WHITE PAPER

Open Source vs. Proprietary CMS

Which is right for my organization?

Open Source vs. Proprietary CMS

What is an Open Source platform? What is a Proprietary platform? What are the strengths and weaknesses of each solution, and how can you make the best choice for your organization once you've decided to create or change your web presence? To help answer these questions, this document will compare the facets of each model, and then guide the reader through some of the major concerns and challenges an organization faces when evaluating web solutions. This document will not determine which solution is right for your organization's needs, but rather it offers a series of points for consideration and a demonstrative case to help you make an informed decision. By the end of this article you should be able to:

- Understand the major features of "open" and "closed" systems
- Identify the strengths and weaknesses of these systems
- Recognize "The Big 4" questions an organization should consider when evaluating a solution
- Know what additional considerations play a role in solution selection
- Apply the lessons of a case study to your own organization's case

Understanding Open Source and Proprietary Systems

Much has changed over the last 10 years for both Open Source and Proprietary solutions, and before your organization commits to a specific solution, it is important to understand both models. We'll start this examination by looking at Open Source and Proprietary software as a whole and later narrow down to CMS solutions specifically.

What does Open Source really mean in 2010?

Today when we hear "Open Source," many people think: "it's free to download" and "modifiable without limits." While this is true, the concept of "Open Source" goes deeper than these two observations.

At its heart, Open Source software is all about collaboration and freedom. With Open Source solutions, the software is yours to modify as you see fit, and it extends to meet your organization's specific needs. You can expand the solution's capabilities by leveraging its community – often community members will release customization code to the rest of the community under the same license as the software itself. While you do run into a fair bit of abandoned projects, or end products that have serious bugginess affecting them, there are some really impactful Open Source projects that have greatly enriched all of our technology experiences.

Many people think that going "Open Source" requires you to be a programmer or super techie. However, with the number of organizations supporting Open Source systems today, that doesn't have to be the case. As organizations continue to adopt an Open Source system, new components are built specifically to create a more useable product for the non-technical end users.

Despite the large umbrella under which Open Source solutions fall, not all solutions calling themselves Open Source operate in the same manner:

Types of Open Source Solutions

- **Free Open Source Software (FOSS)** – This software is free to acquire and all code is available to users.
- **Commercial Open Source Software** – In this model all source code is available, but only after purchasing a license to use the software. Users can still use the code to modify or customize the core tool, but licensing fees are frequently used to fund the solution's development or for support from the company directly.
- **Partial Open Source Software** – This model is a hybrid between truly Open Source software and Proprietary software. In this model, the solution has both closed source components that cannot be viewed or modified by end users as well as components that are completely Open Source. Solutions in this category can be either commercial or free solutions.

In order to most effectively navigate the world of Open Source, it is important to understand how these solutions fit into the broader picture of software platforms.

What Does "Proprietary" Really Mean in 2010?

Proprietary systems have been forced towards system interoperability over the last several years and to no longer consider themselves a monolithic solution that will serve everyone's needs. Most mature software products and platforms have now developed a package to allow end users to extend the functionality of the solution either via a Software Developer Kit (SDK) or through a series of web services and APIs. This functionality allows you to extend the solution without the explicit assistance or approval of the product company.

SalesForce.com serves as an excellent example – while the software itself is not Open Source, the company has really driven the charge to create an open platform by creating web services and APIs. By offering these connectors to its systems, SalesForce has created a product that allows for customization to meet your organization's specific needs without releasing their proprietary code to the world. Similar to Open Source solutions, it has also developed a market for those who innovate and implement on its platform and created a development community.

Let's now take a look at some of the major similarities and differences between Open Source and Proprietary solutions.

Comparing Open Source and Proprietary Solutions

While there are no "right" or "wrong" answers to the Open Source vs. Proprietary solution, your organization will likely have a "best fit" answer. To help determine which model most suits your organization's needs, the following section compares each model.

