BIBLIO COMMONS

2019

The Future of Drupal

Why 2019 Might be the Year for Public Libraries to Re-Consider Drupal (if They Haven't Already)

Prepared by:
Beth Jefferson, CEO & Co-Founder, BiblioCommons
Teresa Harris, Marketing Manager, BiblioCommons

Table of Contents:

For years, Drupal was a go-to open source content management system (CMS) for public libraries, alongside nonprofits, government and corporate sites of all sizes. Many libraries started on Drupal 6 and have migrated to version 7, while others chose Drupal more recently. There is no definitive tracking for library websites, but based on our knowledge, we'd estimate that at least a quarter of public library websites are currently (or have recently been built) on Drupal.

And yet, the migration to Drupal 8 has proved slow, as many organizations — public libraries included — are beginning to reconsider whether it's still the right choice for them. Initially released in 2015, Drupal 8 represents a fundamental shift from previous versions in terms of technical architecture, and this radical shift could be what's preventing many organizations from adopting the latest version.

It's time to re-evaluate and ensure that Drupal remains the right solution

Drupal founder Dries Buytaert announced at DrupalEurope 2018 that come November 2021, Drupal 7 will no longer be supported by core maintainers with fixes, security releases, or enhancements. Now, all organizations currently either on Drupal or considering it as an option are being forced to decide: do we migrate to Drupal 8 — or take this crossroads as an opportunity to reconsider altogether?

In this paper, we will focus exclusively on (and quote extensively from) the opinions of other long-time Drupal builders and contributors, including agencies, freelancers, and in-house developers, and share their recent questions and concerns, such as:

- The challenges and costs of migrating from Drupal 7 to 8
- The new complexities of Drupal 8
- Many of the longstanding challenges of Drupal 7 that have yet to be acknowledged
- The shift in Drupal's priorities
- Whether or not trends are in favour of Drupal
- Available alternatives

We realize that every library will face different circumstances and priorities. However, as the importance of the digital experience in organizations' overall sustainability and strategy grows, it's critical public library leadership is aware of what's happening across the technology landscape. The purpose of this paper is to empower public library stakeholders beyond the front-end development team to meaningfully engage in the discussion, and make better, more informed decisions.

Why have so few organizations upgraded to Drupal 8?

Drupal upgrades have always been challenging — Paul Vetch, Strategy Director at Torchbox, one of the largest design build agencies for nonprofits in the UK, calls them "epic," and some developers even liken upgrades to "starting from scratch." According to Vetch, Drupal's six-month release cycle between minor versions has already led to potential data loss and security issues, like upgrade path bugs and access bypass vulnerability.

Migration is a significant undertaking

Unlike most other CMSs, Drupal has not historically guaranteed backwards compatibility from one version to another. Themes, modules, and plug-ins may become incompatible with new versions — as has happened with Drupal 8 — leaving organizations dependent on module contributors to release updates. Many never do.

1. Themes will have to be completely rewritten

In Drupal 8, PHPTemplate was replaced by Twig as the template engine of choice — the biggest overhaul of Drupal theming in a decade. While Twig makes the Drupal theme layer faster and more secure, it's now impossible to run PHP scripts, make database calls or access the file system. This means that, in most implementations, themes will have to be completely rewritten after migration.

> ***"In addition to the revamped architecture, new required build processes, and upgrade difficulties, almost every Drupal site has to completely rewrite its theme."***
>
> — Jeff Geerling, Author and Software developer

"In many cases, this is the straw that breaks the camel's back," writes author and software developer Jeff Geerling. "In addition to the revamped architecture, new required build processes, and upgrade difficulties, almost every Drupal site has to completely rewrite its theme. And for many of the sites I've built and worked on, this is probably where the majority of the effort would need to happen."

2. Transferring content isn't a simple process

Because of custom modules, developers may need to shift data-structure paradigms and workflows, adding significantly to development time and costs. "You have to meticulously frame your business strategies to make content migration worth your while," writes Shankar Iyer of Opensense Labs. "It is important that developers and editors consolidate their work to simplify the migration." Both parties will have to minimize changes to the information architecture and navigation structure of public-facing sites.

3. No automatic upgrade path for Views (and other core modules)

Put simply, the Views module is a user interface to compose SQL-queries, pulling information (whether it's content or users, etc.) from the database and displaying it to the user in the desired format. However, even years after release, the Views module doesn't have an automatic upgrade path in Drupal's core. This means you will need to manually recreate its views on your Drupal 8 site, despite the module being the third-most installed module for Drupal, after Core.

> ***"You have to meticulously frame your business strategies to make content migration worth your while."***
>
> — Shankar Iyer, Opensense Labs

4. Most custom modules will have to be rewritten

Drupal 8 is object-oriented, where Drupal 7 was primarily procedural. Instead of relying on hook-oriented paradigm and procedural programming as it had in the past, Drupal chose to apply object-oriented methodologies and a new framework called Symfony. This decision affected almost all Drupal's main components — from core functionality to its template engine. This also means that a lot of code will now be in classes rather than simple functions. The end result? Most custom modules will have to be rewritten in the Symfony environment — a huge challenge. Everyone considering Drupal should understand that while Drupal's upgrade path will reliably preserve your data, there is no backward compatibility with previous Drupal code. Since there's no backward compatibility in modules, every single custom module has to be rebuilt.

