code{4}lib JOURNAL

Issue 12, 2010-12-21

ISSN 1940-5758

Improving the Drupal User Experience

Drupal is a powerful, but complex, Web Content Management System, being adopted by many libraries. Installing Drupal typically involves adding additional modules for flexibility and increased functionality. Although installing additional modules does increase functionality, it inevitably complicates usability. At the University of Houston Libraries, the Web Services department researched what modules work well together to accomplish a simpler interface while simultaneously providing the flexibility and advanced tools needed to create a successful user experience within Drupal. This article explains why particular modules were chosen or developed, how the design enhanced the user experience, how the CMS architecture was created, and how other library systems were integrated into Drupal.

By Rachel Vacek, Sean Watkins, Christina M. Morris, and Derek Keller

I. Background

The University of Houston Libraries [1] Web Services Department first started using Drupal [2] in Spring of 2008 when we migrated our intranet from text-heavy, poorly organized and outdated static html pages to a more robust and flexible installation of Drupal 6.

As we were new to the platform and excited about its potential, we quickly began extending the code and adding requested functionality. The new system was well received, looked great, improved internal communication and met the needs of those wanting advanced features. However, as we gained experience in Drupal development practices, some problems were solved with different solutions of varying quality. The system grew in complexity and as a result, the overall user experience began to suffer. Content editors began consistently needing technical assistance from Web Services as the multitude of blocks, views, menus and custom content types were too confusing. Many aspects of the system were so over-engineered that it became more and more difficult to know the purpose of many content types, much less how to output them on a page.

In Fall of 2008, the Libraries began the initial stages of redesigning the main website. Our existing CMS was a custom system built on Adobe ColdFusion and Microsoft SQL Server. Although well liked by most library staff for being robust, flexible and modular, we were uncertain about Adobe's ongoing commitment to support ColdFusion. The technology has aged and it has become difficult to find skilled ColdFusion developers. Within Web Services, one staff member was dedicated to full-time development and improvement of the entire system, thus not allowing for opportunities to work on other projects within the Libraries. It was quickly apparent that we wanted a more open and vibrant CMS platform with an active development community.

After the favorable intranet migration experience and with the growing number of libraries in the Drupal community [3], we chose to move forward and use Drupal as the CMS for our main website. We decided to learn from our initial mistakes and focus primarily on a system that was both powerful and easy to use. At the time, Drupal 7 was still an unstable beta version, there was no definitive release date and many modules had not yet been ported. We decided to use Drupal 6 for primary rollout of the website while minimizing (or at least simplifying) functionality so that a future upgrade to Drupal 7 would be as easy as possible. Our top priorities were to engineer a system that would be a much more pleasant experience for library staff to work with content, and could fluidly handle rapid changes to information architecture.

II. Drupal User Experience (UX)

A critical concern arose during our initial experience with stock installations. Users of all ranges of technical skill were confused and frustrated by unfamiliar Drupal terminology, methodology and poor administrative interface elements.

It quickly became apparent that we needed to adopt a new ideal while planning this second usage of Drupal for the Libraries. We needed to help users edit content quickly and easily, but not force them to think about Drupal and its unfamiliar conventions. Even the wider Drupal community has acknowledged that Drupal 6 doesn't provide a very pleasant user experience, and there are many group initiatives that are attempting to improve it within Drupal 7, such as the DUX7 Project [4]. The Drupal Project Lead, Dries Buytaert [5] summarizes the need for improving the user experience within Drupal:

> Drupal's steep learning curve filters out far too many smart, motivated people who could benefit from Drupal. We see it all the time in the drupal.org forums, in my "State of Drupal" surveys, on Twitter, when talking to customers, and on the web. Even though we've made significant progress with making Drupal easier to use, a lot of work is left to be done. With other content management systems such as Joomla! and WordPress making strides to catch up to Drupal in terms of development flexibility, if we want Drupal to remain competitive, we have a challenge we have to face: we need to create a user experience that makes it easier for people new to Drupal to discover all of its richness and power. [6]

During our research, we discovered one such project that seemed to exemplify a quality user experience: Open Atrium [7], a Drupal installation profile for intranets. For users, any confusing Drupal conventions are hidden behind a thoughtful, cleanly designed and organized interface while allowing developers access to the powerful back-end. This was exactly the idea we wanted to emulate.

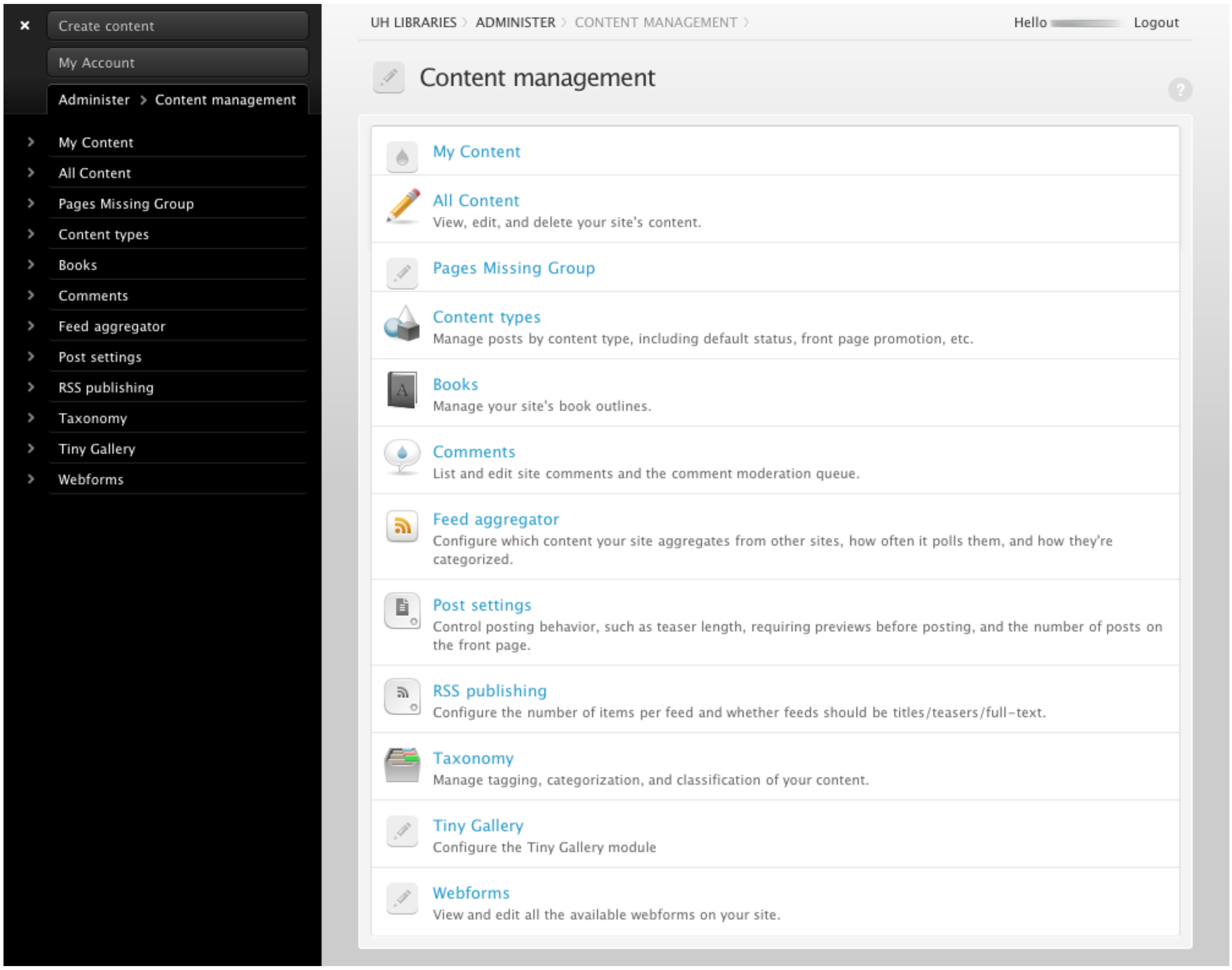
III. Building Drupal

In this section, we share what the most important modules are that we used, created, modified, and why we chose or built them. Not all modules mentioned are referenced because they are either part of core or because they are custom built. Modules are the building blocks to making our library staff and content creators' user experience within Drupal a successful experience.

