A Unified Platform for Archival Description and Access

Christopher J. Prom
University of Illinois at Urbana-Champaign Archives
19 Library, 1408 W. Gregory Drive, Urbana, IL 61802
1 (217) 333-0798
prom@uiuc.edu

Christopher A. Rishel
University of Illinois at Urbana-Champaign Archives
19 Library, 1408 W. Gregory Drive, Urbana, IL 61802
1 (217) 390-3825
rishel@uiuc.edu

Scott W. Schwartz
Sousa Archives and Center for American Music
1103 S. 6th Street, Champaign, IL 61820
1 (217) 333-4577
schwrtzs@uiuc.edu

Kyle J. Fox
University of Illinois at Urbana-Champaign Archives
19 Library, 1408 W. Gregory Drive. Urbana, IL 61820
1 (618) 751-0101
kylefox2@uiuc.edu

ABSTRACT
The archival community has developed content and data structure standards to facilitate access to the diverse and unique sets of archival records, personal papers, and manuscript collections that are held by archival repositories and special collections libraries. However, these standards are difficult for archivists to use and are often implemented in ways that negatively affect materials-handling workflows, depriving archival users of the best possible access to the totality of materials available within an individual repository. The authors propose that archival descriptive problems can be addressed by implementing a web/database application that is tailored specifically to archival needs and can be implemented with little technical knowledge. This paper describes the system architecture of one such tool, the Archon software package, which was developed at the University of Illinois at Urbana-Champaign. Archon automates many technical tasks, such as producing a searchable website, an EAD instance or a MARC record. Although the system utilizes sophisticated algorithms and optimizations, it is easily extensible because most development takes place in an easy-to-use, object-oriented environment.

Categories and Subject Descriptors
H.2.8 [**Information Systems**]: Database Management – *database applications*

H.3.5 [**Information Systems**]: Information Storage and Retrieval – *online information services.*

H.3.7 [**Information Systems**]: Information Storage and Retrieval – *digital libraries*.

General Terms
Algorithms, Management, Human Factors, Standardization.

JCDL '07, June 17–22, 2007, Vancouver, British Columbia, Canada.

Keywords
Archon, Encoded Archival Description, Archival Information Systems, Databases, Web Interfaces

1. INTRODUCTION
Archival records, personal/family papers, and manuscript collections are among the most valuable materials held by libraries and other cultural institutions. By their very nature, such items are unique and possibly of high research and/or monetary value. Taken collectively, they comprise a major source of primary research materials, and many archival records hold continuing administrative or social value to the organizations or people that produced them.

Archival materials are also difficult for staff to describe and for researchers to access. Both problems are a product of archives' uniqueness. Unlike books or serials, a stack of papers or photographs does not contain supplied titles, authors, or other descriptive data. Nor do archival materials necessarily come to the repository bearing a coherent pattern of arrangement. Furthermore, the many parts of any archival aggregation retain their full significance only when described with reference to the context of their original creation and their relationship to other records of the same or similar provenance.[1] Once materials are arranged, described, housed, and shelved (a time consuming procedure which archivists euphemistically call "processing"), researchers may have to travel great distances to access the materials, assuming they know about or can identify content of interest.

When archivists describe archival materials, they mean to alleviate these problems, but the notion of archival description holds two competing ideas in a somewhat uneasy balance. ISAD(G), the General International Standard for Archival Description, notes that the purpose of archival description is "to identify the context and the content of archival materials in order to facilitate their accessibility [6]." On the one hand, archivists

[1] In archives, the term 'provenance' refers to "the relationship between records and the organizations or individuals that created, accumulated, and/or maintained and used them in the conduct of personal or corporate activity [2]."

are admonished to describe materials by making reference to the essential elements of the materials themselves, that is, by 'identifying their context and content [13].' Doing so ostensibly allows the professional to maintain an impartial record of evidence about how the records were created, so that researchers can authenticate evidence or establish the veracity of the information that they contain [7]. On the other hand, archivists have been criticized for being unduly bureaucratic in exercising the function of writing archival descriptions (commonly known as 'finding aids'). More to the point, they have been accused, with some justice, of subordinating user needs to an idealized notion of archival objectivity [3]. Others have noted that in the past technological obstacles prevented archivists from sharing descriptions of their information outside of their repositories [16], a fact that certainly encouraged a proliferation of local practices and a lack of standardization.

2. ARCHIVAL DESCRIPTIVE STANDARDS

To address these issues, the archival community developed data-structure and content standards. Although archivists arrived somewhat late to the game, these standards built on existing library practices and emerging technologies, ensuring their speedy adoption.

In 1985, the Society of American Archivists published the MARC-AMC (Archives and Manuscripts Control) format to allow for the bibliographic description of archival materials [22]. The AMC format was later integrated into MARC21. AMC was the fruit of SAA's National Information Systems Task Force, but MARC was and remains inadequate to capture the full context and content of archival description [9]. To address its shortcomings, groups operating under the sponsorship of the Society of American Archivists and various grant agencies developed Encoded Archival Description (EAD). The early adoption of EAD by the archival community was facilitated by a decision to use the XML syntax and SAA's development of strategic training and public information forums for archivists.

Since the development of EAD in the late 1990s, the community has also published a comprehensive set of recommendations regarding the content of descriptive records: Describing Archives: A Content Standard (DACS) [2]. DACS tells an archivist how to write a good descriptive record or finding aid, and EAD tells him or her how to structure the information. As a result, the archival community has generally, but not exclusively, agreed that the production of an XML document conforming to the EAD Document Type Definition, and containing information written in compliance with the DACS recommendations, is the *sine qua non* of good descriptive practice and archival professionalism.