5. Many contributed modules lack a complete upgrade path

According to developers, porting code to Drupal 8 is a big deal. There are so many changes that many are even tempted to rewrite code from scratch. This is exacerbated by poor

documentation for a lot of Drupal 8 — most of which is much, much worse for contributed modules. Developers often resort to reading source code in order to figure out how things are supposed to work because there is no documentation and no examples.

Moreover, upgrade paths are still experimental and some contributed modules might not have a complete path. If contributed modules provide a path, data stored by a previous version will be migrated to Drupal 8; however, if a Drupal 8 port is not available, then functionality has to be built or the module has to be ported.

Owing to the challenges above, the migration of themes, modules and plugins to Drupal 8 has been particularly slow. Much to the disappointment of Drupal's community, even three years after its initial release, Drupal 8 still does not offer a wide range of plugins. Instead, developers are forced to write custom modules from scratch to implement functionality that was more easily deployed in earlier versions.

Drupal's value proposition for smaller organizations has changed

Drupal has been a go-to CMS of large enterprises for a long time, thanks to its scalability and flexibility. As the Enterprise Content Management (ECM) market is set to grow to US$94.6 billion by 2024, Drupal has stated that it will be positioning itself to own a segment of that market, leaning on its powerful information architecture, multilingual capabilities, and more to secure it.

> ***"Drupal was once an 800-pound, open-source CMS gorilla that has since become an 800-pound monkey on our back... squishing our productivity and squeezing our bottom line — and Drupal 8 hasn't made things better."***
>
> —Paul Vetch, Strategy Director, Torchbox

This marks a major departure from Drupal's roots. Back in 2011, Drupal was the "safe" choice for smaller organizations and nonprofits. "No one ever got fired for choosing Drupal" says Paul Vetch. However, according to Vetch, once an "800-pound, open-source CMS gorilla," Drupal has since become an "800-pound monkey on our back, squishing our productivity and squeezing our bottom line — and Drupal 8 hasn't made things better."

It's clear—Drupal's headed upmarket

Many used to believe Drupal was a fantastic option for nonprofit organizations with unique needs and limited budgets — BiblioCommons even used Drupal as the backbone of BiblioWeb back in 2010 (we've since made the switch to WordPress and haven't looked back.) With the introduction of Drupal 8, that trend has shifted. In 2016, at DrupalCon New Orleans, Drupal's founder and project lead Dries Buytaert explained that "we're more about big sites and less about small sites". Elsewhere he's been quoted as saying: "I see us as being the next large open source business model to reach $1 billion in revenue, like Red Hat. We're on the IPO track — even though it's still early days, but we are getting ready."

> ***"We're more about big sites and less about small sites... I see us as being the next large open source business model to reach $1 billion in revenue, like Red Hat. We're on the IPO track — even though it's still early days, but we are getting ready."***
>
> —Dries Buytaert, Founder & Project Lead, Drupal

So Drupal's headed upmarket — a move precipitated by its commercial ecosystem.

"You used to be able to find lots of freelancers and small shops who were interested in working with smaller organizations on smaller projects," writes long-time Drupal developer and community member David Snopek. "However, many of the freelancers I know have gone on to work at big shops and many of the small shops have grown or merged with others. And they are looking for big projects." And this doesn't just affect Drupal 8 — it's now harder for smaller organizations with limited budgets to find help with Drupal 7, and much harder for any small nonprofit who was successful with Drupal 6 to move to Drupal 8.

Bottom line: Drupal is increasingly moving to the enterprise space, making its value proposition increasingly questionable for nonprofit organizations.

Drupal 8's major rewrite — new, shiny, and unproven

The upcoming Drupal 8 release represents a significant rewrite of the software and a major change in the architecture with a move to the Symfony framework. In the words of Petr Palas, Founder & CEO of Kentico Software: "All that proven Drupal 7 code is heading towards the waste-bin." The Drupal community throws away all of that existing tested and proven code and replaces it with new, shiny, unproven code.

Palas goes on to say:

> "As we all know, all too well, no amount of testing can replace the crucible of real-world use. Inevitably, customers struggle with issues of stability, security, and performance as well as user experience in the early months."

And it doesn't stop there. Another downside of a major change is that implementation best practices change when the architecture does. As a result, many experienced Drupal 7 developers need to relearn key concepts, and experiment and learn from experience to determine best practices.

> ***"All that proven Drupal 7 code is heading towards the waste-bin."***
>
> —Petr Palas, Founder & CEO, Kentico Software

Object-oriented programming presents a steep learning curve

In introducing Drupal 8, Buytaert conceded that it comes with a steep learning curve:

> "The advantages and disadvantages of object-oriented programming are well-understood. The disadvantages are size, verbosity, the amount of work it takes to write (including the design planning that goes into it) and slower performance. For people new to object-oriented programming there may be a steep learning curve; some of the key programming techniques, such as inheritance and polymorphism, can be challenging initially."

