An Architecture for Multischeming in Digital Libraries[⋆,⋆⋆]

Aaron Krowne, Edward A. Fox

Digital Library Research Laboratory
Virginia Tech
Blacksburg, VA 24061, USA
{akrowne, fox}@vt.edu

Abstract. In this paper we discuss the problem of handling many classification schemes within the context of a single digital library concurrently, which we term *multischeming*. We discuss how to represent which category describes an object in the digital library in this system, as well as the workings of the browsing process which is performed by the user. We motivate this problem as related to digital library interoperability, and propose an architecture for representation of classification schemes in the digital library which solves the problem. We also discuss its implementation in the CITIDEL project.

1 Introduction

The use of classification schemes to organize information for retrieval and storage has a long history. Especially in the last century, classification began to receive more methodical treatment within the library science community [1–3, 14, 15]. Chiefly because of economic pressures, Dewey created his Decimal Classification system in the late 19th century [4]. Its subsequent adoption and standardization sparked a wave of theoretical and practical advances in classification. Many new classification systems appeared, including the Library of Congress Classification (LC), Ranganathan's Colon Classification (CC) [14], the Bliss Bibliographic Classification (BBC), and many others. These and similarly-inspired schemes have provided a standard, consistent, and expansible means for library patrons to efficiently find what they want (or simply browse).

These classification systems, meant for general collections, have inspired smaller-scale efforts within more specific domains. This is in spite of the fact that the general schemes are "universal" and aim to have full coverage of all human knowledge. The causes of this are chiefly social: the community most interested in making the narrow subject domain more detailed is relatively small,

[⋆] This research work was funded in part by the NSF through grants DUE-0136690, DUE-0121679, and IIS-0086227.

[⋆⋆] Permission to reproduce this document, in full or in part, is hereby granted, provided that (1) attribution is maintained, and (2) any modifications are made clear as such.

and further, the community which administers the general schemes is not the same as the community which is continually pushing the frontiers of knowledge within the subject domain, necessitating revisions.

Examples of domain-specific classification schemes are the Medical Subject Headings (MeSH), the ACM Computing Classification System (ACM CCS), and the American Mathematics Society Mathematical Subject Classification (AMS MSC).

Classification schemes are now a nearly ubiquitous element of digital libraries. Since these digital libraries usually serve narrower communities than the general public, they typically contain materials from specialized subject domains. Therefore, they tend to utilize domain-specific classification schemes, like the ones listed previously. Further, the small scale of these communities (which may be fragmented globally and organizationally) and their focus on domain-specific work rather than standardization of the knowledge they produce has meant that multiple alternative schemes will often be in use. Lastly, ontologies not even intended for use as classification schemes may be profitably used as such, thus adding to the confusion. It is from this setting that our work emerges.

This paper is organized as follows. In the second section, we describe more specifics of the problem in the interoperable digital library setting, as well as some possible basic solutions and some requirements of a better solution. In the third section we describe how scheme-invariance is represented. In section four we present the scheme index, a structure necessary to make this representation work. In section five we discuss how the system was implemented in CITIDEL. In section six we introduce the distinction between scheme-level and category-level mappings. In section seven we discuss inter-scheme mapping quality. In section eight we present data and results from the CITIDEL implementation of the system. In section nine we discuss our system and a related system. In section ten we discuss limitations and future work, and then follow with concluding remarks.

2 DL Classification in an Interconnected World

Digital libraries (DLs) rarely exist in isolation. The Internet has naturally become the entry point to the modern digital library. However, the Internet also facilitates interconnectedness between digital libraries. This interconnectedness can and will be used for a variety of purposes, including distribution of digital library services (*federation*) and direct sharing of content (*harvesting*). Facilitating this interconnectivity is the motivation and goal of the Open Archives Initiative [12]. The need for federated services between digital libraries was the driving force behind the development of Dienst and the Networked Computer Science Technical Reference Library (NCSTRL) [10, 11].

The Computing and Information Technology Interactive Digital Educational Library (CITIDEL) is a digital library which makes heavy use of harvesting from other digital libraries to build its catalog of content. Thus we deal chiefly with the scenario of harvesting of content in this paper. The issues and solutions we discuss should, however, be applicable to all tasks that are distributed across digital libraries, as we discuss what is fundamentally a knowledge management issue.

We already have discussed how nearly every digital library will likely have some sort of classification scheme to organize its content. However, we also pointed out that there is no standard classification scheme. Thus, the individual digital library often will have to select arbitrarily from a field of alternative classification schemes. This implies that differing classification schemes will be used by different digital libraries, even though their content domains overlap significantly. With interconnectivity, this poses a problem: categorizations supplied by one digital library may be meaningless to another. This shuts out those resources from utilization through services which build upon classification.

