Welcome to a New Paradigm

Content Management Systems

Libraries are about content: acquiring it, storing it, indexing it, retrieving it, and presenting it. Content management systems (CMS) help libraries accomplish these tasks on the Web by providing a back-end structure for a Web site so that the authors can focus on content. Unlike a traditional Web site, where HTML defined both the content and the formatting in a single document, a CMS uses databases and newer Web languages to store content and define formatting separately. This means that instead of using this:

```
<H1>This is the Title</H1>
```

a CMS might use this:

```
<div style="title">$site_title</div>
```

So why should you care? The real power of a CMS is that when you separate content from formatting, you can more easily change either without having to recode your entire Web site.

Instead of storing the text in the code for the page, a CMS instead uses a reference to a variable stored in a database. Similarly, the formating for the title in this example is defined in a style file instead of within the page code. So why is this so important? This separation of content and formatting is the base technology that makes possible everything discussed herein. When content is approached in this separated fashion, it can be presented for editing in a graphical WYSIWYG (What You See Is What You Get) environment. If the content is edited and stored outside of the Web site itself, you can assign different users different levels of permission for editing content. Furthermore, since adding content doesn't involve editing the code of the Web site, it becomes much easier to make more frequent updates. In the following chapters, you will learn how Drupal, an open-source content management system, makes use of the separation of content and formating to let you create a powerful, dynamic library Web site.

Benefits of a CMS

Content management systems can be defined by three common attributes beyond the separation of content and formatting:

- They provide a framework for creating, managing, and publishing Web-based content.
- They provide a secure environment with managed user roles.
- They provide extensions for enhanced capabilities.

Though small library Web sites may have only a few pages, larger sites can run to hundreds or thousands of pages. CMS sites are based on dynamically generated chunks of content that can be more easily displayed for online management (see the sidebar HTML vs. PHP). This is especially important given the ease with which new content can be created in a WYSIWYG environment by users who may not have been able to participate in a code- and file-based system. With a CMS, work can be distributed across multiple authors without sacrificing site security: a department head can publish news updates directly to her or his page, while another staff member can write content that has to be approved before it is displayed. Finally, most of the available CMS applications can be extended to accomplish new tasks or provide new data types through small plug-in modules that meet a specific need.

HTML vs. PHP

A traditional HTML-based Web site is built from many files, one for each page to be displayed. Files like `index.html`, `hours.html`, `contacts.html`, and so on are stored in directories. In a CMS, the separation of content into a database means that our example pages have changed from static files stored on a drive into dynamically generated views that are built from a database at the point of need. What was `hours.html`, a file that held both the information about the hours the library is open and a page layout definition, is now replaced by a general template file such as `pagetemplate.php` and a content chunk about the hours of service. The .php extension here refers to the PHP programming language on which many CMS applications are built. *PHP* is a recursive acronym that stands for "PHP Hypertext Preprocessor": literally, a language that does things to hypertext Web content before it is processed by the browser. This means that when your browser encounters a PHP file, it interprets the code stored within and dynamically renders the hypertext code to be displayed. If you view the source of a PHP-based page, you will see the resulting HTML code and not the PHP that preprocessed the HTML.

Most modern Web applications are really just highly specialized versions of this basic CMS definition. For example, a blog is a CMS that is focused on making it easier to add content frequently, while a wiki is more concerned with keeping track of users and their edits. General CMS applications, often called portals, strike a balance between the three attributes listed above to create programs designed to help you manage a large Web site. As with any other type of software, there are many different possibilities available from which you can select the best CMS for your situation. While choice is nice, the prospect of downloading and installing multiple systems to test them probably sounds a bit daunting. Do not despair: Open Source CMS has collected dozens of systems and provides full administrative access to each so that you can try out all of them in one place. Three years ago, we evaluated the portals available and selected Drupal as the CMS with the best balance of usability and power.

Open Source CMS
http://opensourcecms.com

Why Drupal?

Drupal, the hammer that makes all your Web endeavors look like nails.

To call Drupal a CMS might undersell it a little bit. The Drupal community often refers to Drupal as a content management framework. This is intended to convey the idea that Drupal is not a fixed system but a framework on which you can build your own systems. When we started using Drupal, we were looking for a CMS, so we were looking at products like WordPress and Mamba. Other groups approach Drupal as an alternative to rapid Web development tools like Ruby on Rails or CakePHP, neither of which would be considered a CMS. Drupal offers a nice learning curve as you move from using it as a CMS to using to using it to develop your own Web applications. You can use it out of the box as a very nice, dynamic Web site. As you learn more about configuring it from the Web interfaces, though, it becomes very flexible. Then when you are ready to start writing your own code, it provides you a powerful set of internal APIs (Application Protocol Interfaces) to streamline your coding.