Advantages and Disadvantages of Each Model

Comparing Open Source and Proprietary solutions goes beyond "Open" vs. "Closed" models. The table below will highlight the general advantages and disadvantages of each system, and while not all-inclusive, it is designed to give you an overview of what each model entails.

Open Source - Advantages	Proprietary Software - Advantages
• Price point • The availability of the source code and the right to modify it • The right to redistribute modifications and improvements to the code • No single entity on which the future of the software depends • Potential pool of developers greatly exceeds what a Proprietary solution can draw from • Code visible to the world, preventing development in a vacuum • Provides a new forum for democratic action	• Predictable releases and consistent feature development • Entity to hold responsible for bugs, errors, and updates • Customizations can be added directly to base ensuring future stability of enhancements • More consistent training options • Easier access to support
Open Source - Disadvantages	**Proprietary Software - Disadvantages**
• No guarantee that development will continue • Quality of work can vary by developer • May have significant problems connected to intellectual property • Not always aware of open source projects' existence	• Higher start-up costs • Single company releasing patches • Vendor owns software

In summary, with the "openness" of Open Source comes the opportunity for extensive customization, a free dialogue between developers, and high visibility that encourages innovation. At the same time, unlimited openness is accompanied by challenges, particularly when an organization undertakes more complex customized solutions. If no single person is responsible for owning the upgrade path for existing users to new versions of the platform, then users may encounter obstacles such as irregular releases and a possibility for incompatibility between upgrades and customizations.

Examples of Each Model

To better understand how these models translate into products you may already use, here are some examples of popular solutions.

Open Source - General	Proprietary Software - General

• Operating Systems
  ○ Linux
• Web Browsers
  ○ FireFox
  ○ Chrome (partial)
• Email Clients
  ○ Thunderbird
• Office Productivity
  ○ Open Office
	• Operating Systems
	  ○ Windows
	  ○ OSX & iOS
	• Web Browsers
	  ○ Internet Explorer
	  ○ Safari
	  ○ Opera
	• Email Clients
	  ○ Outlook
	• Office Productivity
	  ○ Microsoft Office (Word, Excel, PowerPoint)

Open Source - General	Proprietary Software - General
• Drupal - http://drupal.org/ • Joomla - http://www.joomla.org/ • WordPress - http://wordpress.com/ • WebGUI - http://www.webgui.org/ • Plone - http://plone.org/ • MediaWiki - http://www.mediawiki.org	• Ektron - http://www.ektron.com/ • SharePoint - http://sharepoint.microsoft.com • SoapBlox - http://www.soapblox.net/ • Sitecore - http://www.sitecore.net

"The Big 4" Considerations

Let's shift our focus from a "bird's eye" perspective of Open Source and Proprietary software to the area of Content Management specifically. When you are considering a CMS, there are several distinctions to make when deciding which model better fits your organization. Here are the most important considerations, or "The Big 4":

Budget

Before your organization begins any new project, it is imperative that you evaluate how much you can budget for that undertaking. In addition to concrete financial costs, your budget needs to reflect a project's timeline and employee availability. Each element does not exist in a vacuum, and each must be evaluated together to fully understand your budget.

For example, if you know you will have 40 content authors and 15 chapter sites, using proprietary software you may already face licensing fees of 70k or more just for software, which could consume your entire budget.

Adopting a free, Open Source system on the other hand may reduce your initial financial investment, but may require additional employee training time, which adds expense and can delay a project.

In both models, you must ask: "how much will it cost to actually deliver my vision?" A proprietary system may have more features out of the box, but you may be locked into working with their developers for any customizations and will be limited to the APIs or constraints of the framework. Open Source systems grant you the absolute freedom to develop your customizations, but how much customization is required to build components the proprietary system may already deliver?

For example, if you know that you have some very specific needs in terms of social networking and document collaboration that are not included with an Open Source system, but are supported through proprietary offerings. The predictability of cost in implementing out of the box features might sway the decision as long as it does not conflict with your organization's philosophy.