An illustration: if a harvested object is marked as a “nut”, but “legume” describes the set of things that we call “nut” in our DL, we have a *classification collision*[1]. Now the user doesn’t know whether to look in category “nut” or “legume” to find this object. Further, if they are simply browsing and not looking for a specific object, their task has been complicated by the need to scan over the contents of two categories. In this case, it is clear that we have no provision for interpreting and acting upon the fact that “nut” *really is* “legume”[2]

Indeed, in many cases, different classification schemes are really describing the same “universe” of objects. This overlap of the universes of content of digital library collections is, in fact, why harvesting is done in the first place. One method of coping with this is to simply enforce standardization. Besides the fact that this is more easily said than done (and implies a significant wait), it is a misguided solution: domains of content (and classification schemes) often overlap only *partially.* We want to keep open the option of digital libraries sharing some subset of content which is the intersection of their domains, even if this intersection is not the same as the union. This mirrors the overlap in fields of knowledge. Those working with these fields have unique ontologies to describe and organize overlapping ideas or objects; these ontologies must accommodate their overall system of knowledge which others do not possess. Inasmuch as classification schemes are ontologies, we would lose something by their standardization.

On the other hand, we could fix the problem by supporting many classification schemes in parallel. Resources would simply appear under the schemes they were originally classified under (and no others). The result of this, however, would be a perpetual scavenger hunt through all of the schemes to find any resource, its original classification scheme being arbitrary. Furthermore, it is not really reasonable to expect users to memorize more than a few classification

[1] Note that in this terminology, it is the “description space” of the categories that is colliding, not the resources themselves. The result of this as the resources are concerned is that they become “far apart” in browsing space, rather than nearby, as would be preferred.

[2] To further compound the problem, note that it will grow in proportion to the number of schemes used by the digital libraries we harvest from. Now the user is in the position of having to guess whether the object they are looking for is classified as a “nut”, “legume”, “bean”, or “seed”, ad nauseum.

schemes which describe the same universe of content, nor is it likely this is practical. We are faced now with the problem of the user *only* being acquainted with one scheme when their desired resource is classified under another, and therefore "hidden".

Another possibility is to add category re-mapping (classification conversion) to the harvesting pipeline, where metadata transforms are often already present. This has a whole host of drawbacks. Firstly, it presupposes that interoperability is *only* harvesting, when in fact it also could be other forms of federated services (such as searching or browsing). Secondly, it adds transforms even when implementers have relied on not needing them (such as when building services upon pure Dublin Core). Third, category transformations are very complicated, and certainly are not implemented easily in XSLT [18], the standard system for transforming XML. Fourth, it is a bad idea to make lossy transformations permanent. Finally, if classification conversion mappings are produced, most of the work in using our more flexible system has already been done.

None of these possible solutions leads to an optimal outcome. We need to accommodate both arbitrary schemes used during classification, as well as user familiarity with (and preference for) one or a small number of schemes at access time. Thus, we need our system to "know" about the semantic equivalence of categories between schemes, and for it to act accordingly. Digital libraries can, and must, smooth over these organizational incompatibilities. We propose a system that affords a *scheme-agnostic* digital library; one where neither classification nor retrieval forces the selection of a particular classification scheme.

3 Multischeming Representation

How would a successful system "act accordingly" in the context of semantic equivalence of categories? We know that when we ask to see objects in category X, we really want to see all objects in X, all objects in categories that X *is the same as*, and all objects in categories that are *subsets* of these. Essentially, we want the user's browse request to be translated into a wider request – what their request really *would be* if they knew all the classification schemes in our DL and how the content areas they describe are related to each other. We call this a *multischeming system*[1].

To do this, the system needs information about the semantic relationship between schemes – connections between categories across schemes. This information takes the natural form of *mappings*.

The mappings are defined by the maintainers of the digital library, and are of the logical form "X is a Y", where X and Y are categories. Note that this takes care of the notion of subset; if X is a Y, but Y is not an X, then the category X is a subset of the category Y. This is our basic mapping primitive, and it is the

[1] This is not to be confused with *multiclassification*, which is the assigning of more than one category to a particular object. Our multischeming system does, however, subsume multiclassification.

only foundational element we need: to define equality, we simply tell the system "X is a Y" and "Y is an X".

We can represent these associations with a directed graph (Figure 1). "X is a Y" corresponds to a node, labeled X, and a node labeled Y, with an arrow *from* Y *to* X. This may seem a little backwards from the natural language statement, but it makes sense for information retrieval: when we are "at" node Y and want to know which nodes we can "get to" (or "see"), logically the question is answered by following the arrows *out of* Y.

One of the things we will store in our system, then, is links between category "nodes" of classification schemes which represent classifications we can "see" from each node. To represent the fact that an object has a particular classification (or classifications), we must separately store pointers between objects and these category nodes (this part is just standard classification). An object may have pointers to more than one node (multiclassification).

