

An Intelligent Method for Searching Metadata Spaces

Introduction

This paper proposes a manner by which databases containing IEEE P1484.12 Learning Object Metadata can be effectively searched. (The methods outlined here may also be applicable to other information spaces.) The concepts of metadata, metadata spaces, and categorical filtering are introduced. A simple rule for narrowing a large number of search results down to a single digit number using the categorical filtering method is postulated. Issac Levi's *epistemic utility function* is introduced as a simple cost / benefit manner of recommending categories by which to filter a search. Finally, a manner in which the proposed categorical filtering rule and Levi's epistemic utility function could function together to decrease the mean time necessary to locate an object is explained.

Definitions

Metadata

The introduction to the IMS Metadata Specification Draft defines metadata in the following way.

Metadata ("data about data") are descriptive labels used to index resources for use. "Use" includes activities like resource management, discovery, and delivery. Examples of metadata range from simple soup can labels (which describe cans' ingredients) to rich indexing schemes employed by libraries (which describe books, titles, authors, subjects, locations within the library, etc.). Metadata systems make the process of using resources more efficient and effective (Wason, 1999).

Although "data about data" is a recursive definition it is probably the best that can be provided. Metadata describes other data. To understand why this description is necessary or even useful, consider the following example.

When search engines first began indexing the web, users were excited by the ability to perform full-text searches for their topics of interest. One could find just about anything. However, as the web has continued to explode in size the unscalability of full-text searching has made our lives miserable. (For example, a full-text search for the term "granularity" turns up 18,000 results at AltaVista.) So if full-text searching is not the answer, what kind of search approach is? A can of soup suggests the answer.

Imagine going to the store to pick up a can of chicken soup for a sick friend. Once inside the store you locate the soup aisle and find, to your horror, a row of unlabelled, silver cans shining toward you. How can you be sure to get a can of chicken soup? Only by opening can after can and investigating the contents until you find chicken soup (assuming you know what it looks like).

This kind of "full-ingredient" search is the approach search engines initially took to indexing the web -- open each page, examine it's contents, and record them individually for later searching. But it is evident that I can't find chicken soup this way unless I know it's ingredients. It is also evident that if there is another kind of soup with similar ingredients I may confuse the two. Finally, it is evident that there must be a better way to shop for soup, and therefore, a better way to search for instructional resources (Wiley et al., 1999).

Thus metadata is, as described in the IMS definition, "descriptive labels used to index resources."

Metadata Spaces

In his conceptual work entitled *Metadata Spaces*, Tom Wason (Wason, 1997) described each metadata field (for example, "author") as a dimension in an n -dimensional space, where n equals the number of metadata fields in the system. Each object for which metadata is stored is "addressed" by the metadata values associated with it. The object is then "located" at that address and can be discovered later. Additional dimensions create additional space in which to store objects. Preserving orthogonality between dimensions creates the most efficient space possible (largest number of addresses with the smallest number of fields). An example of storing common objects in common n -dimensional spaces should help illustrate.

Imagine trying to park a car (store an object) on a narrow street (in a one dimensional space). The number of cars that can be safely parked on the street (the number of available spaces or "addresses") is limited -- let's say that 5 cars can be parked on this street. Once you have parked your car (stored your object), you may remember the location of your parked car by remembering where you parked relative to the length of the street (by a position as figured from some unit of measure). Figure 1 illustrates this idea.



Figure 1. Five automobiles parked on a street, illustrating five objects stored in a one dimensional metadata space.

Now what if you need to park 20 cars instead of just 5? Then you might use a parking lot instead of a street. That is, you might add another dimension to your storage space -- in this case add *width* to the *length* of the street. (See Figure 2.) And if you need to park 80 cars then you might use a parking garage, i.e., add another dimension to your storage space, this time adding *height* to *width* and *length*. (See Figure 3.) Every time an orthogonal dimension of length u units is added to the space, the number of objects that can be stored in the space with a unique address is increased by a factor of u .



Figure 2. Twenty automobiles parked in a parking lot, illustrating 20 objects stored in a two dimensional metadata space.

