A Multipurpose System: The development of the University at Albany Library Systems Wiki

Eugenia Kim

Eugenia Kim is a graduate student in the Master of Science program in Information Science at the University at Albany. She holds an Associate of Arts in Liberal Arts from Simon’s Rock College at Bard, a Bachelor of Science in Electronic Media, Arts, and Communications from Rensselaer Polytechnic Institute, and a post-baccalaureate minor in Computer Science from Tufts University. Prior to attending University at Albany, Eugenia worked in information technology support and web design. Her research and professional interests include digital preservation, East Asian archives, academic and special libraries, and web technologies.

ek959665@albany.edu

Abstract

A growing trend in recent years for American academic libraries is the use of Web 2.0 applications to create and enhance both internal and external websites. In 2009, the University at Albany Library Systems group decided to replace its file server and intranet portal with a combined system that was more collaborative and easier to navigate and access for staff members. To implement this system, Library Systems compared open source and commercial wiki software to create a prototype ‘Library Systems Wiki’ for their department. Based on current literature regarding the use of wikis by academic libraries, this prototype takes a different approach in its choice and use of software. The wiki also has in-house technical specialists as developers that allows for greater system complexity. This paper outlines the systems design and implementation as well as future plans for the wiki. Particular concerns include user customization, the interdependent
relationship between organizational schemes and graphical interface, and the migration of born-digital documents. This prototype will be thoroughly tested to investigate the possibility of its use by all groups in the University at Albany Library.

Introduction

Over the past decade a growing trend in American academic libraries is the use of Web 2.0 software to create and enhance both internal and external websites. Library professionals are particularly interested in using blogs and wikis to promote interaction and collaboration between staff and student users, store information, and format websites. In 2009, the University at Albany Library Systems department decided to combine its file server and intranet portal into a wiki-based system called the ‘Library Systems Wiki’. Library Systems applied for and was awarded an Academic and Special Libraries Section of the New York Library Association (ASLS)/Ridley Book Bindery Student Internship Grant to hire a graduate assistant to implement the wiki system in conjunction with a team of in-house technical specialists. The University at Albany Library then provided matching funds for the project. Work on the project commenced in November 2009 and will continue into June 2010.

This paper presents the multiple phases of the project: research, testing, implementation, organization, and deployment. It is intended to provide a detailed description of the development cycle for reference by other parties. The initial design request required that the wiki system needed to fulfill multiple purposes: archives, library, project collaboration tool, and intranet system. These challenges raised the question of how to balance hardware and software needs with arrangement and preservation concerns. Through this documentation, the author will demonstrate that Library Systems created a unique solution that draws on both technological expertise and archival practices.

What is a Wiki

Wiki technology began with the advent of Ward Cunningham's WikiWikiWeb in 1995 (Clark and Mason, 2008) but it was not until the first decade of the twenty-first century that libraries began making use of the software. The word “wiki” comes from the Hawaiian word for “quick” (Leuf and Cunningham, 2001) that implies the relative speed with which a user can interact and modify a wiki. Traditionally a wiki engine consists of a public or privately hosted server to store its information and an interface for viewing and editing material. A wiki is usually an individual or discrete set of “pages” that provide information and can be configured to look and function like a standard website, embedded as features into other web technology, or have other features such as video and scheduling options embedded into them. A commonly known example of the use of wiki technology is the free encyclopedia Wikipedia (http://www.wikipedia.org). Another growing use of wiki technology can be seen in collaborative wiki-based services such as
PBWorks (http://www.pbworks.com). For the purposes of this paper, the term ‘wiki’ will be used to refer to the information container and its contents while ‘wiki engine’ will refer to the software program and infrastructure behind a wiki.