A. Standard Drupal Modules

Admin Module

One of the bigger causes of confusion and frustration for end-users is Drupal's administrative back-end pages. The Admin Module [8], used with the clean administrative theme Rubik, separates all administrative navigation into a clean and concise side menu that is much more familiar to the casual user familiar with other CMSs. The Rubik theme was set for all editing screens, giving a consistent interface through which to work with content.



My Content

All Content
View, edit, and delete your site's content.

Pages Missing Group

Content types
Manage posts by content type, including default status, front page promotion, etc.

Books
Manage your site's book outlines.

Comments
List and edit site comments and the comment moderation queue.

Feed aggregator
Configure which content your site aggregates from other sites, how often it polls them, and how they're categorized.

Post settings
Control posting behavior, such as teaser length, requiring previews before posting, and the number of posts on the front page.

RSS publishing
Configure the number of items per feed and whether feeds should be titles/teasers/full-text.

Taxonomy
Manage tagging, categorization, and classification of your content.

Tiny Gallery
Configure the Tiny Gallery module

Webforms
View and edit all the available webforms on your site.

Figure 1. Drupal administration interface using the Admin Module

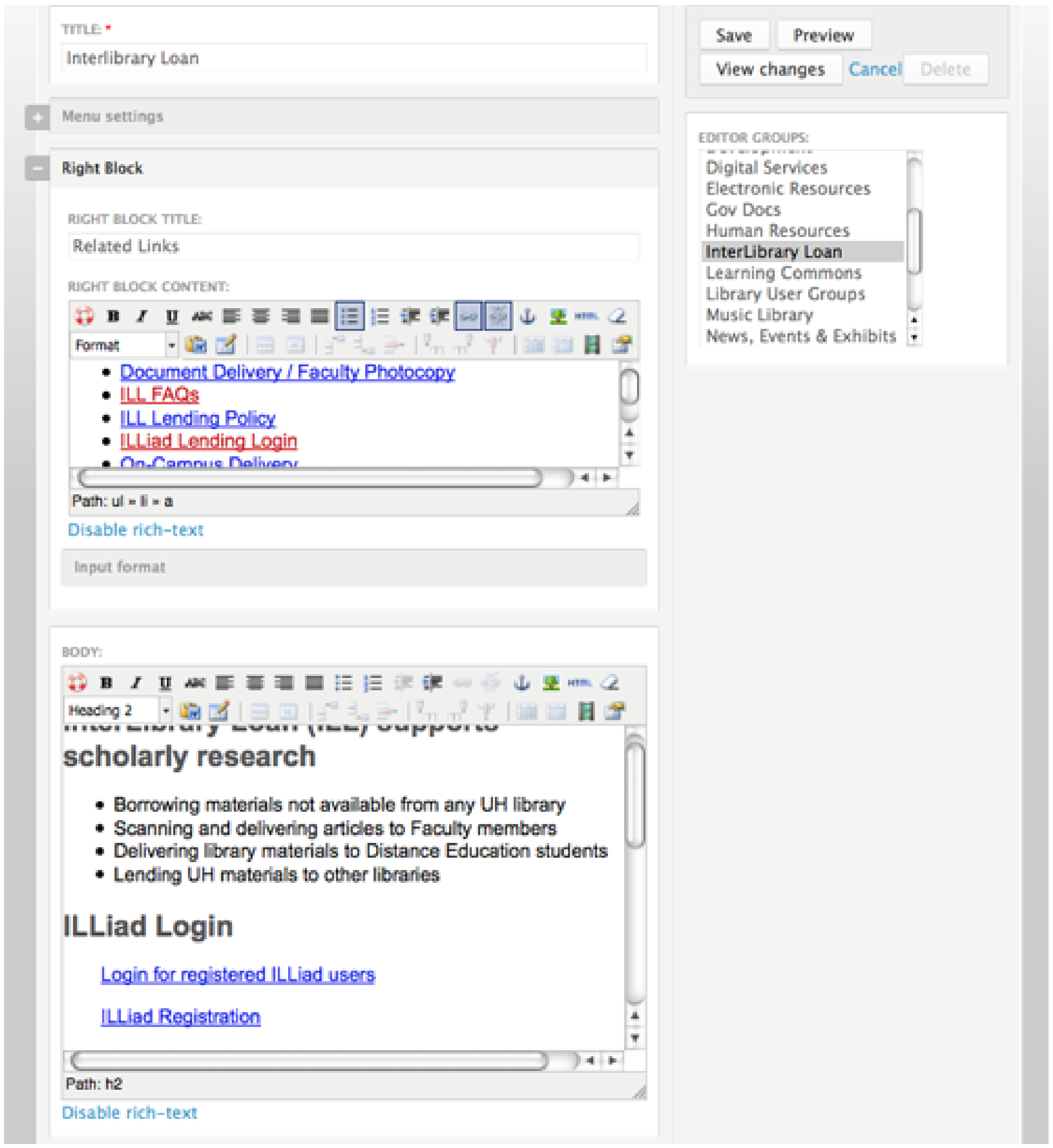
Book Module

The committee responsible for assembling content for the new website did so in an extremely structured, hierarchical way in order to ease maintenance and tracking over time in the future. Drupal's built-in Book module specializes in hierarchically sequenced content, making it a natural fit to implement this defined tree-like outline. Used alongside the Menu Breadcrumb [9] and Pathauto modules, the Book module provided a flexible way to manipulate content hierarchy while retaining auto-defined breadcrumb navigation trails with easily readable URLs. The Books module also was able to output the entire tree hierarchy as a linkable sitemap. It became trivial to create or delete subpages as well as move a page and its children to a different area of the structure.

Certain pages on the site were set to use the PHP input filter to run simple blocks of PHP code. The site map content, for example, was generated by a few lines of modified code that used the Books module API functions to browse the site structure and build the map.

Content Construction Kit (CCK) Module

Another valuable Drupal module is the Content Construction Kit (CCK) [10], which adds a flexible interface for creating custom fields attached to content nodes. For the Libraries' website, a right-hand side content area was needed on pages to communicate information such as related links or contact information. Since we knew that library staff already struggled with creating and maintaining Blocks on the intranet, we needed a simpler solution. We added a set of CCK fields called "Right Block Title" and "Right Block Content" to the Book Page content type. Drupal template files (tpl) were then created to customize the output of these field nodes and CSS was used to style appropriately. In short, users could create a content block on the right-hand side of the page without having to create, configure, and place the block on a page.



