UNIVERSITY OF HELSINKI

Web Content Management Software

Usability & Performance

Seppo Hannu Ilari Sinisalo
Master of Science Thesis

Helsingin YLIOPISTO – HELSINGFORS UNIVERSITET – UNIVERSITY OF HELSINKI

Tiedekunta/Osasto Faculty of Science	Laitos – Institution Department of Computer Science

Tekijä – Författare
Seppo Hannu Ilari Sinisalo

Työn nimi – Arbetets titel
Web content management software: Usability & Performance

Oppiaine – Läroämne
Computer Science

Työn laji – Arbetets art M. Sc. Thesis	Aika – Datum 5.1.2018	Sivumäärä – Sidoantal 44

Tiivistelmä – Referat

Welcome, in this thesis, some of the higher ranked, popular web content management software (CMS), namely Drupal, WordPress, Joomla and Plone, are compared by usability, from a developer's perspective, and by performance of the resultant site build with these CMSs, to find out, among other topics, about their potential to build websites to different needs. This thesis tries to discover if a CMS exists, in this selected group, that is a clear choice above the others in both usability and performance. A substantial portion of source material for this research comes from measurements, and small demo systems built and used, in addition to any literature sources used and experience garnered from career as a web developer. In this thesis, we provide an overview of these four selected CMS; their characteristics, statistics and how they measure up to each other. And so doing, expand upon the still narrow research done in this field.

T.

T.

CCS-classification:
Human-centered computing > **HCI**
Information systems > **Data management systems**
Information systems > *World Wide Web* > **Web applications**

Avainsanat – Nyckelord
CMS, content management system, Drupal, Joomla, Plone, WordPress, usability, performance

Säilytyspaikka – Förvaringställe
Kumpula Science Library

Muita tietoja – Övriga uppgifter

Table of contents

Glossary

In this section, some of the more ambiguous terms used in this thesis are explained.

CMS - Abbreviation for Web content management system.

Template - In a CMS, a template refers to a document that defines a web pages html structure and appearance.

Theme - It’s a site template that defines how a site looks, the name differentiates depending on the CMS.

SEO – Search engine optimization. The purpose of SEO is to make a website to appear more regularly and with higher rank in search engine searches.

Usability - In this thesis usability is an umbrella term housing a host of other usability related philosophies like human-computer interaction, computer-human interaction, man-machine interaction, user performance, ease of use and other such schools of usability science.

1 Introduction

Nowadays the purpose of a website is to display up-to-date information and relevant content for its targeted audience. This is even more so for companies that need to adapt to a rapidly changing business landscape with ad hoc price changes while accurately reflecting their position and products on their website. It's preferred for such sites to also take growth of the business into account in the form of expanding product catalogues, promotion events and support for different forms of media, and answer these needs with scalability. This is where a web content management system, abbreviated to CMS for rest of the thesis, steps to the fore with its structured data and content types combined with modular, reusable elements and open source support in the form of free libraries and components [Pol12]. This thesis focuses on comparing different open source CMS platforms by their usability against their performance.

In website production, the relatively new holistic approach to content management where a framework software with linked database is used to build up a whole website from a highly customizable template, is starting to be the minimum standard for any site with the need to keep its contents up to date, which is to say, almost any site at all. To keep the costs down, the content manager responsible for their site needs the power to keep the content up-to-date themselves with relative ease. But that in mind the website production process with any CMS can get unwieldly, when it should be as lightweight as possible. This can be achieved by not having a lot of overhead on top of the normal development costs by including valuable usability features to help utilize any functionalities developed.

This thesis attempts to shed some light on the capabilities of different CMSs by comparing their usability from the developer's point of view and the performance of CMS built websites. This comparison is performed with data gathered from relevant papers in the field [PRD12]. These two attributes are chosen because they are equally important requirements for almost any successful website. The usability of a CMS is one of its most important attributes as it directly affects the quality of a resulting website and the speed with which it is produced. Therefore, CMS usability is researched from the developer's perspective. Usability for a developer can be measured in terms of how easy and quick a system is to use and learn. On the other hand, the importance of good and stable performance cannot be understated, as it is needed to constantly and consistently service as many visitors as possible in a satisfactory manner. If a website is found underperforming, which is quickly noticed even by a novice to website development, the bad performance quickly overshadows any gains in usability. One perspective to evaluating the performance of this thesis is to find out from live websites, that how well they perform in page load time measuring tests. The usability of these CMSs is also evaluated in terms of how they measure up to usability standards in observing methodology and guidelines, that are gathered from usability papers and my own experience from my job as a frontend developer with Drupal CMS platform. [BM01]

This thesis is paced as follows: first we introduce the reasons and research principles for this thesis. Then we present the four CMSs under inspection in this thesis and how they function. After that we find out about usability in CMS, how it affects user performance, and how usable are the CMS compared. Then we measure website performance and compare the results with other papers on the field with similar measurements. Finally, we list resources used for this work.

