Information Discovery Insights Gained from MultiPAC, a Prototype Library Discovery System

Alex A. Dolski

At the University of Nevada Las Vegas Libraries, as in most libraries, resources are dispersed into a number of closed "silos" with an organization-centric, rather than patron-centric, layout. Patrons frequently have trouble navigating and discovering the dozens of disparate interfaces, and any attempt at a global overview of our information offerings is at the same time incomplete and highly complex. While consolidation of interfaces is widely considered to be desirable, certain challenges have made it elusive in practice.

MultiPAC is an experimental "discovery," or meta-search, system developed to explore issues surrounding heterogeneous physical and networked resource access in an academic library environment. This article discusses some of the reasons for, and outcomes of, its development at the University of Nevada Las Vegas (UNLV).

The case for MultiPAC

Fragmentation of library resources and their interfaces is a growing problem in libraries, and UNLV Libraries is no exception. Electronic information here is scattered across our Innovative WebPAC; our main website, our three branch library websites; remote article databases, local custom databases, local digital collections, special collections, other remotely hosted resources (such as LibGuides), and others. The number of these resources, as well as the total volume of content offered by the Libraries, has grown over time (figure 1), while access provisions have not kept pace in terms of usability.

In light of this dilemma, the Libraries and various units within have deployed finding and search tools that provide browsing and searching access to certain subsets of these resources, depending on criteria such as

- the type of resource;
- its place within the libraries' organizational structure;
- its place within some arbitrarily defined topical categorization of library resources;
- the perceived quality of its content; and
- its uniqueness relative to other resources.

Alex A. Dolski (alex.dolski@unlv.edu) is Web & Digitization Application Developer at the University of Nevada Las Vegas Libraries.

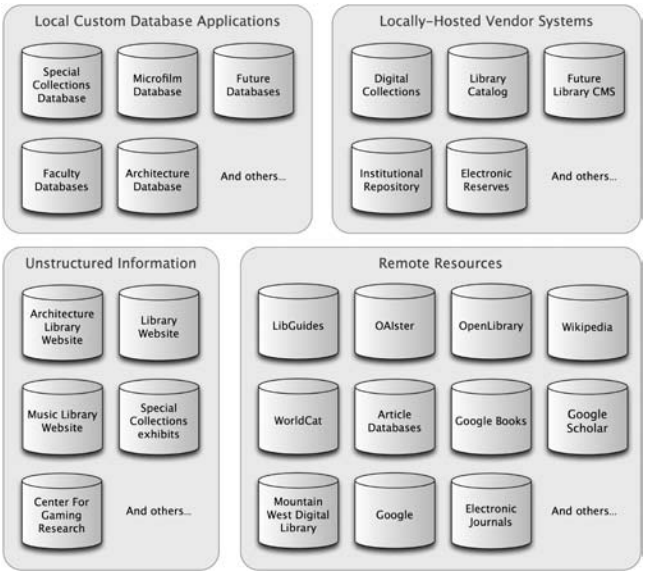

Figure 1. "Silos" in the library

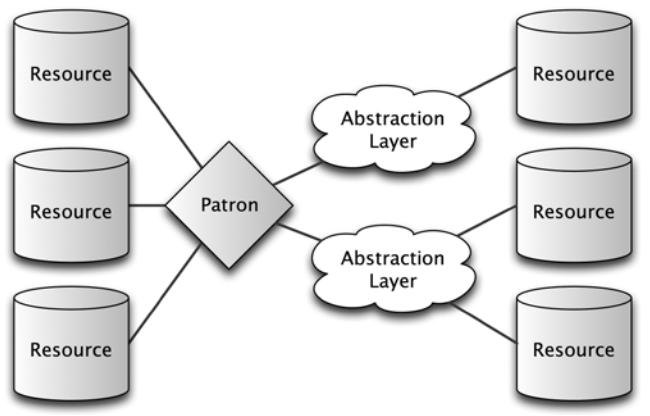

Figure 2. Organization-centric resource provisioning

These tools tend to be organization-centric rather than patron-centric, as they are generally provisioned in relative isolation from each other without placing as much emphasis on the big picture (figure 2). The result is, from the patron's perspective, a disaggregated mass of information and scattered finding tools that, to varying degrees, each accomplishes its own specific goals at the expense of macro-level findability. Currently, a comprehensive search for a given subject across as many library resources as possible might involve visiting a half-dozen interfaces or more—each one predicated upon awareness of each individual interface, its relation to the others, and

the characteristics of its specific coverage of the corpus of library content.