3. IMPLEMENTATION PROBLEMS

MARC-AMC, EAD, and DACS were intended to facilitate better access to the diverse and unique sets of archival records, personal papers, and manuscript collections that are held by archival repositories and special collections libraries. It is commonly argued that EAD can facilitate more effective reference interactions because data is presented to a user in a more consistent, easily-sharable format [11, 25]. This is undoubtedly true, if only because it is impossible to exchange information that is not consistently structured. But it is equally possible that EAD implementation problems have precluded many institutions from using the standard or from sharing information in a fully-useful fashion.

Laura Millar, for instance, recently speculated that North American practices of description—which typically take place after processing non-current records—might not ensure that archival materials can be effectively used to hold an organization or individual accountable for its actions, because essential contextual information is missing from many records [15]. Elsewhere, author Prom has noted some prosaic problems: that archivists have trouble adopting encoding tools [17], that institutions have encoded documents in incompatible ways [18], and that many web pages generated from EAD documents are less than optimally accessible [20]. Others note that only EAD files meeting stringent 'best practice guidelines' will be fully searchable [10], and that few archival websites meet the needs of novice researchers [23]. Those in the broader community recommend that archivists incorporate user-centered design principles into finding aid design [12] and provide better information about archival creators and users [24].

Each of these authors suggest ways that archival description could be improved, but all of them beg a fundamental question: How do the ways in which archivists implement descriptive standards affect the amount of materials that archivists are able to arrange, describe, and service? There are two areas in which this question can be addressed: processing and access.

3.1 Processing

Until a collection is processed, it cannot be described, and until it is described, that description cannot be placed on-line. Given the complexity of most EAD encoding and publication options, it seems possible that current practices are at least partially responsible for the huge processing backlogs that have become an all-too-common feature of the typical archival repository. In their now seminal study of processing backlogs, Mark Greene and Dennis Meissner argue that descriptive standards are not to blame, but their analysis is not based on any hard, conclusive data [5].

Greene and Meissner do illustrate how the *processing* standards used by many archives negatively affect their output (measured as the number of cubic feet processed per year.) Archivists typically process with ideas of provenance and original order firmly in mind. Wherever possible, the archivist strives to retain or recreate original order, lest their evidential value be destroyed [4]. The two essential primers on arrangement and processing stress ways in which archivists should use levels of control, series-level arrangement, intellectual order (hierarchy), filing structure, and physical reorganization to preserve or reassemble the original order of records or manuscripts, thereby preserving their evidentiary value. These values make processing an inherently conservative and potentially time-consuming process. Greene and Meissner argue that many repositories misapply the principles and, as a result, over-process collections.

But their analysis says little about the application of descriptive standards, such as EAD, once basic processing has been completed. Has the application of descriptive standards exacerbated the problems they identify? Some preliminary evidence from a study conducted by Prom found a strong correlation between the application of descriptive standards such

as EAD and larger processing backlogs, as well as lower processing productivity. For example, repositories that use XML editors to create an EAD finding aid classify 58% of their collections as unprocessed, while those that use word processors to create descriptive records show only 37% in a similarly inaccessible state [19].

Since correlation cannot be equated with causation, one cannot say that using the standards causes the backlogs, but the coincidence should give the archival profession some reason to consider the possibility. Until creating an on-line finding aid and sharing it with appropriate content aggregators is as easy as using a word processor, the archival profession is unlikely to significantly improve access to the totality of records and papers stored in a repository. As the Donna McCrea, archivist at the University of Montana, recently wrote to author Prom:

> [W]hat was once love for EAD has turned almost to hate. The need to pay attention to perfect encoding and nit-picky details, combined with the constant updates to "best practices" that have to be retrospectively applied to all my finding aids, are sucking up all my time and keeping me from focusing on collection development and processing. Plus there is no way I can encourage small institutions to adopt EAD—It's way too difficult and time consuming for them to have to deal with!

3.2 Access

Even if McCrea and her archival colleagues were able to create a descriptive record with ease, the collections it describes would not necessarily be optimally accessible to researchers. After a collection is processed, the archivist has an obligation to share its descriptive information as widely as possible to ensure the greatest possible research access to the record's content.

The profession is currently doing a marginal job in this area. Prom's study found that the typical institution provides descriptive information on-line (in any form) for only fifty percent of its collections. Thirty percent of the institutions in his sample provide on-line descriptive information for ten percent or less of their collections.

Encoding descriptive information in MARC, EAD, or any other format can only partially meet our obligations to make collections accessible. Under current workflows, once a file has been encoded in EAD, it must be loaded into a separate search and retrieval system. Although the North American community has begun to develop tools that simplify MARC and EAD encoding, such as the Archivist's Toolkit (www.archiviststoolkit.org), such tools will not solve the fundamental problems that prevent most archives from effectively publishing archival descriptive information online. EAD files currently must be manually converted to another format (such as HTML) or indexed by a complex third-party application—a task that is well beyond the skills of even the most computer-literate archivist, much less the monetary resource base to which he or she might have access.

Europeans have made much greater progress in developing and implementing on-line tools that both encode and automatically publish archival information, but such tools use a much more rigorous system of classification and provenance than do US repositories [14].

Aside from the practical problem of publishing finding aids, an institution using EAD must firmly ground its implementation strategy in fundamental archival doctrines and carefully assess the true needs of both archival users and the reference staff who assist users. Without such a dual assessment, access to the overall holdings of a repository will be frustrated rather than facilitated.

Over the past several decades, archives have taken two disparate approaches to developing descriptive records. Those operating in the so-called 'public archives' tradition (i.e. the many repositories holding government and institutional records) have created systems to collectively describe groups of materials that are related by provenance (relationship to a common creator). Those working in the 'historical manuscripts' tradition (i.e. manuscript curators) have developed systems that (like bibliographic cataloging) index items using subject terms [21]. The developers of EAD consciously sought to meld these two traditions. EAD's structure includes areas that accommodate both subject indexing and information about creators and context under which records were created.