The concern for preservation of orthogonality between dimensions is purely a concern of efficiency. If dimensions demonstrate any degree of correlation at all, then they are to some degree representing the same information. Since the objective is to create as many unique addresses with as few dimensions as possible, orthogonality of dimensions is a desideratum of a set of metadata fields.

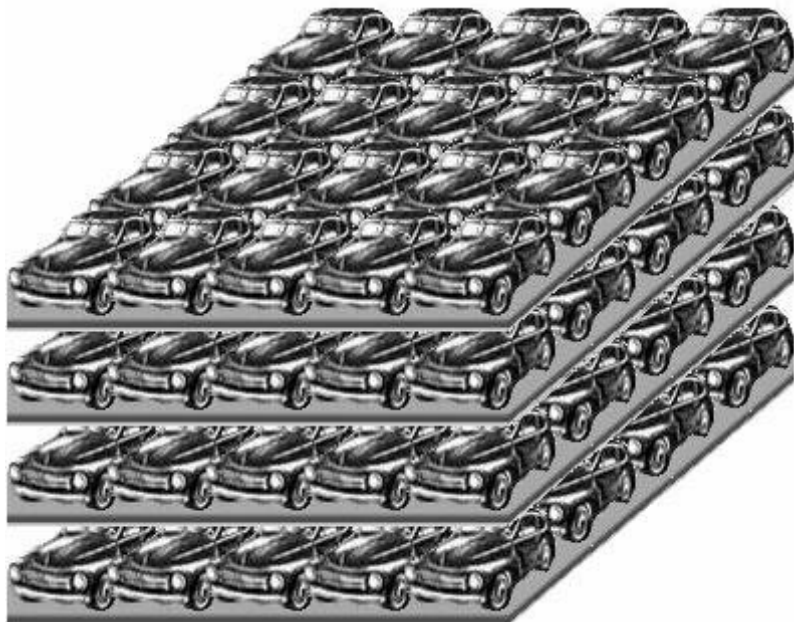


Figure 3. Eighty automobiles parked in a parking garage, illustrating 80 objects stored in a three dimensional metadata space.

An example of a "unique address" of an object in a set of metadata fields might look like Table 1. Instead of x, y, z coordinates, these coordinates are named for the various metadata fields. And instead of having numerical values, these fields have appropriate alphanumeric values. Figure 3 displays the comparison between the familiar x,y,z coordinate and the unique address provided by orthogonal metadata fields.

<i>PowerPoint Presentation Metadata</i>	
Title	How to alienate people and make lots of money
Author	Bill Gates
Creation Date	April 15, 1999
Format	Microsoft PowerPoint (.ppt)

Table 1. Sample set of "coordinates" or a "unique address" of a PowerPoint presentation.

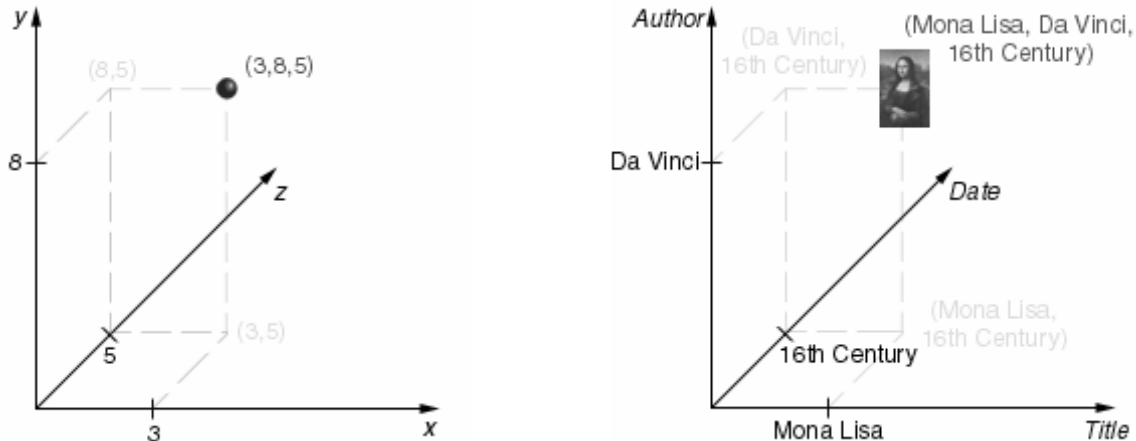


Figure 3. The image on the left shows the location of a point in familiar three dimensional space. The image on the right shows the location of an instructional resource in a three dimensional metadata space.