TITLE: *

Interlibrary Loan

Menu settings

Right Block

RIGHT BLOCK TITLE:

Related Links

RIGHT BLOCK CONTENT:

- Document Delivery / Faculty Photocopy
- ILL FAQs
- ILL Lending Policy
- ILLiad Lending Login
- On-Campus Delivery

Path: ul » li » a

Disable rich-text

Input format

BODY:

Interlibrary Loan (ILL) supports scholarly research

- Borrowing materials not available from any UH library
- Scanning and delivering articles to Faculty members
- Delivering library materials to Distance Education students
- Lending UH materials to other libraries

ILLiad Login

Login for registered ILLiad users

ILLiad Registration

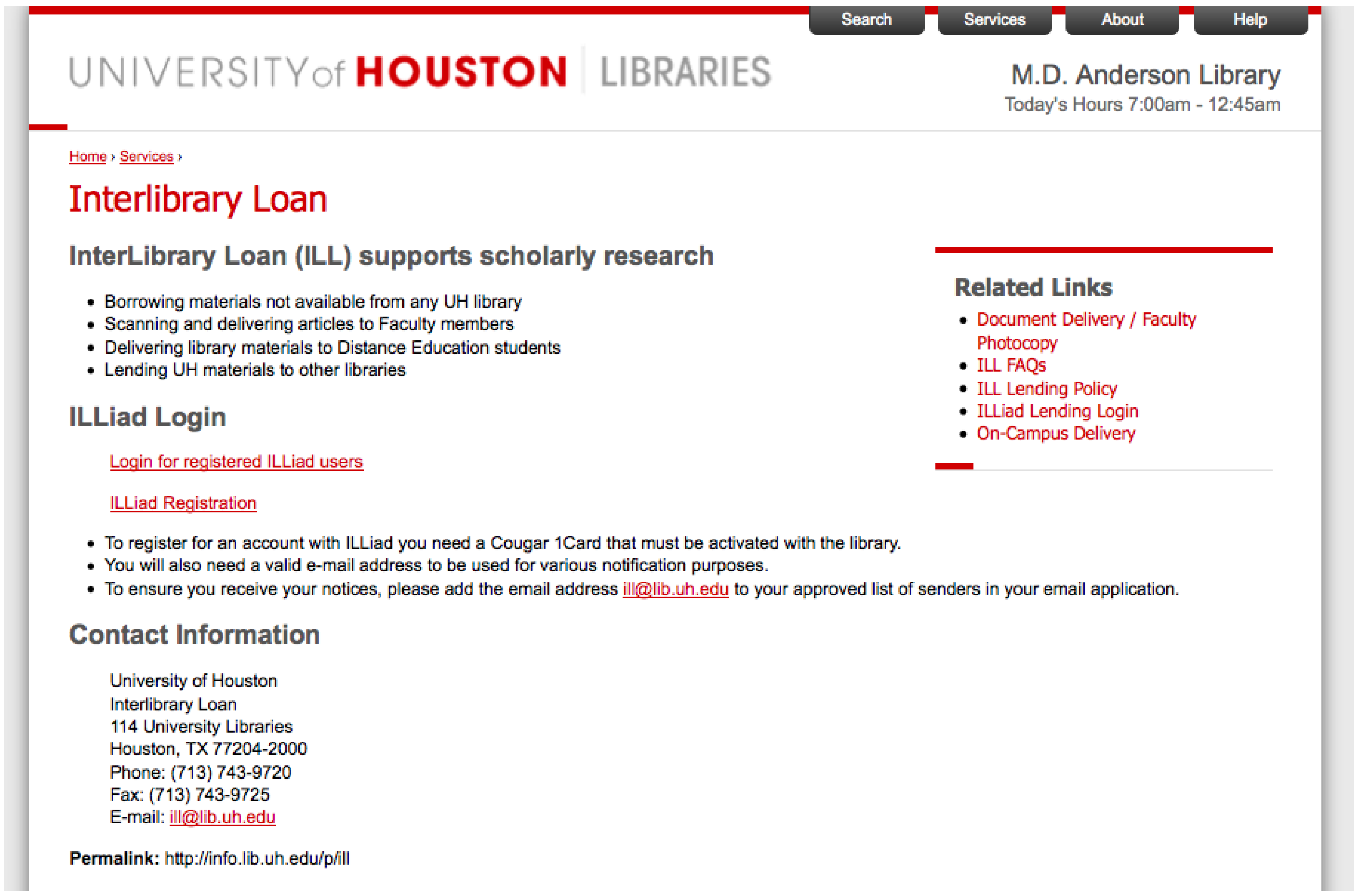
Path: h2

Disable rich-text

EDITOR GROUPS:

Digital Services
Electronic Resources
Gov Docs
Human Resources
InterLibrary Loan
Learning Commons
Library User Groups
Music Library
News, Events & Exhibits

Figure 2. A second Wysiwyg editor is added to the page so users can easily add content to the “Right Block” section without having to interact with the Blocks module.



M.D. Anderson Library

Today's Hours 7:00am - 12:45am

Home › Services ›

Interlibrary Loan

InterLibrary Loan (ILL) supports scholarly research

- Borrowing materials not available from any UH library
- Scanning and delivering articles to Faculty members
- Delivering library materials to Distance Education students
- Lending UH materials to other libraries

ILLiad Login

Login for registered ILLiad users

ILLiad Registration

- To register for an account with ILLiad you need a Cougar 1Card that must be activated with the library.
- You will also need a valid e-mail address to be used for various notification purposes.
- To ensure you receive your notices, please add the email address ill@lib.uh.edu to your approved list of senders in your email application.

Contact Information

University of Houston
Interlibrary Loan
114 University Libraries
Houston, TX 77204-2000
Phone: (713) 743-9720
Fax: (713) 743-9725
E-mail: ill@lib.uh.edu

Permalink: http://info.lib.uh.edu/p/ill

Related Links

- Document Delivery / Faculty Photocopy
- ILL FAQs
- ILL Lending Policy
- ILLiad Lending Login
- On-Campus Delivery

Figure 3. The “Right Block” feature is styled appropriately and displays on right side of the page.

Content Locking Module

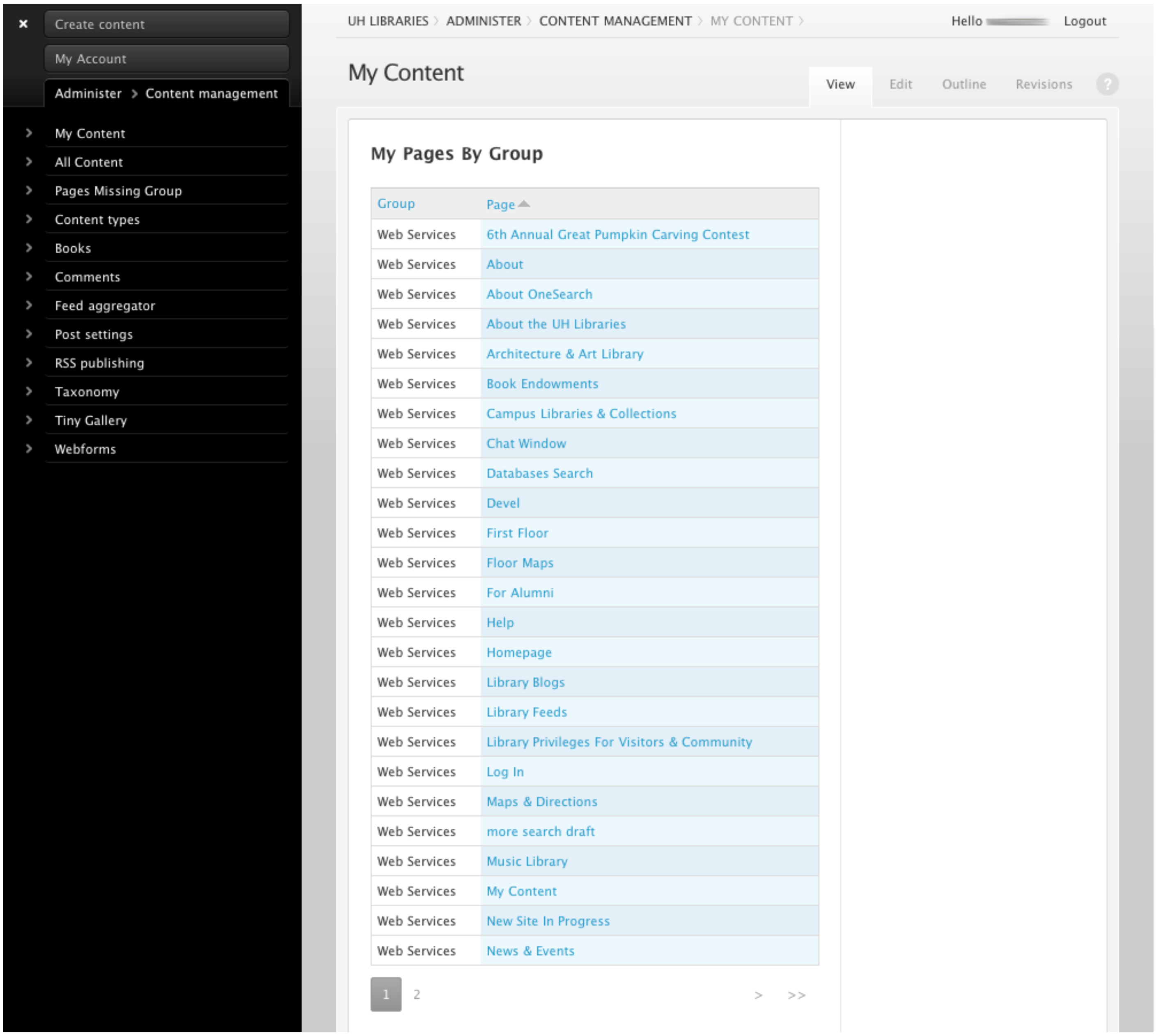
Since we planned for the CMS functionality to be as lightweight as possible to handle very structured content, there was no rigorous control or workflow system put in place. Much of the content was to remain low maintenance and fairly static over time. However, during rush editing periods, it soon became apparent that we needed something in place to help protect users from accidentally interrupting or overwriting the work of others.