On balance, the move to EAD has served to undermine the principles of collective description, grouping of records by common creator, and repository-level description that were the hallmarks of the public archives tradition. This is not because EAD cannot accommodate such practices, but simply because the vast majority of archives treat each collection description as a separate entity. There is currently no effective way to group related EAD instances (such as 75 records series created by one agency), without resorting to manual methods. Nor do most implementations of EAD use authority control, since any so-called authority information (such as creator name, a biographical note, an agency history, or a controlled subject term) lives as a separate instance in each EAD file, no matter how many times the information appears in related collections. Furthermore, many institutions maintain redundant copies of information in MARC, EAD, and possibly other formats, such as word-processing files or card catalogs. Such administrative redundancies have been proven to be inefficient and unsustainable. Perhaps for these reasons, major public archives, such as the National Archives and Records Administration and most state archives, have resisted adopting the standard, as have many smaller institutions.

4. PILOT PROJECT GOALS

In autumn 2004, authors Prom and Schwartz conceived a pilot project to assess the feasibility of a software product designed to overcome many of the implementation problems mentioned in section three. In part, our project grew out of frustration with our own attempts to encode and publish finding aids in EAD format. We certainly experienced some of the problems mentioned above. More to the point, our informal studies indicated that a skilled worker took 20 hours to encode a 100-page finding aid, using standard XML markup tools, on top of the time needed to actually write the collection description and develop a general box listing of its content. This was time we could not afford to spend, given the fact our archives have over 17,000 pages of finding aids for current collections alone, and that we add 700-900 new pages every year. We also have a substantial processing backlog and new unprocessed materials arrive daily. Even if we had the resources needed to encode such massive amounts of data, we had

no means to publish it and make it searchable from the EAD/XML source. Furthermore, we needed to support contextual and authority-control information about the records, and this data needed to live outside of the EAD files.

Our pilot project envisioned a web-based tool that put fundamental archival principles into practice yet could be used by those lacking a detailed knowledge of archival descriptive standards such as EAD and DACS. The tool would provide access to the totality of our collection descriptions and archival context through a fully-searchable and browseable website. In our institution, students process the vast majority of collections. The aim was to make data-entry and encoding user-friendly without adding a costly training component. The application was intended to support rather than undermine archival processing workflows and produce its own searchable website, so that once data was input, nothing further would be required to publish the collection description in several different formats.

Our proposed system relied upon a common data input mechanism (a web browser) and storage medium (a relational database). Archival descriptive records would be exported in multiple formats: as dynamically-generated and formatted HTML, EAD files, MARC catalog records, PDF files, OAI records, RTF files, or any other format needed [1].

Data would be input once but output many different ways. Similarly, any updates or corrections would automatically propagate to the various output formats. We sought to ensure that our website and other systems would always show the most recent data with no manual intervention on our part. For example, EAD files would be automatically harvested by RLG, and MARC records in our Library's Voyager OPAC would automatically update each time a change was made to the underlying record.

In proposing the pilot system, we modestly suggested that the system must support both collection-level and folder-level description. Standard archival theory suggests that any robust program of description should support description to at least five levels [8]; in the end we were able to accommodate these five levels and more by using a recursive data model that is described in more detail below.

Finally, we suggested that the system needed to include true authority control features, so that collections could be easily grouped or regrouped by provenance, creator, subject, or genre. The authority information would be managed separately from the appropriate collection, file, or item descriptions, but could be easily linked to them.

While we briefly considered other options, we ultimately decided to develop our new descriptive tool as a web application using a relational database to store the data. We felt that this would be more easily supported than other options, would cut development costs, and could easily be packaged for other institutions, if our pilot tool proved useful enough to consider for other environments. In addition, we knew that an application built on a relational database would allow us to integrate the authority-control features we required and would ease the process of converting the data to multiple formats. Although many other archives have chosen to store EAD data in its native XML format, our past experience convinced us that XML made much more sense as a data interchange format rather than as a data storage format. Converting from XML to another view of the data, via an XSLT style sheet and XSL processor can be a cumbersome operation, and fewer programmers are skilled in this area than in web scripting and database management.

5. INITIAL DEVELOPMENT AND RELEASE

During the summer of 2005, we researched archival standards, selected essential technologies, and formed a basic development model. A significant amount of time was spent by both the archival staff and the lead developer defining the essential systems requirements. Since the developer had worked on a previous project concerning archival materials, he brought a basic understanding of archival practices to the project. He argued that the application would be most easily supported and widely adopted if it were built on the LAMP (Linux, Apache, MySQL, PHP) software stack, provided that it would also run on other web-servers and databases that use or interact with PHP.

By the end of Summer 2005, we had demonstrated that the concept we had originally outlined could work, and we decided to move the system into production. Late 2005 and early 2006, the code was completely rewritten using an object-oriented model. Additional data elements were defined, and a rudimentary digital library function was added. Data import scripts were written, and the system was debugged prior to the importation of pre-existing information from legacy databases. In August 2006, we launched a production version of Archon, viewable through http://web.library.uiuc.edu/ahx/archon/ and http://web.library.uiuc.edu/ahx/archon/?style=sousa&repositoryid=2 .

Also in August 2006, the University of Illinois released version 1.0 under an academic and research use license and presented the tool at the Society of American Archivists Annual Meeting. The license contains many elements typically included in open source licenses, such as the ability to view and modify source code for local purposes, but does not permit commercial use of the software. Use is free of charge for any not-for-profit agency.