He claims that sacrificing ease of use is necessary in order to create code that will prove "more maintainable, more modular, and more accessible to non-Drupal developers." And yet, this means many smaller Drupal agencies are shifting gears and dropping Drupal altogether.

Why are smaller agencies dropping Drupal?

Many developers maintain that Drupal is altogether too slow to develop with, too hard to use and maintain, and therefore too expensive for nonprofits that need maximum

functionality, at minimum cost. On the Torchbox website, Vetch breaks down specific reasons why they as an agency have abandoned Drupal:

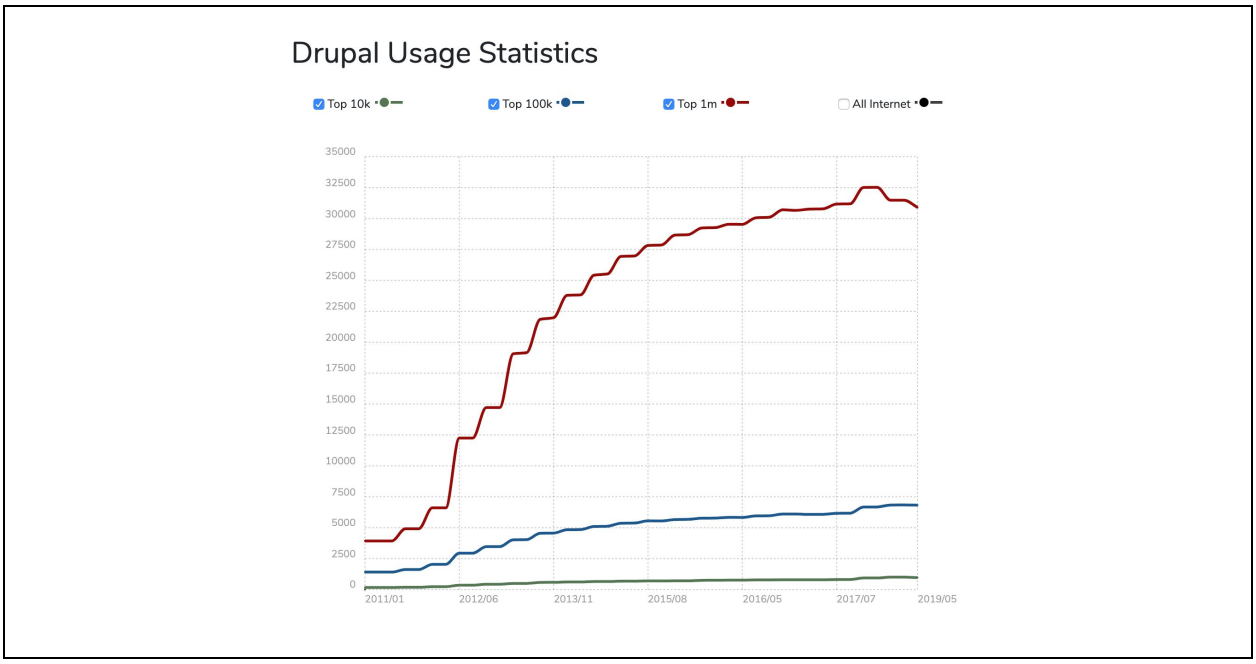
> **"The module ecosystem is a mixed blessing."**
>
> "While they can introduce efficiencies, all too often contributed modules won't fully address the intended use case, and instead add to the maintenance overhead. Compared with Drupal 7, there are 50% fewer modules actively maintained for Drupal 8. That's a significant difference."
>
> **"It's opinionated about markup."**
>
> "Even with the improvements that Twig has brought to the table," writes Vetch, "you still need to be a specialist Drupal frontender to be an effective themer. Just like in 2011."
>
> **"An even steeper developer learning curve"**
>
> According to Vetch, Drupal 8 has a steep learning curve with its adoption of Symfony. Moreover, it requires special care with long-term support and maintenance.
>
> **"The UI is still complicated"**
>
> Despite a significant push to improve usability in Drupal 8, the platform still requires agencies to train users to do basic content management tasks, or expend large amounts of effort customizing the interface to make it user-friendly.