The Content Locking module [11] was a small and simple solution to block concurrent page edits. Any page edited by a user is then locked from others until changes are saved. This has worked well with our expected low volume of users and edits, but a higher volume would likely call for revisiting this system and inclusion of a more complete workflow system.

Taxonomy Module

Only a small percentage of the library staff needs to create and manage content, and a Web oversight committee must approve any major revisions. In lieu of a more traditional and complicated CMS workflow where users are tightly controlled, we opted for flexibility similar to a wiki-type system where content editors have access to edit any page. With protection against deletions or errors in place via restricted Drupal node permissions and revision control, we only needed an easy way to associate users with the pages for which they are responsible.

A commonly used module provided in Drupal core is Taxonomy, which suited this purpose well. A list of content editing groups was defined as taxonomy terms, with some term names based on department and others on particular function. All Book pages are assigned to any number of editing groups, and in turn each user is also manually assigned to a set of editing groups that are most appropriate. For example, someone who is responsible for a computer access policy would be added to the Policy Editing Group. A “My Content” administrative page was then created using PHP code to list all groups to which the user was a member and all pages belonging to those groups.



Group	Page
Web Services	6th Annual Great Pumpkin Carving Contest
Web Services	About
Web Services	About OneSearch
Web Services	About the UH Libraries
Web Services	Architecture & Art Library
Web Services	Book Endowments
Web Services	Campus Libraries & Collections
Web Services	Chat Window
Web Services	Databases Search
Web Services	Devel
Web Services	First Floor
Web Services	Floor Maps
Web Services	For Alumni
Web Services	Help
Web Services	Homepage
Web Services	Library Blogs
Web Services	Library Feeds
Web Services	Library Privileges For Visitors & Community
Web Services	Log In
Web Services	Maps & Directions
Web Services	more search draft
Web Services	Music Library
Web Services	My Content
Web Services	New Site In Progress
Web Services	News & Events

Figure 4. Content creators see the groups they belong to and the pages owned by that group.

As there is no CMS-imposed access control being enforced for editors, using Taxonomy in this fashion is merely a way to help them more easily navigate to the content they care about. This works in a low volume, structured editing environment but does allow for occasional editing accidents that must be reverted.

CAS Module

The UH Libraries has pushed to more fully utilize a trusted single sign-on authentication system so that users only have to enter their username and password once to be logged in to all specified library applications. We successfully implemented CAS, or Central Authentication Service, a web-based system distributed by the Jasig Project [12]. Drupal integrates seamlessly with CAS through the phpCAS library and Drupal CAS module [13].

Diff Module

With the relative lack of editing restriction in the CMS, it is important that we have dependable safety nets in place when things go wrong. The Diff module [14] allows us to see nicely formatted differences in edits across revisions so that problematic edits may be more quickly spotted and reverted.

reCAPTCHA Module

On any website accepting user-submitted input, spam is always a problem. Fortunately, Drupal has many CAPTCHA [15] solutions available. The reCAPTCHA module [16] syncs with the reCAPTCHA web service to protect against web form spam submission, and offers an audio version for those with vision disabilities. This allows us to meet accessibility requirements from the State of Texas.

Views Module

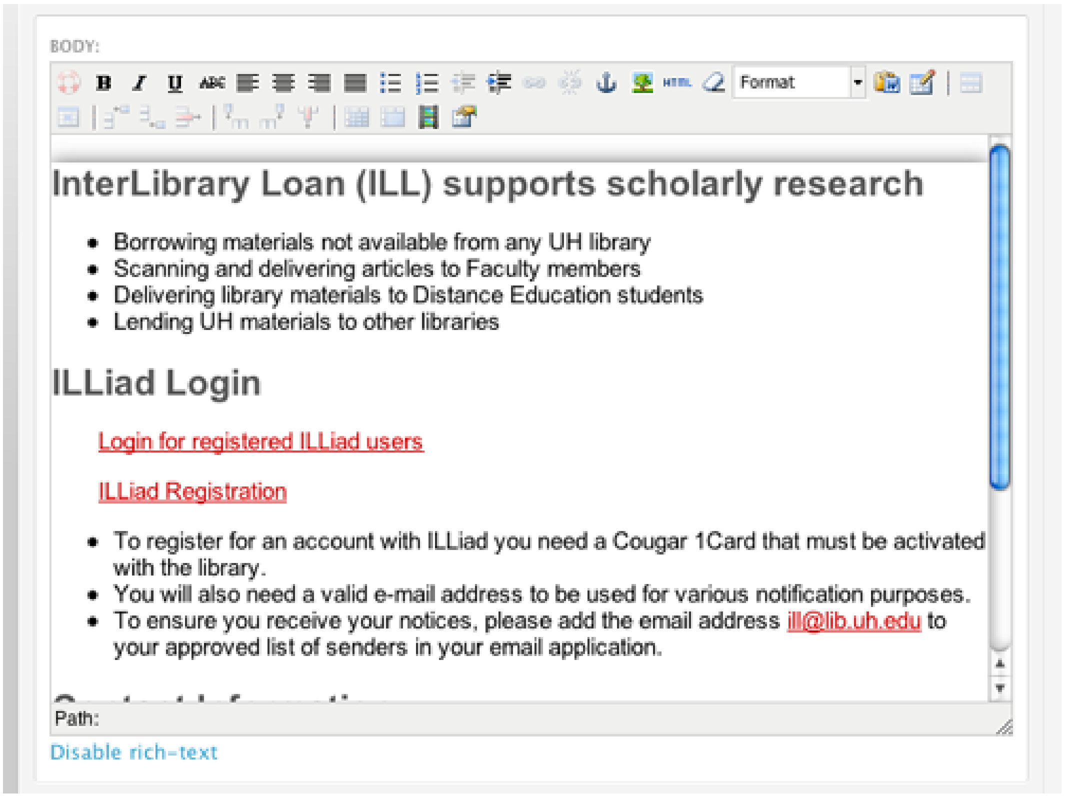
A single page was needed that would both list all past library events and grow as more were created. The Views module [17], a highly valuable smart query builder, easily made it possible to utilize Drupal Events by automatically displaying the expanding list without requiring content editors to manually maintain the page content.