2 Research goals and methods

In the following chapter, we explain the goals and methods for this thesis and discuss the reasons for choosing the scope and perspectives for this research. First, we present what items are left out of the scope of this thesis, then we go through research questions and methods in detail.

2.1 Framing

Quantitative and qualitative scoping is done by reducing the group of multiple different CMS to only four: Drupal, WordPress, Joomla and Plone. This framing is done based first on PageRank page which ranks pages based on google search statistics. In those rankings, all selected CMS scored 9 out of 10 points [Che17]. All these were also in the top ranks of an installation survey with at least 2500 different CMS. The survey consisted of two surveys and a brand familiarity measurement [MJS14]. All these CMSs are also open source projects. Open source CMS are rising in popularity and are clear market leaders on the web CMS segment.

Performance, usability and modifiability are the main attributes, when considering the most business-critical selling points of websites, as they together answer to the question: How quickly a user of variable degree of expertise can produce high performance sites [OBS08]. From these three, performance and usability are under the inspection lens, as modifiability can be considered a subcategory under usability and is discussed in chapters that examine usability together with other usability factors. We could have further specified the type of performance or usability and the situation in which they are measured, but to maintain the integrity of the research we decided against it. Other comparable characteristics would have been availability and security.

Performance is inspected from the perspective of a website produced with a CMS, as the performance needs for development and content management purposes are almost trivial for all CMSs in the frame of this thesis. On the other hand, usability is inspected from the developer’s perspective, as the usability of a website built with a CMS is completely in the hands of the website developer. This responsibility shift is possible through modifiability of every part of the end-product, so it follows that the browsing user experience for the consuming audience is not relevant to CMS usability

The focus is mainly on using CMS for simple web page production as arguably, that is the website type that is becoming the most common. The reason for this is that airy web design punctuates the main points of the site while visitors rarely want to spend time reading large text content but instead prefer their information served through light and efficient navigation structure and in easily consumed format such as short texts, videos and images. A small site is cost efficient when it comes to updating or completely changing the outlook of the site to match new trends. On the contrary, maintaining a large-scale e-commerce site is not really the forte of a CMS.

2.2 Research questions

In pursuit of the sweet spot between the efficiency of utilizing these frameworks and the performance of the end-product we find ourselves approaching the following questions:

- RQ1: From the perspective of usability and user performance for developers, what are the weak and strong points and use cases of each of the frameworks, in this thesis, on creating and maintaining dynamic web pages?
- RQ2: With website performance and developer usability in mind, is there a best of both worlds solution where we get highly usable and well performing software?

2.3 Research methods

The methods used for researching the questions RQ1 and RQ2 will be:

- RQ1 - analysis of findings in different papers in the field and small practical demo constructions with different CMS platforms and researching sites built with CMS against each other by measuring page load times.
- RQ2 – Analysis of papers concerning CMS usability and performance to arm

us with proven principles and tools to gauge how the CMS in this thesis compare in the fields of usability and performance.

Research is additionally done by comparing charts of website performances between papers and page load times from papers against the results of a performance measurement study done in this thesis. There is also analysis of CMS feature differences by using papers and web sources comparing the usability and performance features of the CMSs in question. This was done to find correlation and analyze possible differences. Lastly some of this research is weighted by my personal experience in website front-end development and Drupal CMS.

2.4 Forewords

The purpose of this thesis is to find out which CMS empowers developers with the best ability to build websites efficiently, and that also have competitive performance statistics. We attempt to achieve this by comparing the usability and performance of different CMS platforms against each other by splitting these two slightly overlapping high tier umbrella terms into smaller components and factors that can be evaluated, measured and compared.

My preliminary reading, software experience and knowledge would indicate that Drupal is best for medium to large sites, WordPress shines in small to medium sites and blogs and Joomla would be best suited for social networking sites.

