

# Content Management Problems and Open Source Solutions

by Seth Gottlieb | Content Management Practice Lead

The open source community has produced a number of useful, high quality content management systems which presents an opportunity to deliver tailored content management solutions without the high licensing or management fees associated with commercially-licensed or hosted software. However, the sheer number of open source CMS projects and the ineffectualness of traditional commercial software selection techniques can make the task of finding the right open source software an intimidating challenge. The strategy of using feature matrices is particularly ill-suited to open source software selection. A more practical approach is to match your needs to a common business problem that others have solved using open source software and engage with the community to learn about their experiences in implementing the solution. Doing so will take advantage of the unique aspects of open source software: the openness of the user community and the transparency of the development process.

The content management use cases that are particularly well served by open source are: informational websites, online periodicals, collaborative workspaces, and online communities. This paper briefly describes some open source projects that have been successfully applied to support these use cases and gives techniques for how to engage with the community. While open source is frequently and successfully used as an alternative to custom development of unique solutions, the use of open source software will be the topic of another white paper or case study.

## Table of Contents

Introduction.....	2
When Open Source.....	3
The Informational Brochure Web Site.....	4
eZ publish 3.7.2.....	5
Mambo and Joomla 4.5.3/1.0.5.....	6
OpenCMS 6.0.2.....	8
TYPO3 3.8.1.....	9
Online Periodical.....	11
Bricolage 1.9.1.....	12
Lenya 1.2.4.....	13
Midgard 1.7.3.....	14
Zope CMF 2.8.5.....	15
Collaborative Workspace.....	17
Alfresco 1.1.2.....	18
Plone 2.1.1.....	19
Wiki as Collaborative Workspace.....	21
MediaWiki 1.5.4.....	22
Twiki 20040904.....	23
Online Community.....	24
Drupal 4.6.5.....	25
phpBB 2.0.19.....	27
Roller 1.3.....	29
How to evaluate Open Source.....	30
Where to Put the Money That You Save.....	31
What to Do About Support.....	31
Conclusion.....	31

## Introduction

Recently Bob Doyle of CMS Review posted on the CM Professionals mailing list the results of some rough analysis that concluded that there may be 1,785 content management systems (CMS) in existence. It is likely that a significant portion of this population is open source software. While there is a broad selection of open source content management software, given the number of dead and immature projects and the absence of marketing resources and analyst attention, the number of options turns into more of a liability than a benefit. The good news is that once you have narrowed in on a subset of viable solutions, the openness of open source software makes the strengths and weaknesses of these applications and the communities behind them much more transparent than commercial offerings. This whitepaper describes a strategy for segmenting the open source CMS market and give overviews of some of the leaders in each of the segments.

The field of content management has become mainstream enough where the term is now familiar within the workplace and the solution market is fairly mature. The pain that content management is designed to address is nearly universal. Ask any business if they have a content management problem. The problems they identify may be vastly different, but they will still be viewed as problems related to managing content. Consequently, content management is an extremely broad term encompassing everything from managing simple web sites through centralized document repositories to highly-specialized applications that support complex, content-centric business processes.

The CMS market, as a whole, is segmented into several primary categories: Web Content Management (WCM), Document Management (DM), Digital Asset Management (DAM), and Enterprise Content Management (ECM), which tries to do it all. This article focuses on the WCM category, which in addition to having the majority of the hundreds of open source CMS projects, also has a vast selection of small and middle market commercial solutions as well as hosted applications. There have been some important contributions to sort out this seemingly boundless, disorienting market; most notably CMSML<sup>i</sup> and CMS Matrix<sup>ii</sup>. These resources compare content management software by feature and are helpful to narrow the field of solutions to a manageable list.

But even the main proponents of CMSML will tell you that a feature list alone will not select software. The existence of a feature only tells you part of the story. How a feature has been implemented determines its utility within a business context. For example, a generalized feature such as "workflow," which many CMS claim to have, can mean very different things to different people. In some CMS, the workflow feature consists of a simple approval mechanism where a user with a particular role has the responsibility of transitioning an asset from a "working" state to a "live" state. Another workflow implementation might allow the definition of several states and transitions and guards that guide content through a many stage lifecycle. Yet another workflow implementation might consist of task lists that are only loosely tied to their related assets. Depending on the needs, any of these "workflows" may be the perfect solution.

Even in open source software, which tends to be more honest about feature lists (because it is easier to verify if a feature exists and because an admission of a weakness is an invitation for someone to contribute a solution), the existence of a feature does not necessarily mean that it is useful to you. Throw in the marketing gamesmanship that occurs between competing commercial software companies and feature-by-feature comparisons become even more suspect.

This paper takes a different approach that addresses common content management business problems or use cases<sup>\*</sup>. By doing so, and then categorizing the solutions by the use cases that they have successfully solved, one can not only get to a short list of solutions but also identify a community with the same needs with which to work collaboratively. This is particularly important in open source software where the community and the direction of the software can be more important than its current capabilities. If your company's content management problem fits into one of these archetypes, you can leverage the experience and work of other companies, not only in their software selection, but also in the feedback that they put back into the product.

The risk of adopting a software solution is greatly reduced if you are not the first to use the software in the way that you intend to use it. The open source software model creates an open forum for identifying, collaborating, and learning from other companies who have faced similar challenges. Moreover, the nature of open source makes it more sensitive to evolution by the community that uses it than closed source software. Through mail lists and other forums, one can connect with other users of an application and find out what they are doing with the software, what modifications have proven effective, what modifications were painful, and where the software is going.

---

\* Here, the term "Use Case" is not used in the traditional software development sense but rather the high level business context of what the software is used for.

What follows is a list of some archetypal business problems coupled with brief descriptions of some open source solutions that have evolved within these contexts. In each of the archetypal use case, this paper looks into three areas of functionality (content creation, management, and presentation) and identify characteristics that these business problems require.

#### Note about the ratings used in this report:

This report does not attempt to rate the general quality or usefulness of the content management systems reviewed. As stated earlier, that depends on what you are trying to do with the software. Also, it is often easy to change what you don't like about the software. However, this report does rate the resources that are available to help you with your implementation. In particular, the report grades the comprehensiveness and availability of documentation and the energy and helpfulness of the community. The rating system is course grained with only three levels: below average, average, and above average.

Rating Legend		
Below Average	Average	Better than Average
		

#### When Open Source

First, let's briefly discuss when and why you would consider an open source. Open source content management software is most frequently used in small to medium sized web sites with very common requirements (such as corporate identity websites and departmental intranet sites or online magazines rather than large product websites with hundreds of thousands of pages) and as a foundation for building unique, highly-customized solutions (such as Amazon.com which uses open source components such as Perl, MySQL, and the Mason templating engine).

Interestingly, the interconnectedness of the web and technologies such as portals, search, and RSS, can make having several small and medium sites more practical than trying to standardize on a single central platform<sup>iii</sup>. Large companies do not need to construct a monolithic intranet platform if smaller departmental sites can be integrated into a comprehensive intranet. Regardless of whether there is one system with multiple sub-sections or multiple sites, the same challenges of management and organization exist. While no commercial vendor has solved the problem of programmatically taming large volumes of poorly managed content in one repository, the technology that drives basic web sites and web-based content applications, such as the types described later in this paper, has become commoditized. Market leaders in the commercial space have tried to differentiate by adding features that clutter their products, making it hard for them to deliver a simple solution that meets a basic need. Open source creates an opportunity to target a common problem directly and save money that can be redirected toward training and creating better content. Open source can also enable more flexibility to manage the evolution of the application as requirements change.

Contrary to claims made by the commercial software industry, when used to solve targeted, common content management problems, open source software is not necessarily more expensive to customize and integrate than commercial software. In fact, all content management software requires at least some degree of customization. The idea that everything comes out of the box is a myth. Most industry analysts agree that you should expect to spend two to three times licensing costs in customization and integration. In the case of a web content management system, at the very least presentation templates need to be developed to support the desired layout and branding. For internal systems such as collaborative workspaces, configuration and customization effort tends to be less unless custom modules are needed. Open source content management systems tend to have a modular architecture to encourage communities to extend the core application with add-ons. This can make the addition of a new feature an administrative tax rather than a software development project.

Companies that tend to build custom applications to satisfy their unique needs are also leveraging open source frameworks and components. Doing so not only reduces licensing and development costs, but it also lessens reliance on a single proprietary software vendor. Heavily-customized proprietary software is difficult for software companies to support, makes compliance with forced upgrades expensive, and is vulnerable to deprecation of the underlying product.

## The Informational Brochure Web Site

By population, the largest category of sites on the Web is clearly the basic corporate informational or identity web site. These are the corporate brochure web sites that companies maintain for marketing purposes. While many of these web sites are static HTML, the price point for low-end CMS has lowered the hurdle to achieve the benefits of content management, in particular in the area of distributed authoring and workflow where non-technical users can update the web site without assistance. Many CMS can deliver this basic functionality and, through the addition of modules, more sophisticated functionality such as e-commerce, online forums, newsletters, and blogs.

The primary requirements of these systems are a flexible presentation system that can support the visual design and branding of the company, and ease of use. Because these informational web sites do not change rapidly and they are supported by a small group of users, features like workflows and versioning can be very simplistic. The size of the site and volume of information allow for a page-based design with minimal requirements for content reuse. The key is simplicity. If the system requires more than a couple of hours of training, the goal of distributed authoring is unlikely to be met. Instead, the burden of updating the web site will fall on the few who take the time to learn the new software – probably the same group of HTML developers that were the bottleneck in maintaining the previous static HTML site.

### Content Creation

Online brochures are best served by a page-based, rather than an article-based, or file-based content model. Non-technical users essentially just need to be able to create and edit pages. A WYSIWYG-editing environment and immediate preview functionality are critical. File management usually consists of uploading and linking PDF, images, and occasionally multimedia files. Content reuse and versioning are less critical than in sites with large editorial processes that pump out high volumes of content. Because of their largely European origin and adoption, internationalization of these solutions is particularly strong.

### Management

Typically these sites are managed by the marketing organization with a limited number of contributors. Workflow can be as simple as a two-step approval process. From time to time, the site may need to be reorganized and expanded so users need to be able to “place” pages within a site hierarchy and have the site navigation update automatically.

### Presentation

A corporate brochure site needs to be well designed and reflect the brand of the company. The supporting CMS cannot impose undue restrictions on the structure or visual design of the site. CMS in this category must have a flexible, easy to learn templating architecture. Other important requirements on the presentation side include search index-friendly URLs and an internal search engine to ensure that content is easy for the user to find. Site traffic reporting, support for email marketing, and standards-compliant HTML are also important in this category.

The open source CMS that have delivered particularly well in this category are OpenCMS, Magnolia, Mambo, TYPO3, and eZ publish. All of these applications have the capability of more sophisticated uses such as e-commerce and extremely dynamic informational web sites. However, most installations of these applications power simple structured web sites. The LAMP offerings (Mambo, TYPO3, and eZ publish) have a benefit of being easier to host on inexpensive hosting packages than Java or Python-based solutions.