Current Implementations of Wikis in Academic Libraries

As part of the research phase for the Library Systems Wiki project, a number of articles regarding the implementation of wikis in academic libraries were reviewed by the author. Several universities that published details of their wiki creation process and results were: Antioch New England University (Clark and Mason, 2008), Iowa State University (Haupt, 2008), University of Houston (Ravas, 2008), and Georgia State University (Glogowski and Steiner, 2008). The team at the University at Albany also examined the University of Minnesota Library Staff site as an example of an intranet wiki. Common reasons for using wikis that emerged immediately were the needs for organizing and storing knowledge, updating old websites and portals, and the use of open source software. Each library group prioritized user needs and emphasized the importance of being independent of an information technology (IT) department. Specific software used by said universities were PmWiki, Seedwiki, OpenWiki, and MediaWiki, the latter being the system used for Wikipedia. None of these systems were considered for use by the Library department, which will be further explained in the Technology section.

Primary concerns for all the referenced libraries were security, usability, functionality, and training. Interestingly, cost was not mentioned as a deciding factor. Most wikis began with a single purpose and then evolved to accommodate other interests such as interactive project management and collaboration. Systems were generally maintained and set up by a group of library personnel who were not technology specialists. However there was very little mention of how existing digital-born documents were migrated into wiki systems on a large scale. Only the University of Houston briefly described how they transcribed their music library manual to the new wiki. Thoughts on preserving the information over time even after wiki systems evolve or disappear were absent as well. Nonetheless, each library seemed fairly satisfied with the use of wikis in serving their purposes.

Our Research and Challenges

The need for a Library Systems Wiki started with the goal of simplifying search and access to documents on the current file server. Over five years’ worth of born-digital documents had accumulated on the server without an official naming scheme or clear provenance. Folders had not been culled of irrelevant or outdated information and many documents were not updated frequently enough. Additional information was spread out across two other sites, one being the library-wide intranet portal and the other being the student intern guide. Since the Library Systems group is a fast-paced multi-tasking environment, it was essential to be able to find information quickly and easily.

The research process began with the gathering of user needs from each staff member
within the department. Based on feedback from informal interviews, a user need survey was conducted within the six member department. The results can be seen in the table below:

**Table 1 – Survey Results**

<table>
<thead>
<tr>
<th>Preferred User Needs</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Effective Search Function</td>
<td>100%</td>
</tr>
<tr>
<td>Quick Documentation Access</td>
<td>100%</td>
</tr>
<tr>
<td>Logical File Categories</td>
<td>100%</td>
</tr>
<tr>
<td>Page History</td>
<td>33%</td>
</tr>
<tr>
<td>Authentication and Permissions</td>
<td>67%</td>
</tr>
<tr>
<td>Links to External Websites</td>
<td>17%</td>
</tr>
<tr>
<td>Collaboration and Interactivity Capabilities</td>
<td>33%</td>
</tr>
<tr>
<td>Ability to Make Announcements</td>
<td>17%</td>
</tr>
<tr>
<td>Easy Document Upload and Conversion</td>
<td>50%</td>
</tr>
<tr>
<td>Advanced Administration Options</td>
<td>17%</td>
</tr>
<tr>
<td>Open Source</td>
<td>42%</td>
</tr>
<tr>
<td>Customizable Interface</td>
<td>17%</td>
</tr>
</tbody>
</table>

The most widely stated needs were: effective search function, quick access to documentation, logical file categories, and authentication/permissions. Secondary concerns were how to easily upload and convert documents, page history, and ways to collaborate, comment, and add to others’ work. The remaining features were requests specific to their proposer.

One advantage at the outset of the project was that Library Systems provides for all the IT needs of the library. Therefore team members had both advanced technology skills and an understanding of library services. A wiki systems project team was assembled that consisted of the director of Library Systems, database administrator, server administrator, web developer, and grant-funded graduate assistant. The graduate assistant was involved with project management, documentation, research, testing, organization, interface design, and user training. The student had previous experience in constructing a wiki knowledgebase for another IT department. All team members were regularly informed of each other’s activities and frequently participated beyond their specialization.

