D-Lib Magazine
December 2003

Volume 9 Number 12

ISSN 1082-9873

Open Archives Data Service Prototype and Automated Subject Indexing Using D-Lib® Archive Content As a Testbed

Larry Mongin
Research Programmer
Indiana University School of Library and Information Science
<lmongin@indiana.edu>

Yueyu Fu
Doctoral Student
Indiana University School of Library and Information Science
<yufu@indiana.edu>

Javed Mostafa
Yngve Associate Professor of Information Science and Associate Dean
Indiana University School of Library and Information Science
<jm@indiana.edu>

Introduction

The Indiana University School of Library and Information Science opened a new research laboratory in January 2003; The Indiana University School of Library and Information Science Information Processing Laboratory [IU IP Lab]. The purpose of the new laboratory is to facilitate collaboration between scientists in the department in the areas of information retrieval (IR) and information visualization (IV) research. The lab has several areas of focus. These include grid and cluster computing, and a standard Java-based software platform to support plug and play research datasets, a selection of standard IR modules and standard IV algorithms. Future development includes software to enable researchers to contribute datasets, IR algorithms, and visualization algorithms into the standard environment. We decided early on to use OAI-PMH as a resource discovery tool because it is consistent with our mission.

D-Lib Magazine Archive Structure

We are using the D-Lib Magazine archives [D-Lib Magazine] as a dataset for our

prototype for several reasons. D-Lib Magazine is provided via open access for non-commercial, educational use with few restrictions (see D-Lib Magazine access terms and conditions at <http://www.dlib.org/access.html>. The content of the articles is of interest to the software developers, which is convenient when we are developing the software. Also D-Lib Magazine has a metadata file associated with each article. Each article is stored in a directory with the article HTML file, images, and a .meta.xml file that contains metadata about the article.

D-Lib Metadata

Since March 1999, D-Lib has created an XML metadata file associated with each article published in the magazine. The metadata format is related to, but not quite the same as Dublin Core [Dublin Core] which is the metadata structure used by the Open Archives Initiative.

Below is an example D-Lib metadata file from an article by José Canós et al. in the January 2003 issue [Canós 2003]:

```
<?xml version="1.0"?>
<!DOCTYPE dlib-meta0.1 SYSTEM "http://www.dlib.org/dlib/dlib-meta01.dtd">
  <dlib-meta0.1>
  <title>Building Safety Systems with Dynamic Disseminations of Multimedia
Digital Objects</title>
  <creator>Jose H. Canos</creator>
  <creator>Javier Jaen</creator>
  <creator>Juan C. Lorente</creator>
  <creator>Jennifer Perez</creator>
  <publisher>Corporation for National Research Initiatives</publisher>
  <date date-type="publication">January 2003</date>
  <type resource-type="work">article</type>
  <identifier uri-type="DOI">10.1045/january2003-canos</identifier>
  <identifier uri-
type="URL">http://www.dlib.org/dlib/january2003/canos/01canos.html</identifier
>
  <language>English</language>
  <relation rel-type="InSerial">
      <serial-name>D-Lib Magazine</serial-name>
      <issn>1082-9873</issn>
      <volume>9</volume>
      <issue>1</issue>
  </relation>
  <rights>Jose H. Canos, Javier Jaen, Juan C. Lorente, and Jennifer
Perez</rights>
  </dlib-meta0.1>
```

Most of the elements map into Dublin Core [DC]. The relation field has to be selected for "URL". The creator field has to be combined for all the authors. D-Lib has a few fields that don't map into Dublin Core. This is a common problem with the limited map of Dublin Core.