3 CMS

A CMS could be described as a software tool, and as such, it offers an interface for managing website content with add, publish, edit and remove functions. A typical CMS is an online software written in some scripting language like PHP or Python and run on a server computer with installed webserver and database. The scripting language used and the set of compatible databases can vary between different CMS. [Dru17 user guide]

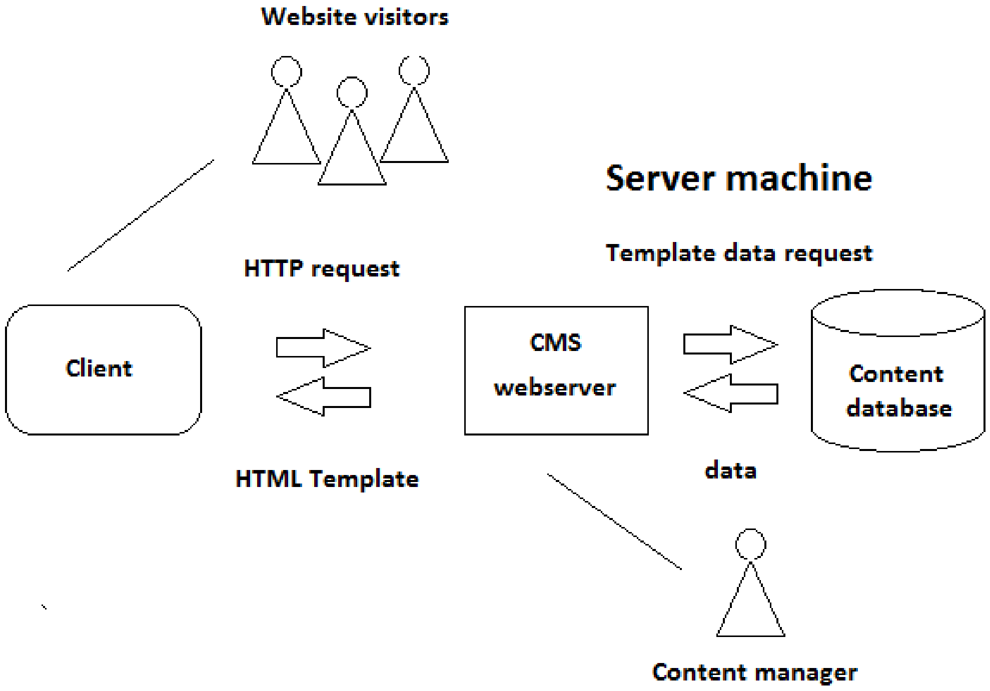
In the following chapter, we present the CMS selected for this thesis, then we discuss what CMSs generally are and how do they work. Analysis is done from the perspective of Drupal, WordPress, Joomla and Plone CMSs. This selection is based on arguments presented in chapter 2.2 Framing.

3.1 How do they work?

The common CMS architecture consists of web software and a database, with a HTML website front-end accessible from an online website. The software part serves the webpage and it appears as any other HTML website, but addition to that it includes an administrator portal or overlay that can be accessed either from the view served to general user or an admin URL and is usually behind admin credentials (Fig. 1). Different CMS platforms manage this differently. Some allow the user to edit the page from the same view they are browsing [Wor17]. Others require the user to create new content with customized templates or modules [Dru17, Plo17, Ope17]. In the administrator view the user can alter, add and remove the content shown on the page.

As with any hosted web page, when a user goes to an address of a page built on a CMS the request goes to the webserver, which serves the correct templates required to render the page as HTML and adding any content to it from the content database. Then the package is returned as a response to the browser which renders it for the user. [Plo17] When compared to a site without CMS, a site with one helps with keeping web content up-to-date by significantly reducing the time and effort that needs to be committed per instance when adding or changing content. For example, in a typical site, to change the content of a page with new images, text and adding a new component like a quote box, a developer needs to go to the HTML of the page and change all those items manually including content links, all the text spread out in multiple HTML tags and create or find a plugin implementation for the quote box and add it to the page. With a CMS, contrary to the previous scenario, any user unversed in programming or web development with only a little learning done to utilize the framework can log in to the admin-interface choose the correct page and content, then write the new text there, choose the new image to replace the old one and enable, the quote box component discussed earlier, from the content management options or download one from the open source community market if such plugin isn't installed yet. This can be accomplished from anywhere with an internet access and a browser.