Categorical Filtering

Categorical filtering is a powerful way of searching through information which is divided into categories. Metadata fields can be thought of as categories for this purpose. Instead of performing a simple keyword search of the full-text of the objects themselves, the search is performed by selecting category after category, cutting away the objects that don't fall under that particular category. The method I am proposing here begins with a simple keyword search against all metadata fields that is then refined through the process of categorical filtering.

I have assumed that fewer, more relevant search results to dig through are better than thousands, although I understand that too few results are just as problematic. I have contrived a simple formula for determining the number of categories (C) and number of values per category (V) needed to get the initial number of results returned (N) down to a single digit number of results via the filtering process.

$$C = \log_{10}(N)$$

$$V = C + 2$$

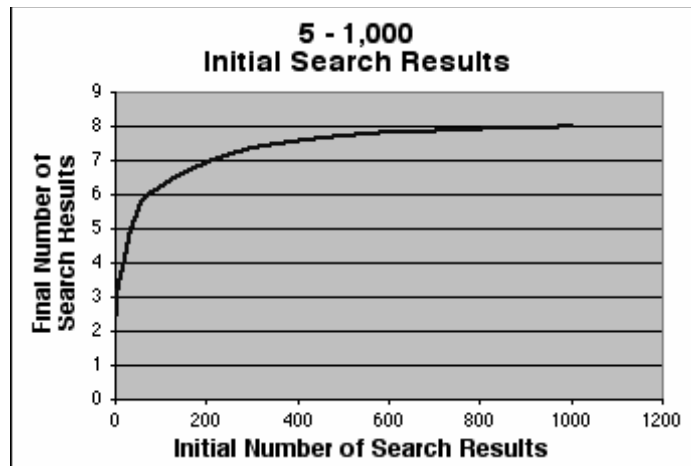
The formula for figuring the number of filtered results (R) is:

$$R = \frac{N}{V^c}$$

As Table 2 and Graphs 1 and 2 show, this approach can reduce a large, unwieldy number of results to a very manageable, single digit number of results quite quickly. This *drastically* cuts down the time necessary to locate a resource.

<i>Number of Search Results (N)</i>	<i>Number of Categories (C)</i>	<i>Average Values per Category (V)</i>	<i>Final Number of Search Results</i>
100	2	4	7
1,000	3	5	8
10,000	4	6	8
100,000	5	7	6
1,000,000	6	8	4

Table 2. The number of categories and values per category necessary to get a large number of search results down to single digits.

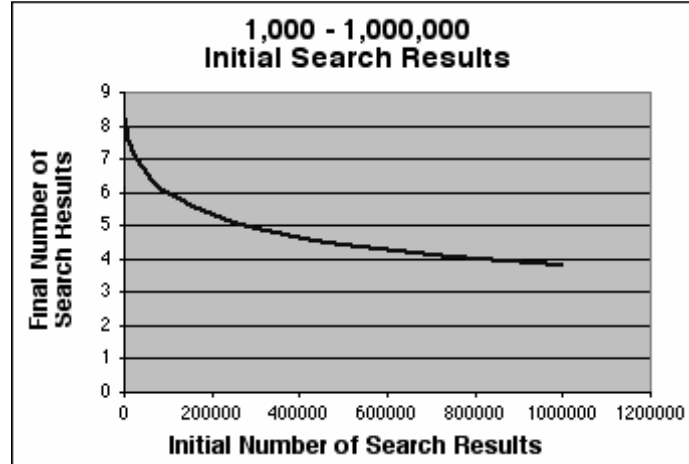


Graph 1. As the initial number of search results approaches 2,200, the final number of results peaks just above 8.