Pathauto Module

Pathauto [18] is another essential module for Drupal that automatically generates human-friendly URL path aliases for content nodes. A path consisting of “node/123” is not very descriptive and would be more helpful if related to the page’s content. Pathauto works with the Book module so that the book hierarchy of the site is matched in the URL without the content creator needing to think about how to create the URL. For example, having a Book page in the site called Staff Directory that is under the About section would result in Pathauto generating the URL path alias “about/staff-directory.” which makes more sense and is easier to remember than “node/43”.

Wysiwyg Module

The Wysiwyg module [19] enhances usability by replacing the basic text-editing box with a Rich Text Format box. This allows comfortable formatting functionality similar to that of common word processors. The TinyMCE editor was chosen for use with Wysiwyg due to its clean and fast interface, stable performance and customizability. In addition, the add-on TinyBrowser makes it easy for users to browse and upload any of multiple media files to be included on the page. The result is a familiar interface for the simultaneous management of text and media content for a page.



InterLibrary Loan (ILL) supports scholarly research

- Borrowing materials not available from any UH library
- Scanning and delivering articles to Faculty members
- Delivering library materials to Distance Education students
- Lending UH materials to other libraries

ILLiad Login

Login for registered ILLiad users

ILLiad Registration

- To register for an account with ILLiad you need a Cougar 1Card that must be activated with the library.
- You will also need a valid e-mail address to be used for various notification purposes.
- To ensure you receive your notices, please add the email address ill@lib.uh.edu to your approved list of senders in your email application.

Figure 5. The editing window using Wysiwyg and TinyMCE

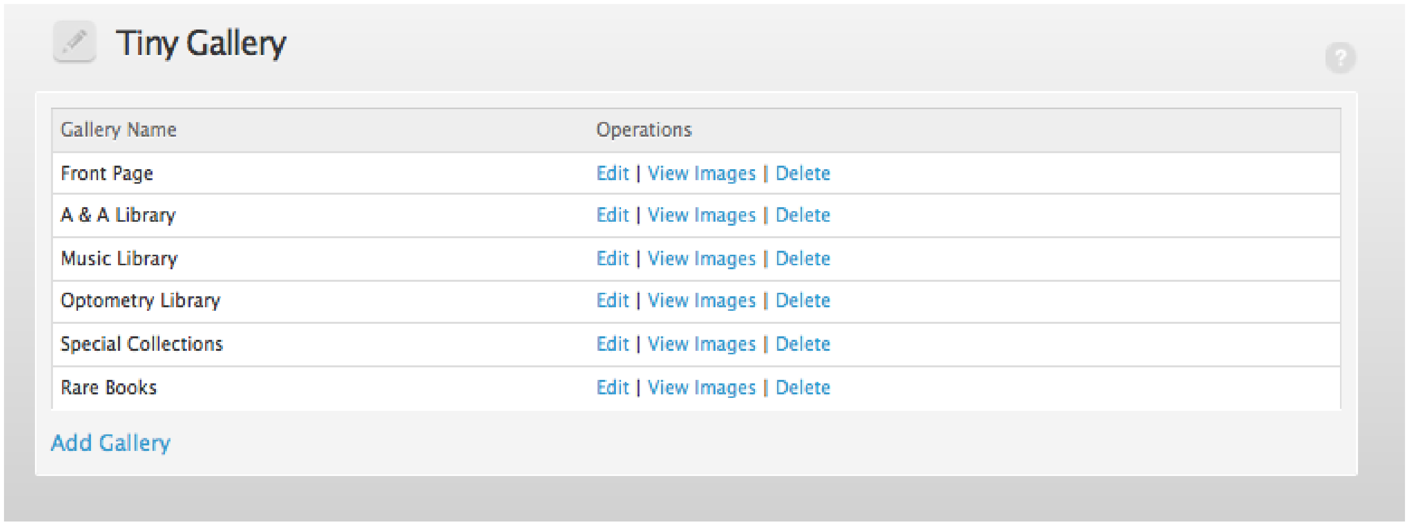
PHP Code In Content Fields

A Drupal “developer” role was created and assigned permission to use PHP code within content node fields (although generally frowned upon and a potential security concern if not handled properly). If small bits of functionality are needed in a single area of the site that do not justify the time or work to create a custom module, PHP code is used when approved on a case-by-case basis. In the future, any further extension of that code will then likely be transferred into a proper Drupal module.

B. Custom Built Drupal Modules

Tiny Gallery Module

Library content editors needed a simple image gallery that would scroll through selected images uploaded by various content providers. Many existing Drupal gallery modules were either too limited or simplistic, while others provided so much functionality that the content creation process became confusing. As a result, we created the Tiny Gallery module to fit their needs simply and easily.



Tiny Gallery

Gallery Name	Operations
Front Page	Edit | View Images | Delete
A & A Library	Edit | View Images | Delete
Music Library	Edit | View Images | Delete
Optometry Library	Edit | View Images | Delete
Special Collections	Edit | View Images | Delete
Rare Books	Edit | View Images | Delete

Add Gallery

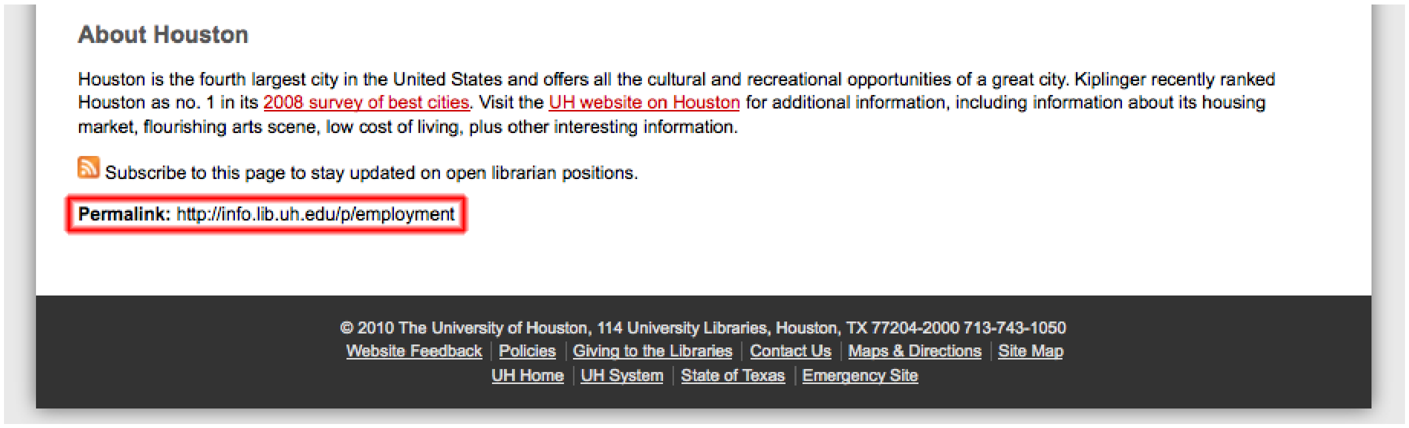
Figure 6. The admin part of the Tiny Gallery Module lets users add additional galleries or edit existing ones.