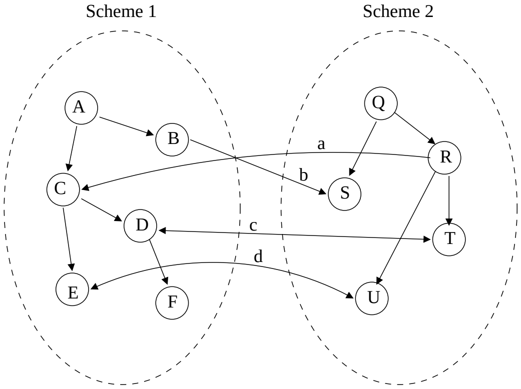


Fig. 1. A graphical representation of mappings between two schemes and their categories. A single arrow indicates that the destination category is a subset of the source. Arrows going both ways ("c" and "d") indicate that the source and destination categories are equivalent (they have equivalent content domains). Lettered arrows are inter-scheme mappings created by the DL maintainer; the other arrows are implicit in the hierarchy of the scheme.

The operation of browsing then works like this: the user selects a node (category) from a classification hierarchy with the intent of retrieving all objects which are classified as belonging in that category. This category is represented as a node, as discussed above, which we will call the "root node" of the browsing query. The system then does a look-up of this root node in the database and finds all nodes that the root node *points to*. This set is then merged with the set containing just the root node, and the search for objects is done on the augmented set. That is, the digital library translates the query "give me all objects in this category" to "give me all objects in this category or categories that are subsets of it"[1]. The category graph representation contains enough information for this question to be answered quite mechanically and succinctly in a standard relational DBMS.

4 Workhorse of the System: The Scheme Index

To make the above work, we need a structure which gives us the critical piece of information of all categories that should be considered in addition to each root query category. We call this structure the *scheme index*.

This index allows us to make the assumption that all the category nodes are linked together as fully as could possibly be determined by the set of "X is a Y" statements encoded by the system maintainers. What we mean by this is that, if "X is a Y", and "Y is a Z", the system should immediately have access to the transitive fact that "X is a Z" via the scheme index.

This notion should be extended to relationships not only once removed, but n-times removed; the "distance" between nodes in the classification graph is purely accidental. We are concerned with semantics the graph represents, which is invariant regarding the number of arrows between connected nodes.

However, the distance between nodes is a practical matter, as would translate to extra work for the system maintainers and/or for the database system[2]. It should be enough to declare that a category, X, which contains categories A, B, and C, "is a" new category Z. The maintainer should not have to explicitly state that "A is a Z", "B is a Z", etc.

With categories of classification schemes as nodes in a large graph, and edges of the semantic mappings between them as described above, it is clear that the solution to our problem is to automate the transitive closure of this graph, at which point it becomes a functional scheme index. In other words, if it is possible to travel through this graph from category node A to category node B, then the transitive closure algorithm will create an arrow directly from A to B. The algorithm does this for all pairs of nodes, which will complete the semantics of our inter-scheme mapping layer.

This mapping layer effectively becomes a semantic "index" over our many schemes, which can be used as described in the previous section to expand a root category node query into a set of nodes it is semantically equivalent to (as far as browsing is concerned). This set, then, is processed conventionally for the union list of objects which are in these categories.

5 Implemented System

In this section we describe the CITIDEL implementation of the architecture outlined above. A synopsis of the whole process is given in Figure 2.

[1] Note that here we do not mean *proper* subset.

[2] Standard relational database systems do not perform transitive closures, unfortunately.

Multischeming System Initiation Process

1. Extract schemes from foreign and local sources; normalize.
2. Import normalized schemes to database, populate scheme index with parent-child links.
3. Create inter-scheme mappings.
4. Augment scheme index with category links from mappings file.
5. Dump scheme index from database to file.
6. Read schemes into memory as matrix; perform transitive closure.
7. Output new set of category links in SQL insert statement format.
8. Replace scheme index in database by executing SQL from last step.
9. Use index for category browsing tasks in digital library.

Fig. 2. The steps to setting up our multischeme browsing system.

First, the scheme data was crawled from web sites, transformed by a script into a normalized syntax, and then imported into our database. At this stage, each category was given a unique identifier, and thus all categories become part of the same logical space (though they still kept a pointer to their originating scheme, for organizational purposes). The schemes we imported and support within CITIDEL are ACM CCS, ACM/IEEE Computing Curricula (CC, both 1991 and 2001 versions), the CoRR Subject Areas, and the Mathematical Subject Classification (MSC, which has a large computing sub-branch)[1,2].

Also automatically generated at scheme import time are some of the contents of the mapping "index": namely the portion that maps parent to child category *within the same scheme*. This parent-child information is all we need to leverage the transitive closure of our browsing engine such that we will get resource lists (and counts) at a parent node which take into account resources in all of its *descendent* nodes. For example, a category "Science" might have sub-category "Physics", which in turn has sub-category "Condensed-matter physics". With our system, the two implicit "has-child" relationships "Science `has-child` Physics" and "Physics `has-child` Condensed-matter physics" are all that are needed to infer that "Science" contains all resources from "Condensed-matter physics". Hence, our system solves for "free" the normally messy transitive-containment problem in browsing by hierarchical classification schemes.