### Projects Reviewed

	Version	Architecture	License	Support	Resources		
					Book	Online	Community
<b>eZ Publish</b>	3.7.2	LAMP	Commercial/ GPL	Commercial/Community			
<b>Mambo/ Joomla</b>	4.5.3/1. 0.5	LAMP	GPL	Consulting/Community			
<b>OpenCMS</b>	6.0.2	Java	LGPL	Commercial/Community			
<b>TYPO3</b>	3.8.1	LAMP	GPL	Consulting/Community			

**eZ publish 3.7.2**

Architecture	LAMP
Organization	Development controlled and supported by eZ systems
License	Dual Commercial/GPL
Sample Sites	<b>Gallery:</b> <a href="http://ez.no/customers/references">http://ez.no/customers/references</a> <b>See:</b> Johns Hopkins University Malaria Research Institute ( <a href="http://jhMRI.jhsph.edu/">http://jhMRI.jhsph.edu/</a> )
Resources	<ul style="list-style-type: none"> <li>● <b>Book:</b> <i>Learning eZ publish 3: Building Content Management Solutions</i></li> <li>● <b>Online Documentation:</b> <a href="http://ez.no/doc/">http://ez.no/doc/</a></li> <li>● <b>User Forums:</b> <a href="http://ez.no/community/forum">http://ez.no/community/forum</a></li> </ul>

eZ publish is a commercially-supported, dual license web content management system written in PHP by Norway-based eZ systems. eZ publish has many reference sites including ecommerce, educational, interactive media, and corporate web sites. Development of the core product is closely managed by eZ Systems with the community contributing extensions and patches that may later be incorporated into the core product. eZ publish does not have the module development community that Mambo, TYPO3, and Drupal have and most of the modules that are available are not open source. In addition to eZ systems, which licenses and supports the product, there is a large partner network that provides integration and training services. eZ systems is also having an impact on the general PHP community with their introduction of eZ components which competes with the Zend Framework as a standard framework for building PHP applications. If eZ components is successful, the population of developers that are familiar with the technology concepts behind eZ publish, such as the templating framework, will increase.

**Content Structure and Editing**

eZ publish natively supports several simple content types such as pages and various binary formats and as well as complex content types, such as a company (which could be used to support a feature like a partner directory) and a product (which could support pages for products). Content is edited in an XML syntax rather than plain HTML. For example, rather than using an <H1> HTML tag, eZ publish requires a user to use <heading level="1">. Because of this requirement, most publicly-available in-page WYSIWYG HTML editors do not work with eZ publish. However, eZ systems does sell a WYSIWYG editor called Online Editor, which produces compliant XHTML, for \$99 per web site.

**Management**

eZ publish comes with a very basic two-step workflow that can be extended by adding events that affect the state of the asset and also can trigger additional functionality. Every asset within an eZ publish site is managed by versioning and localization systems that allow different versions of an asset to be in different states of workflow simultaneously. Assets within an eZ publish site are easily placed in one or more places within the site hierarchy by associating them with locations.

**Presentation and Layout**

The administration interface allows certain aspects of the site layout (such as navigation style, color palette, etc.) to be configured. Significant layout customizations can be made within eZ publish's skin framework by writing templates in eZ publish's template scripting language and CSS. The scripting language consists of HTML with template tags that are identified with curly braces ({} ) that execute presentation logic. As specialized templating languages go, eZ publish is one of the easier ones to learn, mostly due to the excellent online documentation. URLs are clean and search engine friendly with no query string arguments. eZ publish supports URL "aliases" to make important pages more accessible. The layout and site design of multiple sites hosted on the same server can be totally different.

**Community and Support**

The best place to start with an eZ publish question is the forums, which are extremely active and have a searchable archive. The lists are moderated, so almost all questions get answered, although sometimes there is a delay between the question and the answer. There are several extremely active forum contributors that are credited with hundreds of posts. eZ systems also leverages blogs and RSS for community outreach.



## Mambo and Joomla 4.5.3/1.0.5

Architecture	LAMP
Organization	Mambo is lead by the Mambo Foundation; Joomla is a new community
License	GPL
Sample Sites	<p><b>Mambo Gallery:</b>  <a href="http://www.mamboserver.com/index.php?option=com_content&amp;task=view&amp;id=94&amp;Itemid=123">http://www.mamboserver.com/index.php?option=com_content&amp;task=view&amp;id=94&amp;Itemid=123</a>  <b>See:</b> <a href="http://www.desman.com">http://www.desman.com</a></p> <p><b>Joomla Gallery:</b>  <a href="http://www.joomla.org/content/blogcategory/35/69/">http://www.joomla.org/content/blogcategory/35/69/</a></p>
Resources	<ul style="list-style-type: none"> <li>● <b>Books:</b> <ul style="list-style-type: none"> <li>◆ The Definitive Guide to Mambo by Michell Pirtle</li> <li>◆ Building Websites with Mambo by Hagen Graf</li> </ul> </li> </ul>
	<ul style="list-style-type: none"> <li>● <b>Online Documentation:</b> <ul style="list-style-type: none"> <li>◆ <a href="http://help.mamboserver.com/">http://help.mamboserver.com/</a></li> <li>◆ <a href="http://help.joomla.org/">http://help.joomla.org/</a></li> </ul> </li> </ul>
	<ul style="list-style-type: none"> <li>● <b>User Forums:</b> <ul style="list-style-type: none"> <li>◆ <a href="http://forum.mamboserver.com/">http://forum.mamboserver.com/</a></li> <li>◆ <a href="http://forum.joomla.org/">http://forum.joomla.org/</a></li> </ul> </li> </ul>

Mambo is a GPL licensed Web Content Management System developed and distributed by Australia-based Miro International, a provider of software and services for web publishing and collaboration. In addition to providing consulting services around Mambo, Miro International also provides a hosted Mambo service, called Mambo Corporate Edition, which includes support and hosting in their data centers. Mambo's architecture is based on PHP and MySQL. Recently, Mambo has had some internal political turmoil, which resulted in most of the core developers forking the code base to create Joomla. Since the fork happened so recently, the applications are nearly identical, so this summary covers both.

Mambo's primary design goal is ease of use and they have accomplished this goal with a simple user interface and functionality targeted toward managing corporate web sites and customer extranets. Mambo is quick and easy to set up - especially if the customer is willing to use the presentation templates that come with the software. These templates are fairly design-neutral and have the potential to work nicely with different corporate branding elements.

### Content Production/Editing

Content creators can create static pages and content items using the TinyMCE in-page WYSIWYG HTML editor. The major difference between static pages and content items is that content items are slightly more structured with an introductory text field, which is useful for listing pages. Images are associated with assets through the user interface rather than HTML references. This helps Mambo maintain these relationships and prevent broken image references.

Users have a choice of editing content through the administrative interface or directly on the site. The general guideline is to reserve use of the administrative interface for trained power users to prevent inadvertent mis-configuration. In-site editing is sufficient to make textual changes and is foolproof but does not give access to metadata and advanced publishing features.

### Management

Assets within a Mambo site can be in one of two states: published or unpublished. The absence of content versioning makes changes to published pages immediately visible upon save. However, an author is able to preview from the edit page to verify changes before saving. Mambo has a simple locking mechanism to prevent users from trying to edit the same asset simultaneously. Assets are placed on a site by tagging them for the home page, associating them with menus that drive the site's navigation, or assigning metadata and other categories for dynamic listing pages.

### Presentation

Mambo has a skins framework (called templates) that manages customizations to the presentation templates and allows an administrator to toggle between skins. Installing a skin is easily done through the administrative interface. Templates are written in pure PHP, unlike the proprietary tagging languages that many other CMS use. This is an advantage of lessening the

amount of specialized skill required to customize and manage a Mambo site. The Mambo and Joomla communities have several websites that host downloadable skins and there is an active marketplace for custom skin development. Mambo has a flat URL structure and relies on query strings to request specific pages. There are some work-arounds available to create search-friendly URLs but they are not core to the product.

Mambo comes with an internal search engine that can search through page-based content on the site. Other visitor-facing functionality, such as blogs, a shopping cart, and a photo gallery can be added by downloading and installing modules (called mambots) through the management user interface

### **Community and Support**

While the fork between Joomla and Mambo fragmented the community, the original community was large and vibrant enough that both new communities are healthy. The Joomla project, in addition to taking a majority of core team members is also drawing a sizable portion of the user and third party developer community. Mambo has backfilled with a new core development team. There are several Mambo and Joomla related third party sites that discuss Mambo and innovative ways to use it including MamboForge and JoomlaForge which are like SourceForge for modules. Mambo Love<sup>v</sup> is a community driven site for public relations and advocacy. Both Mambo and Joomla have active forums and do a decent job of notification of security vulnerabilities. The Joomla user forum is excellent with over 1,000 posts per day and a well organized moderator community. Forum participants are generally very polite and helpful, a phenomenon that some attribute to the policy of using one's own picture in the forum profile which inserts a degree of personal accountability for what is said.

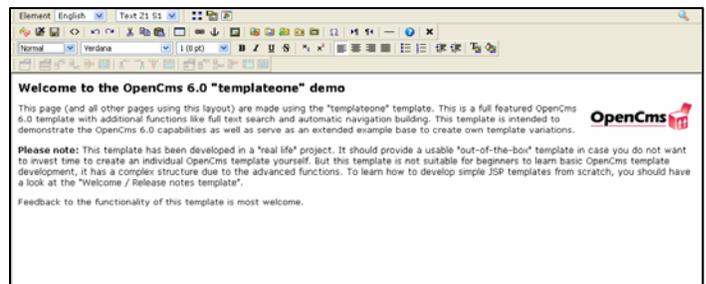
## OpenCMS 6.0.2

Architecture	Java
Organization	Community with leadership and support options from Alkacon Software
License	LGPL
Sample Sites	<b>Gallery:</b> <a href="http://www.opencms.org/opencms/en/support/references">http://www.opencms.org/opencms/en/support/references</a> <b>See:</b> Eastern European Renault sites
Resources	<ul style="list-style-type: none"> <li><input type="radio"/> <b>Book:</b> <i>Building Websites with OpenCMS</i> by Matt Butcher</li> <li><input type="radio"/> <b>Online Documentation:</b> <a href="http://www.opencms.org/opencms/en/download/documentation.html">http://www.opencms.org/opencms/en/download/documentation.html</a></li> <li><input checked="" type="radio"/> <b>User Forums:</b> <a href="http://www.opencms.org/opencms/en/development/maillinglist.html">http://www.opencms.org/opencms/en/development/maillinglist.html</a></li> </ul>

OpenCMS, the most mature of the Java-based open source CMS, was designed to help large teams of non-technical content contributors maintain a dynamic web site. By using a web-based GUI called the Workspace, users interact with a Virtual File System (VFS) that represents the pages and other files that make up a web site. OpenCMS handles issues of concurrency and versioning nicely with its "project" and locking models. OpenCMS is managed by Alkacon software, which sells service and support packages and has a network of certified consultancies that can help with integration.

