Computer Writing and Research Lab

White Paper Series: #050504-1

Towards a Dynamic Community Identity: Transitioning to a CMS-Driven CWRL Web Site

James Brown (jimbrown@mail.utexas.edu)
Mariela Hristova (mariela@mail.utexas.edu)
Thomas Nelson (tjnelson@mail.utexas.edu)
Matthew Russell (mrrussell@mail.utexas.edu)
University of Texas at Austin

4 May 2005

Keywords: textuality, hypertext, markup, literature, pedagogy, literacy, reading

Abstract: This paper explains the rationale for moving the CWRL web site from static HTML pages into a content management system (CMS). It anticipates some of the main changes that this move will entail both in our online identity and in the process of creating and managing web content. students gain a first-hand look not only at hypertextuality but also at textuality more generally, which can then serve a variety of purposes in the teaching of literature.

Introduction

During the 2004-2005 academic year, we have been conceptualizing the transition of the CWRL site into a content management system (CMS) and exploring its implications. In this paper, we will lay out our vision for a CMS version of the site and will explain the justifications for this change, alluding along the way to potential new roles for web developers. First, we look back at several past versions of the CWRL site to gain insights into priorities articulated by previous web developers. We consider problems with structure and design to envision possible solutions. Then, we provide an overview of content management systems and explain the advantages of using Drupal, our CMS of choice. Our discussion focuses on accessibility, distributed authorship and community building. Finally, we explore how content in general and the specific content types in Drupal will affect our virtual identity and will enhance the user experience of our web site.

The Evolution of the CWRL Web Site

While most of this paper is concerned with implementing the CWRL's forthcoming CMS-driven web site, the decisions we made were influenced by previous versions of the web site. The earliest version of the site we will discuss will be iterations of the site administered with Cold Fusion (1999-2003). Subsequently (2003-2005), the web site was composed of static pages created with HTML coding and style sheets. Some of the decisions behind the initial static site (2003-2004) will be discussed below, as will the redesign made the following year.

Bill Wolff's 2003 white paper "Re-Designing the CWRL Web Site: A Look to the Future" explains the logic behind the CWRL static site design. The most pressing motivation for converting the site from its Cold Fusion predecessor was to meet University Web Accessibility standards. The change into static HTML and SHTML pages allowed for several other site improvements as well: simplifying a baroque site structure full of duplications, dead ends, and information-poor pages; implementing css-based design; and improving site usability according to the theories of Jakob Nielson and others. Chief among these additional concerns was simplifying the site's strictly hierarchical organization: the structuring principle that had evolved in the previous site had been to defer information to the hierarchy itself. In other words, a user would have to click through several pages which simply listed links before finding what she was looking for. The Summer 2003 web design team (Wolff, Jennifer Williams, and Tom Nelson) elected to create a text-heavy, information-dense site organized into user-based hierarchies. Rather than click through pages of links, a visitor to the staffing node (for example) would go to a page that included links embedded within a description of the node. This textually-oriented approach seemed particularly appropriate for the primary users of the site: teachers and students of writing and literature.

Although the new site consisted exclusively of static HTML and SHTML pages, the overall look reflected the influence of blogs. This decision was influenced by the blog mania that was beginning at the time (and by the lab's concurrent integration of blogs into its array of resources), but also reflected aspects of the blog ethos that we hoped to integrate into the site: frequently updated, textually orientated, and css-built pages. However, two years of the current generation of the lab site has exposed some limitations inherent in static sites in general. Despite the initial "bloggy" appearance, the static site by definition lacks the database and content management capabilities that make blogs so popular. While the current site is demonstrably more usable than any previous incarnation of the CWRL web site, it still requires an "archon" to maintain its hierarchies and administer content. Changing and adding pages is still a labor-intensive job which requires an administrator or set of administrators versed in the structure and content of the overall site. Moreover, the hierarchies themselves constrain the users of the site, even though they are based on the presumed needs and interests of the users. In our new CMS-driven site we've retained the user-based hierarchies as an apparent structure, though thanks to the taxonomies, a search function, and database structure (all described in greater detail below), these hierarchies are only one way of using the site, not the only way. In fact, as we'll see, the user-based hierarchies of the new site are illusory. They are retained as a metaphor for site structure, but in no way the literal structure itself. The hierarchical metaphor is in essence a vertical rendering of horizontal content.