For the next several months, we added additional data elements and an integrated help system, refined the digital library, and improved the system's overall performance and stability. Version 1.11 was released on February 1, 2007 and represents the most current stable version. It is available for download through the project website, www.archon.org.

6. END-USER FEATURES

Archon's end-user interface includes both a public website and an "Administrative Interface" available only to staff members who have logged in through a link at the bottom of any page in the public website.

6.1 The Public Website

Archon's default public website allows users to search collection descriptions and digital content simultaneously. In addition, users can browse collections by title, name of creator, subject, digital object title, and by archival record group. Each page generated by the default template includes a navigation bar (see figure one).

Figure 1. Archon navigation bar.

Search results are returned for both collections and for digital objects. Users can expand or contract the result lists, as shown in figure two.

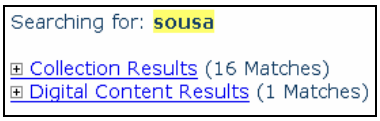

Figure 2. Sample from search results page.

Clicking one of the "Browse by" links produces a series of links, which can then be further browsed.

Browsing by record group is an optional menu item. This feature allows archives to group collections created by a common agency or creator, if they use provenance to classify materials. For example, figure three shows the display of several record series created by the University of Illinois President John M. Gregory.

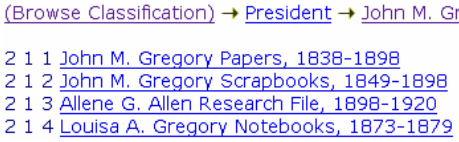

Figure 3. Record group (classification) browsing.

The production of provenance-linked pages is automated by Archon. It is very useful for the archival reference staff, who were trained to find information using the provenance method.[2] This feature is currently not supported by any other archival software package, but many archives have attempted to replicate the feature by linking collections manually in HTML. (For examples, see http://www.oberlin.edu/archive/holdings/finding/ or http://www.marquette.edu/library/collections/archives/.)

The collection and digital object displays produced by Archon are context-rich and provide hyperlinks to other collections and digital objects that are related by provenance, subject, or creator. Figure four shows a truncated collection record for a set of personal papers at the University of Illinois Archives.

[2] When using the provenance method, a researcher or member of the reference staff identifies an agency or individual that might have created records of interest, then browses holdings created by that creator. While the provenance method is not a substitute for good subject indexing, it allows the staff to fill in gaps when subject indexing is inadequate, as it must be for materials as complex as archives and manuscripts.

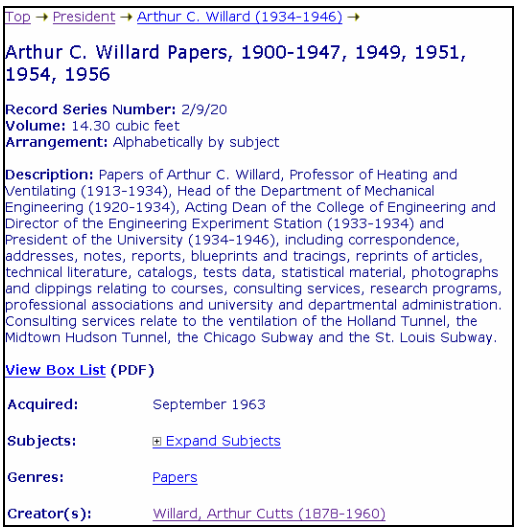

Figure 4. Truncated record with links to contextual data.

The archival user is provided a way to browse the record group system, read a biographical note about Willard, or view collection records for materials about similar subjects. Unlike the traditional archival finding aid, which spreads "collection-level" descriptive information over several pages, Archon provides links to all essential information "above the fold" in the web-browser, so that users do not need to scroll to find essential information.

Archon's public interface also includes links from finding aids to associated digital objects and vice versa, so that users can identify the specific context of any digital object in addition to metadata specific to the item. For example, the view of a digital photograph shown in figure five provides links to the collection, series and box in which the item is found, so that users can see exactly where the item is located, and, more importantly, determine whether items in which they may be interested have not yet been digitized.

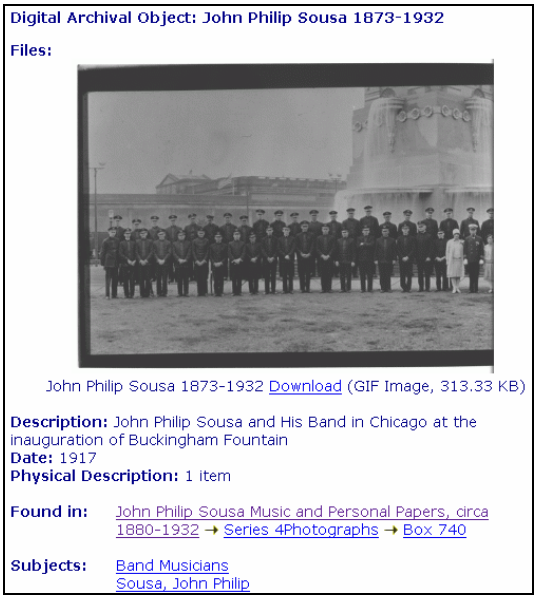

Figure 5. Public view of digital content, showing item context.

The finding aid snippet reproduced in figure six shows the view the user receives after clicking on the 'Box 740' link in figure five. The user can immediately see that an entire box of similar (undigitized) photos exists. Clicking the 'play' (triangle) link will show all items that have been digitized in that box.