The scheme mappings in our system are made up of category mapping statements of the form `contains C1, C2`, for categories `C1` and `C2` (which are of course typically in separate schemes). A simple ASCII text file, `mappings.conf`,

[1] To browse these, see <http://www.citidel.org/?op=cbrowse>.

[2] A wonderful side-effect of using our system is that when schemes are revised, one can smoothly transition to the new scheme by simply loading in both schemes and creating mappings between them. The mappings are easier to create the less that has changed; one could even automate the generation of mappings to same-named categories, then fill in the rest by hand. Optionally, one could drop the support of the old scheme from the user interface, as all resources classified under it will appear under the new scheme automatically.

is made up of lines of this form, and serves as input to the transitive closure portion of our system.

A portion of this file, mapping ACM CCS to CC, is given in Figure 3.

```
# computer vision > image processing and computer vision
contains CC2001.GV11, CCS1998.I.4

# intelligent systems > artificial intelligence
contains CC2001.IS, CCS1998.I.2

# intelligent systems > pattern recognition
contains CC2001.IS, CCS1998.I.5

# fundamental issues in intelligent systems > artificial intelligence
contains CC2001.IS1, CCS1998.I.2

# information management > information systems
contains CC2001.IM, CCS1998.H

# information models and systems > models and principles
contains CC2001.IM1, CCS1998.H.1

# information models and systems > information systems applications
contains CC2001.IM1, CCS1998.H.4
```

Fig. 3. A portion of CITIDEL's inter-scheme mapping file. This is from the section mapping ACM CCS *into* CC 2001. That is, it supplies the data necessary to determine which categories from CCS can be "seen" from each category within CC.

In our system, we map ACM CCS into all other schemes, and this is sufficient for ensuring that all resources classified under ACM CCS will appear when browsing by all other schemes, in addition to resources natively classified under those schemes. However, because we are not yet mapping the other schemes into ACM CCS as well, there are limitations in our instance of the system which have practical consequences we will discuss in the next section.

The next step after importing the schemes and writing the mappings file is to augment the contents of the current mapping index in the database with the contents of the mappings file. This is simple enough. It is handled by a script that reads and parses the mappings file and writes out links of the form `(category_a, category_b)` (where `b` is mapped *into* `a`) to the index table in the database.

After this step, the index is exported to disk, then read by a C-language transitive closure program. This program builds a large in-memory matrix from the data, interpreting category node identifiers as matrix indices. In our system this results in a greater than 6000×6000 matrix. Transitive closure is run on this matrix, producing a new matrix with the necessary transitive links. The program then outputs the results as insert statements of pairs `(category_a, category_b)`, which are used to re-populate the scheme index table in the database. At this point, the index is complete.

Within CITIDEL, the "browse by subject" area leads to a simple tabbed "scheme navigator" interface, where each tab is a scheme. ACM CCS is selected by default (but that can be reconfigured by the user). Below the selected tab is displayed the list of categories at the current level, along with the count of resources in that category (and all categories within it) *as well as all categories in other schemes visible from it*. Other than at the top level in a scheme, a synopsis-style list of resources is displayed below the scheme navigator, once again, containing even resources classified in other schemes, as long as they were mapped into the current category.

When the user clicks on a category, the display reconfigures as is typical for standard category browsing interfaces, showing categories within the newly-selected category, and the resource list is updated to display synopses for resources at the current node or any category node mapped to it via our system.

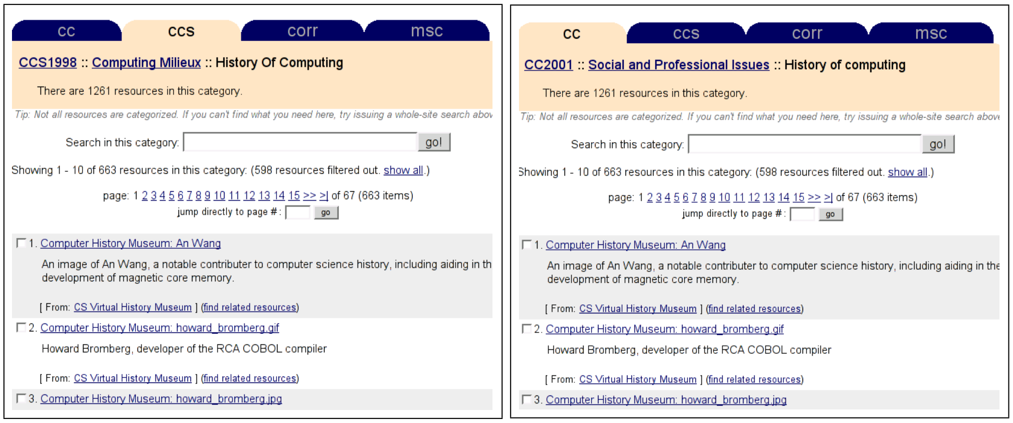
Users also can initiate text searches from any classification category node. This dispatches a search to the search engine, then narrows the returned list of results, using the union of the current category and all categories mapped into it as a filter on the results set.