You might think, "Categorical filtering sounds like browsing through Yahoo!" It is true that you can search Yahoo! by browsing through categories instead of performing a keyword search. It is also true that each time you select a category you cut away all the "search results" that don't fall underneath it. However, the differences between this Yahoo! type of browsing and the intelligent search I am proposing are significant.

First, the Yahoo! search is refined along a single dimension, whereas the metadata space search employs a multi-dimensional refinement mechanism. Refining along several dimensions simultaneously is conceptually equivalent to narrowing a huge n-dimensional space into a tiny n-dimensional neighborhood. Choosing values along the dimensions

basically provides a set of coordinates in the n-dimensional space.



Graph 2. As the initial number of search results increases past 2,200, the final number of results approaches zero.

Second, the intelligent search uses Levi's epistemic utility function to dynamically calculate and recommend the most appropriate dimension to use for refinement. It does this at each stage of the search by making real-time recommendations about which categories to use, further decreasing the amount of time necessary to locate an object.

Intelligent Searching

The Epistemic Utility Function

In his book *The Enterprise of Knowledge* Issac Levi outlines a decision rule he terms the "epistemic utility" function, as follows:

$$Q(a) \geq bM(a)$$

where $Q(a)$ represents the probability that selecting alternative a from a set of choices avoids error, while $M(a)$ represents the informational value of rejecting alternative a . b represents the amount of boldness of the decision maker. (For example, if a person's boldness is high then they will be willing to make decisions based solely on informational value without thought for the truthfulness of the alternative -- so-called "wishful thinking.") The epistemic utility function can be viewed as a type of cost / benefit comparison whose critical points are decided by the constant b .

Before the Search

M -values (the value of not selecting the field in question) can be calculated for each of the metadata fields before a user ever reaches the search page. The value of not searching along one metadata field (dimension) is a function of the number of unique values of that field. For example, in a database with metadata records for 10,000 objects, the field "Title" will likely have close to 10,000 unique values, while a field like "Discipline" may only have 30 or 40 unique values. Thus the value of deciding not to filter the search using the Title category is extremely high (as electing to do so would mean searching through

10,000 entries), while searching through the Discipline field isn't nearly as costly.

Because each field has a different "length" (number of unique values), M-values for each field F ($M(F)$) are calculated by finding the total number T of unique values U , and dividing by the number of unique values in a specific field F .

$$T = \sum_{i=1}^n U_i$$

$$M(F_m) = \frac{U_m}{T}$$

This information (the M-values for each field) can be calculated at the moment the user initiates the search (when he loads the search screen, for example) without any user intervention.

Starting the Search

In the proposed search method, the user would select the three metadata fields s/he feels will be the most useful in conducting the search. The user then rank orders them as first, second, and third most useful. All this happens before the user even enters their search term. From this ranking of metadata fields the Q-values of the different fields are assigned.

For a value v , the first field is weighted $1/1v$, the second field is weighted $1/2v$, the third field is weighted $1/3v$, and the other fields are weighted equally ($1/4v$). These weights can be normalized by the probability mass function

$$S = \sum_{m=1}^n \frac{1}{v} F_m$$

which gives the Q-value ($Q(F)$) for each individual metadata field F as follows:

$$Q(F_m) = \frac{\frac{1}{v} F_m}{S}$$

As long as the number of metadata fields in the database remains constant, which it should if standards are being followed, Q-values can be calculated in advance. (Of course, they can always be recalculated if the metadata set is extended somehow.) The user ranking, then, simply associates the three highest Q-values with the three metadata fields selected, and assigns the standard value to the others metadata fields.

So what do we gain from all these gyrations? Good question.

Stepping through the Intelligent Search

Once the user has selected and ranked the three fields they feel are most likely to help them find the object they are looking for the search term is finally entered. This search term is searched against all the metadata fields in the entire database. But instead of returning the long list of search "results" in which the search term happens to occur, the search engine returns a list of the metadata fields that satisfy the epistemic utility function. That is, it returns the fields for which the Q -value is greater than the M -value times b ($b=1$ by default). These fields are listed in decreasing order of the distance between Q and M (i.e., the most benefit at the least cost first) as headings above the first of several columns, or perhaps in a drop-down box.