The technological expertise of the in-house staff, however, also created a new set of requirements. The most basic prerequisites were local hosting, installation platform, program code type, and database storage. Other preferences included an effective search engine and ability to authenticate using Lightweight Directory Access Protocol (LDAP) or Active Directory accounts. Between the user needs and technology requirements, many systems would be eliminated in the course of finding suitable wiki systems, including many open source options.

There were also other issues that the graduate assistant was principally responsible for investigating. The most basic problem was how to migrate so much born-digital information without any data loss while reducing the amount of time for translating documents into a wiki format. This issue tied in to the primary request of searching for and editing wiki pages quickly and easily. Another request was to compress and rearrange documents that did not always have a clear origin or relationship to other files.
Ultimately the graduate assistant discovered that the three concerns were closely tied together and would be resolved by looking at organization and interface design as an interdependent relationship. However this discovery did not clarify how to satisfy both librarians and technology specialists nor the question of whether wikis will be a viable long term storage and preservation method.

**Research**

In order to compare the myriad of wiki systems, a website called WikiMatrix was used. WikiMatrix contains information on over 100 wiki systems and allows a user to make a comparison matrix. It also provides a wizard to help narrow down the choices for a user. The team set the initial search to look for the following factors: the wiki must be hostable on a local server, use the Windows server platform, be programmed using PHP: Hypertext Preprocessor (PHP), and store files using a MySQL database. This search eliminated at least half of the options right away. From there, other specific features helped filter the results, such as authentication, export, and linking options.

After a few days, a set of eight wiki engines and content management software (CMS) software were discussed in a group meeting: bitweaver, BusinessWiki, Daisy, Drupal, MindTouch, PhpWiki, TikiWiki-CMS Groupware, and WikkaWiki. Based upon the developers’ preferences, the systems selected for initial testing were bitweaver, Drupal, PhpWiki, and TikiWiki. PhpWiki was then removed from the list as it formed the code base for TikiWiki and was therefore nearly the same system.

The first test systems installation was performed on December 22, 2009. All members of the project team were present. The first step was to install MySQL Administrator and Browser (MySQL AB) which is freeware made by Sun Microsystems. These components form the database setup that all the wiki systems would use. MySQL AB performs both administrative tasks and query run. It was installed to a drive on the local wiki server using MySQL 5.1 Server Wizard. During the configuration, the multifunction database option was selected and installed to a different storage area from MySQL AB. Although both MyISAM and InnoDB options were available on the server, the team preferred to use MyISAM. Other specifications included setting the internet port to 3350 and selecting UTF-8 as the text protocol. Connection to the database was made using the ‘localhost’ option.

Four user accounts were created: root, admin, wikiread, and wikiwrite. This is default protocol for the Library Systems department when setting up a new database and is sometimes required by certain software for installation. The first two accounts have administrative privileges, whereas wikiread is for general users who only browse and wikiwrite is for users who can make modifications to the database. These database accounts translate into accounts for use with the wiki systems. Standard privileges for wikiwrite included select, insert, update, delete, create, drop, grant, index, and alter; wikiread has the select option only.

Although the same accounts were used for each
The first system to be installed was TikiWiki. However, when it tried to locate a database connection, it would generally return an error message. Through a combination of automatic and manual install (which involved installing the two SQL files tiki.sql and tiki-secdb_3.0_mysql.sql), TikiWiki was installed once. However, once CGI was switched to ISAPI for installing bitweaver, TikiWiki would not run nor install. For bitweaver, the database settings were similar to the other engine but it asked for a URL and prefix. Until CGI was switched to ISAPI, it also had trouble with PHP coding issues during install. The MySQL Classic option was also selected instead of the MySQLi option. Out of all the wiki systems, it provided the largest range of possible add on packages. Drupal was the easiest system for the team to install. The configuration options were very similar to TikiWiki: DB type - MySQLi, host – localhost, DBname – wiki_dru, etc. Unfortunately there was neither automatic update option nor an immediately apparent method for enabling the clean URLs option, which limited configuration. On the other hand, a wikiread account was not necessary.