### Content Creation/Editing

Prior to the most recent release, version 6.0, OpenCMS had one real content type: a page. Version 6.0 introduced an XML-based mechanism for creating more modular content types but OpenCMS is still very much page-based. Pages are unstructured and are edited using an integrated shareware ActiveX WYSIWYG authoring tool called LeEdit. This model is very intuitive to non-technical users, who will find their word processing skills very useful in navigating the MS Word-like formatting buttons. Pages are associated with templates, which can provide persistent elements (such as navigation, headers, or other dynamic components) and enforces uniformity between pages. OpenCMS has some very useful management features, such as link checkers that verify internal and external links within HTML are valid.



### Management

Within the OpenCMS management interface, "projects" serve the function of branches in a version control system. To edit a page, the page must be checked into a project. Publishing an asset is done by checking the page back into the main branch, a project called "Online." Within a project, pages must be locked before editing. This prevents data loss from multiple users from trying to edit the same asset. OpenCMS workflow is little more than a task assignment system. Tasks are created within projects and passed around until they are marked "complete." Tasks are not tied to individual content assets and there is very little data associated with them: only initiator, owner, date due, priority, and a collection of comments.

### Presentation and Layout

Each page is associated with a presentation template. Presentation templates can be written using JSP (and provided tag libraries), an XML based scripting language, or Velocity. The OpenCMS controller servlet pairs up the page with the associated display template and executes the presentation logic. In addition to using the tag libraries to retrieve content asset from the repository, the JSP programmer is free to call other libraries and perform any logic necessary.

### Community and Support

OpenCMS hosts a developer list with a searchable archive. The community is generally responsive in answering basic questions. However, users with large support needs are encouraged to purchase professional support from Alkacon software. Documentation for the latest version 6.0 is not as good as version 5.0, which has several language translations and a professionally-written book. The German documentation set is the most thorough of the Version 6.0 translations.

**TYPO3 3.8.1**

<b>Architecture</b>	LAMP
<b>Organization</b>	Community with leadership from the TYPO3 Association
<b>License</b>	GPL
<b>Sample Sites</b>	<b>Gallery:</b> <a href="http://www.typo3.com/References.1249.0.html">http://www.typo3.com/References.1249.0.html</a> <b>See:</b> Dana Farber / Harvard Cancer Center ( <a href="http://www.dfhcc.harvard.edu">http://www.dfhcc.harvard.edu</a> )
<b>Resources</b>	 <b>Book:</b> TYPO3: <i>Enterprise Content Management</i> by Rene Fritz et al.  <b>Online Documentation:</b> <a href="http://typo3.org/documentation/">http://typo3.org/documentation/</a>  <b>User Forums:</b> <a href="http://typo3.org/documentation/mailling-lists">http://typo3.org/documentation/mailling-lists</a>

TYPO3 is a solid framework for rapidly deploying intranet and external web sites. The core download contains necessary functionality for small teams to manage structured web pages and file-based content. TYPO3's modular architecture enables rapid extensibility through downloadable and custom modules that can be deployed through the complex yet powerful administration GUI. The TYPO3 web site has an extensive library of modules that enable functionality-like form processing and mailing lists. There is an active development community that is managed by the TYPO3 foundation, which is lead by project founder Kasper Skårhøj. TYPO3 is widely used to support commercial web sites, particularly in Germany and other Northern European countries where large corporations use TYPO3 for sub-sites such as product and country affiliate sites, and medium companies for their primary corporate web site.

**Content Structure and Editing**

Pages within a TYPO3 site consist of sub-elements that can contain straight text or combinations of text, tables, and images. This design increases the degree of page content structure (creating more options for reuse) but breaks the word processor paradigm of a single WYSIWYG-enabled form field that non-technical users find so intuitive and efficient. The text elements translate into a paragraph with a heading (which can be left blank) and a body. The elements with images allow the user to select an image to be displayed along with the element. There is also a free-form HTML element for greater user control. The TYPO3 core does not come with a WYSIWYG editor, although there are several downloadable modules that provide this functionality by replacing the text area field using HTMLArea, an open source component that uses JavaScript.

Once a page has been constructed, it can be edited through the administration interface and also within the actual page. A user with an active session and sufficient privileges will see pencil icons near dynamic text indicating the ability to edit the page fragment. This in-site editing model is often popular with business users because they can edit text as they see it on the site. The downside is that once the change is made, it is live. There is no staging area for others to review.

**Management**

The workflow that comes with TYPO3 is very basic. In its current incarnation, a workflow consists of a task associated with a content asset. When creating a new workflow definition, the administrator selects a group of users that can be assigned the task, and another group of users that can finalize the workflow and publish the asset. When assigned a task, a user can add notes to the task, set the priority, and assign the task to another eligible user. Users that are part of the group allowed to finalize the workflow have the option to publish the asset. When an asset is in workflow, it is marked as hidden and put into a working folder so it is not visible on the publicly-facing site. Finalizing the workflow flag removes the hidden flag and moves the asset to a predefined place within the site hierarchy.

**Presentation and Layout**

Templates are applied hierarchically to content within a site. A template selected for a page will be applied to all the children pages that do not have templates directly assigned. Templates can also inherit attributes from their parents. Creating templates is a well-documented process of inter-mixing blocks of static HTML with a declarative language called TypoScript and CSS. TypoScript looks like a properties file that specifies the page object and objects within the page. TYPO3 comes with an auto-parser to expedite the conversion of static HTML mockups into templates referenced in TypoScript. TypoScript can take some time to learn. Because it is a non-procedural language, TypoScript can be difficult to debug, especially when the parser encounters errors.

Dynamic plugins, which are written in PHP, are the way to build dynamic functionality into pages and provide a familiar development platform for a PHP programmer. Features like site search and password-protected pages come out of the box and many more are available for free download.

All pages are rendered through TYPO3's controller page, which retrieves a page object based on a query string parameter and applies the appropriate template. Because of this design, every page on a TYPO3 site has essentially the same URL. This might prove problematic for search crawlers. However, there is an extension called RealURL that uses Apache mod\_rewrite to change TYPO3 query string based URLs to search engine friendly URLs.

**Community and Support**

The TYPO3 project is a particularly well organized and active project. Development is managed under the TYPO3 foundation which consists of Active Members, who regularly contribute code, and Supporting Members (either companies or individuals), who pay membership fees. Active Members have the right to vote on matters. Supporting membership seems to be more of a vehicle for sponsorship. The development of TYPO3 is divided into two categories: core and extensions. Management of TYPO3 is divided into teams for Core Functionality, Extension Review, Innovation, Security, Bug Filtering, GUI and XHTML, Documentation, Community Communication, and Marketing. The core team, led by project founder Kasper Skaarhoj, is currently working on seven active projects each with its own project lead. Core bugs are managed in a Mantis-based bug tracking system that anyone can register to and submit bugs and new ideas. Extensions, of which there are many, are rated for their maturity and quality although not all are rated.

Community-based support is delivered mainly through a set of mailing lists which are archived on the TYPO3.org site. The archive interface has the appearance of a forum with sortable columns and metrics for the number of views and replies. Responses to questions are generally prompt and helpful. Many of the 740 documents are written in English although the main documents are also available in French, German and Danish.

## Online Periodical

The Internet bubble, with its explosion of content-based companies combined with the immaturity of the CMS market, led to the development of many new CMS solutions. Some of these solutions turned into commercial products, such as C|Net's homegrown CMS becoming Vignette Story Server; others were donated to the open source community. Examples of these news-oriented sites include the online version of a print newspaper or magazine, an e-zine, and a news aggregator web site.

The key problems that news publishing systems were designed to solve are high volumes of new content, editorial control, multiple channel publishing (Web, email, syndication, wireless), incoming news feeds, and content aggregation. These applications also must support high content volumes and heavy site traffic. Separation of content and layout is critical because the volume of content makes manual layout of each page prohibitively expensive and content needs to be reused within the site and into different channels. For example, the title and summary of an article must appear on multiple pages throughout the site: on the front page, on a topical section of the site, and then in the archives of the site.

CMS that are designed to support news-based sites include Bricolage, which was originally built for Salon magazine; Lenya, which was developed for NZZ Online; and Midgard. Each of these products supports a concept of structured articles that can be placed in categories. Articles are distinct from pages in that they are temporal – they have a lifecycle, whereas pages are intended to be permanent. Structure allows display templates to do things like break the article into multiple pages, intersperse article text with advertisements and other content, and display lists of articles along with their summaries. Structured content also support different publishing formats such as RSS. Categorization enables presentation templates to list related articles dynamically.

Because users of these CMS specialize in creating and publishing content (as opposed to other business contexts where maintaining the web site is a side job), a steeper learning curve is acceptable as long as it comes with more powerful functionality. Characteristics of an appropriate CMS are as follows.

### Content Creation and Editing

The content model should be article-based with the capacity to create, store and retrieve thousands of articles. Content should be structured and associated with appropriate metadata to support features like personalization and related articles on the display side. While WYSIWYG editor and preview are important, authors should not get overly consumed about the text/image layout for each article. In true newsrooms, reporters do not even have control over the titles of their articles, let alone how they appear on a page. Frequently, the roles of writers and graphic producers are separate and their output needs to converge within an article by the time it is published. Editors will need a way to escalate the importance of an article and affect what appears on the home page and section landing pages, but much of the content will be dynamically placed using presentation side business logic.

### Management

Workflow must support an assembly-line type of environment optimized for efficiency and throughput. The editorial process can involve several roles including copyediting, legal review, and editorial. Often, workflow is used to deliver writing assignments to writers. Article-based content has a lifecycle that ends when it is moved into an archive area or completely retired from the site. Wherever possible, content, including images and text, should be reusable.

News sites also frequently process feeds from external sources. This content must be imported into the CMS, processed (such as assigning metadata or placing the article within a taxonomy), and then published.

### Presentation

If they are successful, news sites will generate a lot of visitor traffic. Consequently, performance on the presentation side is critical. The CMS should have some sort of caching strategy that optimizes repeated accesses of the same content. Web site analytics and integration with banner ad systems are also critical business requirements. News sites also must publish to different formats for syndication (RSS and NITF) frequently and provide support for other devices (WML).

### Projects Reviewed

	Version	Architecture	License	Support	Resources		
					Book	Online	Community
<b>Bricolage</b>	1.9.1	Perl/Postgres	BSD Style	Consulting /Community	●	○	●
<b>Lenya</b>	1.2.4	Java	Apache	Community		○	●
<b>Midgard</b>	1.7.3	LAMP	GPL	Community/Consulting		○	●
<b>Zope/CMF</b>	2.8.5	Python	ZPL (Apache-style)	Commercial/Community	●	●	●

**Bricolage 1.9.1**

Architecture	Perl/Postgres
Organization	Community
License	BSD-Style
Sample Sites	<b>Gallery:</b> <a href="http://www.bricolage.cc/about/sites/">http://www.bricolage.cc/about/sites/</a> <b>See:</b> Macworld ( <a href="http://www.macworld.com">http://www.macworld.com</a> ), World Health Organization ( <a href="http://www.who.int/en/">http://www.who.int/en/</a> )
Resources	<ul style="list-style-type: none"> <li> <b>Book:</b> <i>The Mason Book</i>, by Dave Rolsky and Ken Williams (O'Reilly 2002) - has an Appendix which discusses installing and using Bricolage</li> <li> <b>Online Documentation:</b> <a href="http://www.bricolage.cc/docs/">http://www.bricolage.cc/docs/</a></li> <li> <b>Article:</b> Content Management with Bricolage, by David Wheeler. Perl.com, August 27, 2004</li> <li> <b>Mailing Lists:</b> <a href="http://www.bricolage.cc/support/lists/">http://www.bricolage.cc/support/lists/</a></li> </ul>

Bricolage is a community-driven, commercially-supported web content management system based on Perl and PostgreSQL. Bricolage was started in 2000 to support the Salon magazine web site and is focused on web publishing with strong functionality in workflow and the ability to support high traffic web sites such as Macworld. Similar to InterWoven TeamSite, Bricolage's publishing templates build static HTML files to be deployed to a web server. This is great for high traffic web sites because there is no need for synchronous dynamic page assembly. However, sites wishing to have transactional or personalization capabilities interspersed with managed content may need to do some additional integration, such as publishing to XML and then using some other dynamic templating mechanism on the presentation tier.