Our library website serves as the de facto gateway to our electronic, networked content offerings. Yet usability studies have shown that findability, when given our website as a starting point, is poor. Undoubtedly this is due, at least in part, to interface fragmentation. Test subjects, when given a task to find something and asked to use the library website as a starting point, fail outright in a clear majority of cases.[1]

MultiPAC is a technical prototype that serves as an exploration of these issues. While the system itself breaks no new technical ground, it brings to the forefront critical issues of metadata quality, organizational structure, and long-term planning that can inform future actions regarding strategy and implementation of potential solutions at UNLV and elsewhere. Yet it is only one of numerous ways that these issues could be addressed.[2]

In an abstract sense, MultiPAC is biased toward principles of simplification, consolidation, and unification. In theory, usability can be improved by eliminating redundant interfaces, consolidating search tools, and bringing together resource-specific features (e.g., OPAC holdings status) in one interface to the maximum extent possible (figure 3). Taken to an extreme, this means being able to support searching all of our resources, regardless of type or location, from a single interface; abstracting each resource from whatever native or built-in user interface it might offer; and relying instead on its data interface for querying and result-set gathering. Thus MultiPAC is as much a proof-of-concept as it is a concrete implementation.

Table 1. Some popular existing library discovery systems

Name	Company/Institution	Commercial Status
Aquabrowser	Serials Solutions	Commercial
Blacklight	University of Virginia	Open-source (Apache)
Encore	Innovative Interfaces	Commercial
eXtensible Catalog	University of Rochester	Open-source (MIT/GPL)
LibraryFind	Oregon State University	Open-source (GPL)
MetaLib	Ex Libris	Commercial
Primo	Ex Libris	Commercial
Summon	Serials Solutions	Commercial
VuFind	Villanova University	Open-source (GPL)
WorldCat Local	OCLC	Commercial

Table 2. Some existing back-end search servers

Name	Company/Institution	Commercial Status
Endeca	Endeca Technologies	Commercial
IDOL	Autonomy	Commercial
Lucene	Apache Foundation	Open-source (Apache)
Search Server	Microsoft	Commercial
Search Server Express	Microsoft	Free
Solr (superset of Lucene)	Apache Foundation	Open-source (Apache)
Sphinx	Sphinx Technologies	Open-source (GPL)
Xapian	Community	Open-source (GPL)
Zebra	Index Data	Open-source (GPL)

Background: How MultiPAC became what it is

MultiPAC came about from a unique set of circumstances. From the beginning, it was intended as an exploratory project, with no serious expectation of it ever being deployed. Our desire to have a working prototype ready for our Discovery Mini-Conference meant that we had just six weeks of development time, which was hardly sufficient for anything more than the most agile of

