A Scalable Knowledge Base: the eLibrarian 2.0 Project

Eloisa Borah, Librarian, Head of Public Services
Rosenfeld Library
Anderson Computing and Information Services
UCLA Anderson School of Management
eloisa.borah@anderson.ucla.edu

Tim Carlson, Programmer, Project Manager
Software Development Group
Anderson Computing and Information Services
UCLA Anderson School of Management
tim.carlson@anderson.ucla.edu

Summary

When a popular service was facing an overwhelming level of demand and becoming a victim of its own success, a technology solution was sought to come to its rescue. The original eLibrarian service used email to have the reference librarians provide students with research strategies within a 24-hour turnaround. The current eLibrarian 2.0 system is entirely Web-based, and provides new functionalities that increase the capacity of the service, while under the constraint of a steady-state number of librarians. These enhancements include an auto-populated knowledge base and a recommender system.

eLibrarian – What is it?

The eLibrarian service provides assistance to remote users. A team of reference librarians respond to requests for help in locating information. According to the *eLibrarian Guidelines*[1], the librarian recommends the best databases and other resources for the need at hand, pre-tests the database search, outlines this search in a brief tutorial, and may include a sample search result to provide confirmation that the research strategy is on target. Responses are delivered within 24 hours of the initial request.

In keeping with the teaching mission of an academic library, the eLibrarian service does not do the actual research required by the course work, as a research-on-demand service would do in a non-academic setting. What eLibrarian provides is the research strategy necessary to locate the information requested, and thereby the instruction to learn to satisfy this information need in the future.

Evolution of the Concept

The ubiquitous Internet and the “anywhere, anytime” expectation of information seekers has made the reference desk just one way to seek help from a librarian. Reference help via email started almost as soon as email accounts started proliferating, and other online options, such as chat reference, have been added to the list of access options at most libraries. This is the case at the Rosenfeld Library.

However, the original email reference service at the Rosenfeld Library had been under-utilized, so it was re-designed from providing ready-reference to providing research strategies. Providing research strategies, under the eLibrarian, took more librarian time, however, it proved to be exactly what our users wanted. With eLibrarian, the user

perceives the librarian as their information partner. The demand for the service increased dramatically.

Improvements made to cope with the increasing demand included a Web front-end that included a Web form to submit new requests. However, it became evident rather quickly that a broader technology solution was necessary to manage the increasing workload, and a plan needed to be put in place.

Problem Solved!

Before going into how we solved the problem, it is important to describe the collaborative environment that is the Anderson Computing and Information Services, or ACIS. The Rosenfeld Library is part of ACIS as a result of a successful convergence of library and computing services[2], which was solidly in place by the time UCLA Anderson School moved into its new physical complex in 1996.

In the ACIS environment, librarians, computer support technicians, and programmers sit side-by-side in the workplace, and are involved in collaborative projects to improve our clients' abilities to access, gather and manage the information they need.

Our clients are full-time, fully-employed, and Executive MBA students at the UCLA Anderson School of Management, which is a top business school, in both U.S. and international rankings.

The plan to find a technology solution to manage eLibrarian's unbridled popularity started with a series of planning retreats attended by all the librarians at the Rosenfeld Library, the ACIS director, and a programmer from ACIS Software Development Group at ACIS. At the initial retreat many options were considered, which even included the dire options of discontinuance and rationing access. However, very early in the meeting the focus narrowed to investigating technological solutions that would help librarians work better with the staff time available.

A management process, adopted by the ACIS Software Development Group for its software engineering methodology, called Rational Unified Process (RUP), was engaged to identify components of the service delivery flow and the technology solutions that were most cost effective to implement. RUP's excellent requirements gathering techniques, its iterative development lifecycle, and it provision of fine quality document templates were among many elements that made RUP one of this project's success factors.

The resulting move to eLibrarian 2.0, facilitated by a librarian and a programmer as lead developers, was made in stages, with testing and assessment after each roll-out. It started with streamlining the email-based environment to deal with non-eLibrarian communication, and eventually moved to an entirely Web-based environment that removed that bottleneck altogether.

Summary of Functions

Under eLibrarian 2.0, the productivity of the reference librarians is enhanced in many ways. Primary among these are a more structured inquiry form, an eLibrarian Workspace on the web, and a new Web-based eLibrarian Knowledge Base, which is automatically populated with completed inquiries and replies, provides search capabilities, and makes recommendations to requestors.