Choosing an OAI Repository Program

The Open Archives Initiative [OAI] is an open standards, open source group. They offer

a number of tools, most of which are available free of charge, to help implement repositories and harvesters. We wanted a program that is easy to install, one implemented in Java, and one with which it is easy to load metadata files. The program we chose was the "Rapid Visual OAI Tool" (RVOT) from Old Dominion University [Old Dominion RVOT]. RVOT is a stand alone Java program that includes a lightweight http server. It is easy to install as a user program. RVOT includes several mapping procedures to convert metadata into the rfc1807 native metadata format [rfc1807]. Sets are supported. The program also includes an interactive user interface to map metadata fields into the native rfc 1807 format. There is some concern that the native file/directory structure might not work for large datasets, and we may need to migrate to a database repository in the future to ensure efficient performance.

Below is an example an example of the rfc1807 metadata record for the Canos et al. D-Lib article:

```
  TITLE:: Building Safety Systems with Dynamic Disseminations of Multimedia
Digital Objects
  AUTHOR:: Jose H. Canos
  AUTHOR:: Javier Jaen
  AUTHOR:: Juan C. Lorente
  AUTHOR:: Jennifer Perez
  ORGANIZATION:: Corporation for National Research Initiatives
  DATE:: January 2003
  TYPE:: article
  ID:: http://www.dlib.org/dlib/january03/canos/01canos.html
  LANGUAGE:: English
  RELATION:: D-Lib Magazine
  COPYRIGHT:: Jose H. Canos, Javier Jaen, Juan C. Lorente, and Jennifer Perez
```

This rfc1807 record maps into a RVOT Dublin Core record that uses a similar syntax but substitutes rfc1807 tags for Dublin Core tags. Author maps to Creator; Organization maps to Publisher, and Copyright maps to Rights.

Below is the RVOT Dublin Core record for the Canós article:

```
  TITLE:: Building Safety Systems with Dynamic Disseminations of Multimedia
Digital Objects
  CREATOR:: Jose H. Canos
  CREATOR:: Javier Jaen
  CREATOR:: Juan C. Lorente
  CREATOR:: Jennifer Perez
  PUBLISHER:: Corporation for National Research Initiatives
  DATE:: January 2003
  TYPE:: article
  IDENTIFIER:: http://www.dlib.org/dlib/january2003/canos/01canos.html
  LANGUAGE:: English
  RELATION:: D-Lib Magazine
  RIGHTS:: Jose H. Canos, Javier Jaen, Juan C. Lorente, and Jennifer Perez
```

Note that the rfc1807 fields [rfc1807] are quite similar to Dublin Core XML used in OAI-PMH.

Extracting Data from D-Lib XML Files

Most of the OAI-PMH fields can be directly mapped from the D-Lib meta.xml files that describe each article. The main field missing from the D-Lib metadata file is a subject terms field. D-Lib includes a <meta> tag in the <head> element of every article, using the same three keywords. Since term association is a major part of a search service we decided to use IR algorithms to compute subject (keyword) terms for each article.

Below is an example of the metadata XML file resulting from our use of the IR algorithm to compute keywords:

```
<article>
  <title>
  Building Safety Systems with Dynamic Disseminations of Multimedia Digital
Objects
  </title>
  <creator >Jose H. Canos</creator>
  <creator> Javier Jaen</creator>
  <creator>Juan C. Lorente </creator>
  <creator>Jennifer Perez </creator>
  <publisher>Corporation for National Research Initiatives</publisher>
  <date>
      <month>January</month>
      <year>2003</year>
  </date>
  <type>article</type>

<identifier>http://www.dlib.org/dlib/january03/canos/01canos.html</identifier>
  <language>English</language>
  <relation>D-Lib Magazine</relation>
  <rights>Jose H. Canos, Javier Jaen, Juan C. Lorente, and Jennifer
Perez</rights>
  <subject>train javax interfaces driver size aspect components ejbs home page
http station states
       evacuation screen stations tunnels section decision transportation
  </subject>
  <xmluri>01canos.xml</xmluri>
</article>
```

Most of the fields, except for the subject fields, were derived directly from the D-Lib metadata file.