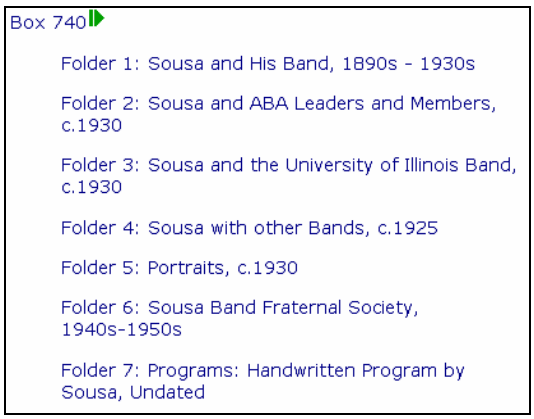

Figure 6. Finding aid/context for item shown in figure five.

6.2 The Staff (Administrative) Interface

Archon's staff interface simplifies the management of collection information, box and folder lists, and digital objects by linking all management capabilities directly to the public website. Once a staff user has logged in to the system, pencil icons appear next to all items that contain editable content. Figure seven shows the edit link for a collection record. In addition, authenticated users see the actual room numbers, ranges, and shelves where materials are stored, facilitating retrieval of items for users. They are also provided several staff views of that data, including MARC and EAD.

Figure 7. Editing link in authenticated view of public website.

After clicking a pencil icon, users are taken directly to the staff interface with information preloaded for editing. Depending on the level of access granted to the particular staff user, certain functions may be disabled.

For collection records, users can enter or edit all of the fields of description that are recommended in DACS and supported by EAD. In addition, they can define the locations where items are permanently held and link creator authority files and controlled subject terms to the collection record. Figure eight illustrates the basic collection editing interface.

Access to basic descriptive information (such as title and collection identifiers) is provided at the top, and more detailed information is provided by opening the relevant section in the bottom portion of the screen. For example, box locations can be provided in the "Location Information" and controlled-subject terms can be linked under the "Subjects" area.

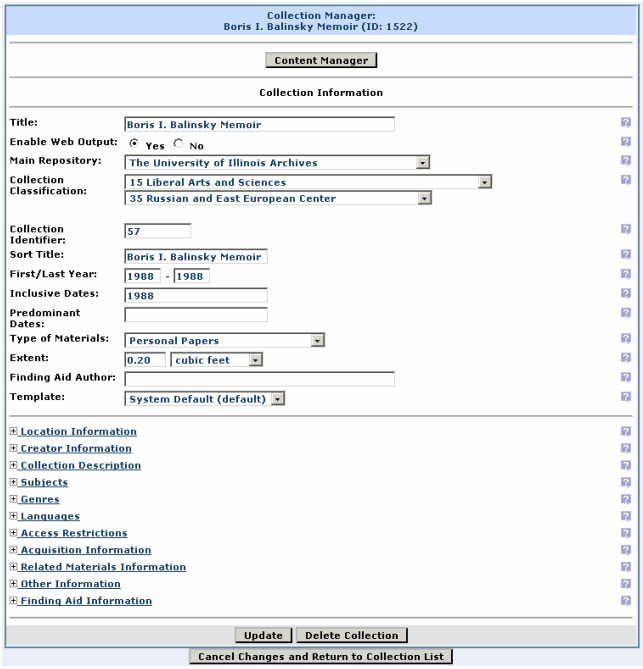

Figure 8. Collection editing interface.

From the staff member's point of view, one of the most innovative and user-friendly aspects of Archon is the ease with which complex operations, such as locating and applying controlled-subject terms, can be accomplished. If a user wishes to apply a controlled-subject term, s/he simply opens the subject area and begins typing any portion of a term which he or she suspects may exist in the controlled vocabulary. The master list of terms (UIUC has over 6,000 in its vocabulary) is quickly filtered into a selection area (using database calls and javascript functions), and the user simply double clicks the term to apply it to the collection. Figure nine shows this interface in more detail.

Figure 9. Filtering mechanism for authority information.

Archon's staff interface includes many other features designed to ease the management of archival information and digital objects. For example, Archon includes a "Content Manger" which allows staff to list the series, boxes, and folders that comprise a collection. Once lists are entered, they automatically appear on the public website as a finding aid, and they are correctly encoded into EAD format for potential data aggregation and exchange.

For cases where it is not possible to fully enter 'legacy' finding aids, such as word-processed box lists, Archon provides a way to link to external documents, such as PDF files. In these cases, a collection-level MARC record and EAD file are still automatically produced by the system, ensuring that the institution can share basic data about all of its collections, rather than some information about a select few. More information

about Archon's features may be found in the user manual on the project web site, at www.archon.org/reports.php.

7. REQUIREMENTS AND INSTALLATION

In order to function, Archon requires a web server (of any type) running PHP 5.0 or higher and a MySQL or MSSQL database server. Other database servers could be implemented by adding a small module to the database platform. The program is distributed with an automated installer that runs on Windows to simplify the process of installation; the install packet uploads files to the web server and creates the database structure. Archon can be easily configured and put into production within five minutes. This characteristic is crucial because the system will be most useful at institutions with relatively weak technical support. Archon is free of charge under an academic and research use license and the technical details of installation are handled easily. This makes Archon accessible to even the most financially strapped and/or technically-limited institutions (a substantial subset of archives.)

8. TECHNICAL INFORMATION AND API

From the developer's perspective, the Archon system is easy to implement and configure. After several overhauls, the current version (1.11, as of this writing) is built upon an object-oriented API. The API provides an extensive interface to the database and abstracts much of the development and error checking out of the scope of the casual developer.

8.1 Abstraction

The Archon code base has been logically subdivided into four groupings: the database module, the Archon API, administrative interface modules, and public interface modules/output templates. To simplify system maintenance and extensibility, it is possible to make significant changes at higher levels of the system without any work at the lower (and conceptually more complex) levels, such as the database platform or API.