Tiny Gallery displays a transitioning slideshow with each image providing a title and URL that, when clicked, links the user to a content page with related information. The module itself is created as a single Drupal block which then displays on any number of a given set of URL paths as defined in the Tiny Gallery configuration. The single block design keeps the administrative block page small and reduces overall block processing time.

Permalink Module

The usage of the Book and Pathauto modules often resulted in Drupal URL's that were lengthy and cumbersome depending on the depth of location within the site hierarchy. These URL's were difficult to handle when printed in physical library materials, where readers need to type the addresses into their web browser via keyboard instead of a simple mouse-click or copy and paste.

Web Services created the Permalink module to add a custom, shortened path field to Book pages based on the title of that page. The path links are auto-populated when content types are created, but can be manually changed afterward by the content editor. If a link has already been created with that title, an incremented number is simply appended to the end of the path. In addition, permalinks are available as fields and can be added to any content type, not just Book pages.



About Houston

Houston is the fourth largest city in the United States and offers all the cultural and recreational opportunities of a great city. Kiplinger recently ranked Houston as no. 1 in its 2008 survey of best cities. Visit the UH website on Houston for additional information, including information about its housing market, flourishing arts scene, low cost of living, plus other interesting information.

Subscribe to this page to stay updated on open librarian positions.

Permalink: http://info.lib.uh.edu/p/employment

© 2010 The University of Houston, 114 University Libraries, Houston, TX 77204-2000 713-743-1050
Website Feedback | Policies | Giving to the Libraries | Contact Us | Maps & Directions | Site Map
UH Home | UH System | State of Texas | Emergency Site

Figure 7. When the page is created, users can either use the automatically generated permalink or can manually create one that is more appropriate for their needs. In this example, the Employment page has a simple Permalink that matches the page title.

IV. Integration with Other Library Systems

Drupal may do many things well, but it doesn't do everything. We have many other web-based applications in the Libraries, and one of the challenges was seamlessly integrating all the various systems with the new CMS for both application and end users. The first four of the five modules listed below are custom-built modules.

Homepage Search System (HSS) Module

Along with the new main website, we launched a discovery platform: Summon [20], branded as OneSearch. OneSearch was to become the primary search option on the main Libraries' homepage. However, we needed to display multiple search box options to users, not just OneSearch, and do it in a way that wasn't confusing to users by presenting too many search boxes. Unfortunately, no existing Drupal modules met our specific needs.

Web Services created the HSS module for the homepage and all the branches of the Libraries to easily administer the homepage search box, allowing editors to create any number of separately ordered tabs, each with appropriate title, description and input form. Where no HTML exists for a particular search form to be copied and pasted, Drupal provides the ability to create those forms as Drupal Blocks that can then be simply chosen from within the HSS module.

While the submission of the Summon search form currently redirects the user to the separately-hosted Summon website, there are plans to utilize the Summon API through AJAX to return form results directly within the appropriate library or branch homepage.

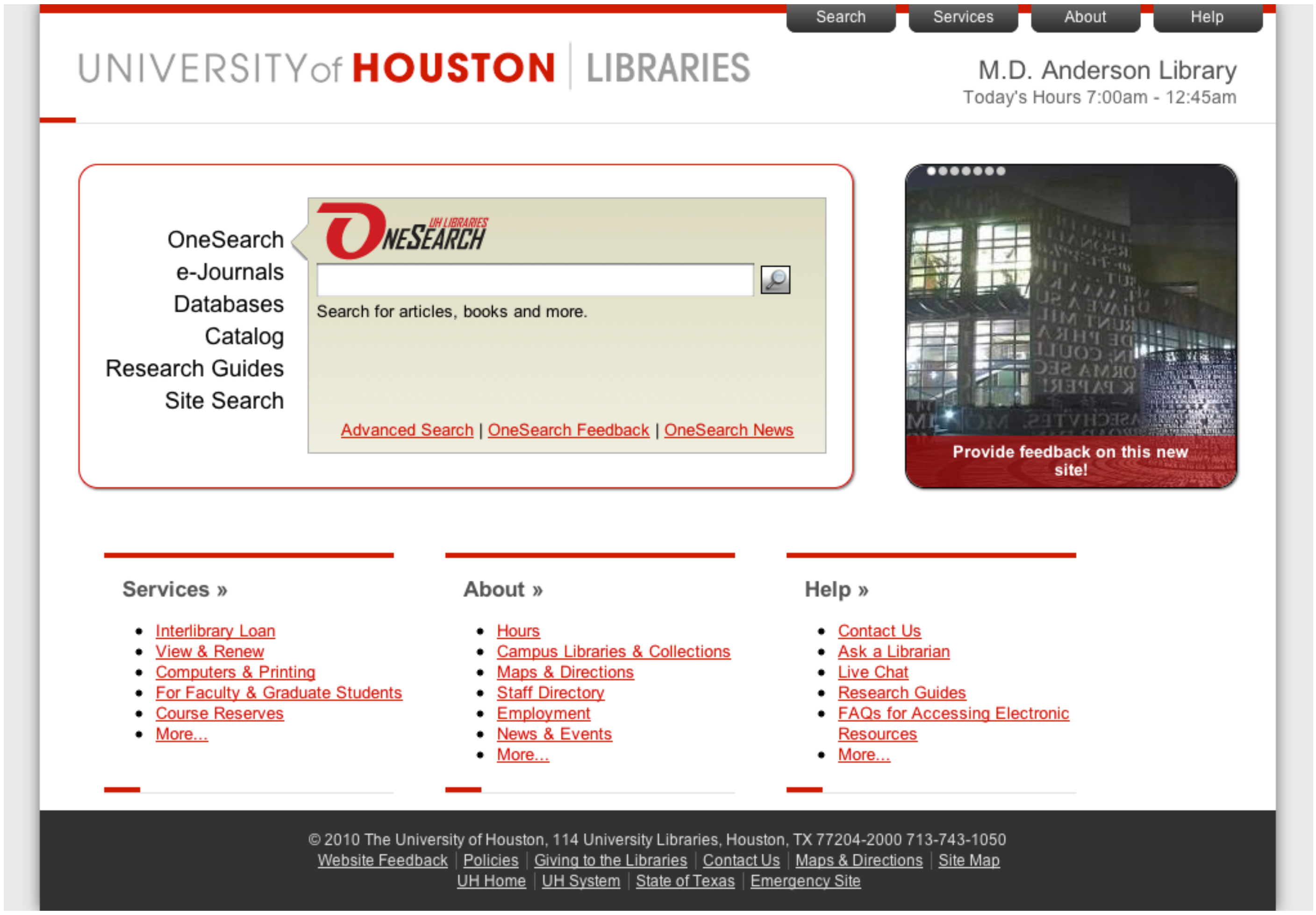

Search
Services
About
Help
UNIVERSITY of HOUSTON | LIBRARIES
M.D. Anderson Library
Today's Hours 7:00am - 12:45am
OneSearch
e-Journals
Databases
Catalog
Research Guides
Site Search
UH LIBRARIES ONESEARCH
Search for articles, books and more.
Advanced Search | OneSearch Feedback | OneSearch News
Provide feedback on this new site!
Services »
Interlibrary Loan
View & Renew
Computers & Printing
For Faculty & Graduate Students
Course Reserves
More...
About »
Hours
Campus Libraries & Collections
Maps & Directions
Staff Directory
Employment
News & Events
More...
Help »
Contact Us
Ask a Librarian
Live Chat
Research Guides
FAQs for Accessing Electronic Resources
More...
© 2010 The University of Houston, 114 University Libraries, Houston, TX 77204-2000 713-743-1050
Website Feedback | Policies | Giving to the Libraries | Contact Us | Maps & Directions | Site Map
UH Home | UH System | State of Texas | Emergency Site