Figure 1 - CMS overview



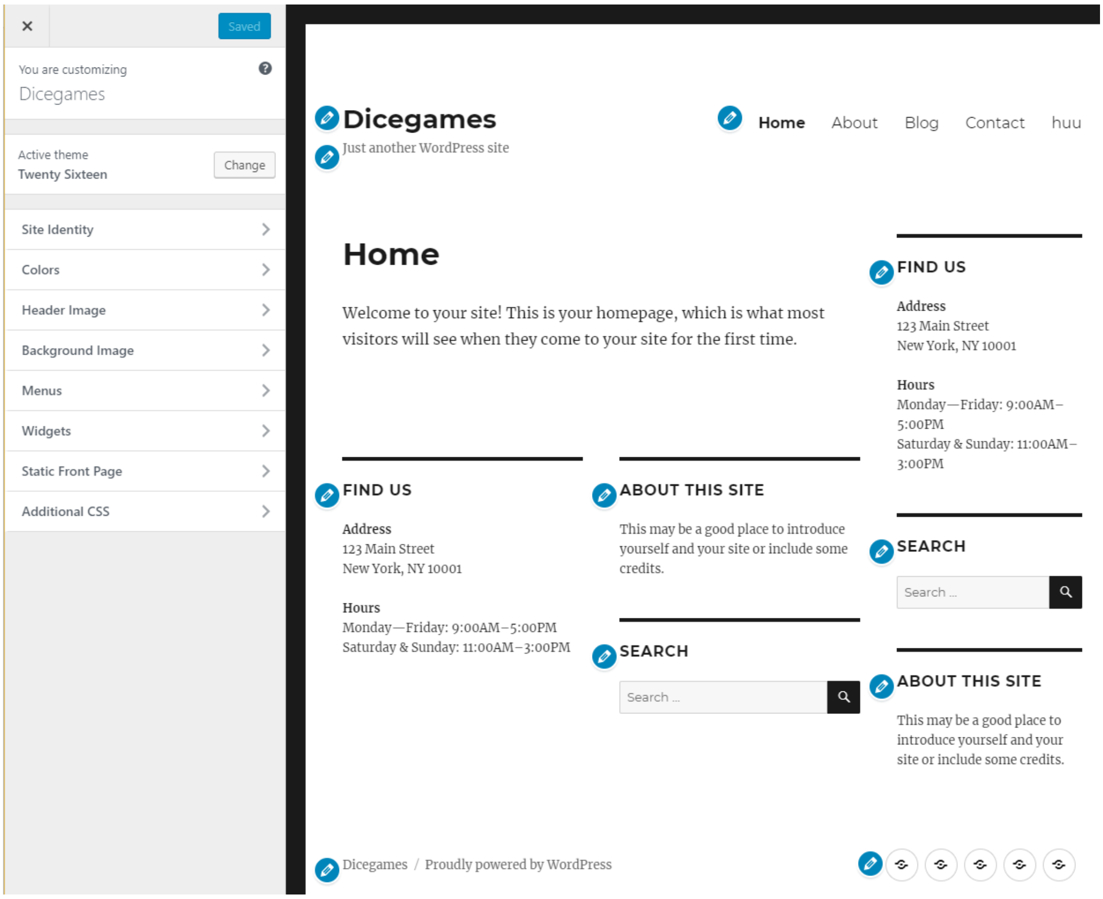
3.2 Website production with CMS

This chapter is based on information and findings gathered from the main webpages and documentation of the four CMS of this thesis Drupal, Plone, Joomla and WordPress and their respective webpages [Dru17], [Ope17], [Wor17] & [Plo17].

When starting a new project with a CMS there are several main areas to consider. There are multiple questions a developer encounters and needs to answer, they are: Which template and theme to use? How to accomplish the desired page in terms of what kind of visual and functional components are needed? Which parts of the site design repeat elsewhere in the site and should be developed as reusable components? Can those parts be found from a marketplace more cost efficiently? After that the problem is how to design the site navigation so that it’s simple to use but meets all the necessary requirements to display site content. After the site is configured and running changes and added features need to be tested. This testing consists mainly of visual testing which aims to verify that the site works and looks as it should. After the CMS is set up, the publishing is usually fast and simple, for example in Drupal the publishing to production is done by changing settings to production mode and then simply pressing the publish button. Setting up the website itself to be visible works the same way as for any other site.

The common core features of open source CMS provide management tools for content, media files, menus, user and group access permissions. Additionally: Version control, heavily customizable templates, localization support, content search tool, easy updates and flexible workflow are common features. CMS also usually boast a rather fast publish cycle where changes made appear to other users within seconds while allowing uninterrupted user experience.

Figure 2 - WordPress admin-ui overlay example



Welcome to your site! This is your homepage, which is what most visitors will see when they come to your site for the first time.