With the continued threat of legislation that would ban social Web sites from schools and libraries, we wanted to find a way to continue to provide access to social Web applications in an educational setting.[1] There is no dearth of open-source projects looking to duplicate popular Web services like del.icio.us, Flickr, and the like. But we were also looking for a Web application to help us manage our projects and track issues and problems. The fear was that we would end up with a huge number of applications to download, install, and maintain on our servers.

We had already been using Drupal for a number of tasks, including our public Web site, extranet, and a shared electronic resources purchasing system. As we continued to work with the system, it became obvious that we could use Drupal for almost anything we needed: shared department calendars, project management, bookmarks, knowledge base, and more. With a staff of three librarians, our system cannot become experts on many different Web applications. Drupal provides a stable and flexible platform upon which we can base our Web development.

The major project that we have been working on in Drupal is a next-generation library portal called Fish4Info. This site features a library catalog built in Drupal along with book reviews, pathfinders, an events calendar, and much more.

Who Is Drupal?

As with most successful open-source projects, Drupal's greatest asset is its community. Drupal was created by Dries Buytaert as a Web board on a shared dorm Internet connection, developed into the Web site drop.org, and finally distributed as an open-source Web platform.[2] Through the development of Drupal 5 and much of Drupal 6, Dries continued his studies full-time as a PhD student in computer science, spending his days working on Java and his nights guiding the development of Drupal.[3] Since graduation, Dries has found venture-capital funding and has formed Acquia to provide commercial support to the Drupal community, much as Red Hat provides support for Linux users.[4]

Unlike the larger-than-life personalities that dominate some open-source projects, Dries has acted more as a guide than a dictator. He spends much of his time offering guidance, reviewing patches, and helping to settle debates. Perhaps this is because of the open nature of the Drupal community that Dries has fostered, but the success of Drupal is mostly due to the number of talented developers who contribute their work to help move the project forward. Most of the active developers in the Drupal project work as Web site developers and have a real stake in the growth of the project. By developing and sharing code, they make each other's work that much easier. But much of the development is done with a "scratch your own itch" mentality: projects are most often done when they are going to be used by the developers coding them. In addition, many of the new and cool features that get put into Drupal end up being invisible to the lay user and can be appreciated only by users who are themselves building Web sites.

To counter the developer-centric current within the Drupal community, Jeff Robbins of Lullabot started the meme "Drupal Will Save the World," which seeks to put an idealistic spin on Drupal development—the thought being that Drupal can provide a powerful activist platform for the disadvantaged.[5] Drupal proved its worth as a community tool in 2004, when staffers in the Howard Dean campaign used Drupal to create DeanSpace, later named CivicSpace, which is widely used in the political realm for campaign sites. But to really save the world, Drupal needs to be more accessible to disadvantaged communities who do not have the technical background to build these Web sites. Along these same lines, the University of Minnesota Libraries stepped in to do formal usability testing on Drupal 6, seeking to greatly improve the usability of its interfaces.[6]

The result of the pairing of a strong development community with a focus on usability is a powerful framework that comes with a usable front-end CMS.

Notes

1 Andy Carvin, "New Federal Legislation Would Ban Online School Networks in Schools & Libraries," www.pbs.org/teachers/learning.now/2006/05/new_federal_legislation_would_1.html (accessed Feb. 15, 2008).

2. Drupal, "History," http://drupal.org/node/769 (accessed Feb. 15, 2008).

3. Lullabot, "Drupal Podcast No. 32: Dries Buytaert," www.lullabot.com/audiocast/drupal_podcast_no_32_dries_buytaert (accessed Feb. 15, 2008).

4. Dries Buytaert, "A Drupal Startup," http://acquia.com/node/8 (accessed Feb. 15, 2008).

5. Jeff Robbins, "How Drupal Will Save the World," www.lullabot.com/articles/how_drupal_will_save_world (accessed Feb. 15, 2008).

6. Chad Fennell, "Drupal Partners with U of M Libraries—Formal Lab Usability Testing of D6, http://drupal.org/node/204667 (accessed Feb. 15, 2008).