Wolff's white paper is, as the title indicates, a memo to future CWRL web designers, to warn them against repeating past mistakes. While we've decided to supplant the static site with a dynamic CMS site for reasons laid out in this paper, we do so with his injunctions in mind. The previous Cold Fusion site was, as Wolff says a "structural disaster," difficult to administer and not compliant with accessibility guidelines. Our switch to Drupal only moves further away from these problems: all the content automatically becomes accessible, the database opens the site up to multiple structures, and by distributing authorship it is in fact easier to administer. Like the previous site, the Drupal site uses stylesheets to determine appearance, so that we can continue our commitment to cutting edge design. Adopting Drupal will stop obligatory annual redesign of the site.

Visual Design Logic and Structural Problems with the Second Generation Site

In the summer of 2004, the developer team attempted to improve upon the nodal structure of the static site by further reducing the visual clutter and repetitive displays of information. Even though the design of the second static site was thought of as extending the benefits of the first static site, it also repeated several problems. Indeed, in light of the improvements that Drupal seems to offer, we suggest that a static site model will not be useful in the future.

The design team sought to constrain the number of links that were present on every page of content. Vertical menus, listed on the left of all pages, were created as a way to reflect information within the six nodes, listed horizontally across the top. Mirroring underlying structure, the design, and especially the design of the nodes themselves, attempted to anticipate the way that particular audiences would like to move through the site by fixing content in particular nodes. The "Research" node, for example, was intended to contain all of the professional development, academic resources and research links available through the site, acting as a clearinghouse of timely and regularly updated information that would encourage repeated visits. In this sense, the design was intended to give a kind of narrative coherence to the site as a whole.

Even though the redesign attempted to consolidate control over structure, it became apparent that the overhaul of the first static site did not sufficiently address fundamental problems. The second site design team had hoped to anticipate content additions by imagining an overarching and unchanging nodal vocabulary that was made up of both audiences ("Students," "Teachers") and resources ("Technology," "Resources"). Without a consistent scheme, the web developers who worked with the site discovered that the nodes presented a number of classificatory limitations as new content stubbornly resisted fitting into the existing nodes. In addition, changing the site in order to fit the appearance of new content was cumbersome, since the main horizontal menu was image-heavy and required time-consuming work to change.

All of these changes had to be made by web developers who were also attempting to regulate content. In this top-down hierarchy, where web developers acted as gatekeepers, decisions about appearance, were effectively and inexorably tied to decisions about structure: Web developers were still developing the appearance and visual look of a site while attempting to regulate its already existing structure. As a representation of the multiple identities of the lab, the contradictory and occasionally debilitating design and structure decisions decreased the ability of the lab to effectively represent these identities to various audiences. Decisions about the proper location for new content had decisive and unavoidable effects on every other node. Subsequently, these decisions actually ended up highlighting absent content rather than drawing attention to the information that was actually present.

These problems led to a further distancing of the web developers both from the intended value of the site and from the content that inhabited it. Although web developers have an obligation to build and regulate the lab's web site, they found it difficult to act also as browsers or effective users of the site they were constructing. Questions as to the final nodal home of new content were always contested, and attempts to anticipate these problems by providing deliberately ambiguous nodal names did not function as anticipated. This last problem was one that made the constitutive and regulative roles of the web developers appear to work against each other: web developers were struggling with the unrelenting task of categorizing content, and these category decisions had unavoidable effects on the multiple identities of the lab.

The continuing problems with the site were not reflective of the tireless efforts of the web developers, but a function of the static site itself—of its inherent limitations as a structural model. In considering the static site, we find that the idea of nodes could indeed still be useful for interacting with the users of the web site. As we are discovering in Drupal, a content management system is able to effectively mimic the nodal design of the second static site by offering an apparent hierarchical ordering of the site. Yet this appearance is, for the most part, a convenient fiction that does not reflect the database structure of Drupal. Indeed, the dissociation of design and structure that Drupal provides could potentially avoid the problems of the second static site while retaining its overall, audience-driven approach with nodes. As we shall see, this dissociation allows users the opportunity to individually engage with the lab's multiple identities. Thus, the real benefit of Drupal is that it fundamentally reconfigures the nodal structure of the site. With Drupal, content is readily sorted by users, who become the true 'nodes' of the site. Users are more free to participate in the development of the site, and web developers are more free to act as regulators of content without getting bogged down with difficult decisions about the site's overarching structure.