**Content Creation and Editing**

All content within Bricolage is either a "Story," representing structured information, or a "Media" document, representing a binary attachment. A Story is a flexible construct consisting of elements such as pages, paragraphs, insets, and user-defined structures. Each element can have sub-elements and custom defined fields, which can be a text field, text area, radio button, check box, pulldown, select box, or date. As of version 1.8, WYSIWYG HTML editing can be achieved by downloading htmlArea (a third party BSD-licensed WYSIWYG editor written in Javascript with new development being done under the project Xinha) and enabling some lines in Bricolage's configuration file. Using this framework, an administrator can define any content type through the UI. Advanced users of Bricolage typically use the "Super Bulk Edit" mode, which enables the editing of complex data structures in a single text field by using special syntax to delineate the sub-elements.

**Management**

Workflow follows a metaphor of "desks." Each desk represents a step in the workflow, or a state of a document. On the participant side, a desk also represents a role in the editorial process. Participants in the process monitor a desk, check out items for review, and then promote the items to the next desk in the workflow. This model has several advantages: by routing through "desks" rather than individuals, the system can more easily respond to bottlenecks created by increased volume or the absence of individuals; individuals can easily be substituted in the process; and it is easy to see lists of documents that are in a particular state. Much of workflow can be configured through the UI by administrators. There is also a rich API (in Perl and SOAP) for workflow integration.

**Presentation**

Presentation templates are written using Perl's powerful HTML::Mason library within the Bricolage's browser-based UI. Presentation templates are associated with content element types and categories and are workflowed like content assets. There is also support for Template Toolkit and HTML::Template templating. During a preview and when the site is published, Bricolage starts a "burning" process where articles are coupled with presentation templates to create static files in the mark-up language of your choosing. Of course, the standard is HTML, but it is just as easy to write to XML or WML formats. These files are then served up by any web server. During the burning process, it is also possible to write to a relational database using Perl DBI. By publishing to XML or a relational data store, Bricolage-managed content could be used in dynamically-rendered site to support features like personalization. Because Bricolage is not installed on the display web server, visitor-facing functionality such as site search must be provided externally. Any search engine capable of indexing a static HTML site would work well here. Apache Lucene, ht://Dig, and Swish-E are popular open source solutions. Google site search works well too.

**Community and Support**

While the API is fairly well documented, Bricolage user documentation is somewhat lacking. Fortunately, this is compensated by one of the friendliest and responsive mail lists in open source. The community leadership group is small, but dedicated and is generous with answers. There is a growing number of external participants on the mail lists which is lessening the demand for the project committers. In looking at the Bricolage community, it is also important to look at the Perl Mason community because Mason is such an integral part of implementing a Bricolage system.

## Lenya 1.2.4

Architecture	Java
Organization	Community
License	Apache
Sample Sites	<b>Gallery:</b> <a href="http://lenya.apache.org/community/live-sites.html">http://lenya.apache.org/community/live-sites.html</a> <b>See:</b> Neue Zürcher Zeitung ( <a href="http://www.nzz.ch">http://www.nzz.ch</a> )
Resources	 <b>Online Documentation:</b> Lenya Wiki, <a href="http://wiki.apache.org/lenya/">http://wiki.apache.org/lenya/</a>  <b>User Forums:</b> <a href="http://lenya.apache.org/community/mailling-lists.html">http://lenya.apache.org/community/mailling-lists.html</a>

Lenya is a Java/XML-based open source CMS that is licensed by the Apache Software License and has functionality targeted for web content management including workflow, separation of content and layout, and access control. In the words of project founder Michael Wechner, "The idea of Lenya is to offer a CM Framework by enhancing Cocoon, which incubated Lenya prior to release 1.2 in September 2004 by CM components, but also a CMS 'nearly out of the box.'" The Lenya project was once hosted by the Swiss systems integrator Wyona. After having difficulty getting people to contribute to the project because of its close ties to a single company, Wyona donated Lenya to the Apache Software Foundation. This achieved the goal of making Lenya more neutral and increased overall interest in the project. The Lenya project is now a top-level Apache project and it is making great strides towards its original vision. The largest installation of Lenya is the online version of one of the largest German speaking newspapers in the world: Swiss-based Neue Zürcher Zeitung (NZZ Online). In addition to NZZ Online, several universities and a few software companies have adopted Lenya to manage their web sites.

While the user interface is a little less polished and sophisticated than some of the other open source CMS, Lenya's architecture makes it a very good starting place for a custom solution designed to meet a unique set of business requirements.

### Content Creation and Editing

In its out-of-the-box state, Lenya uses the in-site editing model where editors and readers have different views of the same site. The editor/author view has an administration menuing structure at the top of each page, where the user can edit pages, process workflow, manage users, and perform other administrative functions. Lenya ships with two WYSIWYG editors: Kupu and BXE. BXE works only with Mozilla-based browsers. BXE appears to be a more powerful tool and is integrated with Lenya's link management functionality.

Lenya's link management functionality allows the application to be aware of links between content managed pages and to notify the user if publishing or removing a document will result in broken links on the site. Lenya's asset management system allows assets (such as images) to be associated with a page so when the page is published, associated content is also deployed.

### Management

Lenya contains a customizable workflow engine that uses an easy XML format to define workflows that consist of collections of states and transitions. Standard one- and two-stage workflows ship with Lenya. Users can be notified by email about pending approvals and all workflow activity is logged and auditable. Workflow events such as publishing or deactivating a page can be scheduled.

The Lenya presentation system maintains separate areas for staging and live so a single server can be used to run the entire application if publishing and viewing loads are modest.

### Presentation

Lenya's presentation system is built on the powerful Java/XML framework Cocoon and CSS. Cocoon blocks are also available to the presentation template developer for dynamic functionality, such as form handling and personalization. Cocoon has a powerful new framework called Flowscript, which supports complex transactive functionality such as multiple page forms. For search, Lenya integrates well with the popular Apache Lucene search engine.

### Community and Support

Over the last couple of months, activity on the Lenya project seems to have slowed. There have been occasional announcements of new Lenya websites deployed but not a significant amount of new development. However, the community around Cocoon, the technology on which Lenya is based, is going very strong. The next major release of Lenya 1.4, which will persist content to a Java Content Repository, is still in early Alpha and seems to be stalled.

**Midgard 1.7.3**

Architecture	LAMP
Organization	Community
License	LGPL
Sample Sites	<b>Gallery (Case Studies):</b> <a href="http://www.midgard-project.org/midgard/1.6/casestudies/">http://www.midgard-project.org/midgard/1.6/casestudies/</a>
	<b>Also see:</b> <a href="http://www.playbill.com">http://www.playbill.com</a> , <a href="http://www.govt.nz/">http://www.govt.nz/</a> , <a href="http://www.cmswatch.com">http://www.cmswatch.com</a>
Resources	○ <b>Online Documentation:</b> <a href="http://www.midgard-project.org/documentation/">http://www.midgard-project.org/documentation/</a>
	● <b>Mailing Lists:</b> <a href="http://www.midgard-project.org/development/projects/aegir/support_discussion.html">http://www.midgard-project.org/development/projects/aegir/support_discussion.html</a>

Midgard is a collection of projects that, in aggregate, compose Midgard CMS. At the core is the Midgard and MidCOM's framework, which handle low-level data access and supports the addition of modules. Consequently, Midgard has a modular architecture where components can be swapped in and out. For example, there are three administrative interfaces for Midgard: Aegir, Asgard and Spider. This enables flexibility in configuring, managing, and extending a Midgard-based system. Midgard's primary strength is as a framework. The modular architecture and extensibility make it an efficient and effective platform for PHP developers to customize and deploy web-based content applications. The Midgard architecture is also being developed to integrate with other technologies such as the Java Content Repository (JCR JSR Spec 170). However, unless customized, the Midgard user interface is frequently considered not intuitive to non-technical users. The development and usage of Midgard is primarily in Northern and Central Europe although the web site and most of the community dialog is written in English. Over the last couple of years, the Midgard project has been very active due to a few European consultancies that actively drive development as they implement Midgard-based solutions for their clients.

**Content Creation and Editing**

Midgard's primary content type is an article, but there is also support for pages (either static, which is just content or active, which contain PHP code). By default, an article can only belong to one topic but that can be changed, although not easily. Midgard uses htmlArea as a WYSIWYG editor. Development of htmlArea has been discontinued and resumed under another project called Xinha. Now, the increasingly popular TinyMCE editor is available.

**Management**

Articles are organized within a hierarchical topic tree. Workflow is very basic and consists of a simple approval. Articles can be scheduled for publication. Midgard supports versioning and is able to lock records to handle concurrency between content authors. The access control model supports groups but does not have a concept of role, which makes it easy to assign a specific set of privileges to a group.

**Presentation**

Presentation templates in Midgard, called "styles," mainly consist of PHP code and a few custom tags to access variables. In many ways, this system is advantageous because PHP programmers can be immediately productive on the system without having to learn yet another custom tagging language. The Midgard presentation engine is installed as an Apache module and achieves very high levels of performance. Templates are modular, consisting of sub-elements, and are applied hierarchically to the content tree.

**Community and Support**

Midgard is lead by Henri Bergius, one of the project's originators. Henri is also active in OSCOM (the Open Source Content Management organization) and is highly visible promoting and explaining Midgard in his blog and at various open source conferences. There are several companies across Europe that sell services around implementing, hosting and supporting Midgard-based solutions. The coverage of the documentation is relatively thin with a couple good articles with screen shots and a majority of topics containing only a paragraph. Documentation is an area where the project is actively looking for contributions. The mailing list, however, is more helpful.