The system operates on a hierarchical model of abstraction. Because the system has grown to a large code base (nearly 3 MB of PHP code and associated libraries), it would be nearly impossible to develop extensions or customizations if every detail needed to be known and handled at all levels. In Archon, the hierarchy of abstraction from lowest to highest level is the database platform, the API, and the administrative and output modules, which are equivalent from the developer's point of view. The vast majority of developers will never need to use the database platform or the API, and only a few will wish to touch the administrative interface modules. As discussed in section 8.1.4, modifications to the output module/templates can be made with very little effort. A simplified view of the system architecture is provided in figure 10.

8.1.1 Database Module

The database module is the portion of the system that communicates directly with the database server. It also provides functions which retrieve data about the database, its structure, and its contents.

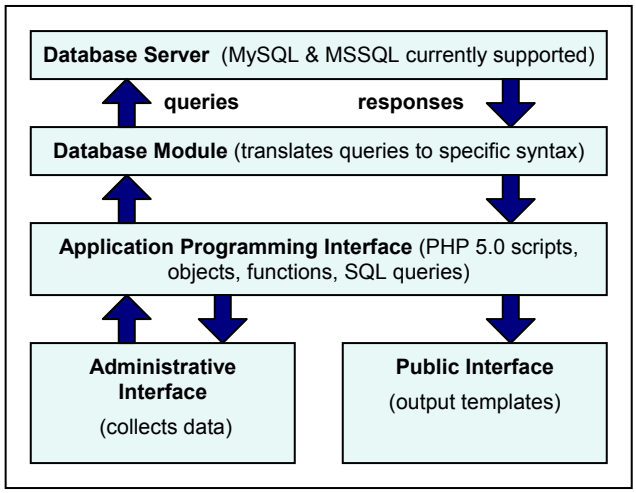

Figure 10. Simplified system architecture.

When we decided to release the software under an academic use license, we decided that the system should support whatever web technologies the end user has available. This meant that it was necessary to support the most commonly-used database servers because we could not predict which database platform archival institutions might have available for local implementation. Unfortunately, each of the common database servers uses a slightly different syntax and/or set of commands to create tables, insert, update, and remove data, or perform other common functions.

The database module in Archon abstracts these details from the developer. The database module handles any necessary manipulation of queries automatically, without any concern at the higher levels of the system, i.e. in the API, or the administrative and output modules. Archon comes bundled with database modules that support MySQL 4.0 or higher and MSSQL 7.0 or higher. Other database platforms, such as Oracle, could be supported by defining a new class in Archon's db folder.

8.1.2 API

The Archon API interfaces directly with the database platform and is a powerful set of over 300 functions that handle the management of data in Archon. The API abstracts the specifics of retrieval, searching, input/output, and data protection from the higher modules, protecting data from malicious attacks and ensuring its security. It also generalizes functions to optimize efficiency both from the perspective of speed and storage. Executive functions in the API return either true or false (if false, the API makes a detailed error message available to the calling function). This allows complex database operations to be executed with one statement and minimal error checking.

Effectively, this means that developers working with the output templates or administrative interface do not need to have any knowledge as to the structure or the implementation of the database. They simply need to read the documentation of the API to find the details about how to call the necessary functions and methods. Furthermore, the API was carefully crafted so that function and variable names are predictable. Wherever possible, they match database table and column names, and they can be easily learned. Non-technical staff will be able to assist with or even fully customize the end-user interface, provided they understand HTML and a minimal amount of PHP or another scripting language. The

API is fully documented through the project website, at www.archon.org/doc/.

8.1.3 Administrative Module

The administrative interface modules comprise the means by which Archon's system administrator and archival staff can manage the system. For example, the administrative interface is used to input collection data and manage user accounts. Administrative modules are accessible only to authenticated users, and permission levels may be tailored to any specificity needed, either for user groups or for individuals. It is important to note that the administrative interface modules do not actually execute any database manipulation but instead generate the interface to gather the data to do so. The Archon API executes instructions that manipulate data. It performs all error checking to ensure that data complies with standards required by the application.

8.1.4 Output Module and Templates

Output modules are responsible for the public view of data in the system. They also make collection descriptions, digital objects, and other information available to third-party data services, so that these services can aggregate data stored in Archon with that in other systems, such as regional archives projects or local bibliographic databases. One set of output modules produces a searchable website; another, MARC records; a third, EAD files. Others (with a slightly higher level of complexity) generate result sets, such as search hits or lists of controlled subject terms.[3]

The output can be easily modified by customizing a small number of relatively simple files found in Archon's 'styles' and 'templates' folders. Basic changes to fonts, layouts, colors and other basic properties are made in the styles folder. The styles folder contains a 'default' folder. Copying this folder and modifying its contents allows archival staff or systems administrators to define a local style, which can then be selected via a drop down menu in Archon's administrative interface, as shown in Figure 11.

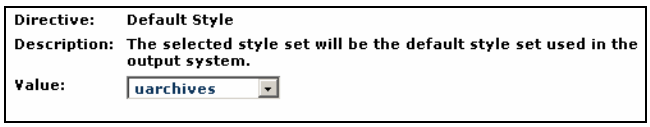

Figure 11. Selecting a default style.

More extensive changes to the output can be made by modifying files in the 'templates' folder. Institutions can define a default template set and additional templates as needed. Different templates can be applied to different collections, if desired, using Archon's Administrative Interface.

While styles control basic page properties, templates control the precise data that is included on a particular output page (such as a result set or the elements of a collection record). Repository staff can reorder content, display data based on conditions, or add additional information by referring to objects in the Collection class. The code in these files is easy to understand, making it possible for staff with relatively little technical knowledge to become involved in the development process. Figure 12 shows the code which places an archival collection's main descriptive information into the output stream.