Advantages of using a CMS

A CMS will best facilitate the generating, managing, and organizing web content by people without prior web development experience. To effectively integrate all processes related to web site construction, content management systems separate web sites into three independent elements: design, structure, and content. Customizable templates are employed for the generation of each web page to control the visual design of the site. The structure is manipulated often through a series of interfaces that manage structural elements such as headers, footers, navigation menus and breadcrumbs. Lastly, the content is submitted through web forms primarily as plain or html-enriched text, or by uploading media files. To be handled properly by the system, each unit of content might have several attributes (title, category, etc.) that are recorded in the database along with the actual data.

We have found Drupal to be optimal for our purposes for reasons discussed above, and because it includes features supporting the generation of content, as well as online discussions and user management. In addition to being compatible with our hardware, this CMS is supported by a large external developer community, which will facilitate our customization efforts and reduce the burden of extensive in-house revisions. Most importantly, Drupal treats design, structure, and content in ways that are advantageous to web development in the CWRL.

The University of Texas and the CWRL focus extensively on offering a meaningful online experience to all types of audiences. That is why standards compliance and accessibility rank high among our web development priorities. A CMS offers us the chance to benefit from the centralization of the information architecture of the site. Because the information architecture is built into the system, the implementation of accessible design templates will consistently generate pages that are standards compliant. Such an approach eliminates the requirement that contributors of content know how to make their materials accessible, allowing them to focus exclusively on the substance of their contributions.

Other advantages of using a CMS for our site relate to authorship and maintenance. Since the updating or adding of content can be done through the use of web interfaces, anyone can easily contribute materials, without having to funnel them to a small group of web developers who spend an inordinate amount of time updating the site. The new site will still require updates, but the web developers can now play the role of coordinators. They will identify various gaps in content or outdated information and request updates by the appropriate authors. Drupal supports more streamlined collaboration, and a CMS will allow for a greater sense that the CWRL web site is a community text. The easy uploading of content can encourage CWRL members to contribute more than they would under the current arrangements of the static site, perhaps taking the initiative for developing new content areas and participating more actively in the construction of our community identity.

The most drastic change in our transition to a CMS, however, is the rethinking of the concept of "content," which will inform our practical choices in transferring the CWRL site over to Drupal. The static site consists of pages for which an individual developer decides how the content should be displayed and how it fits into the overall context of the site. Static content offers extensive control to the developer, but it places technical and decision-making requirements in terms of knowledge of HTML and a vision for the overall logic of the web site's organization. Content in Drupal, on the other hand, is much more narrowly construed to mean "chunks" of information that reside in a database as separate records. The descriptive attributes of each "chunk" are related to it in the database and serve as the basis for its dynamic organization. Thus, contributors need not have any technical skills to easily submit information. However, they need to have an understanding of how the overall site organization works in order to use the content attributes effectively. What makes such content dynamic is that the exact placement is performed on the fly based on the assigned attributes.

Dynamic content management allows web developer to shift roles from regulating to moderating. Web developers need to regulate static content because they have to decide how to integrate every new page into the site as a whole. Regulation requires them to at least attempt some sort of structural consistency. Dynamic content management allows them to concentate on moderation. That is, when a contributor submits some information and assigns a few attributes to it, the content is immediately and automatically integrated based on those attributes. Furthermore, we might expect contributors to be accurate in assigning subject-specific attributes. To better integrate the submitted content into the overall site, web developers can assign more attributes to it or place it within relevant hierarchies within the site, while leaving a significant part of it in the hands of individual contributors, thereby allowing the identity of our site to emerge as a community effort.

The main advantage of dynamic over static content management is, in terms of developer roles, greater consistency. With the use of static content, various interpretations of the site structure over time can lead to a lack of consistency, baffling users' sense of the site's organization as they run up against pages added by different developers. Moreover, many pages would logically fit in more than one location on the site, so any developer is faced with the obligatory choice to make a final decision and reflect the other possible placements at most through the use of contextual links, i.e. hyperlinks for specific phrases within the text. While contextual links can be useful in connecting to other web sites or more detailed information on the linked concepts, they are usually not perceived by users as structural pathways to move through a site. Drupal visually separates navigation links and contextual links. It automates navigation, so contributors would not need to improvise when trying to integrate new content within the site.

Content Types: Taxonomies & Hierarchies

In our first test of Drupal content types, we attempted to lift one of the nodes of the static site and place it into the Drupal installation. In order to see the differences between the various content types, we recreated the "Teacher" node using three of the types: books, pages, and stories. As we plugged content into the Drupal installation, we discovered surprisingly little difference among these content types. The "create content" page explains them as follows:

> **Book Page:** a book is a collaborative writing effort: users can collaborate writing the pages of the book, positioning the pages in the right order, and reviewing or modifying pages previously written. So when you have some information to share or when you read a page of the book and you didn't like it, or if you think a certain page could have been written better, you can do something about it.