Address
123 Main Street
New York, NY 10001

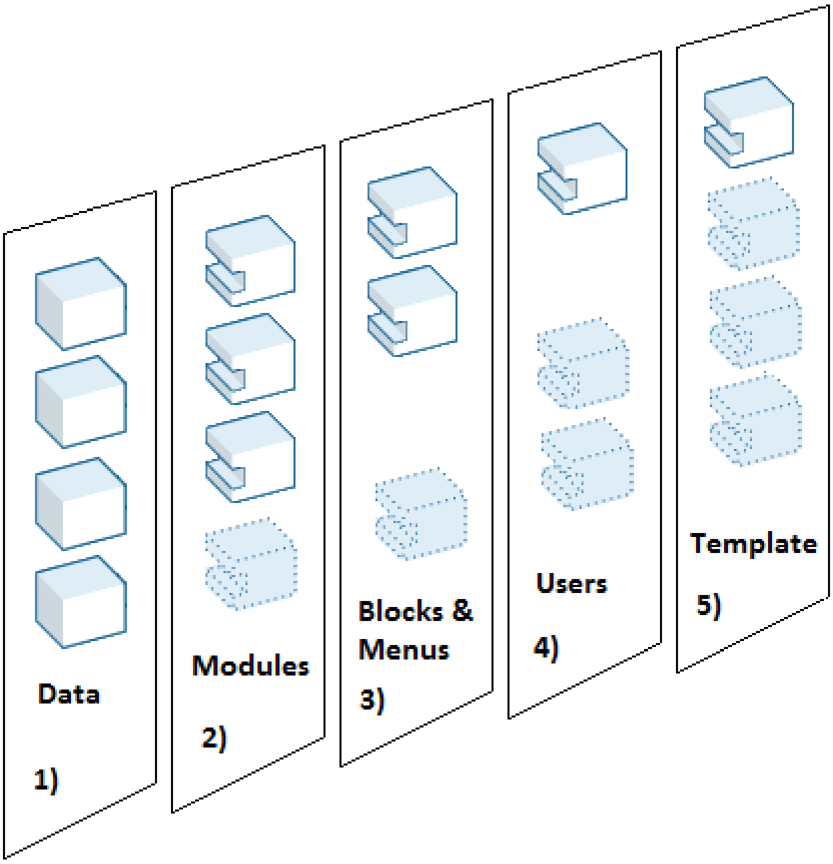
Hours
Monday—Friday: 9:00AM–5:00PM
Saturday & Sunday: 11:00AM–3:00PM

This may be a good place to introduce yourself and your site or include some credits.

Like discussed in the earlier chapter, CMSs have, in addition to the view shown to site visitors, an admin overlay or portal (Fig. 2) through which a content manager can edit the content on the site. When in this overlay mode, the page content and modules have edit links where the content can be edited on the fly. This admin view is for managing page content, navigation menu structures and access settings for the website.

CMS have multiple layers that are used in the creation process and throughout the lifecycle, they are somewhat alike but a general idea can be attained by looking at the Drupal version (Fig. 3). The topmost layer consists of templates (Themes in Drupal) that comprise of CSS stylesheets, any media files, XML, PHP (Twig & XHTML in Drupal). There is usually a user layer which defines the access and rights of logged in users. The site navigation and common structure is defined by menus as the links the menus provide are the usual way to move from one view to another. Other common layer is called Modules and extensions; they are small or large independent modular plugins, like a calendar, that can be added as is to a designated place in one of the templates mentioned earlier or it can be a search engine optimization tool or image and video gallery feature.

Figure 3 - Drupal workflow



3.3 Joomla

The Joomla CMS was chosen to be one of the technologies used to build a CMS website construct. The construct is installed with Microsoft WebMatrix 3.0 and run locally.

For me Joomla installed quickly with the website running in fifteen minutes. Joomla offers about 65 languages and it has localization features for multilingual sites out of the box. Joomla is beginner friendly with integrated help system on each page. Joomla also supports LESS, jQuery and its core templates are built with bootstrap [Ope17].

3.3.1 Joomla extensions

Joomla has five types of extensions that are used to build websites: Components, Modules, Plugins, Templates and Languages.

Components are pluggable mini applications that are on the far complex end of the Joomla extension types spectrum. Components can be responsible for things like login, form building or an e-commerce shop. They work at the front and admin backend, each triggered by a specific menu item.

Modules are usually small and flexible extensions that add some functionality or content to a page. These modules appear usually as content holders like a footer section on a typical page. Modules are assigned on menu item (as in page) basis. Modules can be linked to components: the “latest news” module, for example, links to the content component “com_content” and displays links to the newest content items. Modules can also function alone as standalone static HTML or text.

Plugins are essentially event handlers. They provide functions which are triggered by plugin events. Any extension can fire custom events in addition to the core plugin events. When a particular event occurs, all plugin functions of the type associated with the event are executed in sequence.

Templates are responsible for markup and contain the PHP needed to structure the content. There are templates for the front-end and the back-end of Joomla where front-end templated display content for users and back-end template control how management functions are viewed in the admin portal.

Translations are used to manage localization for a website. These plugins range from organizing your translation files to translating all your pages with automatic translation tools.