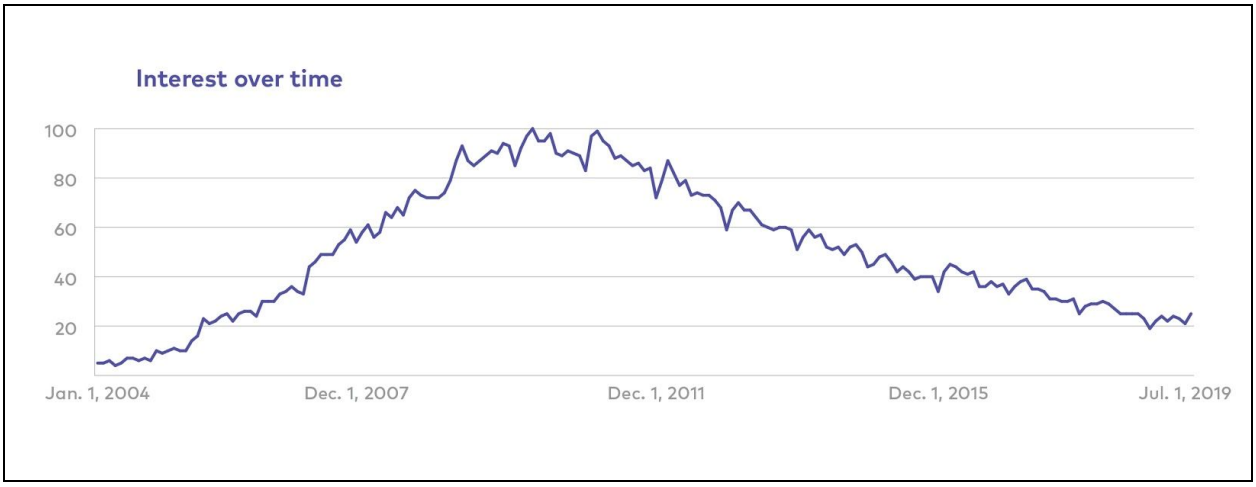
In short, Drupal 8 brings with it added complexity, without removing any of the complexity that existed in previous versions.

Drupal 8's non-existent auto updates means maintenance is increasingly complex.

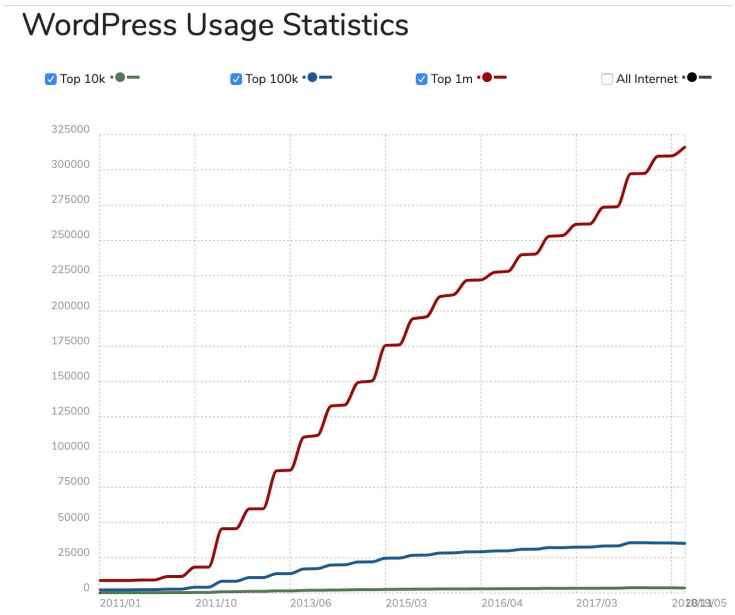
Overall trends are not on Drupal's side

On the surface, Drupal trends appear static, or in a slight decline. The truth is, Drupal is declining in usage by almost any metric. Since Drupal 8 was released in late 2015, Drupal's overall use has stalled at around 1.2 million websites. DrupalCon, Drupal's hallmark international event centered on the use of the platform, attendance peaked in 2014, and has been declining since. Drupal Core downloads have steadily decreased since 2015, and Google search trends reveal that Drupal has hit its lowest ranking since November 2005.





In contrast, WordPress trends have been steadily positive.



Source: https://trends.builtwith.com/cms/WordPress

Get a list of 1,917,404 websites using Drupal which includes location information, hosting data, contact details, 614,752 currently live websites and an additional 1,169,290 domains that redirect to sites in this list. 1,302,652 sites that used this technology previously and 14,939 websites in Canada currently using Drupal.

Site Totals

Total Live	614,752
1,169,290 additional website redirects[?].	
Canadian Live Sites	14,939
Live and Historical	1,917,404
Top 1m	3.17% 31,677
Top 100k	6.96% 6,964
Top 10k	10.05% 1,005

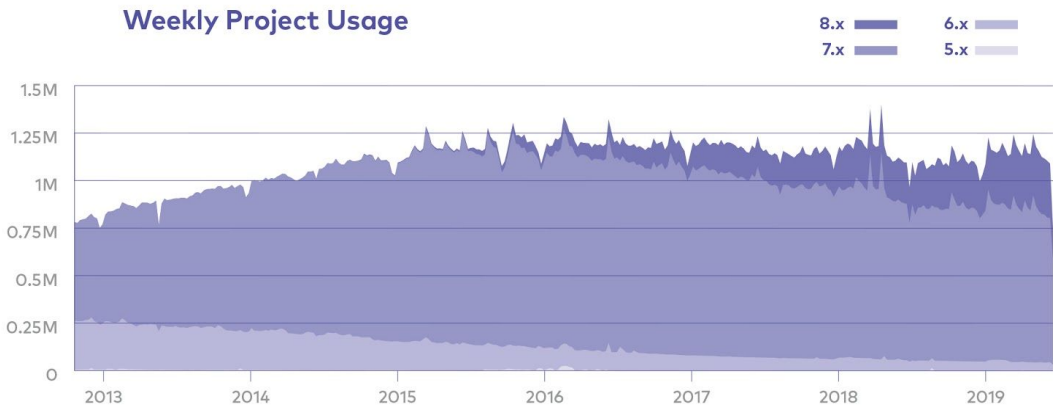
We know of at least 24,808,989 live websites using WordPress.