**Zope CMF 2.8.5**

<b>Architecture</b>	Zope/Python
<b>Organization</b>	Community lead by Zope Foundation; commercial support from Zope Corporation and others
<b>License</b>	Zope Public License (ZPL)
<b>Sample Sites</b>	<p><b>Gallery:</b> <a href="http://www.zope.org/Resources/ZopePowered/">http://www.zope.org/Resources/ZopePowered/</a></p> <p><b>Also see:</b></p> <ul style="list-style-type: none"> <li>◆ <a href="http://www.boston.com">http://www.boston.com</a></li> <li>◆ <a href="http://www.signonsandiego.com">http://www.signonsandiego.com</a></li> </ul>
<b>Resources</b>	<ul style="list-style-type: none"> <li>● <b>Books:</b> <ul style="list-style-type: none"> <li>◆ The Zope Bible, by Michael Bernstein and Scott Robertson</li> <li>◆ The Zope Book, by Amos Latteier and Michel Pelletier</li> <li>◆ Dive into Python, by Mark Pilgrim</li> </ul> </li> <li>● <b>Online Documentation:</b> <ul style="list-style-type: none"> <li>◆ Zope Book Online Edition</li> <li>◆ Blogs and web sites</li> </ul> </li> <li>● <b>Mailing Lists:</b> <a href="http://www.zope.org/Resources/MailingLists">http://www.zope.org/Resources/MailingLists</a></li> </ul>

Zope is a web application server and web application framework for the Python programming language. With roots tracing back to managing a newspaper classified ads application, Zope has a rich history in the news business and has several characteristics that make it conducive as a foundation for building custom publishing applications. Its internal object database (ZODB), built-in security, and object orientation lead to rapid development of sophisticated applications. CMF, or Content Management Framework, is a set of extensions that support basic content management functionality such as the definition of content types, workflow, and content indexing.

While there are more “out of the box” ready extensions available such as Infrae Silva and Plone (described later), Zope CMF is currently being used to manage sites such as Boston.com (the online presence of the Boston Globe and primary informational web site in the Northeast) and SignOnSanDiego (the online presence of the San Diego Union-Tribune) as well as many other media sites. Zope is available as GPL-licensed open source software, but Zope Corporate also sells commercial licenses of the software that have been specialized for specific industries.

Like a J2EE application server, Zope's Product framework makes it easy to install modules into the system. This coupled with an active community of module developers, makes it easy and inexpensive to deploy new functionality to a Zope site.

**Content Creation/Editing**

At the very core of Zope is a folder structure where content objects can be placed. Folders and other objects can have properties that are “acquired” by sub-folders and contained objects. This behavior is helpful in maintaining metadata because individual content items can be automatically tagged just by putting them in the appropriate folders. Zope integrates with several WYSIWYG editors but Zope's Epox editor and Kupu are among the most frequently used. Zope has a framework called Archetypes that allows developers to create efficiently complex content types, as well as interfaces for managing them, with just a few lines of Python code. Zope also has native support for WebDAV and FTP and this feature is frequently used to facilitate feeding content into the system or uploading images.

**Management**

Workflows in Zope are composed of states and transitions that can be defined in the ZMI with minimal programming skills. The state/transition model is very powerful and flexible and is fairly common in CMS applications.

States determine the visibility of assets and what actions can be taken on those assets. Transitions control the shift of assets between states and provide events to trigger scripts (such as emailing, flushing cache, etc.). There are several very good tutorials on configuring and extending workflow. Task lists and notifications are not a core part of the product, but there are tutorials on how to set them up.

**Presentation**

Zope's templating system is called Zope Page Templates (ZPT) which uses a language called TAL (Tag Attribute Language) and METAL (Macro Enhanced Tag Attribute Language). CMF introduces a "skins" framework to Zope, which allows content types to be associated with page templates for display. Zope's indexing system (ZCatalog) is closely integrated into the application and provides search services for both internally-facing sites and the management interfaces. Scaling a Zope-based application to support a high traffic environment is usually done through various load balancing and caching strategies. In extreme load conditions, Zope based sites are published to a static HTML site to be served by basic web servers.

**Community and Support**

The Zope and Python developer communities are extremely active because Zope is such a popular platform on which to build applications. In addition to having elegant mechanism-building applications, Zope's license, the ZPL, is extremely liberal and allows for proprietary applications to use and distribute Zope. The movement from Zope 2 to Zope 3 has fragmented the community to some degree because most products built on Zope are not easily migrated to the new Zope 3; but Zope 3 is where all the new and interesting core development is. Still, the Zope community is so large that there are 24 individual mailing lists that specialize in different types of topics. The most popular list is [zope@zope.org](mailto:zope@zope.org) which is for general users. Zope Corporation sells support packages. The mailing lists are active but not moderated, so some questions do go unanswered. Documentation on the Zope site is very good and there are a lot of other sites and blogs that have useful information for customizing and extending the product. The most useful book on Zope is the out-of-print *Zope Bible*.

## Collaborative Workspace

Another innovation of the Internet economy was a dynamic collaborative work style that required a central repository for teams to share documents and information. While email and file servers are the most popular tools to solve this problem, web-based collaboration systems are highly effective and excel in areas where traditional tools fail. Collaborative workspaces make large volumes of content manageable by recording metadata that makes it easier to browse and search the repository. A browser-based interface facilitates collaboration across the corporate firewall to support distributed teams. A centralized repository makes the authoritative version of a document clear. Versioning and audit trail help track the lifecycle of a document. Access control enforces rules of what information teams can share and what information is private. Teams also need to coordinate schedules, document discussions, and make announcements.

Take those workspaces in aggregate and they magically become a knowledge base, or so goes the theory. In reality, to make project-working deliverables into reusable and reference-able, manual work needs to be done to identify good examples, clean them up, and promote them within the community. There also needs to be a culture of contribution where individuals are recognized for creating resources that are of value the company.

Collaborative workspaces can take very different forms. Because most project teams deal in documents, a system that focuses on file management capabilities is often needed. In this case, a system like Plone, which is particularly good with uploading, modifying and updating files, is very useful. But if the project team is collaborating around simple text, the possibilities expand. For example, many successful collaborative efforts have used wikis such as Wikipedia or Tsunami Help, and documentation teams for countless open source projects.

### Content Creation and Editing

Collaborative workspaces should make it as easy as possible to add and update files. If it is easier and quicker to use email rather than the collaborative workspace, the collaborative workspace will fail. If the content is file based, it is helpful to enable users to save their content within their client application (as in Microsoft Word) directly onto the server. This solves the problem of a person's local copy being more up to date than the copy on the server. That said, if the content is not stored in such a way that it can be easily found (in the proper place and with good metadata), the repository will deteriorate rather than improve over time and the workspace will never live up to the goal of being useful as a knowledge base. If content is primarily file based and unstructured, it is unrealistic to expect any reuse other than cut, paste, and save-as.

### Management

Historically, these systems have tended to be either full of outdated material (the nickname of one system was "haystack") or totally empty. Managing the system so that the repository improves, rather than degrades, over time is difficult and requires commitment from the organization. In general, the most successful strategy is to make things easy to add but have good workflow that allows people to promote valuable assets to accessible areas. Security can also be a key issue, especially when project teams cross corporate boundaries. For example, a consultancy that hosts project workspaces for integrated client/consultant project teams needs to be careful that one client is not able to see another client's project files.

### Presentation

The single most important feature for collaborative workspace is search. Users usually prefer to search rather than browse for content. The search engine must be able to index all the formats managed on the workspace and have reasonable response times.

### Projects Reviewed

	Version	Architecture	License	Support	Resources		
					Book	Online	Community
<b>Alfresco</b>	1.1.2	Java	Commercial/ MPL	Consulting/Community			
<b>Plone</b>	2.1.1	Python	GPL	Consulting/Community			
<b>MediaWiki</b>	1.5.4	LAMP	GPL	Community			
<b>Twiki</b>	20040904	Perl	GPL	Consulting/Community			

**Alfresco 1.1.2**

<b>Architecture</b>	Java
<b>Organization</b>	Commercial
<b>License</b>	Mozilla Public License and "Shared Source" components
<b>Sample Sites</b>	No externally facing sites available
<b>Resources</b>	<p> <b>Online Documentation:</b> <a href="http://www.alfresco.org/mediawiki/index.php/Main_Page">http://www.alfresco.org/mediawiki/index.php/Main_Page</a></p> <p> <b>Forum:</b> <a href="http://www.alfresco.org/forums/">http://www.alfresco.org/forums/</a></p>

Although Alfresco is a newcomer to the open source CMS market, the progress that the project has made and the history of the team makes Alfresco worth mentioning. In June 2005, VC-backed Alfresco Software announced the first public release of what they called was the first open source Enterprise Content Management System (ECM). Adding intrigue to the story was the fact that most of the development team came directly from Documentum, including Documentum co-founder John Newton. The claim of being the first open source ECM is debatable. There are other more established open source projects that are designed to handle web and file-based content. Furthermore, not all of Alfresco is open source – several of the features that enterprises would need, such as group based security and clustering, are shared source and require a monthly subscription fee to use. Nevertheless, the amount of stability and functionality that the Alfresco team has achieved in such a short time is impressive. Currently, Alfresco's core strength is in document management with only trace support for web content management. WCM functionality is promised for later releases as well as support for Wikis and Blogs.

**Content Creation and Editing**

As a document management system, Alfresco is easy to use due to the multiple ways that files can be written and read from the system. The most intuitive interface is Microsoft's Common Internet File System (CIFS) protocol, which allows a Windows user to mount the Alfresco repository as a normal network file share. Through CIFS, users can open, edit, and move files without even knowing that they are using a document management system. Behind the scenes, Alfresco handles locking, versioning, and metadata extraction and can execute rules such as sending emails or adding categories. The more open WebDAV protocol is also supported. Alfresco does not yet support page-based content and does not have the extensive library of add-on collaboration modules that more mature open source CMS have.

**Management**

All advanced functionality and administrative control is done through the Alfresco web client. Here, users with sufficient privileges can define content rules and manage security in folders. The web client also displays metadata and versioning information about content. Alfresco supports the concept of "Aspects." The general idea is that an "Aspect" is a general set of attributes or capabilities that can be assigned to an object without relying on inheritance through the class hierarchy. In Alfresco, there are "aspects" like "versionable" or "categorized." These concepts allow content types to be very simple and, if they desire, users can add attributes to a single instance of a content asset.

**Presentation**

Alfresco's search is powered by the open source search engine Lucene and Open Office, which is able to extract text from many file formats and make them available to the Lucene search engine. Support for Microsoft Office and PDF file formats is the strongest. Alfresco can be extended to handle other file formats. Folders, or "spaces" in Alfresco terminology, can have different views or "dashboards," which are defined by creating templates using the FreeMarker templating syntax. It is likely that the FreeMarker templating engine will be a central part of the WCM support that is planned for future releases.

**Community and Support**

Alfresco is a closed project with development of the core product solely originating from Alfresco Software employees. The Alfresco welcomes outside contributions of bug fixes and add-on modules, but it remains to be seen whether a development community will form around the project to make these contributions significant. Documentation is another area where an active user and development community would benefit.