[Ope17]

3.3.2 Joomla workflow

When building a website with Joomla, the common workflow goes as follows, you need to have a theme for your site to have its own look and feel. The efficient way to go is to search for a suitable template from Joomla marketplace or any other site with Joomla templates on offer and then either purchase or get a free template. Download the template, then install and apply it through the Joomla extensions menu in the admin user interface to your site. Usually there are modules and plugins that come with the theme, upload those to Joomla with the template, then enable them from template manager, module manager and plugin manager so that the new theme starts working. In Joomla sites, almost everything dynamic present on the pages are modules. Modules need to be assigned a specific position from the template. There is usually a documentation about the features and a map of the module positions that are available in templates gained from theme and template marketplaces.

Customizing templates

Adding modules to the site is achieved through the menu manager where you need to create a new menu module with name, content type, position and the pages it is shown in. Joomla pages are called articles and they are accessed by a menu that is linked to them in the backend. To add menu items, first create menu modules with the type single article, then create an article in the article manager and link it to the menu identified by title. Last assign the menu items to the chosen menu in the menu manager [Ope17].

Adding modules to the articles

To add a module to a page, it needs to be enabled, assigned to a module position in the template of a selected theme and configured to appear in the pages wanted. The look can be usually customized from the backend with selected CSS parameters [Ope17].

If the pre-contained module positions aren’t desirable they can be customized with JDoc include statements. Then it needs to be set up in a template-details XML file which defines what stylesheets, template positions, JavaScript and media files are needed [XiYu10]. This kind of customization requires rather extensive programming knowledge and understanding of web-based technologies.

3.4 WordPress

WordPress has a fast and easy installation process when installed with web matrix, which took less than 20 min to download, install and fill a form to create a database. They advocate simple experience for end user without cluttering the software with needless options from the user's angle. The basic software has less customization features for niche use-cases. The philosophy behind this is that its core features are targeted towards the needs of 80% of the developer community. The remaining 20% of developer needs should be satisfied with third party plugins which are supported with sophisticated plugin system by WordPress. WordPress is also search engine optimization (SEO) friendly and it has tools to create responsive mobile sites [Wor17]

3.4.1 WordPress workflow

There are two ways to create WordPress sites, both of which recommend downloading an existing template on which to build upon. These can be templates from either WordPress template marketplaces or other users sharing templates to the web.

The first way is to create a site from scratch. WordPress custom themes need a new theme folder in your WordPress project folder and a stylesheet and an index file all which need to be activated with the new theme at the WordPress backend. WordPress core, contributed and Purchased extensions can be used to extend the theme project with added functionality.

The other way is using a site building module with drag 'n drop page construction with elements belonging to the theme, and then customizing the resulting site if needed. An experienced user can find and install just about any feature as a high-end plugin and set it up in few minutes, as long as one is happy with the default customization options that come with the theme. One such example is an online shop including product setup with price and description.

Hard customization, for example, adding a new column with new module positions to a theme, needs customization with PHP code in the form of WordPress action hooks that allow custom script execution at any part of the core code. The other way for this is to add the functionality as code directly to the themes component files. It’s also the point where the user needs to have programming skills and where most of the upsides of the easy customizability and site building with WordPress disappear [Wor17].

3.5 Drupal

Drupal styles itself as a collection of parts and tools that can be used to create any kind of website. Drupal is best for medium to large sites which can justify the overhead that comes with component oriented architecture with extensive potential to utilize reusability and modularity in its content types. Drupal is highly customizable but it comes with the price tag of steep learning curve. Drupal uses Twig templating language that is built over PHP which makes template customization much faster. Drupal is also SEO friendly by having all its content crawlable by web crawlers. Drupal is the CMS of choice for medium to large websites [Dru17].

The installation tools and process for Drupal is well documented but more cumbersome than for the other CMS presented in this chapter. Drupal requires you to install your own database and have that database server running before you start. Then you head to the browser and run an install script from the web installer. [Dru17]

3.5.1 Drupal workflow

Drupal essentially has its workflow split into several layers like in the picture (Picture 3). They are as follows:

1) Data: Anything that needs to be displayed on the site must be an input as data
2) Modules: Functional plugins built on core functionality that are part of Drupal core or contributions from the Drupal community. There are thousands of contributed Drupal modules in the Drupal module repository [Dru17].
3) Blocks & Menus: Blocks provide the output from a module, or can be created to display whatever is needed. They can be placed in various Regions in the Theme template layout.
4) User Permissions: Layer where users are assigned roles with different permissions to content.
5) Themes: Comprises of XHTML, CSS and Twig templates to establish the look

> and feel of the site and provides function hooks to allow control over how modules generate their markup at rendering. [Dru17]

There is a lot of configuration and defining to be done like setting up content types and structures before starting to use Drupal to manage content for your site. Publishing content in Drupal is simple and doesn't have the sophistication in its publish workflow that some of the other CMS do.