Generating Keywords from D-Lib Articles

Here is the term selection algorithm we used:

- Extract tokens from HTML article text with Java callback parser.
- Drop terms that contain special characters except terms that end with punctuation (. , ? ;).
- Convert characters to lower case.
- Drop terms less than 4 characters in length.
- Drop terms in stop word list.
- Add term to term frequency matrix.

- When terms in document >100, start new document.

- When the end of the file is reached, process the term by document matrix as follows:
    - Compute df (document frequency matrix).
    - Drop terms that don't appear in at least 10% of documents.
    - Compute tf/idf (Term frequency (tf) is the number of times a term occurs in a single document. Inverse document frequency (idf) is a measure of the distinctiveness of this term across the document space.) tf/idf = tf/log(N/df).
    - Select n terms with highest tf/idf weights.
    - Compress tf/idf matrix.

tf/idf

Each cell of the term by document matrix is the raw frequency count of the number of times a term occurred in a particular document. The overall frequency matrix is the number of documents in which each term is found. These two arrays are used to compute tf/idf. The terms are about 25% of the tokens in the HTML document after rejecting terms with special characters, tokens less than four characters, and terms in the stop word list.

tf/idf = tf/log(N/df)

The tfidf(termArray) method computes tf/idf from the raw frequency arrays.

Below is a snippit of sample Java code for this process:

```
tf/idf = tf/log(N/df)
```

The tfidf(termArray) method computes tf/idf from the raw frequency arrays.

Below is a snippit of sample Java code for this process:

```
public void ctfidf(){
  int i,j;
  int N = docs;
  for (i=0;i<nterms;i++)
    for (j=0;j<N;j++)
    if(htdf[i][j] > 0 && hdf[i]   > 0){
      htdf[i][j] = htdf[i][j] * Math.log(N/hdf[i]);
      //System.out.print(hterms[i]);
      //System.out.println(htdf[i][j]);
    }
    else
      htdf[i][j] = 0;

  }
```

The tf/idf method provides a way for weighing terms in a document space. A term that

occurs in every document is not a very useful search term and a term that appears very infrequently is also probably not that useful [Salton, McGill: 1983].

Creating a Document Space within a Single Article

An important feature of this OAI-PMH data service is computing keywords from the text of the articles. The tf/idf algorithm requires a document space; a term by document matrix. Because paragraphs in an HTML document are difficult to define, we decided to use term proximity to define the document space. Each set of n terms (where n=100) in the current code is treated as a document. Tokens that passed the screen represented about 25% of the terms in the original document. A typical paragraph is about 400 words; hence, 100 selected terms is close to the size of a paragraph in English prose. Table 1 below illustrates the conversion data from five D-Lib articles.

Table 1. Conversion Data from Five Sample Articles

On Making and Identifying a "Copy" *Norman Paskin, International DOI Foundation*	Computed Keywords	Relevance Data	Metadat
Building Safety Systems with Dynamic Disseminations of Multimedia Digital Objects *José H. Canós, Javier Jaén, Juan C. Lorente, and Jennifer Pérez, Polytechnic University of Valencia*	Computed Keywords	Relevance Data	Metadat
MOAC - A Report on Integrating Museum and Archive Access in the Online Archive of California *Richard Rinehart, University of California, Berkeley*	Computed Keywords	Relevance Data	Metadat
iVia Open Source Virtual Library System *Steve Mitchell, Margaret Mooney, Julie Mason, Gordon Paynter, Johannes Ruscheinski, Artur Kedzierski, and Keith Humphreys, University of California, Riverside*	Computed Keywords	Relevance Data	Metadat
Open Archives Activities and Experiences in Europe: An Overview by the Open Archives Forum *Susanne Dobratz and Birgit Matthaei, Humbolt University, Berlin*	Computed Keywords	Relevance Data	Metadat

Table 2. Computed Keywords

	Paskin	Canos	Rinehart	Mitchell	Dobratz
1	document	train	page	programmer	united
2	blue	javax	broad	chakrabarti	expectations
3	schema	interfaces	physical	phrase	offered
4	hence	driver	direct	learning	second