Within the first column are listed the values (filtered through the metadata field selected in the heading) for every item discovered via the keyword search. The other columns are empty. When a value from within the column is selected, the Q -values and M -values are recalculated without replacement, that is, without the metadata field represented by the current column. A new list of fields that meet the epistemic utility test is generated, and placed above the second column. The second column is then populated with the results of the initial search narrowed by the value selected in the first column. That is, the query is performed again, this time with two constraints (keyword = "some interesting word" and metadata field one's value = "the value you clicked on"). This process continues until the user finds an option that satisfies their information need. An example should clarify the process.

Sample User Experience

A user logs into the search screen, and proceeds to rank the metadata fields in the following order: Discipline, Author, and Age Level. The user then sets the threshold at which he wants to begin viewing results. Finally, he initiates a search for the term "gas." After appropriate calculations (as described above) and searching happen in the background, the heading Discipline appears in the drop-down box above the first column, and the values "Chemistry", "Geology", and "Physics" appear in the column beneath the heading. The user clicks on the value "Chemistry." Q and M -values are recalculated and the initial search is re-performed with the additional criterion that the field Discipline must equal "Chemistry." The number of search results is now narrowed to a third of the number originally returned (assuming that the metadata values are randomly distributed). The heading "Age Level" appears as the top of the second column and the values "K-3", "10-12", "Undergraduate", and "Graduate" appear in the second column. The user continues to select values and refine the search until he finds the information he is looking for in the bottom preview pane. Figure 4 shows a sample user interface.