[3] Within the next year, we plan to add output modules for OAI and METS.

```
if($objCollection->Scope)
{ ?>
    <p><b>Scope and Content</b></p>
    <p><?php echo($objCollection->Scope); ?></p>
<?php  }
```

Figure 12. Sample code from an output template

Full information and instructions for modifying Archon's output templates are available in the Administrator's Manual available through the project web site (http://www.archon.org/reports.php).

9. NOTABLE API OPTIMIZATIONS

Some of the most innovative aspects of Archon from a developer's point of view are the optimized algorithms used to return complex, recursive result sets. These algorithms are essential elements in Archon because they ensure that archival information is served to the end user in an acceptable amount of time. It is our belief that the optimizations made by author Rishel make the system suitable for the searching and display of large amounts of archival data and may have applicability with other recursive data sets. We intend to conduct further tests to verify these suppositions and will refine the algorithms as necessary in future releases of the software.

One of the most important optimizations concerns Archon's CollectionContent class. This class is used to load data into PHP objects. The objects can then be referenced in the output templates for the pages describing archival collections. For example, the generation of a finding aid is one of the key components of Archon, but it has the potential to become a very resource intensive process. Finding aids can range in size from one to several thousand items because some archival collections are hundreds of boxes in size. As one might imagine, the process of loading and preparing these finding aids into HTML or EAD format is computationally expensive.

Before analyzing the algorithms to handle the generation of finding aids, it is necessary to consider the data structures involved. To maintain a sound and logical data organization, the "collection content" table in the source database uses a recursive data structure, where one piece of content contains an identifier that points to its parent. This indefinite degree of nesting permits Archon to accommodate any organizational model a particular archival institution may follow in its finding aids. A snippet of the class definitions for Collection and CollectionContent follows:

```
class Collection
{ …
   /**
    * @var CollectionContent[]
    */
   public $Content = array();

   …}
class CollectionContent
{  …
   /**
    * @var Collection
    */
   public $Collection = NULL;
   /**
    * @var CollectionContent[]
    */
   public $Content = array();
   /**
    * @var CollectionContent
```

```
    */
    public $Parent = NULL;
…}
```

When a finding aid is requested, a collection object is loaded. It contains a Content class variable which is an array of the "root-level" CollectionContent objects. The root-level content has its Parent variables set to NULL (because they have no parent), and their Content variables contain an array of all Content "children' that they directly contain.

Because the CollectionContent data is recursive and similar to a tree in structure (like a standard XML instance), the initial algorithm (shown below) that was used to load a collection's content utilized a slightly modified depth-first search traversal. All of the root-level content was loaded, then a DFS traversal was performed for each 'node.' This process allowed Archon to mimic XML-like properties in a relational database structure, and it was hoped that the process would return result sets in an acceptable amount of time. (We have simplified the PHP code and database query somewhat to improve readability.)

```
function traversal_DisplayCollection($id)
{
  $objCollection = New Collection($id);
  $objCollection->Content = traversal_RecurseContent($id, 0);
}
function traversal_RecurseContent($collectionid, $containedbyid)
{
   $query = "
    SELECT *
    FROM
      tblCol_Content
    WHERE
      CollectionID = '$collectionid'
     AND
      ContainedByID = '$containedbyid'";
   $result = $db->query($query);
   while($row = $db->fetch_array($result))
   {
      traversal_DisplayContent($row);
      traversal_RecurseContent($collectionid, $row['ID']);
   }
}
```

For a test collection that contained over 10,000 content entries, loading a finding aid to a user's desktop took over two minutes. This was clearly an unacceptable result. After some investigation, it became clear that latency in making thousands of queries to the database server was causing the problem.

The current version of Archon utilizes a "dump-and-sort" algorithm, which makes one query to the database, selecting all the content for a given collection, and then sorts the data into the proper structure. This alleviates the overhead of many recursive calls. This method of loading takes approximately 3 seconds for the same collection (using our development machine, which acts as both web server and database server, and includes a single 2.4 GHz processor). The following code is a simplified form of the current dbLoadContent() function of the Collection class.

```
public function dbLoadContent()
{
   $query = "
    SELECT *
    FROM
      tblCol_Content
    WHERE
      tblCol_Content.CollectionID = '$this->ID'";
   $result = $db->query($query);
   while($row = $db->fetch_array($result))
   {
// If Content[$row['ID']] is already a CollectionContent, for example,
// in the case where a child was found before the parent, we don't  //
want a new instance, but we do want to run the constructor.
      if(($this->Content[$row['ID']] instanceof CollectionContent))
      {   $this->Content[$row['ID']]->CollectionContent($row); }
      else
      { $this->Content[$row['ID']] = New CollectionContent($row);}
      $this->Content[$row['ID']]->Collection = $this;
// If the current CollectionContent has a parent, add it to Content[]
//of the parent.
      if($row['ParentID'])
      {
// If the parent has not been found yet, make a new
// CollectionContent instance for it.
         if(!($this>Content[$row['ParentID']]
instanceof CollectionContent))
         { $this->Content[$row['ParentID']] =
New CollectionContent($row['ParentID']); }
         $this->Content[$row['ParentID']]->Content[$row['ID']] =
$this->Content[$row['ID']];
         $this->Content[$row['ID']]->Parent = $this-
>Content[$row['ParentID']];
      }
   }
}
```

Another optimization that stemmed from this improvement is the caching of data whenever possible. Many functions in the Archon API take the form getAllx() where x is some set of data (for example, Creators, Subjects, or even Collections). Throughout the execution of a script, many functions are called, and some may call the same getAllx() that has already been called elsewhere. Archon has an abstracted system to cache the data retrieved from these calls such that, so long as the database table from which this data was loaded has not been changed, any subsequent calls will return the cached version of the data, saving a great deal of processing time.