Philosophy

When choosing a solution, organizations may also consider whether that solution falls in line with the group's overall mission and ideals. An organization might hold in high esteem the aspect that the application is created democratically with each member weighing in and donating their time for the good of all to benefit, for example. The Open Source movement reflects many ideals/goals of mission-based organizations in terms of giving back to the community and sharing knowledge and

resources for the betterment of all. These are intangibles that each organization really needs to think about as it determines how organizational philosophies will impact a decision.

Innovation / Motivation Factor

Entering the world of Open Source software is still somewhat adventurous. You need to plan well, find good traveling partners, and recognize that your organization will deal with a product to which thousands of people have likely contributed. In the proprietary software environment, you can likely contact the tech support guy who has fielded a hundred other similar questions, and if needed could bring in the development team that built the code in question. If there's a problem, a fix will likely be on its way. In the Open Source world you would either need to identify the module author and await a code update, or hire a developer/use internal developer to make the required changes and then submit the module update back to the community for review and acceptance.

Solution Complexity

When considering what kind of CMS will meet your organization's needs, you must also come to an understanding about the degree of complexity involved in the solution (particularly when needs can't be met "out of the box" and require varying degrees of customizations). There's a fine line between "hacking" code to get it to what you need it to do and building extensions using a recognized and agreed upon framework. Many mission-based organizations may not have full time development groups. Some consider the website a bill that comes due every 4 to 5 years. The Open Source option typically requires organizations to have a more hands on approach or to find a very good partner to keep things working smoothly.

Additional Considerations

Once you have tackled "the Big 4," it is important to expand your search for a solution to encompass a wider set of factors.

Company & Product Stability

Before committing your organization to a specific solution, you must evaluate the stability of the product itself and the company behind it. After all, why would you want to spend months getting your site set up just to have the company behind that solution go out of business?

Recommended Questions:

- How many years has the company and product been in existence?
- What is the likelihood that development of the product will continue?
- What other organizations are using the software?
- What kinds of vendor resources exist for site maintenance and troubleshooting?

Developer/Community Activity

Similar to stability, when choosing a solution it is helpful to evaluate how the community as a whole has embraced the solution. Open Source solutions should have a robust, active community routinely

contributing to the solution, and Proprietary solutions should have an active member base that can provide information, assistance, or even paid services should they be required.

Recommended Questions:

- How active is the community on the solution-controlled site?
- How many other sites and organizations support the solution?
- Does the solution have regular meetings/conferences for the community to get together?

Existing Feature Fit

Next, you need to begin to evaluate your organization's requirements for the new site. Create a list of your required features and compare that against the available feature list of each solution. Any feature required that is either not present or does not meet your expectations ("gaps") may need to be customized to fully meet your needs.

Recommended Questions:

- Which solution meets the most important requirements out of the box?
- Which feature "gaps" can be postponed till a future version of the site?
- How much time, effort, and financial resources will be required to complete any needed customizations?
- Are there any systems (member/contact database, financial system, email system) that will need to integrate with the CMS?

Internal Resources

Embarking on a web project can involve a lot of jobs – from content creation and organization through design and build. Before selecting a solution, you should start by evaluating what in-house resources are available to assist with the project as well as any limitations those resources may present. This is also a good time to consider whether your staff has the skill set necessary to work with the solution, or if they will have the time to learn a new platform. For example, some organizations only have a .NET developer and would not think of using something written in PHP – it is important to establish this information at the outset.

Recommended Questions:

- What tasks can be completed in-house?
- What technologies are currently supported by in-house staff and hardware?
- What amount of staff time can you commit to learning a new system, should you choose an unfamiliar platform?

Cost

Cost is a major concern for organizations, but it is important to understand the difference between initial costs and total costs of implementation. Organizations are commonly drawn to free, Open Source solutions because of the price point – free is cheap, right? In fact, "no cost to acquire does" not mean "free."