The testing of these systems then commenced on December 23, 2009. Because TikiWiki was not usable, only Bitweaver and Drupal were tested. Bitweaver proved to have a well-laid out GUI but many of the basic features like file upload required so much server and PHP configuration that it was decided that the system was not time-efficient. After several days of testing, Bitweaver proved to require too much troubleshooting in order to make basic functions such as file upload work. Bitweaver was then abandoned in order to examine other systems.

The team then moved to Drupal testing. The initial ‘wiki’ content type was defined using recommended modules and configuration settings according to the Using Drupal book from O'Reilly. Most of the configuration settings worked smoothly but some of the revision history options failed to work.

Other modules were also installed, such as “What you see is what you get” (WYSIWYG)editing, chat options, and table of contents. Some modules were operational “out of the box” while others required tweaking. In general it seems that Drupal will not only require adding modules to enhance functionality, but that it is not immediately obvious which modules to install. Overall, Drupal was very simple and stable in its operations. Its only flaw was the amount of customization that was required in terms of features and interface.
Because the options were so limited between bitweaver and Drupal, the team decided to look at three commercial options: Confluence, DotNetNuke, and Mindtouch. Confluence is specifically aimed at knowledge management and has been used by the graduate assistant for other projects in the past. It is also the wiki system used by the University at Albany for its university-wide wiki service. DotNetNuke and Mindtouch have both a commercial and an open source option that are convenient for Library Systems.

Testing for Confluence was done via a 30-day trial offered by Atlassian. Using their Wiki Markup, it is fairly easy to create and link pages together. The primary difficulty lies in finding the proper markup code to use for a particular function. Confluence also makes use of plugins, leading to similar issues that were encountered with Drupal. The layout code looks like a combination of HTML and PHP that could be useful or difficult depending on staff expertise. Like many content management systems, Confluence makes it possible to create a custom stylesheet and provides a WYSIWYG option for changing color schemes. The most useful tool was the Notation Guide for wiki markup reference: http://confluence.atlassian.com/renderer/notationhelp.action?section=all#.

This reference page made linking and setting up pages extremely simple, the built-in search engine was fairly quick and stable, and a page revision history was also available.

The DotNetNuke testing was also performed using an online trial option. The initial impression was that the interface was very simple, but hard to edit in Layout mode. Although the Stylesheet Editor facilitates directing editing of CSS, it does not seem to provide the same levels of modification found in other systems. The other customization options were standard for a content management system in that one adds and deletes modules for changing functionality. The biggest disadvantage to using DotNetNuke was that making a Wiki page required creating a new page, then adding the module to a content pane. Updates did not always hold, adding links was not that intuitive, and it was a bit cumbersome
to add new pages. Thus while the overall interface was visually pleasant, the software was weak in usability for its primary purpose.

Mindtouch was unofficially dubbed “simple of the simple” by the wiki team for its very clean interface. However the Core freeware was not an easily installable option in terms of the required platform needs. Useful features of the program were built-in tutorials and templates for site configurations: intranet, extranet, knowledgebase, etc. However, it was not possible to edit title details from within the page editor that used a customized form of wiki-markup language (WML). A positive aspect of Mindtouch was that it was possible to embed most media including video. There were many logos and skins, CSS and HTML options for users as well as Google Analytics and graphical linking. Convenient features included the ability to add attachments without going into ‘Edit’ and automatically generated Table of Contents on the side bar.

As a final step before selecting a wiki engine, pricing was investigated for Confluence and MindTouch. It was immediately apparent that initial and maintenance costs could be quite steep for a small organization.

