descriptor", borrowed from the image database literature, to represent a document's structure and content according to elements of interest to groups of users.

The main advantages of this approach are: (i) shadows isolate domain-relevant elements of a document from its format; (ii) shadows follow interoperability standards, enabling their exchange and machine consumption; and (iii) shadows were implemented as XML database instances, thereby serving as a homogeneous basis for processing arbitrary sets of documents in multiple formats, allowing summarization, indexing, and semantic annotation via the shadow base.

Shadows are a flexible and format-independent way of immersing documents in the semantic Web. Our ongoing work is exploiting this, by constructing networks of shadows, and trying to establish metrics to indirectly index and compare documents.

Acknowledgment

This work was partially supported by the Microsoft Research FAPESP Virtual Institute (NavScales project), the Brazilian Institute for Web Science, CNPq (MuZOO project), PRONEX-FAPESP [18], CAPES (AMIB project) and CNPq[19].

References

[1] A. Santanchè, M. Mota, D. P. Costa, N. Oliveira, and C. O. Dalforno, "Componere – web authoring based on components," in *Proc. of XV Brazilian Symp. on Multimedia and the Web*, 2009.

[2] T. Hey, S. Tansley, and K. Tolle, Eds., *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Redmond, Washington: Microsoft Research, 2009.

[3] J. Kahan, "Annotea: an open RDF infrastructure for shared Web annotations," *Computer Networks*, vol. 39, no. 5, pp. 589–608, Aug. 2002.

[4] M. Koivunen, R. Swick, and E. Prud'hommeaux, "Annotea shared bookmarks," in *Proc. of the KCAP 2003 workshop on knowledge markup and semantic annotation*. Citeseer, 2003, pp. 25–26.

[5] H. Cunningham, "GATE, a general architecture for text engineering," *Computers and the Humanities*, vol. 36, no. 2, pp. 223–254, 2002.

[6] R. da Silva Torres and A. Falcão, "Content-based image retrieval: Theory and applications," *Revista de Informática Teórica e Aplicada*, vol. 2, no. 13, pp. 161–185, 2006.

[7] J. van Ossenbruggen, F. N., and L. Hardman, "That Obscure Object of Desire: Multimedia Metadata on the Web, Part 1," *IEEE MultiMedia*, vol. 11, no. 4, pp. 38–48, 2004.

[8] J. Greenberg, "Understanding Metadata and Metadata Schemes," *Cataloging & Classification Quarterly*, vol. 40, no. 3, pp. 17–36, Sep. 2005.

[9] E. Duval, W. Hodgins, S. Sutton, and S. L. Weibel, "Metadata Principles and Practicalities," *D-Lib Magazine*, vol. 8, no. 4, pp. 1–10, Apr. 2002.

[10] H. Lejsek, F. H. Ásmundsson, B. T. Jónsson, and L. Amsaleg, "Scalability of local image descriptors: a comparative study," in *MULTIMEDIA '06*. NY, USA: ACM, 2006, pp. 589–598.

[11] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, "A survey of content-based image retrieval with high-level semantics," *Pattern Recognition*, vol. 40, no. 1, pp. 262 – 282, 2007.

[12] E. Oren, K. H. Moller, S. Scerri, S. Handschuh, and M. Sintek, "What are semantic annotations??" 2006, technical report, DERI Galway.

[13] J. Euzenat, "Eight questions about semantic web annotations," *IEEE Intelligent S.*, vol. 17, no. 2, pp. 55–62, 2002.

[14] P. Cano, "Automatic sound annotation," in *In IEEE workshop on Machine Learning for Signal Processing*, 2004, pp. 391–400.

[15] A. Kiryakov, B. Popov, I. Terziev, D. Manov, and D. Ognyanoff, "Semantic annotation, indexing, and retrieval," *Web Semantics: Science, Services and Agents on the WWW*, vol. 2, no. 1, pp. 49 – 79, 2004.

[16] S. H. Yeganeh, O. Hassanzadeh, and R. J. Miller, "Linking semistructured data on the web," in *Proceedings of the 14th International Workshop on the Web and Databases*, 2011.

[17] T. Grust, M. Mayr, and J. Rittinger, "Let SQL Drive the XQuery Workhorse," in *Proc. EDBT*, 2010, pp. 147–158.

[18] L. Zhang, Y. Zhang, and Q. Xing, "Filtering semi-structured documents based on faceted feedback," in *Proc. 34th SIGIR conference*, 2011, pp. 645–654.

[19] M. V. Cundiff, "An introduction to the Metadata Encoding and Transmission Standard," *Library Hi Tech*, vol. 22, no. 1, pp. 52–64, 2004.

[20] DCMI, "Dublin core metadata initiative," Aug. 2010, http://dublincore.org/metadata-basics/, accessed on 01/2011.

[21] E. Gamma, R. Helm, R. Johnson, and J. M. Vlissides, *Design Patterns*. Boston, USA: Addison-Wesley Longman Publishing Co., Inc., 1995.

[22] M. S. Mota, "Shadows: a new means of representing documents," Master's thesis, Institute of Computing – UNICAMP, May 2012.

[23] ——, "Shadows: a new means of representing documents," Master's thesis, Institute fo Computing, Unicamp, May 2012.

[24] M. S. Mota, J. S. C. Longo, D. C. Cugler, and C. B. Medeiros, "Using linked data to extract geo-knowledge," in *XII Brazilian Symposium on GeoInformatics - GeoInfo*, November 2011.

[25] X. Han, L. Sun, and J. Zhao, "Collective Entity Linking in Web Text: A Graph-Based Method," in *Proc SIGIR*, 2011, pp. 765–774.

[18]Model and Methods in eScience fot the Life and Agricultural Sciences

[19]individual scholarship – process 133815/2010-2