**Plone 2.1.1**

Architecture	Zope/Python
Organization	<ul style="list-style-type: none"> <li>◆ Community with leadership from the Plone Foundation</li> <li>◆ Commercial support options from Enfold Systems and others</li> </ul>
License	GPL
Sample Sites	<b>Gallery:</b> <a href="http://plone.org/about/sites">http://plone.org/about/sites</a>
Resources	<ul style="list-style-type: none"> <li>● <b>Books:</b> <ul style="list-style-type: none"> <li>◆ <i>The Definitive Guide to Plone</i> by Andy McKay</li> <li>◆ <i>Plone Live</i> by Michel Pelletier and Munwar Shariff</li> </ul> </li> <li>● <b>Online Documentation:</b> <ul style="list-style-type: none"> <li>◆ <a href="http://plone.org/documentation">http://plone.org/documentation</a></li> <li>◆ <a href="http://api.plone.org/">http://api.plone.org/</a></li> </ul> </li> <li>● <b>Mailing List:</b> <a href="http://plone.org/contact">http://plone.org/contact</a></li> </ul>

The default Plone site is immediately usable if you want a collaborative workspace a' la Microsoft SharePoint, eRoom, or a basic informational web site and is frequently used as a workgroup collaboration portal. Registered users can upload articles, files, calendar events, and web pages into personal and group folders that are searchable using Zope's indexing and searching functionality. There is also commenting functionality that enables threaded discussion around any content asset. Users are able to grant or deny other users from accessing their folders and files by using Zope's powerful access control system, which is simplified by the Plone user interface. Zope's acquisition functionality makes access control policies cascade down into subdirectories unless specifically overridden, similar to Windows folder sharing. While Plone can be used as a foundation to build any web site, a collaborative workspace is the most common and straightforward use of the software.

**Content Creation/Editing**

There are several ways to add/update file based content on a Plone site. The default and most commonly-used method is by using the upload functionality of the web interface. This works well for one-at-a-time uploads. Plone also supports content addition through WebDAV and FTP, through which Plone looks like a simple directory structure of a file share. Of course, adding content in this way causes metadata to be left blank and this can be a problem. One very useful tool supported by Plone is called ExternalEditor. ExternalEditor is a client side Python script that gets launched by your browser when you click on the pencil link next to an asset link of an ExternalEditor-enabled Plone site. ExternalEditor stores the file in a temporary local directory, and launches the client program of your choice (such as Microsoft Word for a .doc file). ExternalEditor then monitors the file and, whenever it is updated, sends a copy up to the server. As long as your local client program is open, the file on the server is displayed with a lock icon, which prevents version conflicts.

In addition to file-based content, Plone allows authors to write web pages and add news items and other content types (easily extensible through Zope's archetype framework). There are additional modules for functionality like blogs and a wiki.

**Management**

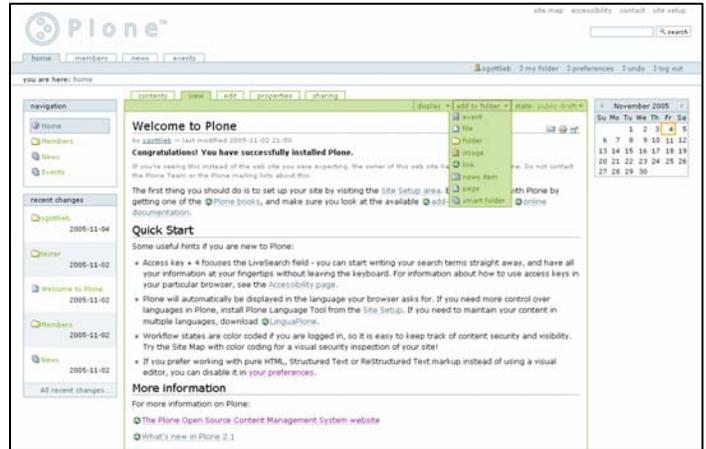
Plone has a powerful security model. Each user has his own folders and folders can be created for groups. Users are able to allow other users to share files and sub-folders within their user and group folders. Plone's workflow is based on Zope's DCWorkflow and can be used to submit assets for public view and can be part of a process to share reusable assets across teams. While Plone does not have native support for versioning of content, this feature is available through modules such as PVSF or Plone2.

**Presentation**

Plone comes enabled to index all of the Microsoft file formats for search. Additional formats, such as PDF, are index-able by adding some extensions to the core applications. There is a feature called "Smart Folder," which is like a dynamic saved search. Most UI configuration (such as branding the site with images, palette, and font, and configuring workflow) can be done through the administration interface. Deeper modifications require familiarity with the Python programming language, Zope's Page Template language TAL (Tag Attribute Language), and CSS.

## Community and Support

The Plone community represents a large and active portion of the overall Zope community. From time to time, the Plone community holds conferences in the United States and in Europe. Several companies also sponsor Sprints where developers get together and work intensively on a specific set of features. Recently, the Plone community has received an infusion of investment through the Goldegg project which is designed to refine Plone's architecture and prepare it for Zope 3. The books available for Plone are well written and informative. The mail lists are responsive and are archived on GMANE. General Google searches are also useful in getting answers.



## Wiki as Collaborative Workspace

While wikis are not generally considered true content management systems, they do solve many content management problems. A wiki can be extremely effective in facilitating the collection of information into a basic web site. They excel at creating online documentation and are frequently used by open source projects for that purpose. Wikis tend to grow organically and require less training and planning than do more structured content management systems. A successful wiki requires what has been called "wiki culture".<sup>v</sup> Of course, in many situations, a wiki can be totally inappropriate. For example, when the LA Times tried to harness the opinions and passion of their readership with an experimental editorial news wiki, the site was quickly defaced with obscenities<sup>vi</sup>. The following questions are useful in determining whether a wiki is right for you.

- ◆ Do you trust your users to manage the content? Or, said another way, are your users motivated to improve the quality of the resource? If so, go wiki.
- ◆ Is your organization engaged in a taxonomy initiative to organize content? If so, a traditional CMS model might be preferable because you can delegate an authority to organize and tag content according to the corporate taxonomy. The organization of a wiki is totally organic. People create pages, people notice redundancies, people merge pages. It is a chaotic process that eventually gravitates towards order (or doesn't).
- ◆ Which is worse: having something wrong appear on the site (like a typo or incorrect information) or discouraging/impeding someone from publishing an idea or some information that has the potential to help others? If the goal is to harness the creativity and knowledge from as many people as possible, a wiki might be the right tool.
- ◆ Do you intend to reuse content in an automated way? Content in wikis is totally unstructured and, therefore, difficult to reuse in other applications. Many web CMS have structured content (including metadata) that allows assets to be used by different sites in different ways.

**MediaWiki 1.5.4**

<b>Architecture</b>	LAMP
<b>Organization</b>	Community very closely aligned with the Wikipedia
<b>License</b>	GPL
<b>Sample Sites</b>	Wikipedia. As knowledge management projects, few come close to the success of Wikipedia which has become a de-facto authority on the Web. By keeping the hurdle to contribute very low (from a skills and process perspective), Wikipedia has attracted informative contributions from a broad population of experts. Unlike DMOZ, which has a hierarchy of editors, anyone can submit a post to Wikipedia. If you can capture the magic of the community contribution, MediaWiki, the software that runs Wikipedia, may provide equal success.
<b>Resources</b>	<ul style="list-style-type: none"> <li>● <b>Online Documentation:</b> <ul style="list-style-type: none"> <li>◆ <a href="http://meta.wikimedia.org/wiki/Help:Contents">http://meta.wikimedia.org/wiki/Help:Contents</a></li> <li>◆ <a href="http://en.wikipedia.org/wiki/How_to_edit_a_page">http://en.wikipedia.org/wiki/How_to_edit_a_page</a></li> <li>◆ <a href="http://wikipedia.sourceforge.net/doc">http://wikipedia.sourceforge.net/doc</a></li> <li>◆ <a href="http://www.mediawiki.org/wiki/Documentation">http://www.mediawiki.org/wiki/Documentation</a></li> </ul> </li> <li>● <b>User Forum:</b> <a href="http://mail.wikipedia.org/mailman/listinfo/mediawiki-">http://mail.wikipedia.org/mailman/listinfo/mediawiki-</a></li> </ul>

MediaWiki is one of the easier to use wikis and because of the success of Wikipedia, is very familiar and intuitive to web-savvy people. Wikipedia also demonstrates MediaWiki's ability to handle large volumes of content and traffic.

**Content Creation/Editing**

The MediaWiki wiki syntax is well documented and easy to learn. By adding heading tags ("==" in the syntax), a page can be broken up into multiple sections, each with its own editing capability. In this way, two people are able to edit different sections of the same page simultaneously. This feature is very effective for rapid collaboration around a single document. MediaWiki can upload attachments such as images and other files.

**Management**

MediaWiki has a functionality to monitor pages (via RSS) and have dialogs around the content. Any page can be assigned categories that can be added on the fly and are not restricted by a controlled vocabulary. Automatically-generated category pages list all the pages tagged with a category.

**Presentation**

MediaWiki has basic search functionality for all page content. However, uploaded files are not indexed. As shown by Wikipedia, MediaWiki is capable of handling very high traffic. There is no automated navigation within MediaWiki. Pages can be placed tagged with categories, which allow them to be listed on special category pages. There is a navigation box but editing it requires editing a configuration file on the file system and some other configuration.

**Community and Support**

The MediaWiki project is driven by Wikipedia which is probably the largest and most successful instance of a wiki. On the usability side, the popularity of Wikipedia has helped MediaWiki to evolve and has also built a large population of competent users of the technology. Not surprisingly, the documentation for MediaWiki is excellent.

**Twiki 20040904**

Architecture	Perl
Organization	Community, Consulting
License	GPL
Sample Sites	<b>Case Studies:</b> <a href="http://www.twiki.org/">http://www.twiki.org/</a> <b>Also see:</b> <ul style="list-style-type: none"> <li>◆ Disney</li> <li>◆ British Telecom</li> <li>◆ Yahoo!</li> </ul>
Resources	 <b>Online Documentation:</b> <a href="http://twiki.org/cgi-bin/view/TWiki/WebHome">http://twiki.org/cgi-bin/view/TWiki/WebHome</a>  <b>Twiki Support Web:</b> <a href="http://twiki.org/cgi-bin/view/Support/WebHome">http://twiki.org/cgi-bin/view/Support/WebHome</a>

Twiki is promoted as an enterprise collaboration platform. The target market is internal corporate wikis and, based on their testimonials page, Twiki seems to be having success in this area.

**Content Creation/Editing**

In addition to basic wiki functionality, Twiki also enables structured content, which are like database tables. Having a simple database enables Twiki applications such as To-Do Items and a FAQ. However, the increased flexibility and capability comes at the cost of ease of use. Twiki's wiki markup is considerably more complicated than MediaWiki and other more basic wikis.

**Management**

A Twiki site is broken into webs. Within each web there are topics that further organize content. Twiki has an access control system that enables an administrator to grant or deny access at the topic or web level. However, if access control is an important requirement, it is quite possible that a wiki is the wrong tool to use because, in general, wiki's are based on trust and sharing rather than restrictions.

**Presentation**

Like most wikis, Twiki has a built-in search capability that indexes all page content. The look and feel of a Twiki site can be modified through a skinning framework called "Twiki Skins." Twiki Skins has a templating framework that allows includes and basic variable manipulation.

**Community and Support**

Judging from the large number of corporate success stories (in high profile companies) on the Twiki site, Twiki's install base appears to be extensive. Twiki community-based support is delivered through a section of the Twiki site called "Support Web." Knowledgeable members of the Twiki community subscribe to updates to pages on the support web and answer questions. There is also a network of consultants and programmers that are available for hire.