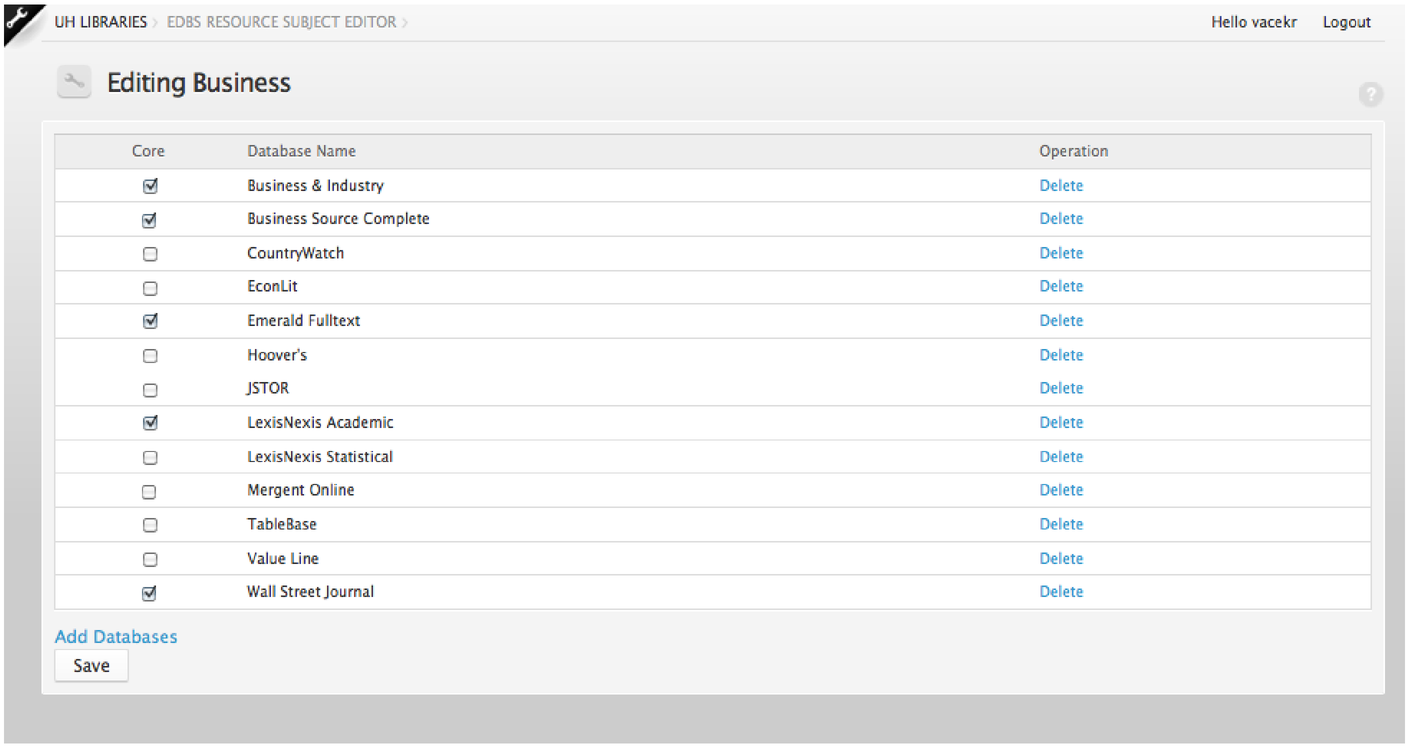
Figure 8. The University of Houston Libraries homepage displaying the Homepage Search System Module (left), as well as the Tiny Gallery Module (right), and Hours Module (top right).

Electronic Database System (EDBS) Module

Like most libraries, we needed electronic resources and their associated metadata from our integrated library system (ILS) [21] to display on the website. A custom module called EDBS was created to integrate with Drupal so that any change in the cataloged databases would then be reflected within the Drupal site with little staff interaction or redundant entry of data. The module reads a MARC-format text file output from the ILS consisting of all of the available databases and related metadata.

It then automatically performs operations such as setting categories, marking databases for full-text and media formats, and the displaying of new databases within a specified timeframe. As the MARC file lacked subject fields as part of the record, the module provides administrative pages for subjects to be manually created and assigned to databases.

The EDBS module [22] was an instant success with librarians and catalogers due to the easy user interface and integration between systems, which resulted in a much simplified workflow process and reduction of manual intervention.



Core	Database Name	Operation
✓	Business & Industry	Delete
✓	Business Source Complete	Delete
☐	CountryWatch	Delete
☐	EconLit	Delete
✓	Emerald Fulltext	Delete
☐	Hoover's	Delete
☐	JSTOR	Delete
✓	LexisNexis Academic	Delete
☐	LexisNexis Statistical	Delete
☐	Mergent Online	Delete
☐	TableBase	Delete
☐	Value Line	Delete
✓	Wall Street Journal	Delete

Figure 9. EDBS has a user-friendly interface for librarians to choose what databases they want listed under a particular subject.

Library Hours Module

The Drupal community provides a few modules to handle schedules and display hours of operation, but an existing system in the UH Libraries already managed that information with an established workflow. The Library Hours module [23] was created to integrate with that system and make the entire process much more efficient. An hours text file is output on request from the ILS, processed within Drupal and then displayed on a page with a custom user interface to display hours of operation in a variety of formats. An advanced feature allows the creation of a block to be placed within the Drupal theme to display the current day's hours for a particular library branch.

This automated integration has helpfully reduced the workload on the Web Services Department for this previously tedious task, and has resulted in faster and more accurate display of scheduling for the Libraries.

Staff Directory System (SDS) Module

The main goal of the SDS Module was to provide a means to sync staff information not only over other Drupal installations, but other web services the UH Libraries has implemented as well. The module is implemented with both server and client sub-modules.

The SDS server sub-module houses the main staff directory, editable by our Human Resources Department to keep staff profile information current. Individual staff members may also manage certain fields of their profile including contact information, background, subject area responsibilities, links to sites of interest and other related information.

Along with the Libraries' main site launch, Research Guides created with CampusGuides (the academic and more feature-rich version of LibGuides) [24] were introduced. Links to librarian subject and course guides are dynamically pulled from the SDS server and displayed on their individual profile pages.

The SDS client sub-module is for use on other Drupal installations. The client uses the RESTful API component of the SDS server to pull profile information for display within the staff directory page, caching this list for future use if the SDS server somehow becomes unavailable.

The SDS module has greatly eased the process of distributing staff information, and library staff themselves have been much more happy with only needing to update their profiles in a single place.

Blogs

Blogs have become a popular and valuable means of communication between staff and patrons of the Libraries. An installation of WordPress [25] has been in place with good reviews from content editors due to its pleasant usability experience, and many blog entries and comment discussions have populated the installation. One issue that caused some thought was whether to migrate this system from WordPress to Drupal.

After some research, we found the Blog module included with Drupal core severely lacks in functionality and finesse. We initially planned to recreate WordPress features within Drupal, but after weighing the costs of time, effort and potential user disruption, we decided to simply integrate our WordPress content into Drupal.

The Libraries' blogs are updated frequently. While the content and editing process for our main site is rigid and the Drupal backend is lightweight, we wanted the more structured workflow that WordPress offers which would more appropriately handle the volume and type of usage. Content editors continue to manage blog content in their familiar interface, while Drupal pulls certain entries such as news, emergency information or event announcements from WordPress using either RSS or the WordPress API [26] to display the content on the main site.

This architecture allows each application to take advantage of the power offered by the other with the least administrative work for the Web Services Department.