In this manner, CITIDEL allows users to browse by their favorite classification schemes without worry of "missing out" on resources categorized under the other four currently-supported schemes. In addition, they have the option of utilizing more than one scheme if they choose, so they can exploit the strengths of one or the other depending on the information finding task. All of this is entirely without worry of whether the resource desired was classified under this or that scheme. In Figure 4, we give a demonstration of classification invariance on CITIDEL by way of side-by-side comparison.

6 Mappings and Schemes

There are really two logical levels to mappings, though there is no distinction at the implementation level. The category (or "low") level, already discussed, consists of mappings between the categories which are the "atoms" of the multischeming system and the primitive elements of all classification schemes. This is the level the system actually works with. However, we can think of the scheme (or "high") level (Figure 5) as mappings between entire classification schemes; which was really the goal of this entire enterprise.

We can view each scheme as a node in a graph, along with directed edges to other scheme nodes. We draw an arrow from scheme A to B when every category of A is mapped to a category of B. We would expect that normally schemes are mapped both ways (that is, their categories are co-mapped to each other). Optimally, then, we can drop the arrows and replace them with undirected edges.



CCS1998 :: Computing Milieux :: History Of Computing

There are 1261 resources in this category.

Tip: Not all resources are categorized. If you can't find what you need here, try issuing a whole-site search abov

Search in this category: go!

Showing 1 - 10 of 663 resources in this category. (598 resources filtered out. show all.)

page: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 >> >| of 67 (663 items)
jump directly to page #: go

□ 1. Computer History Museum: An Wang

An image of An Wang, a notable contributer to computer science history, including aiding in th development of magnetic core memory.

[From: CS Virtual History Museum] (find related resources)

□ 2. Computer History Museum: howard_bromberg.gif

Howard Bromberg, developer of the RCA COBOL compiler

[From: CS Virtual History Museum] (find related resources)

□ 3. Computer History Museum: howard_bromberg.jpg

CC2001 :: Social and Professional Issues :: History of computing

There are 1261 resources in this category.

Tip: Not all resources are categorized. If you can't find what you need here, try issuing a whole-site search above

Search in this category: go!

Showing 1 - 10 of 663 resources in this category. (598 resources filtered out. show all.)

page: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 >> >| of 67 (663 items)
jump directly to page #: go

□ 1. Computer History Museum: An Wang

An image of An Wang, a notable contributer to computer science history, including aiding in the development of magnetic core memory.

[From: CS Virtual History Museum] (find related resources)

□ 2. Computer History Museum: howard_bromberg.gif

Howard Bromberg, developer of the RCA COBOL compiler

[From: CS Virtual History Museum] (find related resources)

□ 3. Computer History Museum: howard_bromberg.jpg

Fig. 4. A side-by-side comparison of browsing the history categories of the Computing Classification System and Computing Curricula schemes on CITIDEL. Note that the list of resources (the top of which you can see here) is the same in both cases. However, these resources are *only* classified under CCS.

However, we might not be able to do this if, for example, one scheme's content domain is just a small part of that of another scheme.

Using our multischeming system, the digital library maintainer need only make sure this high-level graph is *connected* for the system to be able to guarantee that an object classified under one scheme will show up under all schemes. This implies that a new scheme can be added to the system by making mappings to a single scheme in an already-connected graph. For convenience, all new schemes could be mapped to some canonical scheme C. In this case, the DL maintainer can leverage high familiarity with C to produce mappings rapidly and of a high quality. Alternatively, the DL maintainer can be opportunistic, and map a novel scheme to whatever is perceived to be the most similar scheme in the graph.

7 Mapping Quality

The previous section presented a rosy view of the scheme-level mapping situation. However, even in our CITIDEL implementation, we have not attained this ideal, which raises the issue of *mapping quality*. We introduce some definitions to aid in discussing and understanding this.

- A mapping M from scheme A to scheme B (or of scheme A *into* B) is a set of ordered pairs of the form (a, b), each of which defines a link from category a in A to a category b in B.
- We say that M is *complete* when there exists no $c_a \in A$ such that M lacks a pair of the form (c_a, b). In other words, every category of A is mapped to some category in B.