With Drupal, like in Joomla, if you want to have anything other than a generic web page which structure is completely designed and built by someone else you must customize the templates. There are powerful and effective tools for customization Drupal but they are not too easy to approach. Usually the fastest way is overriding either the default templates that you get right off the bat or finding a third-party template (themes in Drupal) and overriding those [Dru17]. A big downside to this way of customization that Drupal offers is that it requires serious experience and understanding about Drupal, its PHP twig templates, template nesting and overall programming skills.

3.6 Plone

Plone is python based unlike the other three CMS present in this thesis. For me Plone installation was a straightforward process. It requires to run an install.sh script on the downloaded package and it handles the setup from there. You can get the Plone download to start with 3 clicks from their site [Plo17]. It supports enterprise Integration with the ability to connect with CRMs, databases and continuous integration tools. Plone is used by CIA and FBI among other agencies because of its low number of vulnerabilities and Industrial strength security. Plone typically runs on the Python-based Zope application server.

From the start Plone feels ready to use right after installation and has predefined content types to begin with and a user can start to manage content quickly. Plone also offers more inbuilt core functionalities than Drupal for example, and to get these features in other CMSs' they need to be installed separately. A rather large downside is that finding and installing new addons and themes to Plone from the Python software foundation library is more ambiguous than it could be for new and old users alike.

Plone is also compliant with accessibility standards for motor- and sight-impaired individuals [Plo17].

3.6.1 Plone workflow

The site content manager is somewhat limited in terms of what kind of pages can be created right out of the box with Plone. For anything more advanced, a theme, like one in any other CMS talked in the CMS chapter, needs to be downloaded or created, which might come with additional content types. But to do any serious customization like changing theme colors, the user needs at least basic understanding of programming as these changes are done to code snippets. The best practice way to theme Plone is using a theming engine called Diazo. Theming is done by providing a theme file which is defined with only HTML, JavaScript and CSS files. This customization is not from the easiest end of things, but it's a lot more approachable than heavy customization of some of the other CMS. Plone themes provide the slots which the content types fill according to a set of rules defined in a rules file using XML syntax. In Plone pages, called views, including their contents are managed by content types. These are instantiable objects that have a certain look and they can be reused with different parameters. Each of these content types has a schema that describe a set of fields for each instance of the content type. Content type is usually associated with multiple fields and so access to it falls under user permissions management rules that concern those views.

Plone allows for hierarchical publishing with user management separated to content creators, reviewers and publishers if needed, where creators create articles, reviewers either submit it for publisher or send it back and publisher either publishes or rejects. Plone also has layers and adapters. Layers are basically for linking code and other modules to only work when certain conditions are met, like only including mobile code when the site is viewed via mobile device. Adapters are programming tools to adapt some functionality of a class with output similar to the input of some other module.

3.7 Summary

We found out that all four CMS discussed have a large quantity of similarities. They are accessed like ordinary websites by connecting to a server, but in addition to that they respond by serving extensible templates to which data is fetched from a content database. These CMS also have an admin user interface or overlay from which views can be managed by modifying, adding or removing views and content types, but is mainly used for editing and publishing content like text and images. The workflow differs from product to product but the principle is the same. There needs to be data in the content database, templates that use that data in defined content types or displayable widgets/constructs and lastly views rendered according to the templates. Then there are different levels of users who have different access rights to content and content publishing. All these open source CMS have a high extensibility via third party plugins and themes found on the CMSs respective marketplaces either for free or for a fee.

4 Usability

We define usability as the ease by which a user, in the role of a developer in our case, achieves desired and well performing website with a CMS tool. To evaluate usability, it needs to be broken down to smaller individual measurable components, as said in this book [Nei94 p.26]: “only by defining the abstract concept of “usability” in terms of these more precise and measurable components can we arrive at an engineering discipline where usability is not just argued about but is systematically approached, improved, and evaluated (possibly measured)”. Additionally, the physical tasks themselves that are required to operate a system should be broken to individual actions and measured. For example, a use-case where a button is used to change the view, consists of moving the cursor to the button, then interacting with it according to its type and finally waiting that the triggered action resolves [Hol05].

Usability of a system, like a web site, is not necessarily an abstract measurable thing but instead it is the perceived usability of all the users of the system. This individual perception is greatly affected by user characteristics such as age, gender, educational level and general knowhow in technology [BM01]. The effects of a highly usable system can be seen as high user performance and in the effectiveness of newly introduced users. As such user performance can be called a measure of usability.