We have heard the saying “free as in beer,” but in fact Open Source Software is more like “free as in kittens” - it is yours to take, but requires continual attention and cost. Patching, upgrades, research, and testing require employee resources and time, and may even require financial support if you need the assistance of outside organizations.

For content management systems specifically, eventually a major, new version of the software will be released. Your upgrade costs will depend on the amount of customizations you site has and whether features are supported when you upgrade. Any custom component you build will need to be upgraded to work with the new version. In Proprietary solutions, APIs you rely upon may have changed or even been removed. For Open Source systems, any community-provided functionality that you rely upon can be downloaded for free if the developer has already released the upgraded version, but if not, you may need to fund the upgrade directly.

Recommended Questions:

- What is the financial budget for this project?
- What in-house resources are available for this project, and what outside resources would need to be brought in to complete the work?
- What requirements must be achieved by the project’s deadline, and what can be postponed till a future release?

Product Support Model

Should your organization run into any difficulty working with the solution, who can you turn to for assistance? Proprietary solutions usually have a company behind the product providing support, but often these come with varying levels of service and cost. Open Source solutions may have a company behind the tool offering paid support similar to Proprietary solutions, but often the community provides this service instead or in conjunction with a company.

Recommended Questions:

- How much, if anything, are you willing to pay for support?
- Does support need to come from the company who created the solution, or can a 3rd-party company offer support?
- How willing is your staff to work out issues on their own?
- How much documentation and training materials are available to the general public?
- What training resources are available for the solution?

Making the CMS Decision

Now that you’ve considered these factors, here is the process that we use with customers when helping them make a CMS decision. This section will briefly identify each item, and the following section will walk through a fictitious case study for an organization selecting a solution.

Step 1 Define the Project Need

- Why are you building the site?

- Who is it for?
- What does it need to do?

Step 2 Consider "the Big 4:

- Determine your budget
  - As mentioned before, this includes financial costs, timeline, and employee resources
- Weigh your organization's philosophy against potential solutions
- Evaluate your willingness to innovate
- Analyze your solution's complexity

Step 3 Consider Your In-House Strengths /Skills

- It may go without saying, but if you have developers on staff with PHP familiarity, that gives a significant advantage to choose a PHP-based platform.

Step 4 Identify a List of Potential Platforms

- Start with a Long list to begin your evaluation
- As you continue the process, remove platforms that do not fit
- The end result should be a Short List of platforms for full evaluation

Step 5 Evaluate the Short List of Solutions

- Determine Feature Compatibility
- Determine "gaps" between your requirements and the features provided
- And, Determine long-term costs for the solutions

Step 6 Make Your Selection

- This is not a decision to be made in a vacuum, but instead must factor in all of the above steps

Sample Project – DEMO

To better evaluate Open Source vs. Proprietary systems, let's use a sample project and actually walk through the pros and cons of how one organization would select a solution. Certainly some factors are organization-specific, but this demonstration will help to outline the basics of the decision-making process in action.

Step 1 Define the Project Need

D.C.'s group of Environmentally Motivated Organizations, or DEMO, is looking to launch their first website from the ground up. Their staff of six all work to advance the organization's mission, and have now decided that having a web presence is the best method to further reach out to the community.

Step 2 Consider "The Big 4"

Open Source - CMS	Proprietary Software - CMS
DEMO has budgeted between $25,000 - $30,000	DEMO doesn't feel strongly about whether the system they use is shaped by the crowd, or a corporation. It needs to work consistently and provide the features needed within their budget.
Open Source - CMS	**Proprietary Software - CMS**
• Given the legislative work and campaigns that DEMO is planning for 2010, the staff team is not looking to do more than log into a site, make updates, set up alerts, and move on • The team has minimal interest in 'tinkering' with the site	• Simple integration to a donation payment processing tool and an email newsletter service • There's interest in RSS alerts for issues and calendar management / subscription for events • The site is expected to be about 100 pages to start

Step 3 Consider In-House Strengths & Skills

Two of the people on staff consider themselves "techies" and regularly blog, but have limited HTML knowledge. The rest of the staff is comfortable with office software but have no HTML knowledge. No one on staff has any programming knowledge or familiarity with any CMS system currently on the market.