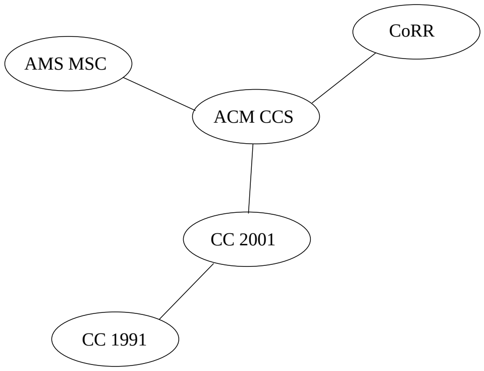


Fig. 5. A scheme-level view of optimal mappings between classification schemes (based on CITIDEL). Here we have formed a connected graph by mapping most schemes to "ACM CCS" and vice versa, except for "CC1991" and "CC2001", which are very close to each other. Note that the undirected edges mean that the mappings go both ways; an object classified with an "AMS MSC" category is viewable from ACM CCS, and vice versa. Since the graph is connected, this is true for all pairs of schemes.

- Let M_{AB} and M_{BA} be complete mappings from A to B and B to A, respectively. We say that $M = \{M_{AB}, M_{BA}\}$ is a *symmetric mapping* between A and B, or that A and B are *symmetrically mapped*.

Thus, a more precise way of saying what was said in the previous section is that if the undirected graph formed by symmetrically mapped schemes in a digital library is *connected*, then every resource classified *anywhere* will be visible *somewhere* in every scheme.

This makes intuitive sense: a complete mapping means a resource classified in a source scheme *must* appear somewhere in the destination scheme; symmetric mappings mean that the same fact applies symmetrically, and a connected graph of such mappings (along with the use of transitive closure in our system) extends this symmetric relationship across all levels of graph indirection. This is what we mean by "completely scheme-invariant classification".

Due to limitations discussed later, in CITIDEL we only have complete mappings (mapping every other scheme to ACM CCS), and have *no* symmetric mappings at the present time. However, at the moment we mostly have resources classified under either CCS exclusively or CCS and X (where X is one of the other schemes). This means we can get "effective" classification invariance with our current set of resources simply by mapping ACM CCS *into* all schemes. Since objects classified natively in other schemes also happen to have CCS classifications, we luck out and do not yet need to map other schemes *into* CCS. This situation is illustrated in Figure 6.

The notion of quality discussed above can be extended. Completeness is a very rough metric; it does not take into account mapped categories which are

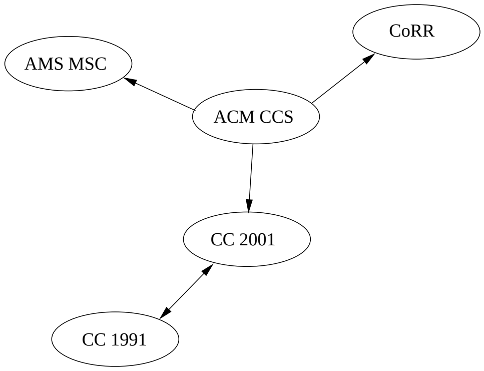


Fig. 6. A scheme-level view of a mapping situation closer to the current one on CITIDEL. "ACM CCS" is mapped *into* most other schemes, which means resources classified under CCS will appear when browsing via the other schemes. For resources natively classified under another scheme to appear under CCS, they must be multiclassified under both schemes (as some of our resources are). Here we have shown the two versions of CC as symmetrically mapped; an economical task due to their similarity. Note that, using this graph, one can intuit where resources classified under one scheme can appear by following the arrows.

mapped to the wrong places or mapped to the same place, for instance. Indeed, the notion of "wrong" may not be amenable to consensus at all. Thus, scheme mappings are ultimately as subjective as the schemes themselves. One could say that scheme mappings inherit the subjective nature of classification itself.

However, there does seem to be room for guiding metrics about the quality of inter-scheme mappings, based on notions of loss of information, precision, or accuracy. We leave development of such metrics for future work.

8 Results

In Table 1, we give statistics for our application of multischeming in CITIDEL. Transitive interscheme links are the most important links which are added by the transitive closure portion of our system, as they extend the category mappings to the set of mappings that is truly needed for scheme-invariant object visibility to work properly. The "virtual classifications" statistics show the number of true classifications that would be needed to match the effect of multischeming in our implementation. Note that we have attained the effect of a nearly ten-fold classification verbosity without storing such a massive quantity of categorizations.

In Table 2, we give object inter-scheme visibility data. This consists of counts of objects visible *natively* in a scheme (those which are classified under categories in that scheme within their metadata), and those which are visible from categories in that scheme through our multischeming system. Clearly the data indicate that most resources, visible only through ACM CCS natively, become visible in all of the other schemes with multischeming.

This serves as a rudimentary proof that a multischeming system can, even without complete and symmetric mappings, make objects exportable to other schemes.

Note that the counts are not all maximal (89,150) due to imperfections in the mappings. Still, the vast majority of resources become "portable", despite the small amount of work we put into the mappings, which we think is a testament to its utility.

These results do not close the book on evaluation: still useful would be studies comparing the end-user retrievability of mapped and unmapped resources (a more precise metric of mapping quality than we have given here), as well as studies of the usability of an interface which presents alternative schemes. Still, we think the figures here and the running system they are derived from are encouraging.