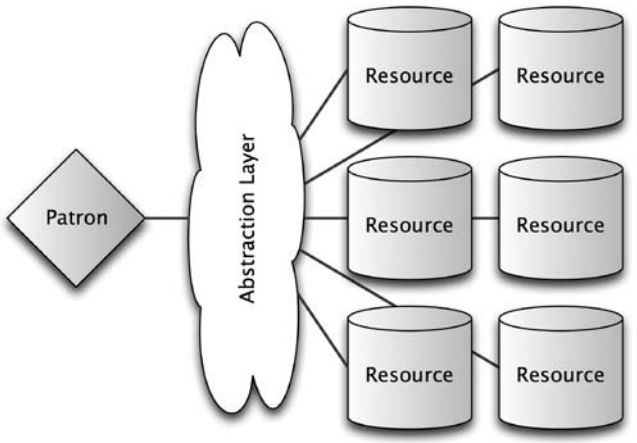

Figure 3. Patron-centric resource provisioning

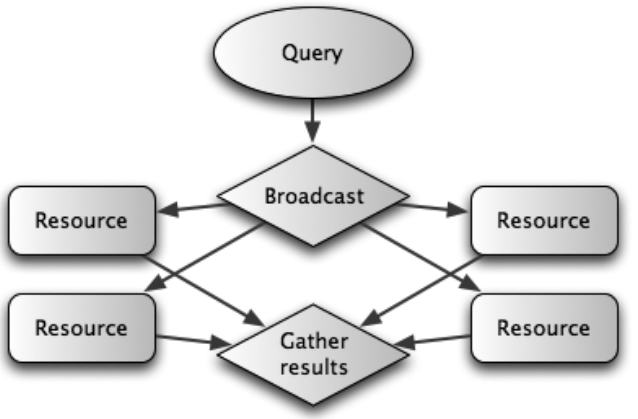

Figure 4. The federated search process

development models. The resulting design, while foundationally solid, was limited in scope and depth because of time constraints.

Another option, instead of developing MultiPAC, would have been to demonstrate an existing open-source discovery system. The advantage of this approach is that the final product would have been considerably more advanced than anything we could have developed ourselves in six weeks. On the other hand, it might not have provided a comparable learning opportunity.

Survey of similar systems

Were its development to continue, MultiPAC would find itself among an increasingly crowded field of

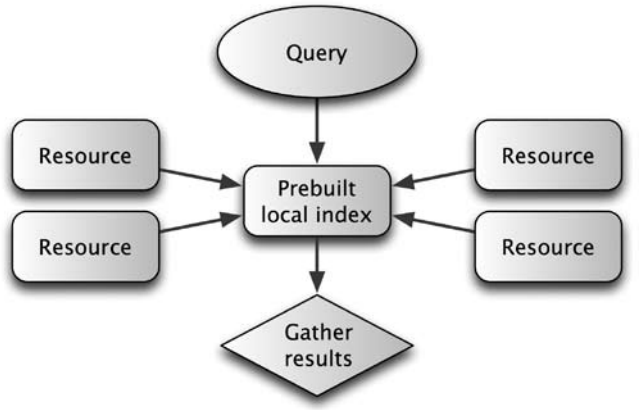

Figure 5. The harvested search process

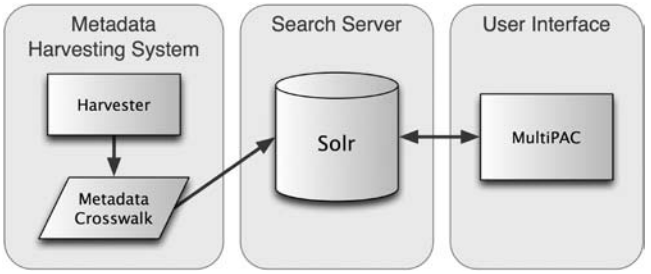

Figure 6. The three main components of MultiPAC

competitors (table 1). A number of library discovery systems already exist, most backed by open-source or commercially available back-end search engines (table 2), which handle the nitty-gritty, low-level ingestion, indexing, and retrieval. These lists of systems are by no means comprehensive and do not include notable experimental or research systems, which would make them much longer.

Architecture

In terms of how they carry out a search, meta-search applications can be divided into two main groups: distributed (or federated search), in which searches are "broadcast" to individual resources that return results in real time (figure 4); and harvested search, in which searches are carried out against a local index of resource contents (figure 5).[3] Both have advantages and disadvantages beyond the scope of this article. MultiPAC takes the latter approach. It consists of three primary components: the search server, the user interface, and the metadata harvesting system (figure 6).