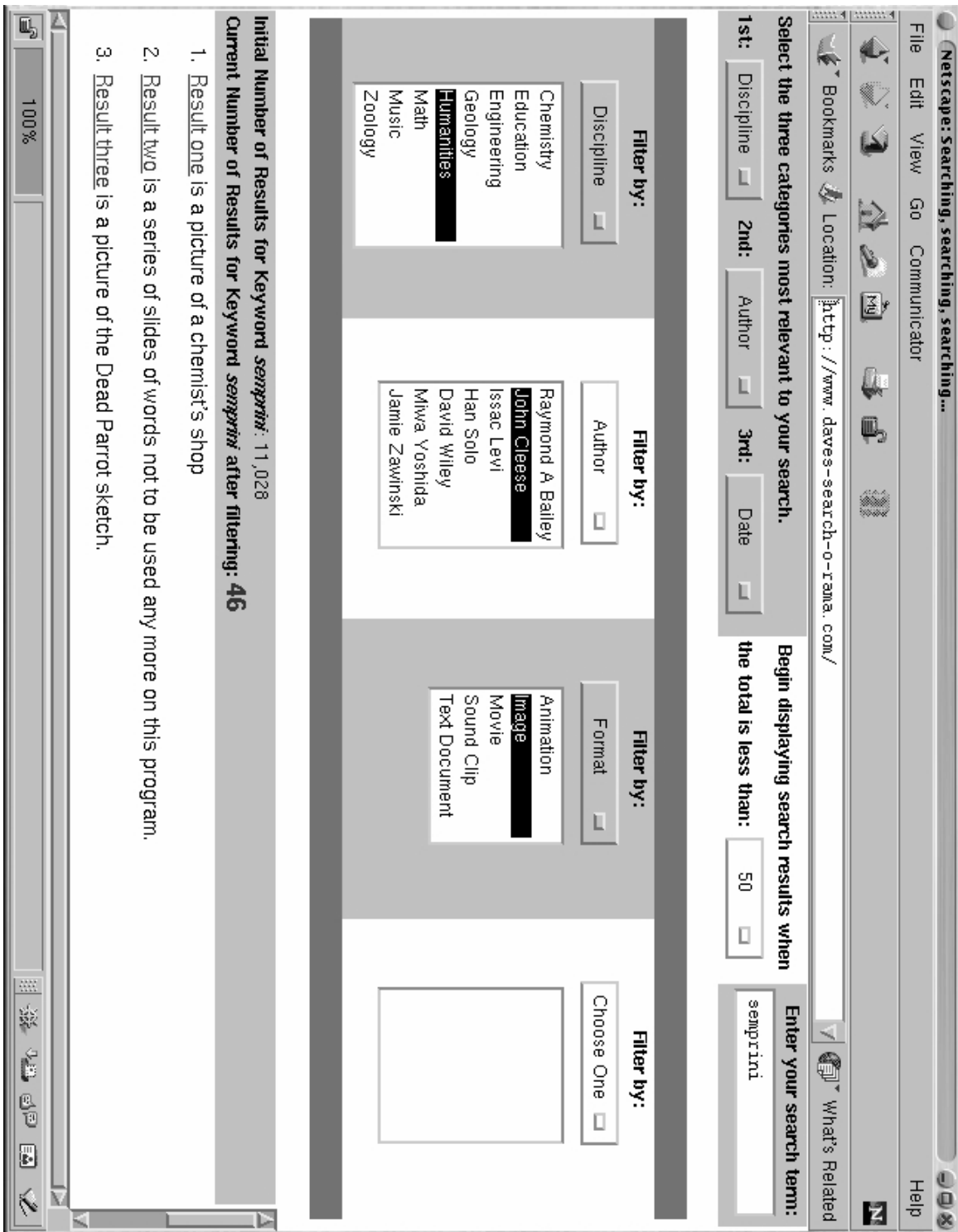


Figure 4. A sample user interface with a search session in progress.

Four Search Modes

Once the categorical filtering approach is understood and the Q and M machinery is in place, a number of search modes can be made available to the user.

1. Classic mode. For the user who insists on scrolling through 10,000 results, a simple keyword query that returns AltaVista-like results could be provided.
2. Speed mode. For the user willing to take some risks in order to save time, a quicker mode of searching could be provided. The user would set a results threshold, for example, 50 results. The search tool would then automate the process of choosing categories to filter along according to the $Q \geq M$ rule, by selecting the category for which the $Q - M$ value is greatest. The tools would iteratively filter the results until the number of remaining results was under the threshold. The user would then see AltaVista-like returns.
3. Filter mode. In this mode the tool does not use the epistemic utility function to make category recommendations. The user simply steps through the search, choosing their own filters as they go. This varies from the search described above in that the user can choose to filter with any of the categories, not just those that satisfy the epistemic utility function. It also requires less processing power per search.
4. Intelligent mode. This is the search technique described above.

Benefits of the Intelligent Search

The intelligent search has several benefits. Perhaps the foremost is that it enables the effective searching of huge repositories of data that might otherwise be too large to search. It does this by means of its drastically shortened time-to-location of object. When the user selects one of X values in the metadata field column, the total results under consideration are diminished by a factor of $X-1 / X$. As mentioned above, this powerful filtering approach can decrease 100,000 initial search results to single digits after filtering through only five or six columns. It does this in a few seconds. This is made possible not only by the power of the categorical filtering approach, but is also due to the n -dimensional nature of the data being searched and the recommendations made by the system using the epistemic utility function.

Another benefit, dependent on the user interface but certainly possible, is the ability of the searcher to constantly see their location within the metadata space. Most progressive search approaches move from screen to screen as they filter the search. By using columns within a single screen the user can know at all times exactly how far into the search s/he is, can easily backtrack, and can quickly see the results of changing a single filter.

Conclusion

This paper provides a conceptual framework for what I believe to be an improved way of searching IEEE 1484.12 metadata repositories. Because repositories like those referred to in this paper are just coming into existence, work in the area of search methodologies and user interfaces specifically for them is vital. I believe the concepts presented herein are clear enough that someone with sufficient time and talent could successfully implement them, so I will be working to find people to aid me in this effort. I believe these techniques will decrease the mean time to location of a satisfactory learning object.

References

Levi, Issac. (1983). The Enterprise of Knowledge. Cambridge: The MIT Press.

Wason, Thomas. (1997). Metadata Spaces. [On-line.] Available:
<http://www.imsproject.org/technical/metadata/History/did054.html>

Wason, Thomas. (Editor). (1999). IMS Metadata Specification Draft. [On-line.]
Available: http://www.imsproject.org/work_public/meta-data_did188.html).

Wiley et al. (1999). Three Common Properties of Efficient Online Instructional Support Systems. Manuscript in preparation.