Table 1. Multischeming statistics for CITIDEL. In this table, "intrascheme links" refers to links between categories in the *same* classification scheme (implicit), and "interscheme links" refers to links between categories in different classification schemes (multischeming mappings). "Virtual classifications" are the effective classifications emulated via multischeming.

Schemes	4
Total categories	6,166
Total category mappings	244
Scheme index total links	12,419
Parent-child links	5,922
Transitive links	6,253
Intrascheme links	10,745
Transitive ancestor links	4,823
Interscheme links	1,674
Transitive interscheme links	1,430
Classified resources	89,150
Classifications	241,723
Average classifications/object	2.7
Virtual classifications	1,968,488
Average virtual classifications/object	22.1

Table 2. Counts of CITIDEL objects natively classified in each scheme, and visible under that scheme through the multischeming system.

Scheme	Native Resources	Multischemed Resources
CCS	89,150	89,150
CC	0	81,947
CoRR	0	81,222
MSC	0	80,138

9 Discussion and Related Work

The architecture we have described above seems to have few drawbacks. The computational complexity does not increase as the digital library grows in size; instead it depends only on the number of categories within all classification schemes understood by the system. This complexity, however, only surfaces at the time of initializing or adding schemes. Processing for transitive closure need only be done offline at these times in order to update the scheme index, and is tractable on current machines for even a large number of categories (for 6000+ categories we found the transitive closure took less than a minute to run on a Pentium III 800MHz machine).

Nor does the addition of more schemes pose an increasing amount of work for the system maintainer; a new scheme need only be mapped to one other scheme for the complete invariance of our system to work.

Multiple classifications for a single object are handled naturally by representing classification as a pointer, which could be one among many, from one object into the set of classification category nodes. Adding more categories to an object poses no computational or architectural challenge to the system.

All of this has been demonstrated to be tractable, workable, and usable within CITIDEL, which is a large-scale setting[1].

There exists a system called "Renardus"[2] [13], which is similar in many respects to the multischeming system we have proposed here. Renardus considers itself a *gateway* or *brokering service*; that is, its goal is to be a go-between for users among many disparate digital library collections. In this spirit, Renardus will direct users to resources from many source collections through its text search and classification browsing systems.

Differing from most of our discussion here, Renardus has a universal subject scope. To provide its classification browsing, all schemes are mapped into the Dewey Decimal Classification (DDC). However, the user is not entirely limited to DDC, as the mapping relationships are exposed at browse time via lists of categories which are related to the current DDC category. These relations are of the type "Narrower Equivalent", "Fully Equivalent", "Minor Overlap With", and "Major Overlap With", and appear in separate lists.

However, upon clicking any of these related category hyperlinks, the user is whisked out of Renardus and to the remote digital library, suddenly browsing in a completely different format. This change in interface could be jarring to users. In addition, it is questionable whether exposing category mapping relations within a category hierarchy is not too confusing in the first place.

While not a multischeming system (it is lacking the *retrieval* half of scheme-invariance), the Renardus architecture has much in common with ours. Its sup-

[1] CITIDEL carries about 450,000 resources in its current catalog, with near 1,000,000 expected by 2004.

[2] See <http://renardus-broker.sub.uni-goettingen.de/>.

port for many mapping relationships goes beyond our containment/equivalence model[1], and is certainly worth considering.

10 Limitations and Future Possibilities

There was a conspicuous omission from the above presentation: how to create the scheme mappings which are the semantic basis of the entire system. Indeed, this is one of the greatest limitations of the current system; we have no elegant method to create the mappings. This is a major reason why we have not created full mappings for CITIDEL, settling instead for "sufficient" mappings, given the characteristics of our collection.

Hence, one of the most attractive possibilities for the future would be to develop a program which would read in a classification scheme in a standard format, then provide a convenient graphical interface for drilling down into the schemes and drawing connections between certain parts. It also would be useful to maintain progress metrics regarding how much of the mapping is complete, and what the fidelity of the mapping is. At the end, the program could write out the mappings in the proper format.

This process could even be bootstrapped by automated inference, utilizing resources classified across schemes to suggest initial mapping links. For our purposes, however, this was not a useful approach to develop, as we did not have many resources which were cross-classified.

This tool could accelerate the development of standard mappings between schemes in certain domains, which could then be disseminated widely, eliminating the need for digital library maintainers to make common inter-scheme mappings.

There have been related efforts which could possibly apply here. As we mentioned earlier, classification schemes are really just a type of ontology. Also qualifying as ontologies are thesauri, which can be seen as a generalization of dictionaries [17]. Work in thesauri has lead to many knowledge-based systems to do cross-language information retrieval, due to their ability to make connections between concepts [6–8, 16]. In fact, the types of relationships exposed by Renardus are precisely the kind one would see in a thesaurus, suggesting that thesauri subsume classification schemes [5, 9] and indicating that methods and tools used to work with thesauri may be applicable to our system. This possible bridging between fields deserves further attention.