The redesign of the new inquiry form makes use of data fields in existing databases on the UCLA Anderson network, identified and enabled by our team's programmer, in order to authenticate the requestor name, email address, and current course load. This eliminates the time it takes librarians to check for requestor's student status, as well as any possible typos in name, email address, and especially in the course title related to the research, as this helps librarians understand the context of the inquiry. The new inquiry form also requires the requestor to select from the four basic types of business information needs[3] (company, industry, management function, or business environment). These required input data, mentioned above, become the key searchable fields in the knowledge base.

The Web-based common workspace automatically assigns each newly received eLibrarian inquiry to the reference librarian on that work shift, using a pre-determined work schedule. The status of all inquiries in the queue are also displayed, along with their different stages of completion. This spares the reference librarians the confusion over which inquiries are actually being worked on, and by whom.

The Workspace also has a convenient link to searching the knowledge base. Reference librarians can easily search for similar requests for a possible time-saving cut-and-paste of relevant research steps from prior responses. The reference librarians can search past inquiries and replies by requestor name, by assigned reference librarian, by course number or instructor, or by keyword within the entire text. Requestors will be offered the search function in a future version of eLibrarian.

There is also an interesting new link in the workspace which allows the reference librarian to check previous requests from the same requestor to find relevant threads, or catch duplicate submissions before precious time is wasted. The workspace allows reference librarians to attach documents, to provide links to relevant websites, and other functionalities that the reference librarians had been using in the previous email-based environment.

The new eLibrarian knowledge base is Web-based, and is automatically populated with both inquiries and replies after each transaction is completed. For confidentiality, personal information is stripped from the text by the librarian during the reply process, before each record is added to the knowledge base.

An exciting "smart" element of eLibrarian is its recommendation system – you may be familiar with the one used at Amazon. After the requestor selects the course he is doing

research for, eLibrarian 2.0 will automatically push on to the requestor's screen inquiries from that same course from the eLibrarian knowledge base. The requestor may view any or all the records offered and still return to complete an original inquiry form. However, if after viewing one or more of the offered records, the requestor finds his information need satisfied and clicks on a button confirming this and exits the system, then the transaction is counted as "answered by the Knowledge Base".

The eLibrarian 2.0 Web-Based Process

This is the perfect time to show you a demo of the eLibrarian 2.0 Web-based processes.

We have previously described the user authentication function and the important role that the request forms play in our system, as well as the recommender system offered by our knowledge base.

We have also described above the time-saving and organizational benefits reaped by the reference librarians from the common workspace and the ability to search past records from the knowledge base.

When the reference librarian completes the response, an email is sent to the requestor that contains a link to the response. The response, in printable format, ends with a brief user survey. To maximize survey participation and response pick-up, the system features a daily offline process that looks at all answered responses without completed surveys. If five days have elapsed without a survey, this program sends the requestor an email reminding them that the response is still available. The program is executed via a UNIX cron process.

Real-time Web reports are planned for a future phase, but currently our solution was to hook up an MS ACCESS front-end to our SYBASE tables. Our assistant then exports the data to MS Excel for reporting. Again, since we try to capture all the "who did what and when" data, there are many possibilities for reports. Among the metrics we file in reports are: use by course, by faculty and by student population group.

Software

The eLibrarian 2.0 system is written entirely in-house. Perl was chosen for its excellent text handling features, its platform portability, and its wide usage and known reliability in Web applications. There is also a minimal amount of Javascript in the system. Although it currently using SYBASE tables, eLibrarian -- as part of a department-wide effort -- is converting to MS SQL Server on August 1, 2005. No other commercial software is used in the eLibrarian 2.0 system.

Technical Design Considerations

We faced many challenging technical design decisions in developing the eLibrarian 2.0 system. Since eLibrarian replaced an existing email based system, one major concern

was that the new version would not entail a lower service or functional level than email. Email is, of course, among the most successful computer applications of all time. The process of replacing an email based system allows a designer to realize how functionally rich and user-friendly most email clients are.

Initially there was concern by some librarians that a Web-based system might be less stable than the existing email platform. We reasoned -- based on general observation -- that the Web platform is as stable as email systems. This has been born out -- albeit subjectively -- by experience. Most of our users, in fact, feel that this system is more robust than the previous release.

To replicate or 'meet and beat' email functionality, the first step was to design functions that corresponded to email behavior. Thus, the user request function, the common workspace, and the user response functions, shown earlier, correspond to a requestor sending and a librarian responding to an email. The ability to forward an email to another librarian is done via the claiming function in the workspace. Email folders no longer exist of course, but the data is organized in the database and can be accessed a number of ways. Email folder searching has been replaced by the search function. Although email folders are very flexible and allow users to organize on the fly, clearly they are an inferior organization of data than a well designed set of relational data base tables.

E-mail's ability to save drafts has its equivalent in the "confirm changes" button in the workspace. One technically challenging area was replicating e-mail's ability to attach documents. Our solution involves storing the "attached" document into a special directory and then sending a link to that document in our response.