Site Totals

Total Live	24,808,989
2,353,161 additional website redirects[?].	
Canadian Live Sites	166,646 Estimated
Top 1m	32.83% 328,299
Top 100k	35.98% 35,981
Top 10k	37.75% 3,775

https://trends.builtwith.com/cms/Shopify

What's most telling is that over 77% of websites on Drupal haven't yet migrated to version 8. In fact, Drupal 8 adoption has still not hit a level of growth which will put it ahead of Drupal 7 any time soon — despite being released in 2015.



This graph showcases how much each version of Drupal has been in use from 2013 - 2019. Source: https://www.drupal.org/project/usage/drupal

There's growing concern about Drupal's ecosystem

It's no surprise then that there is a growing concern about the health and sustainability of the overall Drupal ecosystem, a concern that is reflected in poor documentation. Since the Drupal developer community is much smaller than that of other CMSs like WordPress, fewer people are contributing to shared knowledge, which means fewer solutions to uncommon problems and slow updates to base documentation.

> ***"[Drupal 8 documentation] is a mess of unfinished pages without a clear structure."***
>
> —Dogerthat, Reddit User

One Reddit user complained "[Drupal 8 documentation] is a mess of unfinished pages without a clear structure" and another agreed: "Drupal 8 documentation is rough. I can't seem to find clear directions to accomplish something."

So, what's next for libraries?

There are also significant changes in the technology landscape surrounding Drupal, as a rising number of former Drupal agencies have started to adopt other technologies to address the needs of smaller organizations like libraries that don't need to scale at the level of enterprise companies. These moves are driven by two primary needs:

- **Site building tools.** Libraries require powerful site building tools that are nonetheless simple to learn, and don't require dozens of contributed modules to be installed and configured in order to keep implementation costs down. They'd also prefer to avoid writing a lot of custom code because of limited budgets.
- **Easier updates and maintenance.** Libraries would benefit tremendously from auto-updates because maintaining and updating their Drupal 8 sites can be too manual, too complex and too expensive. Site updates have often become more complex for libraries because of their dependency on third-party libraries and having to juggle ad-hoc updates from contributed modules.

There are hundreds of solutions to choose from. However, they all have their own strengths and weaknesses. Choosing a web platform depends on the use case. Here are a few alternatives that have proved popular among organizations seeking Drupal alternatives:

Static Site Generators

Open source web development technology is not limited to the CMS. A new breed of website creation tool (like Jekyll and Middleman) is gaining popularity as easy-to-use solutions for quickly creating responsive websites. Typically, content is created and stored in text files and compiled into a static site for the server. They usually don't come with a user-friendly admin interface, raising the technical bar. But for developers, they offer significant value compared with a traditional CMS, as they are often easier to develop on, which translates into less of an expense to maintain.

Backdrop CMS

Some developers didn't like the changes introduced with Drupal 8. So much so that two well-known people in the Drupal community — Jen and Nate Lampton — "forked" Drupal, gave it a new website, a new contributors' platform, and then took the platform in a completely different direction, which was perceived as a threat by many Drupalists. Backdrop will be familiar to people with Drupal experience because it is so similar to Drupal, but includes numerous differences in usability and features.

Wagtail

Wagtail is an open source CMS written in Python and built on the Django framework. Built by developers for developers, it offers a fast interface for editors where content can be

created and structured intuitively. Kevin Howbrook, a former Drupal developer, claims that the 'Drupal site builder' role will become more and more obsolete as it becomes best practice to create functionality in code versus using hundreds of modules. That's why he foresees Drupal being supplanted by CMSs like Wagtail in certain cases. He writes: "Why not switch to something that's not only already doing that, but has been doing it for a long time?"

Headless CMSs

Drupal and WordPress are both traditionally "monolithic" CMSs, with presentation baked in via the theme. That means your website must be built "on top" of the CMS; to implement them, you will need to learn and (re)build your website based on CMS rules and processes. Due to the need for more flexibility and freedom, however, many developers have begun decoupling the CMS, using it for content management, editorial, and administrative tools, while implementing a separate frontend component dedicated to the user experience which communicates with the CMS via a web API. This allows organizations to add CMS functionality where they need it in their existing tech stack. This way, the CMS is integrated rather than foundational.

Wordpress

With approximately 60% of websites (like *TechCrunch*, *Walt Disney*, even *The New Yorker*) using WordPress as their CMS, WordPress boasts a massive community. With a larger community comes more people contributing to documentation, feedback, and ideas, making development on WordPress much more rapid. Additionally, the WordPress ecosystem is huge. From its community to its range of plug-ins, WordPress offers customization at significantly less effort than other platforms. And unlike some other solutions, WordPress supports automatic updates, meaning far fewer upgrade- and migration-induced headaches.