> **Page**: if you just want to add a page with a link in the menu to your site, this is the best choice. Unlike a story, a static page bypasses the submission queue.
>
> **Story**: a story is similar to a newspaper article. If stories are moderated, the post will be submitted to the attention of other users and be queued in the submission queue. Users and moderators vote on the posts they like or dislike, promoting or demoting them. When a post gets above a certain threshold it automatically gets promoted to the front page.

These content types do have potential differences, most of which focus on the different forms of collaborative writing that Drupal can facilitate. However, we envision our CMS as having only limited collaborative writing applications. We do see collaborative possibilities for the "forum topic" content types, and we imagine interesting ways in which students and professors could post announcements to the site such as CFPs or conference announcements. For our purposes, books, pages, and stories could be used almost interchangeably.

In our early discussions, we saw the book content type as a way of recreating the "node" design of the static web page. Upon experimenting with books and book pages, we began to see how our Drupal installation would look. There would be a book to serve as an introduction to an audience-based node and then content would fall within that book in the form of book pages. We considered the audience-driven structure of the static page to be useful. Therefore, we decided to create a book or story for teachers, students, proctors, developers, and visitors, and then plug content into these "nodes" via book pages.

At this point in the process, our new site was beginning to look nearly identical to its static predecessor. However, our use of taxonomies was where we saw the greatest opportunities for innovation. By associating content taxonomy terms, we saw CMS as providing a nonlinear way of navigating the site – one that accompanied our vision of audience nodes. Thus, we would provide users with one idea of how they might want to navigate the site, but taxonomies would provide numerous other possibilities.

An example should serve to make this point more clear. The static CWRL site contains a Dreamweaver tutorial. A user hoping to locate this tutorial would have three choices:

1. Locate the "Technology" node and follow the link to the "Tutorials" section
2. Type "Dreamweaver" into the Google search bar

3. Use the sitemap

The first of these three options would be covered in our new design. We would create a book or story called "Technology" and then create a page within it called "Dreamweaver Tutorial." The Google search is something that we found troublesome in the static site as it was typically not a very useful function. The Google search in this example does get us to the tutorial (though not efficiently, due to the noise of instructor pages), but we believe the Drupal search function to be superior because results will be limited to our database content. Currently search results include teacher pages and anything contained on the CWRL web server. This makes Google search results too broad, a problem that will be eliminated in Drupal. The final option of visiting the page's sitemap is a viable one, but seems somewhat cumbersome.

We have placed the Dreamweaver tutorial within the "Student" node, but it is also associated with any number of taxonomies such as tutorials, web design, or software. A visitor to the site would now be able to arrive at content from an array of different paths limited only by the number of taxonomies associated with the Dreamweaver tutorial.

Drupal's ability to provide numerous pathways to content gives the user more freedom and can lessen the role of the designer in determining how one experiences the web site. Yet, the idea of taxonomies raises a host of new questions: How specific should taxonomy terms be? How many of them is too many? Too few? At some point the system could become completely unwieldy and unmanageable due to the number of terms created. We believe the advantage of Drupal is that such questions can remain open-ended, and the flexibility of the system allows design decisions to be a process rather than an endpoint.

Conclusion

Adopting Drupal will stop obligatory annual redesign of the site. Since new styles can be introduced at any time, and since content can be re-organized by any user, our collective energies can be focused on enhancing Drupal for other needs, such as blogs, discussion forums, and internal administration. As we have shown, the historical trajectory of the CWRL site has been plagued by repeated redesigns and "reinventions of the wheel." Transitions from Cold Fusion to the current static site have proven to be arduous and labor intensive. Our hope is that Drupal will provide a more stable infrastructure that can remain in place regardless of changing design and content. Decisions made today are not static in any way, and future developers will be free to rethink the site. However, we encourage future developers to remain within the framework of a database-driven site. We believe that continued work with Drupal is much more efficient than attempting brand new iterations of the site on an annual or biannual basis. Regardless of our initial design decisions, the advantage of a CMS is that new design concerns and architecture decisions can be easily implemented. We have chosen to retain the audience-driven design to provide continuity for current users of the site. It's possible that future groups will not see the need for such a design or that the taxonomies we've created are too broad or narrow. Unlike the static site, Drupal allows for more flexibility in design changes, and future developers will be able to put their own stamp on the CWRL site without engaging in a complete overhaul.