V. Reflections & Successes

Traditionally, throughout various iterations of the Libraries' website, different types of complex CMS back-ends have been built to revisit the challenges of the last. None were unsuccessful at accomplishing their goals, but a common challenge among all was a lack of fluidity with regard to changing ideas toward content itself throughout the Libraries. With this latest rewrite, the Web Services Department decided to approach things from the standpoint that focus and priority of the entire project would be on the information architecture itself.

We chose to use Drupal due to its power, reputation and active developer community. We learned from the mistakes of our intranet project, and have designed a system using the "use only what you need" philosophy. While sacrificing fine-grained controls and restrictions, this lightweight foundation will allow us to expand flexibly with minimal need for rewriting functionality, and should flow much more smoothly with changes in information architecture as the Libraries see fit.

We have overcome a number of longstanding hurdles. There has been a marked increase in interest and participation among content editors as they feel more included in the user experience process, which we hope will continue as we further refine editing features. Integration with other library systems has become even easier, and with automation and simplification freeing up time and budget for more projects, we are excited about further possibilities.

While our implementation of Drupal may not be altogether common in some aspects, it is entirely suited to this site rewrite. The beauty of Drupal is its flexibility in solving various problems. The design we have used for this project would not be appropriate for an intranet installation, for example, but the Drupal development methodology we have learned will definitely let us repurpose Drupal exactly as we would need for upcoming projects.

We have also been pleased with the outreach of interest and support from other libraries in the Drupal developer communities, which was unexpected at this early of a stage in the process. We have received valuable support through other universities and Drupal developer conferences, and have been contacted by other libraries asking for support on projects of their own. This new-found participation is extremely valuable and would have been unlikely had we chosen another software platform. Some of the custom modules mentioned in this article will be available through the Drupal site.

While adopting Drupal as a CMS platform is no easy undertaking, the growing community of skilled and helpful developers is greatly narrowing the learning curve. Drupal may be an intimidating system at introduction, but this can be overcome with patience, planning for what needs to be done, thinking less about the technology itself and more about project needs, and by being conservative with what functionality and features are used.

There may be many different ways to solve a given problem with many different levels of complexity, but an initial investment in simplification and respect for user experience can yield an extremely powerful system run by satisfied, happy users.

References

[1] The University of Houston Libraries, in Houston, TX, is comprised of the M.D. Anderson Library and three branches: the Music Library, Optometry Library, and the Architecture & Art Library. Available from: http://info.lib.uh.edu/

[2] Drupal is a free and open source content management system written in PHP and distributed under the GNU General Public License. [cited 2010 Sept 27]. Available from: http://drupal.org/

[3] Libraries using Drupal. [cited 2010 Sept 3]. Available from: http://groups.drupal.org/node/13473

4] Drupal 7 User Experience Project. [cited 2010 Sept 27]. Available from: http://www.d7ux.org/

5] Dries Buytaert's personal website. He's the original creator and project lead for the Drupal open source web publishing and collaboration platform. [cited 2010 Sept 27]. Available from: http://buytaert.net/

[6] Dries Buytaert's overview of why the D7UX Project is essential. [cited 2010 Sept 27]. Available from: http://www.d7ux.org/about/

[7] Development Seed's Open Atrium product is a Drupal-based intranet and team portal package. [cited 2010 Oct 15]. Available from: http://developmentseed.org/product/open-atrium/

[8] Admin Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/admin

[9] Menu Breadcrumb Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/menu_breadcrumb

[10] Content Construction Kit (CCK) Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/cck

[11] Content Locking Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/content_lock

[12] Central Authentication Service (CAS), provided by the Jasig Community. [cited 2010 Oct 15]. Available from: http://www.jasig.org/cas

[13] CAS Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/cas

[14] Diff Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/diff

[15] CAPTCHA is a program that protects websites against bots by generating and grading tests that humans can pass but current computer programs cannot. [cited 2010 Oct 3]. Available from: http://www.captcha.net/

[16] reCAPTCHA Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/recaptcha

[17] Views Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/views

[18] Pathauto Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/pathauto

[19] Wysiwyg Module. [cited 2010 Oct 15]. Available from: http://drupal.org/project/wysiwyg

[20] Serials Solutions' Summon. [cited 2010 Oct 15]. Available from: http://www.serialssolutions.com/summon/

[21] Innovative Interfaces' Millennium ILS. [cited 2010 Oct 15]. Available from: http://www.iii.com/products/millennium_ils.shtml

[22] We are in the process of moving the EDBS Module to the Drupal Repository under the terms of the GNU General Public License and expect it to be available by April, 2011.

[23] We are in the process of moving the Library Hours Module to the Drupal Repository under the terms of the GNU General Public License and expect it to be available by April, 2011.

[24] Springshare's CampusGuides. [cited 2010 Oct 15]. Available from: http://springshare.com/campusguides/

[25] Up until WordPress 3.0 came out in July 2010, we had been using WordPress Mu which supported multiple blogs and

multiple authors. WordPress Mu was integrated into version 3.0 and we were still able to have multiple blogs and offer the same functionality. [cited 2010 Oct 15]. Available from: http://www.wordpress.org/

[26] WordPress API. [cited 2010 November 17]. Available from: http://codex.wordpress.org/WordPress_API's

About the Authors

Rachel Vacek is the Head of Web Services at the University of Houston Libraries and manages the Libraries' web presence. She presents regularly at local and national conferences and was a 2007 ALA Emerging Leader. She enjoys playing with Drupal, experimenting with mobile technologies, thinking about the future of libraries, and gets teased frequently by colleagues about her social media and World of Warcraft addictions. She can be contacted at revacek@uh.edu.

Sean Watkins is a Web Developer at the University of Houston Libraries. He has been developing web application services since 2002 and spends most of his time typing PHP code. He enjoys building servers, creating new web services, and managing his World of Warcraft guild. He can be contacted at slwatkins@uh.edu.

Christina M. Morris is a Web Developer at University of Houston Libraries and has 14 years of experience in web and user interface development. She's also proud to be a Kingslayer on two different characters in World of Warcraft. She can be contacted at cmmorris2@uh.edu.

Derek Keller is a Web Developer at the University of Houston Libraries with 15 years of tech experience. He intelligently refuses to play World of Warcraft but is somehow accepting of Magic the Gathering. He can be contacted at dkeller@uh.edu.

Subscribe to comments: For this article | For all articles

5 Responses to "Improving the Drupal User Experience"

Please leave a response below, or trackback from your own site.

1. CMS usability: The overlooked part of digital productivity | J. Boye, 2011-04-19

   […] from a recent CMS Expert group meeting11 usability principles for CMS products by James RobertsonImproving the Drupal User Experience Case study from The University of Houston Libraries who will also be speaking at in Philadelphia […]

2. Jason, 2011-07-25

   Great article and comment, but link is dead:
   http://jboye.com/blogpost/cms-usability-the-overlooked-part-of-digital-productivity/

3. Casos de estudio de uso Drupal en bibliotecas « guilleten, 2011-10-24

   […] ejemplo, Vacek, en Improving the Drupal User Experience, se refiere a un trabajo realizado con la herramienta en la red de bibliotecas de la Universidad de […]

4. Casos de estudio de uso Drupal en bibliotecas | Wordpress 3.1.x, 2012-01-02

   […] ejemplo, Vacek, en Improving the Drupal User Experience, se refiere a un trabajo realizado con la herramienta en la red de bibliotecas de la Universidad de […]

5. CMS usability: The overlooked part of digital productivity - J. Boye, 2014-11-10

   […] Improving the Drupal User Experience Case study from The University of Houston Libraries who will also be speaking at in Philadelphia under the headline Therapy for your CMS: Improving the User Experience […]