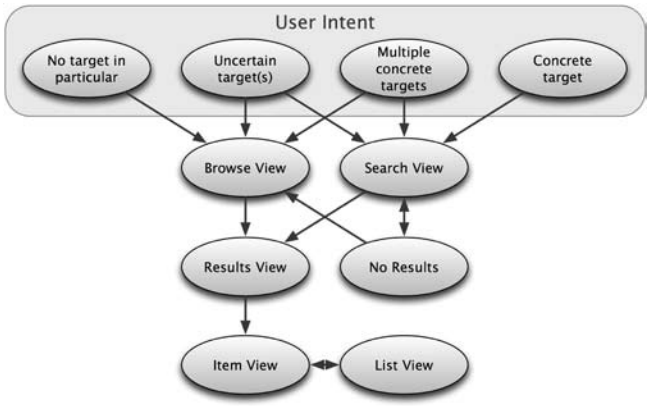

Figure 7. The information-finding process supported by MultiPAC

Search server

After some research, Solr was chosen as the search server because of its ease of use, proven library track record, and HTTP–based representational state transfer (REST) application programming interface (API), which improves network-topological flexibility, allowing it to be deployed on a different server than the front-end Web application—an important consideration in our server environment.[4] Jetty—a Java Web application server bundled with Solr—proved adequate and convenient for our needs.

The metadata schema used by Solr can be customized. We derived ours from the unqualified Dublin Core metadata element set (DCMES),[5] with a few fields removed and some fields added, such as "library" and "department," as well as fields that support various MultiPAC features, such as thumbnail images, and primary record URLs. DCMES was chosen for its combination of generality, simplicity, and familiarity. In practice, the Solr schema is for finding purposes only, so whether it uses a standard schema is of little importance.

User interface

The front-end MultiPAC system is written in PHP 5.2 in a model-view-controller design based on classical object design principles. To support modularity, new resources can be added as classes that implement a resource-class interface.

The MultiPAC HTML user interface is composed of five views: search, browse, results, item, and list, which exist to accommodate the finding process illustrated in figure 7. Each view uses a custom HTML template that can be easily styled by nonprogrammer Web designers. (Needless to say, judging by figures 8–12, they haven't been.) Most dynamic code is encapsulated within dedicated "helper" methods in an attempt to decouple the templates from the rest of the system.

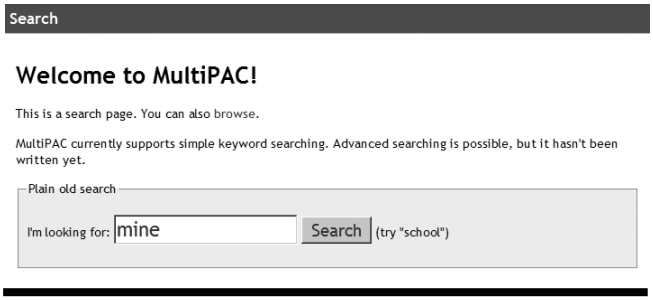

Figure 8. The MultiPAC search view page

Output formats, like resources, are modular and decoupled from the core of the system. The HTML user interface is one of several interfaces available to the MultiPAC system; others include XML and JSON, which effectively add Web services support to all encompassed resources—a feature missing from many of the resources' own built-in interfaces.[6]

Search view

Search view (figure 8) is the simplest view, serving as the "front page." It currently includes little more than a brief introduction and search field. The search field is not complicated; it is, in fact, possible to include search forms on any webpage and scope them to any subset of resources on the basis of facet queries. For example, a search form could be scoped to Las Vegas–related resources in Special Collections, which would satisfy the demand of some library departments for custom search engines tailored to their resources without contributing to the "interface fragmentation" effect discussed in the introduction. (This would require a higher level of metadata quality than we currently have, which will be discussed in depth later.)

Because search forms can be added to any page, this view is not essential to the MultiPAC system. To improve simplification, it could be easily removed and replaced with, for example, a search form on the library homepage.

Browse view

Browse view (figure 9) is an alternative to search view, intended for situations in which the user lacks a "concrete target" (figure 7). As should be evident by its appearance,

this is the least-developed view, simply displaying facet terms in an HTML unordered list. Notice the facet terms in the format field; this is malprocessed, MARC–encoded information resulting from a quick-and-dirty Extensible Stylesheet Language (XSL) transformation from MARCXML to Solr XML.