<table>
<thead>
<tr>
<th>Software Pricing</th>
<th>Confluence Annual Pricing</th>
<th>Mindtouch Annual Pricing</th>
<th>Mindtouch Perpetual Pricing</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$10 / 10 users</td>
<td>$995 / 10 users</td>
<td>$1495 / 10 users</td>
</tr>
<tr>
<td></td>
<td>$800 / 25 users</td>
<td>$2195 / 25 users</td>
<td>$3295 / 25 users</td>
</tr>
<tr>
<td></td>
<td>$2200 / 100 users</td>
<td>$3795 / 50 users</td>
<td>$5695 / 50 users</td>
</tr>
<tr>
<td></td>
<td>$4000 / 500 users</td>
<td>$6395 / 100 users</td>
<td>$9595 / 100 users</td>
</tr>
</tbody>
</table>

Final Selection

Based on the final report and prototype systems, the Library Systems team discussed and voted for their preferred software. The Drupal CMS was selected for its open-source status, staff familiarity with the software, the potential to integrate the wiki with the library website, and high level of customization. Preparing the server for the final implementation was relatively fast and simple because it is possible to migrate a Drupal installation to another location without losing any configuration settings or information. The only components to be reset for the production server were the database settings.

During the testing, it was observed that Drupal regards wikis as a Content Type and is not a single-purpose wiki engine. This characteristic gave the web developer the option of completely defining how a wiki page should function. It also allows for other web services to be fully customized around the wiki Content Type without being tied to a predetermined set of features or doing extensive manipulation of the source code.
Interface and Organization

After the system migration, the first step was to configure the user interface. This process was fundamental to how information would be migrated into the system. It also determined the arrangement and organization of the documents as well maintaining organizational control of new documents. An important rule was to keep the interface as straightforward as possible to facilitate quick creation and editing of content. The document hierarchy was set to extend to no more than five levels of depth or ‘breadcrumbs’ and allowed for categorization listings by both subject and creator-subject relationships.

This primary feature was implemented in Drupal by creating new Content Types that were customized to act as category templates. Content Types are not directly dependent upon the Theme or CSS settings at all which enables the web developer to reserve the graphical interface step for the end. Customized menus were also created to display appropriate links as well as hierarchy trees of pages based on a user type. The latter menu was deceptively tricky to configure as it relies upon manipulation of menu settings and Blocks. Blocks are an essential setting in that they determine the location of a component within a layout as well as which pages the component is displayed on. To further refine menu options, the Menu by Content Type module was installed to limit users to specific category options based on template type. A useful feature of the Menu settings later proved to be the option of multiple menu links pointing to the same nodes. A source of frustration was trying to understand how the Taxonomy module worked. Because it is responsible for the display of child nodes, similar to Confluence, and categorizing content, it became essential for the templates. Ultimately it was discovered that the display of a Taxonomy vocabulary list was dependent on the selection of Content Type.

User accounts and permissions were also key in controlling where information was categorized and simplifying the editing process. Through these settings a user can only have access to certain areas relevant to their purpose thus preventing a person from mistakenly placing their page in the wrong section. Other customizations were performed by modifying the Drupal cascading stylesheets (CSS) and PHP source code. The ability to make changes at the PHP level quickly proved to be a very powerful and dangerous ability. While modifications allowed for a deeper level of customization, it also caused many errors depending on the changes made.

During February 2010 it was discovered that the interface was very dependent on how the information was to be organized. All the vocabulary lists, content templates, menus, and page structures could not be completed without a clear idea of the information structure. A basic structure was established through staff input and a DirTree spreadsheet of the files on the department server. However, given the depth of the file organization, these items were not sufficient to fully determine the best way to organize items. To satisfy staff requests to better locate folders and files they had authored, the graduate assistant recommended the use of a
provenance-based or ‘creator’ menu. Although it is easy to organize recent or overall posts by user in wikis, they are usually arranged by date of modification or creation and as such are not immediately useful. The use of a creator category page that lists relevant items by folder and menu name allows for more efficient browsing of usage by staff members. The menu also cross-references pages so that a document with multiple authors will appear in each author’s feed.

**Deployment**

Partway through the project there was a change in focus from creating a pure prototype for the Library Systems department to accounting for use by other library groups via content templates, user type testing, user permissions. A factor in this change of focus was an expression of interest by other library personnel to use the wiki for upcoming projects. As a result, the initial release date was moved from late Spring 2010 to March 15, 2010. Since the majority of Drupal configuration happened over 2 weeks in February for less than 20 hours, this deadline was considered reasonable. An advantage of this early deployment date is that it will provide usability testing and feedback for creating a final prototype release in June 2010.