10. CONCLUSION

The University of Illinois developed the Archon system to meet the pressing need for improved access to information in the archives of our institutions and to overcome limitations working with EAD in its native format. From an archival user's point of view, Archon automates many technical tasks, such as producing an EAD instance or a MARC record. Staff members do not need to learn technical coding and can concentrate on accomplishing archival work. Little or no training is needed to use the system, assuming the staff member or student worker has at least a passing familiarly with basic principles of archival arrangement and description.

In addition, the public website preserves essential contextual information for users while providing an easily-searchable and browseable interface. The public website can be adapted for local purposes with little or no knowledge of the system's API or the database platform, by modifying the supplied styles and templates and (if necessary) referring to the complete sets of documentation available through the project website. Although the system itself operates using some very sophisticated algorithms and optimizations, Archon is very extensible because all but the most complex development can be done at higher levels of the system.

It provides EAD and MARC output, allowing for interoperability with other local, regional, national, or international systems.

UIUC will continue to support the development of Archon, which is used as a production system for four of our archives and special collections units. For future releases, we plan to add an accessions manager and a research cart feature, so that users can tag items for request, and multilingual support. As of this writing, five additional institutions have adopted Archon, and we are actively seeking additional partners who might provide additional input on desired features and extensions.

11. REFERENCES

[1] Archon: Initial Grant Application, available at www.archon.org/reports.php

[2] *Describing Archives: A Content Standard*. Chicago: Society of American Archivists Press, 2004.

[3] Finch, Elsie Freeman. "In the Eye of the Beholder: Archives Administration from the User's Point of View." *American Archivist* 47 (Fall 1986): 393-407.

[4] Gracy, David B. *Archives and Manuscripts: Arrangement and Description* (Chicago: Society of American Archivists Press, 1977).

[5] Greene, Mark A. and Dennis Meissner, "More Product, Less Process: Revamping Traditional Archival Processing." *American Archivist* 68:2 (2005), 213.

[6] International Council on Archives. *ISAD(G): General International Standard Archival Description,* 2nd edition. Ottowa: International Council on Archives, 1999. http://www.ica.org/biblio/cds/isad_g_2e.pdf

[7] Haworth, Kent M. Archival Description: Content and Context in Search of Structure, in *Encoded Archival Description on the Internet*, eds Daniel V. Pitti and Wendy M. Duff. New York: Haworth Information Press, 2001.

[8] Holmes, Oliver Wendell. "Archival Arrangement - Five Different Operations at Five Different Levels." *American Archivist* 27 (1964): 21-41.

[9] Hensen, Steven L. "'NISTF II' and EAD: The Evolution of Archival Description." *American Archivist* 60 (Summer 1997): 284-96.

[10] Jones, Barbara. "Hidden Collections, Scholarly Barriers: Creating Access to Unprocessed Special Collections Materials in America's Research Libraries. *RBM: A Journal of Rare Books, Manuscripts, and Cultural Heritage* 5:2 (Fall 2004), 88-105.

[11] Kiesling , Kris. "The Influence of American and European Practices on the Evolution of EAD." *Journal of Archival Organization* 3:2/3 (2005), 207-215

[12] Lack, Rosalie. "The Importance of User-Centered Design: Exploring Findings and Methods." *Journal of Archival Organization* 4 (2006), 71-88;

[13] MacNeil, Heather. "The Context is All: Describing a Fonds and its Parts in Accordance with the Rules of Archival Description." In *The Archival Fonds: From Theory to Practice*, ed. Terry Eastwood. Ottowa: Bureau of Canadian Archivists, 1992: 195-225.

[14] [Menne-Hartiz] Menne-Haritz, Angelika. "Enhancing the Effectiveness of Description with Innovative Tools–The Example of MidosaXML, Germany." *Journal of Archival Organization* 3 (2005): 83-95.

[15] Millar, Laura. "An Obligation of Trust: Speculations on Accountability and Description." *The American Archivist* 69:1 (Spring/Summer 2006): 60-78.

[16] Pitti, Daniel V. "Encoded Archival Description: The Development of an Encoding Standard for Archival Finding Aids." *American Archivist* 60 (Summer 1997): 268-283.

[17] Prom, Christopher J. The EAD Cookbook: A Survey and Usability Study. *American Archivist* 65, no. 2 (2002): 257-275

[18] Prom, Christopher J. Does EAD Play Well with Other Metadata Standards? *Journal of Archival Organization* 1 (2002): 52-72

[19] Prom, Christopher J. Optimum Access? Processing in College and University Archives. Forthcoming in College and University Archives: Selected Readings, ed. Christopher J. Prom and Ellen D. Swain. Society of American Archivists Press. Preliminary draft available at http://web.library.uiuc.edu/ahx/workpap/

[20] Prom, Christopher J. User interactions with electronic finding aids in a controlled setting. *American Archivist* 67, no. 2 (2004): 234-68

[21] Pugh, Mary Jo. *Providing Reference Services for Archives and Manuscripts*. Chicago: Society of American Archivists Press, 2005, 88-90.

[22] Sahli, Nancy. *MARC For Archives and Manuscripts: The AMC Format*. Chicago: The Society of American Archivists, 1985.

[23] Scheir, Wendy. First Entry: Report on a Qualitative Exploratory Study of Novice User Experience with Online Finding Aids. *Journal of Archival Organization*, Vol. 3(4) 2005: 49-85.

[24] Szary, Richard V. "Encoded Archival Context (EAC) and Archival Description: Rationale and Background." *Journal of Archival Organization* 3 (2005): 217-227.

[25] Szary, Richard V. "Encoded Finding Aids as a Transforming Technology in Archival Reference Service." *Journal of Internet Cataloging* Vol. 4: No 3/4 (2001): 187-97