Results view

The results page (figure 10) is composed of three columns:

1. The left column displays a facet list—a feature generally found to be highly useful for results-gathering purposes.[7] The data in the list is generated by Solr and transformed to an HTML unordered list using PHP. The facets are configurable; fields can be made "facetable" in the Solr schema configuration file.
2. The center column displays results for the current search query that have been provided by Solr. Thumbnails are available for resources that have them; generic icons are provided for those that do not. Currently, the results list displays item title and description fields. Some items have very rich descriptions; others have minimal descriptions or no descriptions at all. This happens to be one of several significant metadata quality issues that will be discussed later.
3. The right column displays results from nonindexed resources, including any that it would not be feasible to index locally, such as Google, our article databases, and so on. MultiPAC displays these resources as collapsed panes that expand when their titles are clicked and initiate an AJAX request for the current search query. In a situation in which there might be twenty or more "panes" to load, performance would obviously suffer greatly if each one had to be queried each time the results page loaded. The on-demand loading process greatly speeds up the page load time.

Currently, the right column includes only a handful of resource panes—as many as could be developed in six weeks alongside the rest of the prototype. It is anticipated that further development would entail the addition of any number of panes—perhaps several dozen.

The ease of developing a resource pane can vary greatly depending on the resource. For developer-friendly resources that offer a useful JavaScript Object Notation (JSON) API, it can take less than half an hour. For article databases, which vendors generally take great pains to "lock down," the task can entail a two-day marathon involving trial-and-error HTTP-request-token

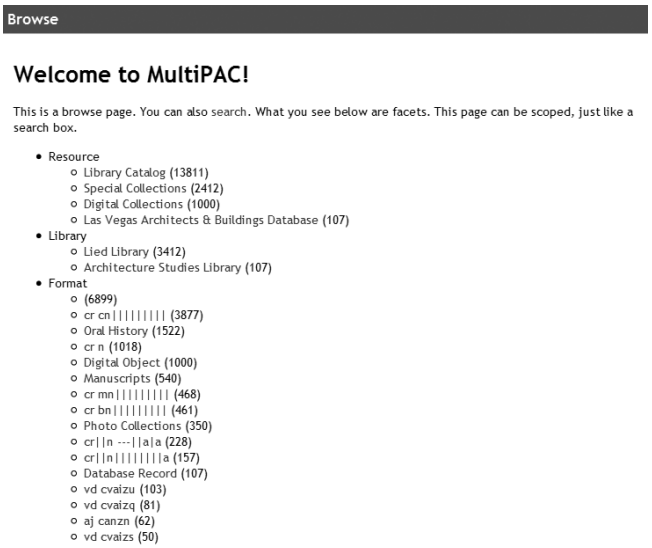

Figure 9. The MultiPAC browse view page

authentication and screen-scraping of complex invalid HTML. In some cases, vendor license agreements may prohibit this kind of use altogether. There is little we can do about this; clearly, one of MultiPAC's severest limitations is its lack of adeptness at searching these types of "closed" remote resources.

Item view

Item view (figure 11) provides greater detail about an individual item, including a display of more metadata fields, an image, and a link to the item in its primary context, if available. It is expected that this view also would include holdings status information for OPAC resources, although this has not been implemented yet.

The availability of various page features is dependent on values encoded in the item's Solr metadata record. For example, if an image URL is available, it will be displayed; if not, it won't. An effort was made to keep the view logic separate from the underlying resource to improve code and resource maintainability. The page template itself does not contain any resource-dependent conditionals.

List view

List view (figure 12), essentially a "favorites" or "cart" view, is so named because it is intended to duplicate the list feature of UNLV Libraries' Innovative Millennium

OPAC. The user can click a button in either results view or item view to add items to the list, which is stored in a cookie. Although currently not feature-rich, it would be reasonable to expect the ability to send the list as an e-mail or text message, as well as other features.

Metadata harvesting system

For metadata to be imported into Solr, it must first be harvested. In the harvesting process, a custom script checks source data and compares it with local data. It downloads new records, updates stale records, and deletes missing records. Not all resources support the ability to easily check for changed records, meaning that the full record set must be downloaded and converted during every harvest. In most cases, this is not a problem; most of our resources (the library catalog excluded) can be fully dumped in a matter of a few seconds each. In a production environment, the harvest scripts would be run automatically every day or so.