11 Conclusion

In this paper, we have introduced classification schemes and discussed their importance in digital libraries. We then discussed how the nature of specialized

[1] Note that in our model, "overlap" relationships cannot be expressed elegantly: to avoid damaging resource recall, the mapper must map a category to multiple destinations if its content belongs to each of them in part.

communities of study and interoperability among digital libraries introduces the problem of "colliding" classification schemes. We proposed a model that accommodates multiple classification schemes in a single digital library such that browsing is classification-agnostic. We discussed the details of the scheme index which is central to implementing this model, and the process of creating the mappings which it relies on. We discussed the implementation of this system in CITIDEL, and its scalability. We discussed a related system (Renardus) and compared it with our multischeming system. We also exposed limitations and discussed future possibilities, where connections might be made to work in thesauri.

We hope that this paper has focused attention on the looming problem of classification collision in the interoperable digital libraries environment, as well as our proposed and implemented system which can be used to solve this problem. We think our system turns this crisis into an opportunity: one for more powerful and flexible digital library features for end users and an improved overall digital library experience.

Acknowledgements

We would like to thank the ACM for sharing their content and classification schemes with us. We also thank colleagues in the Digital Library Research Lab at Virginia Tech for their useful suggestions.

References

1. Bakewell, K. G. B.: *Classification and Indexing Practice*. Linnet Books, Hamden, Conn. (1978)
2. Bengtson, Betty G., Janet S. Hill (eds.): *Classification of Library Materials*. Neal-Schuman Publishers, Inc., New York, New York (1990)
3. Bliss, Henry E.: *The Organization of Knowledge in Libraries*. The H. W. Wilson Company, New York (1939)
4. Chan, Lois M., John P. Camaromi, Joan S. Mitchell, Mohinder P. Satija: *Dewey Decimal Classification: A practical guide*. Forest Press, Albany, New York (1996)
5. Dykstra, M.: LC subject headings disguised as a thesaurus. *Library Journal* 113(4), 42-46. (1998)
6. Eichmann D., M. Ruiz, and P. Srinivasan. Cross-Language Information Retrieval with the UMLS Metathesaurus. In: *Proc. of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Melbourne, Australia* (1998)
7. Gonzalo J., F. Verdejo, and I. Chugur: Using EuroWordNet in a Concept-based Approach to Cross-Language Text Retrieval. In: *Applied Artificial Intelligence*, Vol. 13 (1999)
8. Hull D. A., and G. Grefenstette. Querying Across Languages: A Dictionary based Approach to Multilingual Information Retrieval. In: *Proc. of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ACM SIGIR (1996)

9. Kamps, Thomas, Christoph Hüser, Wiebke Möhr, Ingrid Schmid: Knowledge-based Information Access for Hypermedia Reference Works: Exploring the Spread of the Bauhaus Movement. In: Maristella Agosti and Alan Smeaton (eds.): *Information Retrieval and Hypertext*. Kluwer Academic Publishers, Norwell, Massachusetts (1996)
10. Lagoze C.: Dienst - An Architecture for Distributed Document Libraries. *Communications of the ACM*, Vol. 38 No. 4 page 47 (April 1995)
11. Lagoze, C.: The Networked Computer Science Technical Reports Library. In: *Cornell Computer Science Technical Reports* (July 1996) <http://techreports.library.cornell.edu:8081/DPubS/UI/1.0/Browse>
12. Lynch, Clifford: Metadata Harvesting and the Open Archives Initiative. *ARL Monthly Report* No. 127 (August 2001) <http://www.arl.org/newsltr/217/mhp.html>
13. Neuroth, Heike, Traugott Koch: Metadata Mapping and Application Profiles. Approaches to providing the Cross-searching of Heterogeneous Resources in the EU Project Renardus. In: *Proceedings of the International Conference on Dublin Core and Metadata Applications* (2001) <http://www.nii.ac.jp/dc2001/proceedings/abst-21.html>
14. Ranganathan, S. R.: The Colon Classification. In: *Systems for the Intellectual Organization of Information*, Vol. IV. Rutgers University Press, New Brunswick, New Jersey (1965)
15. Tauber, Maurice F., Edith Wise: Classification Systems. In: Ralph R. Shaw, (ed.): *The State of the Library Art*, Vol. 1(3). The Rutgers University Press, New Brunswick, New Jersey (1961)
16. Verdejo, Felisa, Julio Gonzalo, Anselmo Peñas, Fernando López, David Fernández: Evaluating wordnets in Cross-Language Information Retrieval: the ITEM search engine. In: *Proceedings of the International Conference on Language Resources & Evaluation* (2000)
17. Fellbaum, Christine (ed.): *WordNet: An Electronic Lexical Database*. MIT Press, Boston, Mass. (1998)
18. XSL and XSLT. <http://www.w3.org/Style/XSL/>