5	itself	size	visitors	page	costs
6	distinct	aspect	testbed	wide	most
7	edition	components	richard	michigan	workshops
8	ifla	ejbs	reach	analysis	southampton
9	press	home	held	greenstone	workspace
10	niso	page	bancroft	citeseer	openarchives
11	identical	http	portals	phind	scientific
12	elements	station	models	theme	user
13	resource	states	record	create	pisa
14	common	evacuation	damd	authorities	extend
15	likely	screen	image	description	torii
16	qualfier	stations	object	cdrom	responding
17	reference	tunnels	often	refereed	formats
18	ebook	section	objects	conference	activities
19	open	decision	initial	riverside	workpackage
20	users	transportation	practice	crawling	univ

Analysis of Results of Subject Term Extraction

We selected the top 20 terms based on tf/idf weight for each article. We then read the article and evaluated a simple relevance judgment for each term. The question was whether the term was relevant to the semantic meaning of the article. See Table 3 below.

Table 3. Relevance Judgments

Article/ Key	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	%
Paskin	y	n	y	n	n	n	y	y	y	y	n	y	y	y	n	y	y	y	y	y	70
Canos	y	y	y	y	n	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	90
Rinehart	y	n	y	n	y	y	y	n	n	y	y	y	y	y	y	y	n	y	y	y	75
Mitchell	y	y	y	y	y	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	95
Dobratz	y	n	n	n	y	n	y	y	y	y	y	y	y	y	y	y	y	y	y	y	85

After running the Java program that computed subject terms, we read each article to make a judgment on whether the computed subject terms were relevant to that article. The criteria was not whether the program selected the best subject terms for that text, but rather whether the term generally reflected the semantic meaning of the article. The resulting scores varied from 70-95%. There are some terms in each article set (reach, held, likely) that would not work well at all as search terms. An interesting question for

future work is whether to have a cataloger prune those terms or leave them in assuming that most users of the search interface wouldn't use those terms anyway.

OAI-PMH Harvester Selection/Search Service

A harvester provides the means for collecting metadata from repositories. We wanted a harvester that would be easy to install. We also looked for one that would make it easy to schedule a harvesting job and to manage the harvested data. In addition we wanted the harvester to be open source and implemented in Java. Among the many existing OAI-PMH harvesting tools, we chose OAIHarvester from OCLC. The OAIHarvester is an open source Java application providing an OAI-PHM v.2.0 harvester framework. It provides some Java interfaces to harvest customized metadata formats.

Although the OAIHarvester provides a way for harvesting the metadata, we wanted to offer users a direct view of the repository by adding some search services on top of it. The search services we developed support search functions in the fields of author, title, subject, and publication date. Users can search on any combination of the fields. The search results include the links to the retrieved article, to the authors, and to the full metadata about the article in XML format.

D-Lib Article Browser

We developed an article browser with a search service using as data the D-Lib Magazine articles and metadata. The browser is now running on data in our OAI-PMH repository. We are in the process of linking the repository to an OAI-PMH harvester and will run the browser and search service from the harvested data once that link is complete. For the time being the metadata and subject terms for the articles in the OAI-PMH repository is output to a single XML file using a Java program. A second Java program reads the XML file and outputs the metadata to a MySQL database as a single large table. The Apache Tomcat search servlet queries the database using SQL commands. Apache Tomcat is an open source Java servlet engine. Results from user queries are written to an HTML file and returned to the user. Here is a sample of the browser returning a table of articles from the January 2003 issue:

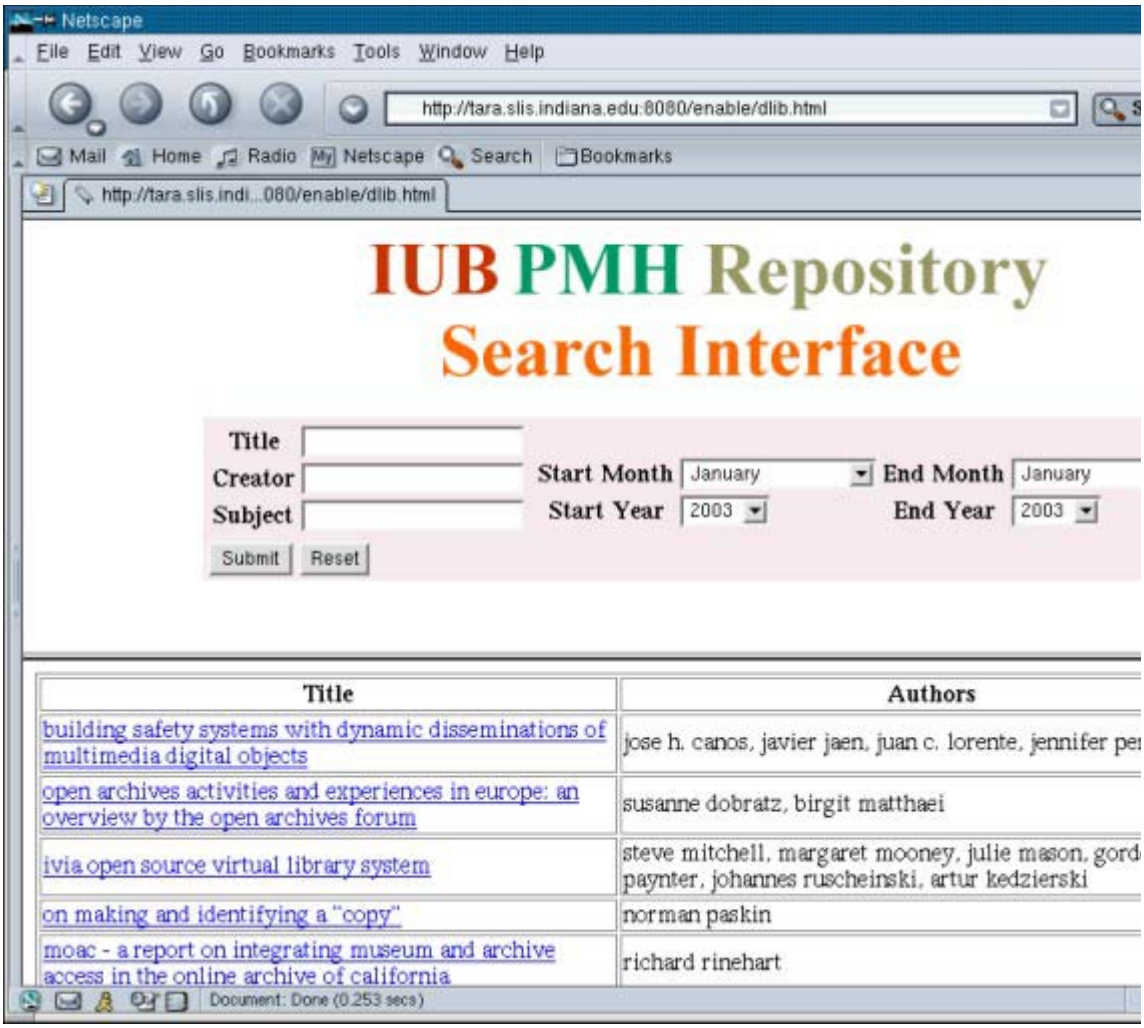

Figure 1. Result of searching for January 2003 articles

A search for the author ("Creator") Canos returns this result:

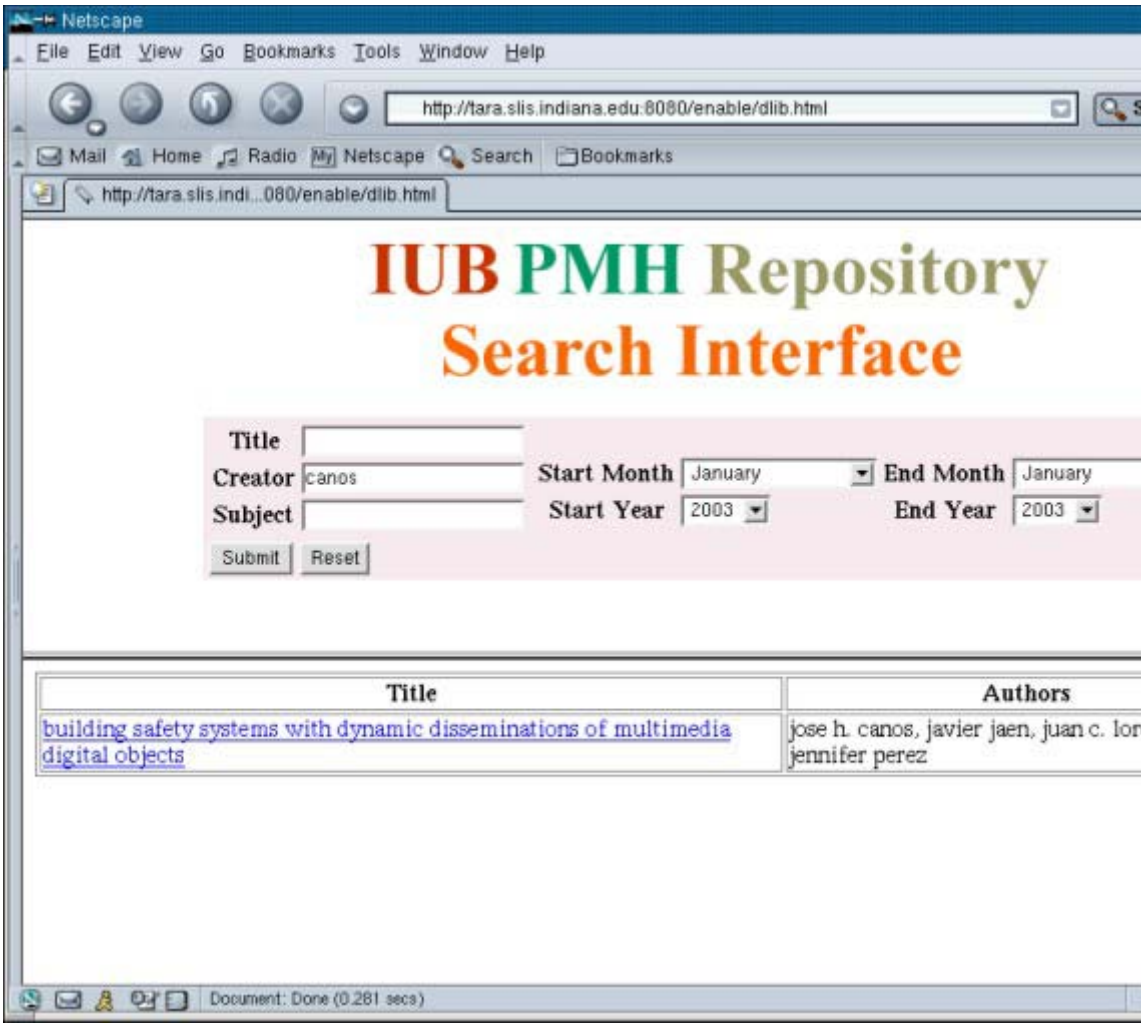

Figure 2. Result of searching for the author ("Canos")

An interesting question is whether this application produces better results than the search function already built into D-Lib Magazine. The D-Lib search service is based on a conventional full text search engine. The main difference between our search service and D-Lib's search service is that in ours the subject field contains terms generated using tf/idf. The browser and search interface can be accessed at <http://enable.slis.indiana.edu:8080/enable/dlib.html>.