In practice, every resource is different, necessitating a different harvest script. The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) is the protocol that first jumps to mind as being ideal for metadata harvesting, but most of our resources do not support it. Ideally, we would modify as many of them as possible to be OAI–compliant, but that would still leave many that are out of our hands. Either way, a substantial number of custom harvest scripts would still be required.

Figure 10. The MultiPAC results view page

For demonstration purposes, the MultiPAC prototype was seeded with sample data from a handful of diverse resources:

1. A set of 16,000 MARC records from our library catalog, which we converted to MARCXML and then to Solr XML using XSL transformations
2. Our locally built Las Vegas Architects and Buildings Database, a MySQL database containing more than 10,000 rows across 27 tables, which we queried and dumped into XML using a PHP script
3. Our locally built Special Collections Database, a smaller MySQL database, which we dealt with the same way
4. Our CONTENTdm digital collections, which we downloaded via OAI-PMH and transformed using another custom XSL stylesheet

There are typically a variety of conversion options for each resource. Because of time constraints, we simply chose what we expected would be the quickest route for each, and did not pay much attention to the quality of the conversion.

How MultiPAC answers UNLV Libraries' discovery questions

MultiPAC has essentially proven its capability of solving interface multiplication and fragmentation issues.

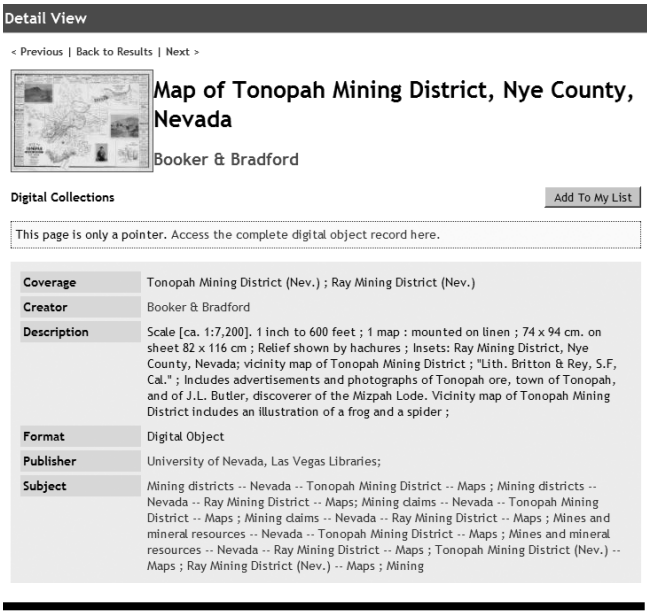

Figure 11. The MultiPAC item view page

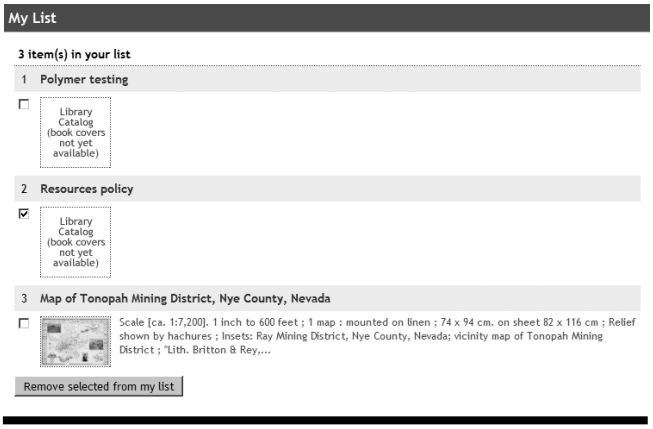

Figure 12. The MultiPAC list view page

By adding a layer of abstraction between resource and patron, it enables us to reference abstract resources instead of their specific implementations—for example, "the library catalog" instead of "the INNOPAC catalog." This creates flexibility gains with regard to resource provision and deployment.

This kind of "pervasive decoupling" can carry with it a number of advantages. First, it can allow us to provide custom-developed services that vendors cannot or do not offer. Second, it can prevent service interruptions caused by maintenance, upgrades, or replacement of individual back-end resources. Third, by making us less dependent on specific implementations of vendor products—in other words, reducing vendor "lock-in"—it can potentially give us leverage in vendor contract negotiations.