## Online Community

While originally designed to support distributed, self-forming communities of interest, community communication tools are finding utility in building customer and employee communities and strengthening customer relationships. A new trend in corporate communications described in articles like the "Cluetrain Manifesto," advocates a CMS to support a dialog rather than a traditional publication model.

The model of a community portal goes back to electronic bulletin boards, which started in 1978 with the pre-Internet, modem-based CBBS and there are incredibly successful examples today such as Slashdot, where readers can submit news items and have lively threaded discussions. There is a diverse set of tools that can support this style of communication. The key requirements center on ease of use, varying degrees of editorial control, search, flexible user profile, and security systems that lower the hurdle to participation but maintain accountability and control.

Modern community portal tools include Nukes, blog tools, and bulletin boards. They all have the capability for users to post an ad hoc article easily and have the potential for readers to post commentary. Open source software is particularly attractive in this area for a number of reasons. First, enabling communication and collaboration between distributed people is an important need within the open source community so there are many very active projects that are pushing the envelope on new concepts and functionality. Unlike in many other areas where open source follows behind commercial software in innovation and community building, open source is clearly in the lead (RSS – Vignette example). The second reason why open source is highly desirable in this area is that success of the community requires a certain level of grassroots adoption that is difficult to predict. Some communities fail, some succeed. With this degree of uncertainty, investing up front in expensive software licenses is risky. You can't return unused enterprise software.

One such example is a software support portal where customers can post questions to a community that is moderated by the software company's support staff. From personal experience, sometimes these are the best resources for support available from the software company. Internally, tools like these are better than sole reliance on group email distribution lists because the content is centrally accessible rather than scattered across individuals' email boxes and vulnerable to deletion. Search turns these threaded discussions into a growing knowledge base.

### Projects Reviewed

	Version	Architecture	License	Support	Resources		
					Book	Online	Community
<b>Drupal</b>	4.6.5	LAMP	GPL	Community			
<b>phpBB</b>	2.0.19	LAMP	GPL	Community			
<b>Roller</b>	1.3	Java	Apache Style	Community			

## Drupal 4.6.5

Architecture	LAMP
Organization	Community
License	GPL
Sample Sites	<ul style="list-style-type: none"> <li>◆ <b>Gallery:</b> <a href="http://drupal.org/handbook/drupal/gallery">http://drupal.org/handbook/drupal/gallery</a></li> <li>◆ <b>LinuxJournal:</b> <a href="http://www.linuxjournal.com">http://www.linuxjournal.com</a></li> </ul> <p>While the print version of Linux Journal follows the traditional magazine format, the Linux Journal web site is more like a group blog of Linux Journal's authors complete with comments</p> <ul style="list-style-type: none"> <li>◆ <b>Corporate departmental intranet site:</b> (internal URL)</li> </ul> <p>This site is used by a technology team to share and discuss articles and other information discovered through research</p>
Resources	<ul style="list-style-type: none"> <li>● <b>Online Documentation:</b> <a href="http://drupal.org/handbooks">http://drupal.org/handbooks</a></li> <li>● <b>User Forums:</b> <a href="http://drupal.org/forum">http://drupal.org/forum</a></li> </ul>

Drupal is commonly described as an “online community in a box” or a “web log plus.” The primary strength of the core product is the ability to post articles on the site easily and the ability to select and install modules from a vast community supported module library. The most commonly-used modules for a community portal are calendar, blog, and forum.<sup>vii</sup>

With its LAMP-based architecture, Drupal performs well even on inexpensive hardware. Drupal is also one of the few PHP based CMS that run on the newest version of PHP (5). Drupal can support a diverse range of web projects ranging from personal web logs to large community-driven sites. Drupal is very easy to set up and administer. Much of the configuration can be done through the administration user interface. Drupal is frequently used for intra/inter-departmental intranet sites where individuals can post articles and other notifications rather than use email distribution lists.

### Content Creation/Editing

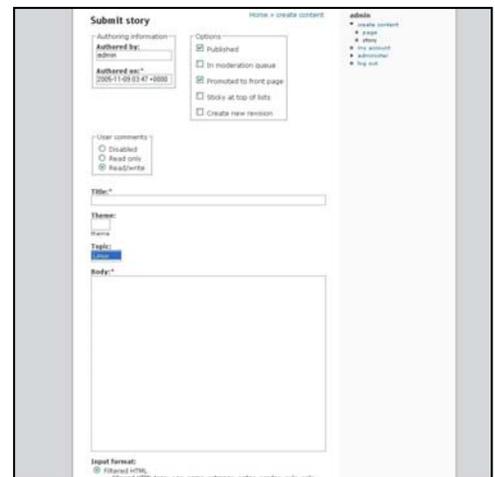
Out of the box, Drupal supports two basic content story types: Page and Story. Story is the most commonly used on Drupal sites. Content is largely unstructured with just a title and a body. More structured content types are available through the addition of modules. Occasionally, popular modules are added into the core product. WYSIWYG HTML editing requires the addition of a WYSIWYG editor such as TinyMCE. Otherwise, users need to be familiar with HTML tags. An option to filter which HTML tags are used in content can help keep users out of trouble. There is preview functionality to verify how pages look when presented on the site. Users with programming skills and sufficient privileges can insert PHP code into their pages.

### Management

Workflow consists of just two states: unpublished and published. More advanced workflow can be added by installing a module such as Drupal Workflow. Site organization is achieved through a category structure of Vocabularies, which consist of one or more Terms. This framework allows content to have multiple facets to be used in search and listing pages.

### Presentation

Drupal has a theme framework that allows administrative users to download a theme from a community supported library and install it. There are settings to allow individual users to personalize their view by selecting from the installed themes. Creating new themes requires knowledge of basic PHP and CSS. RSS XML output is native in the core product and other formats, such as Atom, are available through modules. Search is provided by an add-on module based on the open source search engine Swish-e. While the default behavior of Drupal is to use query string URLs if mod-rewrite is installed, an administrator can select a user “Clean URL” option to format URLs without query strings. Drupal's commenting feature enables users to have a dialog around a page or article.



**Community and Support**

Drupal has a large and vibrant grassroots community but does not have the structure that of Plone and TYPO3. Drupal users and developers are passionate about the software and eager to contribute their time and knowledge to the advancement of the project. Most of the contributions come in the form of add-on modules and skins. Frequently, users post bounties or other requests for these features to be developed. There is also some energy behind improving documentation which is organized into a set of handbooks. The chief resource for support is the Drupal forum. In the past, Drupal has been vulnerable to various security exploits. Drupal is aggressive about patching these vulnerabilities and communicating risks through the Drupal Security Announcements RSS feed.

**phpBB 2.0.19**

<b>Architecture</b>	LAMP
<b>Organization</b>	Community
<b>License</b>	GPL
<b>Sample Sites</b>	<ul style="list-style-type: none"> <li>◆ <b>Alfresco Forum:</b> <a href="http://www.alfresco.org/forums/">http://www.alfresco.org/forums/</a> Alfresco is an open source software start-up co-founded by Documentum founder John Newton to create an open source Enterprise Content Management System. Alfresco uses phpBB to help support a new user community by responding to support questions and soliciting feedback.</li> <li>◆ <b>Brazzil Forum:</b> <a href="http://www.brazzilforum.com/">http://www.brazzilforum.com/</a> BrazzilForum is an online community supported by Brazzil Magazine where members engage in discussions about Brazilian politics, culture, and news.</li> </ul>
<b>Resources</b>	<p><b>Book:</b> <i>Building Online Communities with phpBB 2</i> by Stoyan Stefanov and Mike Lothar</p> <hr/> <p> <b>Online Documentation:</b></p> <ul style="list-style-type: none"> <li>◆ <a href="http://www.phpbb.com/support/guide/">http://www.phpbb.com/support/guide/</a></li> <li>◆ <a href="http://www.phpbb.com/support/tutorials">http://www.phpbb.com/support/tutorials</a></li> </ul> <hr/> <p> <b>User Forums:</b> <a href="http://www.phpbb.com/phpBB/viewforum.php?f=1">http://www.phpbb.com/phpBB/viewforum.php?f=1</a></p>

While an online bulletin board is usually not considered a content management system, bulletin board software can provide functionality to solve the same business problems that companies typically implement a CMS to solve. If the goal is to simply have a web site where people can easily share information, a bulletin board system may be the answer.

phpBB is one of the most popular bulletin board systems available. phpBB is frequently offered by hosting companies as an add-on option to a hosting package. Maybe a victim of its own success, there have been many security threats to phpBB such as the Santy worm and various other exploits. Since then, the application has been totally rewritten with security a primary requirement as well as increased flexibility to add new features.

phpBB is frequently used to power support forums for commercial and open source software instead of, or in addition to, a typical help desk. By having a staff member moderate a forum, questions are sure to be answered, even if there is not a critical mass of participants.

**Content Creation/Editing**

phpBB is designed to make registration and posting simple. Users register to the site and, after responding to an email verification, are able to post messages and replies. A phpBB site can have several subject-based forums. Within a forum, users can post new topics or reply to existing ones. Topics can be monitored by an email notification system where a user can elect to receive an email any time a new post is added.

Formatting is simple but limited with buttons that add special formatting tags (called BBCode) for bold, underline, quote, list, images, links, and code (for fixed width formatting and preserved text formatting similar to the HTML <pre> tag. This is useful for posting code samples on software based forums). There is also an emoticon feature that allows a user to add "smileys" to his post.

**Management**

A single phpBB instance can have multiple forums which can have multiple categories. Forums can be configured to be private and visible to only a certain group of users. Administrators can moderate bulletin boards by removing users and their posts. Topics with information that is frequently requested can be made "sticky," which places them on top of the otherwise reverse-chronologically ordered list. Users can elect to have replies to their posts emailed to them. Administrators can manage members and censor posts including automatically filtering out certain words. In addition to posting and replying to other posts, phpBB supports a polling feature where visitors can rate posts.

**Presentation**

The user interface of phpBB is a typical online bulletin board. phpBB keeps track of which posts a user has viewed and indicates viewed posts with formatting styles similar to an email client. Site branding can be modified by installing styles that can be built from scratch or downloaded from the phpBB site.

**Community and Support**

There are hundreds of instances of phpBB in use today and the community is extremely active. Community support is delivered through an instance of phpBB running on the phpBB web site which has hundreds of posts per day and is well moderated. The forums also support the community's sharing of skins and modules. Users can either post a request for a module that they need or publish a module that they have developed.