Search terms can be entered into the title, creator, and subject text fields. Entering terms into multiple fields is a logical OR. Start and end dates can be selected. If one simply wants to see articles from one or more issues, no terms need to be entered into the text fields. The user has to specify the date range and click on the submit button. Results are returned as a table. The user then selects the title to view the article or selects Metadata to view the Dublin Core metadata for that article.

Conclusions, Limitations and Future Work

To prototype the search service, we are running the search Java servlet from a database built from data in the RVOT repository. The next step is to install an OAI harvester and run the search service from the harvester, which will harvest the metadata from the repository. This step will be completed in December 2003. OAI recommends running data services from harvested data. We will use the OCLC harvester that stores data in a database so the search servlet code will be the same.

The relevance judgments described in this article were binary, and the result was expressed as a simple percent. More work is needed to improve this analysis function. There is almost no overlap between keyword sets across the several articles, which makes vector analysis difficult.

A web crawler needs to be written that can discriminate between full length articles and other features in each issue.

Terms were selected based on high tf/idf weights in the tf/idf matrix. Other selection algorithms need to be explored and analyzed. One selection algorithm by Mostafa is used in Sifter vocabulary generation: Select a term if it ranks in the top n terms in a document(R) and occurs in at least N documents(D) [Mostafa 1998]. We should be able to compare the resulting term vectors using statistical algorithms more sophisticated than simple percentages.

The RVOT OAI-PMH repository program is quite easy to use and will easily meet our needs through the prototype stage of this project. RVOT uses a file format similar to rfc1807 to store the Dublin Core data elements. A mapping function in the server allows the user to select rfc1807 elements to map to specific Dublin Core fields.

The results of computing keywords from text in the articles are quite interesting. We assigned binary relevance values to each term in the keyword list based on a fairly relaxed standard. The question we posed was not whether the keywords were the best terms to represent the meaning of the article. Our criterion was instead whether the term was a reasonable keyword for the article or a part of the article. Results varied from 70% to 95% based on that relaxed criteria.

There are some nonsense terms in the keyword set but there are many terms that do reflect the semantic meaning of the article. One interesting observation is that the keyword sets for each article have little in common with other article keyword sets. That was a bit of a surprise given the tight focus of D-Lib. Another interesting feature of the relevance analysis is that most of the non relevant keywords appeared at the top of each list. The keyword list is sorted by tf/idf weight with the highest values at the top of the list. This result suggests that tf/idf weighting may not be the best method of selecting terms. The tf/idf algorithm does a good job of selecting terms for a specific document but not necessarily a good term for all paragraphs in an article. Keep in mind that we

split each article into multiple documents (paragraphs) for this computation.

Acknowledgment

This work was partially supported through a grant from the National Science Foundation (NSF) (Award#:0333623).

References

[Canós 2003] José H. Canós et al.: "Building Safety Systems with Dynamic Disseminations of Multimedia Digital Objects", *D-Lib Magazine*, January 2003. Available at <doi:10.1045/january2003-canos>.

[D-Lib Magazine] D-Lib Magazine is an electronic publication focused on digital library research and development. It is available at <http://www.dlib.org/>.

[Dublin Core] Dublin Core Metadata Initiative, <http://dublincore.org/>.

[IU IP Lab] Indiana University School of Library and Information Science Information Processing Laboratory, <http://ella.slis.indiana.edu/~lmongin/iplab/>.

[Mostafa 1998] Mostafa, J., Quiroga, L., & Palakal, M. "Filtering Medical Documents Using Automated and Human Classification Methods." *Journal of the American Society for Information Science*, 49(14), 1304-1318, 1998.

[OAI] Open Archives Initiative, < http://www.openarchives.org/>.

[rfc1807] An XML Schema for the rfc1807 metadata format, 2002. Available at <http://www.openarchives.org/OAI/2.0/guidelines-rfc1807.htm>.

[Old Dominion RVOT] Rapid Visual OAI Tool from Old Dominion University, <http://rvot.sourceforge.net/>.

[Salton, McGill 1983] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.