Iterative development is recognized as an IT best practice. eLibrarian incorporated this practice in several ways. First and foremost was our decision to start with a release that featured some key, but easy to implement, enhancements. That release did not involve any new databases, but it did separate the system for internal and external users and also began the practice of pulling data from Anderson databases and minimizing text entry. This iteration allowed us to prove that the quality of data and ability to differentiate internal and external users were enhanced and useful. There have been 17 minor releases since 2.0 went live, incorporating various enhancements. Various design decisions, maximizing code maintainability, have facilitated our ability to iteratively improve the system. Management support, allowing us to develop in an incremental on-going fashion, has also been a major success factor. ACIS uses the Rational Unified Process as its software engineering methodology. Iterative development is a key practice in the RUP methodology.

Another challenge, of special interest in library applications, was finding a way to insure privacy and confidentiality for our requestors, while at the same time being able to retain data in its original form. As seen in the workspace and other functions, one solution was to allow librarians to edit the original requests, including removing any personal information. Although a requestor can see their original request, other requestors only

see edited requests. The ability to edit requests and categories, while keeping the original also allows for better quality of data, and an ability to track how much data improvement is required or performed. Another privacy protection is in the link to URLs for viewing the response, which contain the requestors email address and an internal generated number for security.

Portability and Scalability

In designing eLibrarian, we attempted to maximize its future portability both in terms of computer platform and in terms of multi-library implementation.

From a platform standpoint, Perl can run on a variety of platforms and has been widely used to support large scale applications. UNIX, SYBASE and SQL SERVER are all recognized for their ability to handle large-scale and complex applications.

eLibrarian uses an in-house written enhanced front-end to the Perl DBI module. Perl DBI is Perl's standard data base API which can run against virtually any prominent RDBMS. The SQL calls within the program are all SQL-92 complaint which also increases the system's platform portability.

Several application-level features enhance multi-facility support; in the database, each table has a field called "facility_code". Each library's URL passes the facility code as a parameter to the Perl scripts which then incorporate it in all SQL calls. The facility code may also be used in getting appropriate data from configuration files. Thus with almost no programming, the system can support multiple facilities just by a parameter in the URL.

Another database feature which supports portability is an ability to dynamically define table columns. For example, each facility may want different survey questions and different categories. To accomplish this, there is one table that defines questions or categories for each institution. A corresponding table holds the answer or value for that field. The row with the actual data points back to the corresponding definition of the survey question or category. The programs thus pull category and survey definitions out of the tables for display. Once categories or survey questions are defined and instantiated this way, no library-specific programming is required to handle them across facilities.

Other database features to support institutional portability include: a table to define each institution's personnel, and a table which holds information about a particular library including its home page URL, particular text, turn-around time threshold and so forth.

All page headers and footers are created via function calls. The function calls can be pulled out of a configuration table, so again no special programming, other than the target library creating its function calls, is required.

Separating presentation, business logic and data layers is recognized as a best practice in Information Technology. As already shown, the data layer is well defined in the tables. To support the visual layer separation, the system uses a Perl module called HTML::Template. This module separates logical and visual layers by allowing page layout definitions which include variables. The variables are assigned values in the program and the page template is called for display. The result is neater, more maintainable code than the typical "here document" used in Perl programs for painting html.

What's in the Future?

Among what is envisioned for future versions of eLibrarian is: Web reporting, some aesthetic enhancements, and plans to retro-populate the knowledge base.

Some roll-outs are also planned to test eLibrarian 2.0 at a business school at another UC campus, as well as other business schools across the country. Also a test to see how it functions across subject disciplines in a campus-wide roll-out. A test roll-out to professional schools other than business, such as law, medicine and engineering could also prove the viability of this system. Finally, it has come to our attention that the eLibrarian 2.0 system is a superb knowledge base generator that can be ported to many non-library applications.

Footnotes

[1] UCLA Rosenfeld Library, *Rosenfeld Library eLibrarian Service,* http://www.anderson.ucla.edu/resources/library/eLibrarian_guidelines_Oct2004.pdf.

[2] Bellanti, Bob and Jason Frand, *"*Connectivity and Convergence", *UCLA Librarian*, 48 (1995-96) 24-31. *http://www.anderson.ucla.edu/faculty/jason.frand/researcher/articles/librarian96/page1.htm*

[3] Yeargain, Eloisa Gomez, "Conceptual Analysis of Business Information Needs"*,* (paper presented to the Business Reference Services Discussion Group (now BRASS) at the annual conference of the American Library Association, San Francisco, June 30, 1987).