Do-it-yourself

In order for a built in-house solution to serve you well, it must be feature rich, flexible, extensible and powerful, as well as integrate with other services. And, of course, you'll need to select solid open source tech with good APIs and useful documentation. But that doesn't mean connecting them is going to be fast or easy. Libraries spend a significant amount of time and expense integrating all the parts and pieces and supporting them with a ton of custom code. Everything will work at the launch of the project, but the investment required to maintain it will grow all too quickly.

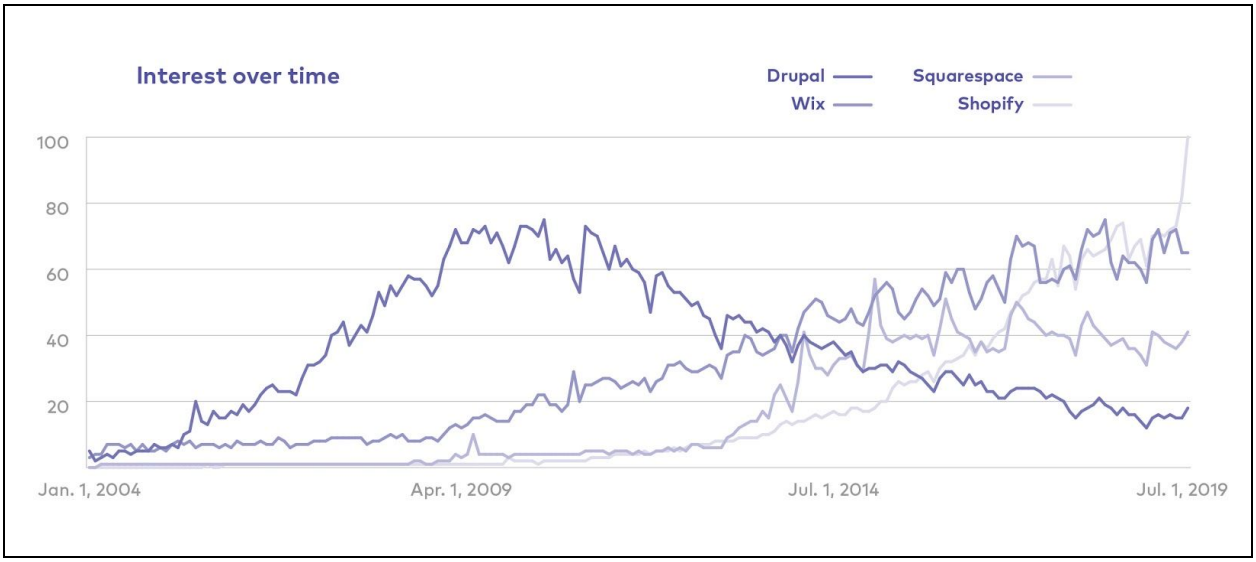
Software as a Service

SaaS, or Software as a Service, solutions are becoming increasingly more common among libraries due to the many benefits for both software providers and customers. SaaS solutions are, simply put, significantly easier to administer, update, support and expand, and offer extensive flexibility and continued support. SaaS companies are dedicated to evolving

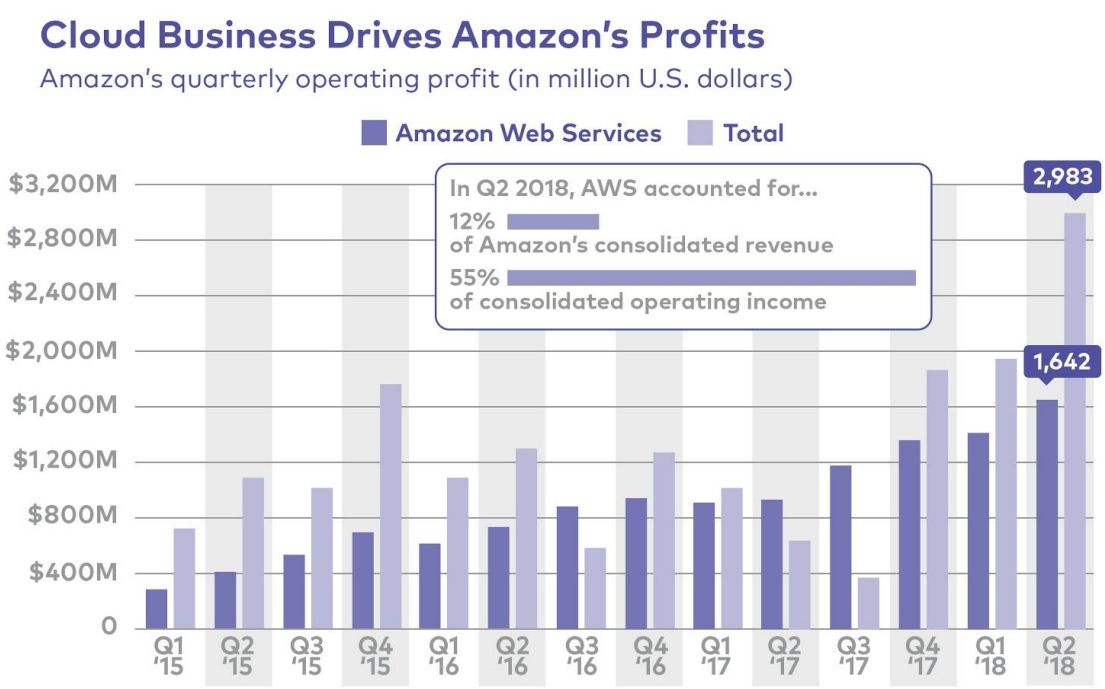
and improving their product over the long term, which customers then benefit from, typically at no extra cost.