**Roller 1.3**

<b>Architecture</b>	Java
<b>Organization</b>	Community
<b>License</b>	Roller Public License (Apache Software License with the words "Apache Software Foundation" replaced with Roller project founder "David M Johnson")
<b>Sample Sites</b>	<ul style="list-style-type: none"> <li>◆ Sun Blogs (<a href="http://blogs.sun.com">http://blogs.sun.com</a>)</li> <li>◆ JRoller (<a href="http://www.jroller.com">http://www.jroller.com</a>) JRoller is one of the primary blogging sites within the Java community.</li> <li>◆ IBM Internal Blogs  IBM hosts over 2,800 internal blogs and is considered a success story in corporate blogging<sup>viii</sup>. IBM internal blogs run on Roller.</li> </ul>
<b>Resources</b>	<ul style="list-style-type: none"> <li> <b>Online Documentation:</b> <a href="http://www.rollerweblogger.org/wiki">http://www.rollerweblogger.org/wiki</a></li> <li> <b>Mail Lists:</b> <a href="http://www.rollerweblogger.org/wiki/Wiki.jsp?page=RollerMailingLists">http://www.rollerweblogger.org/wiki/Wiki.jsp?page=RollerMailingLists</a></li> </ul>

Roller is the most established of the Java-based blogging softwares. Powering sites such as the popular blogs.sun.com and JRoller, Roller certainly qualifies as industrial strength. Roller is designed to host multiple blogs, each with its own formatting layout and style.

**Content Creation/Editing**

Authors can select from different WYSIWYG and plain text editors. The default editor is plain text but preserves basic formatting such as line breaks. There is a spell check feature. In addition to adding text, users can also upload files to reference in their web log rather than having to host images elsewhere. In addition to the web interface, Roller works with blogging clients such as w:logger, MarsEdit, and Ecto.

**Management**

Each web log on a Roller server can be managed as its own site where the author can specify the look and behavior of the web log through an administrative user interface. Roller supports categories so authors can segment their blogs into sub-blogs on a particular topic. Advanced blogging features such as traceback come standard. However, internal corporate blog sites may want to disable this feature so that external sites are not notified of references to them.

**Presentation**

The Roller presentation system is based on the Apache Velocity template engine which makes it fairly simple to create new templates. Theme modifications require familiarity with HTML. Roller comes with an imbedded search engine and a blog aggregator that creates a combine list of entries on a single page complete with a listing of the most active blogs. Roller has a page caching mechanism to support high traffic sites such as blogs.sun.com.

**Community and Support**

Development is lead by project founder Dave Johnson who was hired by Sun to work on Roller to work full time on the Roller project which powers blogs.sun.com. Other contributors and community members run large blog sites and/or corporate blogging servers. In this way, the Roller community could be compared to the early Apache web server community. The release process was developed Sun's release engineer who is also a committer on the project. Response time to questions on the mail lists is good and the documentation is thorough.

## How to Evaluate Open Source

The source code is not the only thing that is open about open source software. Open source happens out in the open. By subscribing to user mail lists and other communication channels, it is easy to learn about what others are doing with the software, which features are good, and which features need work. Reading a project roadmap or the publicly-accessible bug lists will tell you where the project is going, who is driving it, and whether the team is well organized. You can also get a feel for the personalities and the social dynamics of the group.

As you read through the archives, pay attention to questions that do not get answered and who answers the questions that do get answered. Having several people actively posting answers is a sign of a strong community that will survive if one of its principles moves on. Also look at the content of the answers. A reference to a document means that a document exists – a good sign! A long set of step by step instructions may indicate that there is insufficient documentation and processes for creating documentation. It may also indicate that users need to constantly deal with work-arounds rather than actively maintaining the code base. For example, if you see instructions like “comment out the line that says x and add the following code...” it could mean no one is patching in those fixes.

Look at (or have your technical staff look at) the development guides and practices. The better managed projects have functionality roadmaps, a clearly defined release process, coding standards, and use practices like unit tests which automatically verify that additions do not break other parts of the code base. Reading through the developer site should make it clear how the community decides how functionality is assigned to releases and what kind of testing occurs.

Browsing through the bug tracking system will tell you how active the software is being tested and how efficiently issues are being resolved. Do not assume that having lots of issues in the bug tracking system is a bad thing. It means that the software is being used by people that care enough to work with the community to improve it. Look at the content of the issues. Bug tracking systems are also used to record requests for enhancements which indicate how the software is expected to evolve.

Most importantly, you can actually try the software. In many cases, you don't even need to install the software to get a demo. The site [www.opensourcecms.com](http://www.opensourcecms.com) has demo versions of over seventy open source LAMP based CMS including Drupal, Mambo, and Joomla, as described here. eZ publish, Lenya, and phpBB have demo instances of the software running on their sites. As you experiment with the software, involve prospective users in the process. Have them try the software out, list what they would like to change, and how important those changes would be. Give them ownership in the process. Doing so will help them become invested in the solution and increase adoption.

I should note here that I would apply the same recommendations for evaluating commercial software if I could. However, software companies do not expose who is the brain behind the technology, how helpful the tech support is, and how the organization tends to respond to turnover.

## Other Resources

### CMS Matrix

([www.cmsmatrix.org](http://www.cmsmatrix.org))

The CMS Matrix rates hundreds of open source content management systems based on roughly one hundred criteria. A comparison tool allows you to do feature comparisons of up to three CMS. Although this tool is not reliable for actually selecting a CMS, it is useful to narrow down the options that you need to evaluate more deeply. Another useful feature of CMS Matrix is a discussion board where you can get answers and advice.

### The CMS Report

(<http://www.cmswatch.com/cmsreport>)

Like “Consumer Reports,” The CMS Report is a “buy side” analysis of content management software. Because it is not targeted to the vendor market, the CMS Report is able to give open source better coverage than the big software research companies. Currently the CMS Report has in depth reviews of OpenCMS, Zope, Plone, and Midgard. Soon, coverage of more projects will be added. CMSWatch's “Trend Watch” blog is also an excellent resource for getting CMS industry news.

### CMS Review

(<http://www.cmsreview.com>)

The CMS Review has short summaries of many open source and commercial content management systems.

### OSCOM

(<http://www.oscom.org>)

OSCOM is an organization for open source content management developers. The Matrix part of the site is not very useful because it is incomplete and does not go into depth. However, the blog aggregator (Planet OSCOM) is a great way to keep up to date on what is happening in the world of open source

### CMS Mailing List

(<http://mailman.skybuilders.com/mailman/listinfo/cms>)

This list is useful for general content management questions and is frequently used by people interested in open source.

---

## Where to Put the Money That You Save

---

The truth is that technology is not the primary reason why content management initiatives succeed or fail. Success in content management depends on activities such as migrating content, improving business processes, and achieving adoption. With the absence of licensing costs and availability of different support options, investments in the solution can be redirected into factors that have the highest impact on the success of a content management initiative. More time and effort can be spent on prototyping to understand requirements better, managing the project, improving business processes, migrating content, and educating users. In his article "*Spending patterns during CMS implementation*"<sup>ix</sup> James Robertson of Step Two writes that the implementation is just the first of three phases of a CMS project. Implementation is followed by phases of adoption which includes data migration, training, and evangelizing the solution, and enhancement which addresses requirements that were either deferred or realized once the solution was deployed. Robertson recommends setting realistic expectations of time and effort for these second two phases, with the adoption phase lasting up to 12 months, and continuous enhancement.

After the solution is deployed, allow it to evolve. If the stakeholders were involved in the earlier phases of the project, chances are they will feel ownership of the application and think of ways to help it improve once they understand how the application fits into their business processes. Impress on these business users that the application has the potential to evolve and they will be less tenacious on the scope of the initial release.

---

## What to Do About Support

---

Because open source software presents more support options (commercial, consulting, and community), some thought should be given to the support needs of the organization. Many companies, due to policy or habit, always purchase support contracts. In some cases, when tech support is frequently requested, those investments are justified. In other cases, when the software just runs without much need for human intervention, or if the organization has experienced technical staff that has knowledge of the technology, subscription-style support packages make less sense. Frequently, developers find access to the knowledge base and public search engines more expedient and useful than phone or email-based support. If commercial software style support is desired, it may be offered by the company hosting the open source project or by a third party (such as SpikeSource and SourceLabs). The system integrator that helped deploy the solution may also offer different support models ranging from per-use to subscription based.

As you support your implementation, you should also think about what you should be giving back to the community. Contributing back code you developed for your own use is not required but it may be to your advantage to do so. A good rule of thumb is that you should contribute code back when the competitive advantage of sole access is outweighed by the cost of maintaining it. When you contribute back, great things can happen. The code gets reviewed by really good programmers. The code becomes part of the core application so you have less to worry about it when you upgrade. The contribution that you make also has the potential to grow into a new feature that would be useful to you. The application gets better and attracts more users which ensures its future.

---

## Conclusion

---

Open source content management software presents an attractive option for companies looking for a straightforward solution to a common problem. However, traditional methods of software selection are less helpful in evaluating open source than commercial software. Indeed, the vast selection of open source content management software, coupled with the broadness of the category, can make the task of sifting through the possibilities tedious and disorienting. To get your bearings, focus on the business problem and look to see what other companies have used to solve similar problems. Once you do narrow down to a set of viable options, the openness of open source will allow you to learn more about the software than you ever could learn in the commercial world. If you leverage these benefits, open source can reduce the risk of your initiative and provide a solution that can grow with your company.

---

<sup>i</sup> CMSML (<http://www.cmsml.org>) is an initiative first started by the CMSReview (<http://www.cmsreview.com>), OSCOM (<http://www.oscom.org>), and the CMS Evaluation Lab of the University of Washington iSchool () to define an XML specification that CMS vendors could use to describe the features of their products.

<sup>ii</sup> CMS Matrix (<http://www.cmsmatrix.org/>) rates over 100 web content management systems on a matrix of roughly 100 features. Information is volunteered by software developers but there is a policing mechanism where viewers can post corrections based on their real world experience with the software.

<sup>iii</sup> See article *From Enterprise Content Management to Effective Content Management*.

<sup>iv</sup> <http://www.mambolove.com>

<sup>v</sup> See Article WhyWikiWorks on the original Wiki developed by Ward Cunningham.  
<http://c2.com/cgi/wiki?WhyWikiWorks>

<sup>vi</sup> People power backfires for LA Times, Claire Cozens and agencies, Guardian Unlimited, Tuesday, June 21, 2005  
(<http://media.guardian.co.uk/site/story/0,14173,1511300,00.html>)

<sup>vii</sup> Drupal Case Study for Community Portal Websites, Cade Rarick, Jason Moehlmann, Greg Noren.  
<http://drupal.org/node/12356>

<sup>viii</sup> "Blogging at IBM" posting on *This week on IAOCblog.com* by Phil Borremans. International Association of Online Communicators. URL: [http://www.iaocblog.com/blog/\\_archives/2005/3/4/396555.html](http://www.iaocblog.com/blog/_archives/2005/3/4/396555.html)

<sup>ix</sup> Step Two briefing (October, 2005): [http://www.steptwo.com.au/papers/cmb\\_spendingcms/index.html](http://www.steptwo.com.au/papers/cmb_spendingcms/index.html)

---

#### ABOUT OPTAROS <http://www.optaros.com>

Optaros is a consulting and systems integration firm that helps enterprises solve IT business problems by providing services and solutions that maximize the benefits of open source software. Bringing together experts in creating enterprise IT solutions and experts in the power of open source, Optaros plans and builds business systems that give you better value today and increased control in the future.

#### CREATIVE COMMONS LICENSE



This work is licensed under a Creative Commons Attribution 2.5 License

#### CONTACT

**Brian Otis**

VP, Sales and Partnerships

email: [botis@optaros.com](mailto:botis@optaros.com)

phone: (617) 227-1855 x110