Because of the breadth of information we offer from our website gateway, we as a library are particularly sensitive about the continued availability of access to our resources at stable URLs. When resources are not persistent, patrons and staff need to be retrained, expectations need to be adjusted, and hyperlinks—scattered all over the place—need to be updated. By decoupling abstract resources from their implementations, MultiPAC becomes, in effect, its own persistent URI system, unifying many library resources under one stable URI schema. In conjunction with a URL rewriting system on the Web server, a resource-based URI schema (figure 13) would be both powerful and desirable.[8]

Lessons learned in the development of MultiPAC

The lessons learned in the development of MultiPAC fall into three main categories, listed here in order of importance.

Metadata quality considerations

Quality metadata—characterized by unified schemas; useful crosswalking; and consistent, thorough description—facilitates finding and gathering. In practice, a surrogate record is as important as the resource it describes. Below a certain quality threshold, its accompanying resource may never be found, in which case it may as well not exist. Surrogate record quality influences relevance ranking and can mean the difference between the most relevant result appearing on page 1 or page 50 (relevance, of course, being a somewhat disputed term). Solr and similar systems will search all surrogates, including those that are of poor quality, but the resulting relevancy ranking will be that much less meaningful.

Implementation-based	http://www.library.unlv.edu/arch/archdb2/index.php/projects/view/1509
Resource-based (hypothetical)	http://www.library.unlv.edu/item/483742

Figure 13. Example of an implementation-based vs. resource-based URI

Metadata quality can be evaluated on several levels, from extremely specific to extremely broad (figure 14). That which may appear to be adequate at one level may fail at a higher level. Using this figure as an example, MultiPAC requires strong adherence to level 5, whereas most of our metadata fails to reach level 4. A "level 4 failure" is illustrated in table 3, which compares sample metadata records from four different MultiPAC resources. Empty cells are not necessarily "bad"—not all metadata elements apply to all resources—but this type of inconsistency multiplies as the number of resources grows, which can have negative implications for retrieval.

Suggestions for improving metadata quality

The results from the MultiPAC project suggest that metadata rules should be applied strictly and comprehensively according to library-wide standards that, at our libraries, have yet to be enacted. Surrogate records must be treated as must-have (rather than nice-to-have) features of all resources. Resources that are not yet described in a system that supports searchable surrogate records should be transitioned to one that does; for example, HTML webpages should be transitioned to a content management system with metadata ascription and searchability features (at UNLV, this is planned).

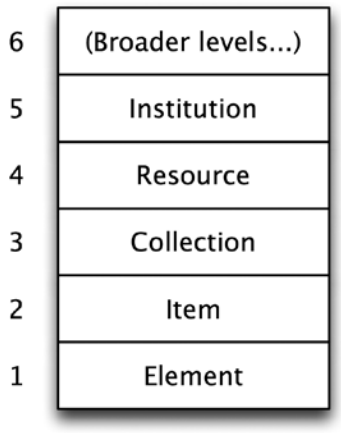

Figure 14. Example scopes of metadata application and evaluation, from broad (top) to specific

However, it is not enough for resources to have high-quality metadata if not all schemas are in sync. There exist a number of resources in our library that are well-described but whose schemas do not mesh well with other resources. Different formats are used; different descriptive elements

Table 3. Comparing sample crosswalked metadata from four different UNLV Libraries resources

	Library Catalog	Digital Collections	Special Collections Database	Las Vegas Architects & Buildings Database
Title	Goldfield: boom town of Nevada	Map of Tonopah Mining District, Nye County, Nevada	0361 : Mines and Mining Collection	Flamingo Hilton Las Vegas
Creator	Paher, Stanley W.	Booker & Bradford		
Call Number	F849.G6P34			
Contents			(Item-level description of contents)	
Format		Digital Object	Photo Collections	Database Record
Language	eng		Eng	eng
Coverage		Tonopah Mining District (Nev.) ; Ray Mining District (Nev.)		
Description		(Omitted for brevity)		
Publisher	Nevada Publications	University of Nevada Las Vegas Libraries		UNLV Architecture Studies Library
Subject	(LCSH omitted for brevity)	(LCSH omitted for brevity)		

are used; and different interpretations, however subtle, are made of element meanings.