This evaluation puts no limitations on the platform chosen, but the selected solution must be easy to use by non-technical staffers.

Step 4 Identify a List of Potential Platforms

DEMO has done some research, and based upon their needs, narrowed the list of CMS systems down to two platforms – Drupal, an Open Source system, and Ektron's CMS400, a proprietary CMS.

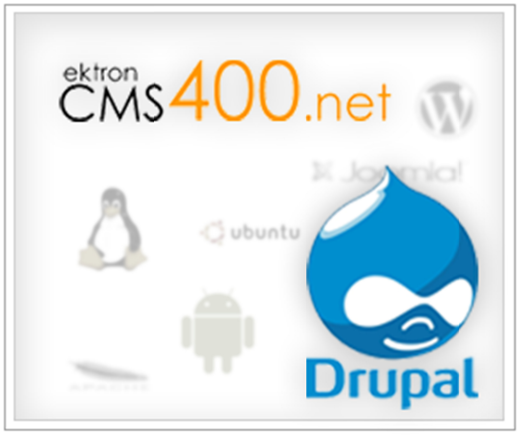

Figure 1. Ektron, a proprietary CMS, and Drupal, an open-source CMS, are compared

Step 5 Evaluate the Short List of Solutions

To determine which solution is the best fit, each of the Additional Considerations will be evaluated:

Company & Product Stability

DEMO considered stability a key area of concern for them, so they dug a little deeper. DEMO considered the following questions:

Drupal	Ektron
How mature is the solution?	
• Originally launched in January of 2001 • Released version 7.x in January of 2011	• Company founded in 1998 • Launched CMS100 in 2001 • Launched CMS400 v8 in 2009
How easy is it to use and maintain the system?	
• Drupal requires regular monitoring and patching of both Drupal itself and of all modules downloaded and used on the site. • Out of the box Drupal is not that easy to use but can be configured to be more usable.	• The selected Ektron hosting partner applies all major patches for you as part of the hosting fee • This comes with a higher monthly hosting fee over Drupal
What other organizations are using the software?	
• Drupal powers major sites like whitehouse.gov and Amnesty International	• Ektron powers a wide variety of sites like Wal-Mart, the United States Golf Association, and the Special Olympics

Developer & Community Activity

Building off of an analysis of the companies and their product stability, DEMO wanted to select a product that had a robust, active community. DEMO considered the following questions:

Drupal	Ektron
How active is the community around the product?	
• 700,000 registered users • 700 design themes available for download • 7,000 contributed modules	• About 30,000 forum posts • 75 community-created widgets and another 65 from Ektron directly
Does the solution have regular meetings/conferences for the community to get together?	
• One annual conference in the US and one in Europe • Washington, DC user group that meets regularly close to the DEMO office • Lots of materials available for free	• Ektron meets throughout the year across the country • Ektron holds one major conference annually • Many resources available with license

Feature Compatibility

DEMO's next step was to compare its list of requirements against the functionality provided by each tool (this process is heavily abridged and simplified for the purposes of this example).

Drupal	Ektron
What requirements are achieved out of the box?	
• Both platforms appear to meet all of DEMO's requirements without the need for custom modules or programming.	
What requirements are not delivered by the solution ("gaps")?	
• All requirements are met with the use of 3rd-party modules.	• All requirements are met out of the box.
Is there a preference for either system?	
• DEMO likes how easy it is to allow site visitors to contribute site content should they ever wish to add that capability.	• DEMO prefers the video embedding process provided by Ektron over the functionality of Drupal, functionality that will come in handy for DEMO's phase 2 where there will be a high focus on creating and distributing viral videos

Internal Resources

Although DEMO's team has no programming or web hosting experience (giving neither option an advantage), DEMO still chose to evaluate this consideration since it may drive future hires as well as impact the project's cost.