SaaS — the rising tide that lifts all boats

SaaS is a model in which the software is licensed and the solution is accessed through the internet. Customers do not install or download any local software; instead, the vendor is responsible for the security, uptime and updates. Over the last 10 years, the SaaS market has grown dramatically, far outpacing that of CMSs like Drupal: it is anticipated to grow at a compound annual growth rate of 21.2% over the next four years, and currently generates roughly $20 billion in quarterly revenue.



The cloud market, which makes SaaS possible, is expected to grow by 17.3% in 2019. Even Amazon, which is known to most as an e-commerce business, looks to it's B2B cloud SaaS service Amazon Web Services (AWS) for 55% of its operating revenue.



Source: https://omoto.io/academy/saas-in-2018/

The reason SaaS providers have seen so much success is that the great ones have invested time and resources into deeply understanding their market, and perfecting their product and the service they deliver. This is particularly true for vertical SaaS companies.

What is vertical SaaS?

Finding a SaaS partner that perfectly suits your needs has become even easier thanks to the rise of vertical SaaS. Vertical SaaS solutions are those that are optimized for a particular industry's needs and workflows. Vertical SaaS providers are cropping up across all industries, most notably healthcare, construction, finance, and cannabis, as these are massive industries that have very particular needs that many horizontal or industry-agnostic SaaS providers aren't able to fully understand or serve.

The *Washington Post*'s Arc Publishing platform is a great example of a vertical SaaS solution that was built specifically to address the needs of digital publishers, and has been utilized by the *New Zealand Herald*, the *Boston Globe*, and the *Chicago Tribune*. Veeva Systems provides cloud-based CRM and content management to the life sciences industry.

BiblioCommons, for example delivers vertical SaaS solutions across web, catalog, events and more to public libraries, whose complexity is often underestimated.

Why is vertical SaaS perfect for libraries?

Public libraries don't compete with each other. Instead, they're vying for their respective communities' time, attention, and loyalty in an era of television streaming, social media, podcasts, and other sources for inexpensive or free content. For this reason, public libraries are well suited to adopting a vertical SaaS solution, given that neighbouring county libraries can use the same service, without fear of competition. In this case, a rising tide lifts all boats: by partnering with large, metropolitan libraries that have the funds to invest in our SaaS product, (like the Chicago Public Library, Boston Public Library, and King County Library System) BiblioCommons has been able to innovate and develop stronger products, which in turn benefits smaller libraries with limited budgets. Of course, the general trend of moving to the cloud brings with it other opportunities and benefits, some of which we explore below.

Continuous product improvements

The high costs of custom code force most libraries to make punctuated software improvements once a year or every few years, depending on the library's size. Popular SaaS platforms, on the other hand, deliver frequent updates to stay current with the needs of their customers. Libraries looking for a schedule of continuous improvement should look to SaaS — Salesforce, for example, undergoes scheduled maintenance twice per month.

Better integrations with other solutions

When it comes to integrating with other solutions, custom software has a slight advantage over SaaS — but it comes at significant expense. You can have custom code built to integrate with any third-party solution that your library uses. Realistically, though, few libraries will find that custom software integrates better than SaaS options. Today's landscape is dominated by application programming interfaces (API) that let them work in coordination with other products. If a developer indicates that a software's API can work with your other software, then integration should occur pretty easily.

Opportunities for social coding and sharing

For libraries, a social context runs to their core. Indeed, it's part of the process of discovery and engagement with their collections and fundamental to their mission. SaaS software offers libraries the opportunity to partake in a public digital space where communities of librarians, readers and learners help each other discover and explore the ideas, information and stories that are the public library's collections and services.

Faster speed of implementation

Building custom software can take months or years of work depending on the size of the library. After finalizing software, you still need to install it and train your staff how to use it correctly. You could easily wait a year or longer before you implement custom software. On the other hand, SaaS software offers a much faster option. Since the software has already been built, you don't have to wait for a team to build it. Even if the software needs a few tweaks to fit your business's needs, you can expect to start using it within weeks or months.

Sharing reduces cost

Even if going the DIY or self-managed route sounds like the best option for your library, the high price of retaining a software engineer could make you change your mind. You can choose to hire a software development company to do the work, but you shouldn't expect the price to fall by a considerable amount. SaaS companies charge less for their services because they distribute costs across their customers. Instead of spending hundreds of thousands of dollars all at once, you may only spend a few thousand each month.

Public libraries' limited budgets make custom integrations not only time consuming, but also financially risky. Often the scope and price of a custom project increases dramatically once it's already underway, leaving libraries vulnerable to either blowing their budget, or ending up with an incomplete product. By contrast, the number one job of a vertical SaaS product is to know which integrations are necessary for that industry from the get-go, and to complete them efficiently to deliver an out of the box experience — with no surprises.