Despite the best intentions of everyone involved with its creation and maintenance, and despite the high quality of many of our metadata records when examined in isolation, in the big picture, MultiPAC has demonstrated—perhaps for the first time—how much work will be needed to upgrade our metadata for a discovery system. Would the benefits make the effort worthwhile? Would the effort be implementable and sustainable given the limitations of the present generation of "silo" systems? What kind of adjustments would need to be made to accommodate effective workflows, and what might those workflows look like? These questions still await answers.

Of note, all other open-source and vendor systems suffer from the same issues, which is a key reason that these types of systems are not yet ascendant in libraries.[9] There is much promise in the ability of infrastructural standards like FRBR, SKOS, RDA, and the many other esoteric information acronyms to pave the way for the next generation of library discovery systems.

Organizational considerations

Electronic information has so far proved relatively elusive to manage; some of it is ephemeral in existence, most of it is constantly changing, and all of it is from diverse sources. Attempts to deal with electronic resources—representing them using catalog surrogate records, streamlining website portals, farming out the problem to vendors—have not been as successful as they have needed to be and suffer from a number of inherent limitations.

MultiPAC would constitute a major change in library resource provision. Our library, like many, is for the most part organized around a core 1970s–80s ILS–support model that is not well adapted to a modern unified discovery environment. Next-generation discovery is trending away from assembly-line-style acquisition and processing of primarily physical resources and toward agglomerating interspersed networked and physical resource clouds from on- and offsite.[10] In this model, increasing responsibilities are placed on all content providers to ensure that their metadata conforms to site-wide protocols that, at our library, have yet to be developed.

Conclusion

In deciding how to best deal with discovery issues, we found that a traditional product matrix comparison does not address the entire scope of the problem, which is that some of the discoverability inadequacies in our libraries are caused by factors that cannot be purchased. Sound metadata is essential for proper functioning of a unified discovery system, and descriptive uniformity must be ensured on multiple levels, from the element level to the institution level.

Technical facilitators of improved discoverability already exist; the responsibility falls on us to adapt to the demands of future discovery systems. The specific discovery tool itself is only a facilitator, the specific implementation of which is likely to change over time. What will not change are library-wide metadata quality issues that will serve any tool we happen to deploy. The MultiPAC project brought to light important library-wide discoverability issues that may not have been as obvious before, exposing a number of limitations in our existing metadata as well as giving us a glimpse of what it might take to improve our metadata to accommodate a next-generation discovery system, in whatever form that might take.

References

1. UNLV Libraries Usability Committee, internal library website usability testing, Las Vegas, 2008.

2. Karen Calhoun, "The Changing Nature of the Catalog and Its Integration with Other Discovery Tools." Report prepared for the Library of Congress, 2006.

3. Xiaoming Liu et al., "Federated Searching Interface Techniques for Heterogeneous OAI Repositories," *Journal of Digital Information* 4, no. 2 (2002).

4. Apache Software Foundation, Apache Solr, http://lucene.apache.org/solr/ (accessed June 11, 2009).

5. Dublin Core Metadata Initiative, "Dublin Core Metadata Element Set, Version 1.1," Jan. 14, 2008, http://dublincore.org/documents/dces/ (accessed June 25, 2009).

6. Lorcan Dempsey, "A Palindromic ILS Service Layer," Lorcan Dempsey's Weblog, Jan. 20, 2006, http://orweblog.oclc.org/archives/000927.html (accessed July 15, 2009).

7. Tod A. Olson, "Utility of a Faceted Catalog for Scholarly Research," *Library Hi Tech* 4, no. 25 (2007): 550–61.

8. Tim Berners-Lee, "Hypertext Style: Cool URIs Don't Change," 1998, http://www.w3.org/Provider/Style/URI (accessed June 23, 2009).

9. Bowen, Jennifer, "Metadata to Support Next-Generation Library Resource Discovery: Lessons from the eXtensible Catalog, Phase 1," *Information Technology and Libraries* 2, no. 27 (June 2008): 6–19.

10. Calhoun, "The Changing Nature of the Catalog."