**Complications**

In the process there were two difficulties that proved to be challenging. The first was getting input and consensus from staff members. Part of this came from the fact that it was hard to explain its utility and importance at levels beyond the technological. This was a crucial realization that helped structure feedback requests more efficiently. The other complication was the technical difficulties with open source software. Amongst all the software options tested, the graduate student observed that the commercial products proved the easiest to install and that open source solutions often had problems during installation or use. Additionally, there was a lack of detailed or consistent technical support for the open source systems. This was very surprising to the wiki team since our review of other library wikis had stated little or no problems with implementing open source wiki software. Although Drupal had a reputation of being easy to install and having a wide developer-user community, it was quickly apparent that there was a steep learning curve for custom applications of Drupal beyond general websites.

Some concerns for the graduate assistant in particular were the fact that there is still no clear method for handling the data migration or how to protect the information over time. Wikis are currently popular with the general public and have a fifteen-year history but it is impossible to determine the total lifespan of the technology. The only option for now is to focus on transferring data as simply as possible and look into exporting options into other formats. Promoting use and interactivity by staff members is also unclear. This task will eventually be the responsibility of the department as the graduate assistant will depart at the end of the project. One possible option is to demonstrate the range and simplicity of the wiki features as well as store key information
that can only be accessed via the wiki. It is more likely that the department will need to research their users’ priorities and demonstrate why the wiki system is important for them. This will include understanding the different user perceptions and logic patterns across users. For example, one person may view “accessibility” as being able to get to information quickly. Another person, however, may be more concerned with the nature of the content or its presentation.

**Future Status of the Wiki**

After June 1, 2010, the maintenance of the wiki system will no longer be the sole domain of a single administrator. To account for this change there is a plan for three types of user accounts: administrators, power users, and users. Administrators will have full control over all settings and will be members of the Library Systems department. Power users will have most settings open to them and will be key personnel within other divisions and departments. This will allow individual groups to oversee their content and options without necessarily contacting an administrator first. ‘User’ will be the default basic account and should satisfy most user needs.

Between March and June 2010 priorities will be placed on data migration and user interface improvement. It is expected that categories and Drupal modules will also undergo modifications as staff members provide more feedback. During the process of migration the question of how to resolve both old materials and newly created materials without duplicate migration will need to be settled. The liquid nature of the wiki design should accommodate for major changes without completing overhauling the system. However there is no guarantee for a smooth transition to the upcoming Drupal 7 release.

In terms of marketing, new strategies will need to be employed other than providing an educational workshop and testing option. As a casual observer stated “How do you make them [users] think that this product is essential to their lives?” Other considerations include the fact that a librarian’s sense of “access” may not be the same as a technologist’s sense of “access” or that a user may be less interested in the infrastructure and more interested in the ease of service. Overall more research will need to be done on target audiences to learn how to speak to their interests.

**Conclusion**

Through our software selection and team composition, we have developed a hybrid wiki system that allowed us to define our concept of a wiki from the ground up. While there are other existing software solutions that provide a wiki within a CMS solution, the team has yet to identify an academic library that makes use of Drupal as a wiki knowledgebase or an engine that would allow a developer to define their concept of a wiki. Although it may seem that this project has spent a long time in development, most work was done over the course of 4-5 months with an average of 5-10 hours a week of work by the graduate assistant and even less by each individual team member. This translates to the equivalent of 1-2 work weeks for a full-time staff member.
It is hoped that through future feedback that the Library Systems wiki can become a functional and important tool for library staff. Like most new wikis this system will also need about a year or so before it can be truly refined and its performance commented upon. The ideal outcome is that the Library Systems Wikis will allow for communication between all library departments and divisions in the University at Albany Library and meet their knowledge management needs.
References