Drupal	Ektron
What technologies are currently supported by in-house staff and hardware?	
• LAMP based solution • Requires external hosting	• Windows Server and .net-based solution • Requires external hosting
What tasks can be completed in-house?	
• All content will be written in-house • Internal graphic design team can create initial design comp • Internal staff will manage and maintain all content post-launch • All other roles will need to be outsourced, including maintenance	

Determine Project Costs

After all of the requirements have been outlined, with the help of experts on each solution, DEMO evaluated what the solution would cost on each platform. They took into account not only the initial cost of the solution, but also what additional costs could arise with each subsequent year. Note: these numbers are for illustrative purposes only. Each project's specific needs and requirements will dictate project costs and plan.

Drupal	Ektron
Software	
$0	$16,000 for 5 administrators
Hosting	
$600/year	$1,200/year
Maintenance & Support	
At cost from consulting firm, estimated $3,000/year	$3,200/year (20% of licensing fee) from Ektron directly with the first year included in license
Solution Design & Build	
$22,000	$22,000
Customization Budget/Reserve	
$2,000	$0
Total Cost of Ownership	
Initial - $27,600 • 3 Year Total - $40,165 (includes budget for upgrade) • 5 Year Total - $47,985	Initial - $39,200 • 3 Year Total - $48,000 • 5 Year Total - $56,800

Overall we see a 3 year TCO difference of approximately $9,000. The initial $16,000 difference in software acquisition costs are flattened out by:

- Customization Reserve of $2,000 in Drupal required to handle any modules that do not perform as desired/required
- Proprietary host vendor performing routine application patching and maintenance
- Refactoring Reserve in Year 3 to cover migration to the next version of Drupal

While these numbers are merely meant to illustrate the point that there are a number of factors to include in estimating the cost of a solution, if cost is a significant driver, total cost should be assessed to understand the long term implications for an organization.

Step 6 Make Your Selection

Ultimately, DEMO's budget, the complexity of the solution (low), and the stability of local user groups/ meet-ups led DEMO to select Drupal. They spoke with several Drupal vendors after sending out a post to NTEN's 501 tech club at NTEN.org. Ultimately, while the organization loved the back-end interface of Ektron, it wasn't enough of a factor to re-prioritize other needs in order to build out the site. DEMO briefly reconsidered their decision a week later when a staff member started talking about the need to have document collaboration capabilities on the site in member only areas, but the team decided it was a better choice to pay for a separate, hosted SharePoint site for $29 a month instead.

Conclusion

As this document has illustrated, choosing between Open Source and Proprietary solutions requires detailed considerations that are organization-specific. Both Open Source and Proprietary platforms have strengths and weaknesses that relate to the "open" versus "closed" nature of their development. The Open Source model encourages community collaboration, frequent innovation, and it offers low initial costs, but it may also require more customization and more risk in terms of the regularity of updates, patches and other changes. Proprietary models typically offer greater predictability and easier access to support, but these benefits may come with higher initial costs and an inability to "tinker" with the product which is owned and maintained by one vendor.

When initiating the process of picking a new web solution that is either Open Source or Proprietary, it is helpful for organizations to use "The Big 4" scheme – Budget, Philosophy, Innovation/Motivation and Solution Complexity – as well as to evaluate other factors such as internal resources, product support and long-term versus initial costs. No "one size fits all" solution exists, which means it is especially important that organizations take time in early stages to determine their "must haves" and "wish lists" in order to make an appropriate match. Time spent on evaluation at the front end will have long-term benefits when it comes to improving an organization's web presence.