Is now the time to adopt a cost-effective, library-focused solution?

Every library has a unique spirit that reflects the staff and the community they serve. However, the primary function of a public library remains more or less the same across the board: to provide access to knowledge, information, and services to support their community. It would be an unnecessary waste of limited resources for each library to build its website from scratch or self-manage a platform when SaaS providers can support customized websites that reflect each library's unique brand, voice, collections and services.

BiblioWeb has evolved over eight years to allow each library to represent their unique flavor through their content, voice, and branding. We take care of the underlying platform security, integrations, and operations, enabling any public library, regardless of size or staffing, to serve its purpose with excellence. As organizations consider alternatives to Drupal to manage their websites, BiblioCommons hopes that more public libraries will consider BiblioWeb as a cost-effective, library-focused solution to supporting and enhancing the library's online customer experience. //

References:

1. (2018) *Why is Drupal now the second most hated platform behind Sharepoint?* Retrieved June 26, 2019 from https://www.reddit.com/r/drupal/comments/84u23f/why_is_drupal_now_the_second_most_hated_platform/
2. Boucher, Stephen. (2019, May 12). *Why Drupal matters.* Retrieved July 3, 2019 from https://obj.ca/article/why-drupal-matters
3. Buytaert, Dries. (2006, May 17). *Backward compatibility.* Retrieved June 28, 2019 from https://dri.es/backward-compatibility
4. Buytaert, Dries. *Locations and Attendance.* Retrieved June 28, 2019 from https://www.drupal.org/association/drupalcon/locations
5. Buytaert, Dries. *Usage statistics for Drupal core.* Retrieved July 3, 2019 from https://www.drupal.org/project/usage/drupal
6. *CVSS Scores For Drupal Drupal Between 2018-01-01 and 2019-01-01.* Retrieved June 28, 2019 from https://www.cvedetails.com/cvss-score-charts.php?fromform=1&vendor_id=&product_id=2387&startdate=2018-01-01&enddate=2019-01-01
7. Editorial Staff. (2019, May 8). *The History of WordPress from 2003 – 2019 (with Screenshots).* Retrieved June 27, 2019 from https://www.wpbeginner.com/news/the-history-of-wordpress/
8. *Google Trends: drupal search term.* Retrieved June 27, 2019 from https://trends.google.com/trends/explore?date=all&q=drupal
9. Kabir Aion, Mainul. (2019, June 21). *Top Websites Using WooCommerce for eCommerce.* Retrieved June 26, 2019 from https://wedevs.com/89524/top-websites-using-woocommerce/?utm_source=blog&utm_medium=hyperlink&utm_campaign=best-ecommecre-platform&utm_content=5
10. Koontz, Christie and Barbara Gubbin. (2010, July). *IFLA Public Library Service Guidelines.* Retrieved June 27, 2019 from https://www.degruyter.com/viewbooktoc/product/43971?rskey=G0DzHu
11. Lansbury, Owen. (2017, November 20). *Have We Reached Peak Drupal?* Retrieved from June 27, 2019 from https://www.previousnext.com.au/blog/have-we-reached-peak-drupal
12. Locke, John. (2011, October 14). *Top 6 reasons Drupal really sucks -- Developer Edition.* Retrieved from June 27, 2019 from https://www.freelock.com/blog/john-locke/2011-10/top-6-reasons-drupal-really-sucks-developer-edition
13. Olivas, Jesus. (2014, May 21). *Why is Symfony in Drupal 8, and how Does that Change Things?* Retrieved from June 27, 2019 from

https://ffwagency.com/learning/blog/why-symfony-drupal-8-and-how-does-change-things
14. Pushpathadam, Tom. *WordPress vs SaaS: The Case for SaaS Website Builders.* Retrieved June 27, 2019 from https://www.verycleverdesign.com/blog/wordpress-vs-saas
15. Spilotro, Chloe. (2018, May 23). *Drupal 8 Migration: There Has To Be Another Way.* Retrieved June 27, 2019 from https://www.zesty.io/mindshare/marketing-technology/drupal-8-migration-there-has-to-be-another-way/
16. *Usage of content management systems.* Retrieved June 26, 2019 from https://w3techs.com/technologies/overview/content_management/all/
17. Valley Voices. (2019, March 1). *Vertical Software Is Growing Up: The 10 Things You Need To Know*. Retrieved June 26, 2019 from https://www.forbes.com/sites/valleyvoices/2019/03/01/vertical-software-10-things-to-know/#fa33e3c24359
18. Vivek. (2019, January 2). *How did the SaaS market perform in 2018?* Retrieved June 26, 2019 from https://omoto.io/academy/saas-in-2018/
19. Widmer, Gerry. (2018, August 1). *What is Drupal? Is Drupal 8 Right for My Business?* Retrieved from June 27, 2019 from https://www.zesty.io/mindshare/marketing-technology/what-is-drupal--is-drupal-8-right-for-my-business-/
20. *WooCommerce.* Retrieved June 26, 2019 from https://en-ca.wordpress.org/plugins/woocommerce/