In this section we research the usability of the four CMS. First we analyze the table of usability features of these CMS (table 1). Then we discuss what factors must be considered when evaluating usability of a system. After that, the focus is on user performance and learning curves, which can be used to measure usability. Then we analyze the usability of the CMS in this thesis with the tools presented. Finally we review the results and findings.

Table 1 – Usability feature table [Soc17]

Features	*Drupal*	*Joomla!*	*WordPress*	*Plone*
Drag & Drop Content	**No** Plugin	**Yes**	**Yes**	**Yes**
Image Resizing	**No** Plugin	**Yes**	**Yes**	**Yes**
Multiple Upload	**No** Plugin	**Yes**	**Yes**	**Yes**
Spellchecker	**No** Plugin	**No** Plugin	**Yes**	**No** Plugin
Style Wizard	**No** Limited	**No**	**No**	**No** Plugin
Subscriptions	**No** Plugin	**No** Plugin	**No** Plugin	**No** Plugin
Template Language	**Yes** Limited	**Yes**	**No**	**Yes**
Undo	**Yes** Limited	**No**	**Yes** Limited	**Yes**
WYSIWYG Editor	**No** Plugin	**Yes**	**Yes**	**Yes**
Extensible User Profiles	**Yes**	**Yes**	**No** Plugin	**Yes**
Interface Localization	**Yes**	**Yes**	**Yes**	**Yes**
Management				
Advertising Management	**No** Plugin	**Yes**	**No**	**No** Plugin
Content Scheduling	**No** Plugin	**Yes**	**Yes** Limited	**Yes**
Inline Administration	**Yes**	**Yes**	**No** Plugin	**Yes**
Package Deployment	**No**	**No**	**No**	**Yes**
Sub-sites / Roots	**Yes**	**Yes**	**Yes**	**Yes**
Themes / Templates	**Yes**	**Yes**	**Yes**	**Yes**
Web Statistics	**Yes**	**Yes**	**No** Plugin	**No** Plugin
Web-based Translation Management	**Yes**	**No** Plugin	**Yes** Limited	**Yes**
Workflow Engine	**Yes** Limited	**No** Plugin	**No**	**Yes**

4.1 Usability features

Usability features comparison data can be found in table 1. The version info is the same as for table 2 and can be found there. First section of the table 1 presents a list of features that enhance the user experience of a CMS. The second part describes what management tools each CMS has access to.

4.1.1 Term explanations

Explanations to the terms found in table 1.

- Multiple upload feature means that it is possible to upload more than one content file at the same time.
- Spellchecker is a feature that checks if spelling of any text content is valid.
- Template language feature means that the software has its own templating language with which the page structure can be modified.
- WYSIWYG editor is an abbreviation for: What you see is what you get, which concretely means that any text formatting done in the live text editor matches the text formatting on the page.
- Content scheduling feature enables a timed content publishing schedule.
- Package deployment means that the software deploys the current version to production with one click and packages it in one package for transferring.
- Workflow engine allows for customized workflow experience for content creation and publishing flow

4.1.2 Table data analysis

With a cursory glance at the table (table 2) we can see that in most cases there is clearly a plugin to provide for almost any usability feature that doesn’t ship with the box. There is no significant difference in what these CMS are capable of with a few plugins. The largest differences being that WordPress doesn’t have a Template Language and that Plone has package deployment.

In conclusion Plone has all the features in this table either as comes-with-the-box or as a plugin, though it should be noted that third-party plug-ins are not necessarily as well tested as the features that are directly included in the CMS project files or published by the CMS team themselves. Drupal ships with the least features but it is highly extensible with plugins. Joomla and WordPress falls in the middle with some features included and some extensible with plugins.

4.2 Strategic usability factors

E-commerce sites die or thrive depending on how well they are implemented in the way of smooth operability that consists of ease of use features and bug free implementation among a host of other usability related attributes. These areas can be characterized with a usability assessment-model (Fig. 4) and ease of use statistics.

Usability is not an abstract thing but rather something that varies from user to user because of differences in that user’s characteristics like gender, age, education and general knowhow in the IT-field. These are strategic usability factors and they also include other aspects that also affect this perception. For example, cultural differences, aesthetic bias, the positioning of elements on areas where the users are or aren’t used to find them and the use of color schemes that appeal to certain audiences more than to others. [BM01]

To improve website usability and performance, this improvement needs to be quantifiable with measurable attributes. A research by J. Palmer suggests that usability in terms of website success can be measured by download delay, site navigation schema, interactivity and responsiveness as key factors. [Jpw02]

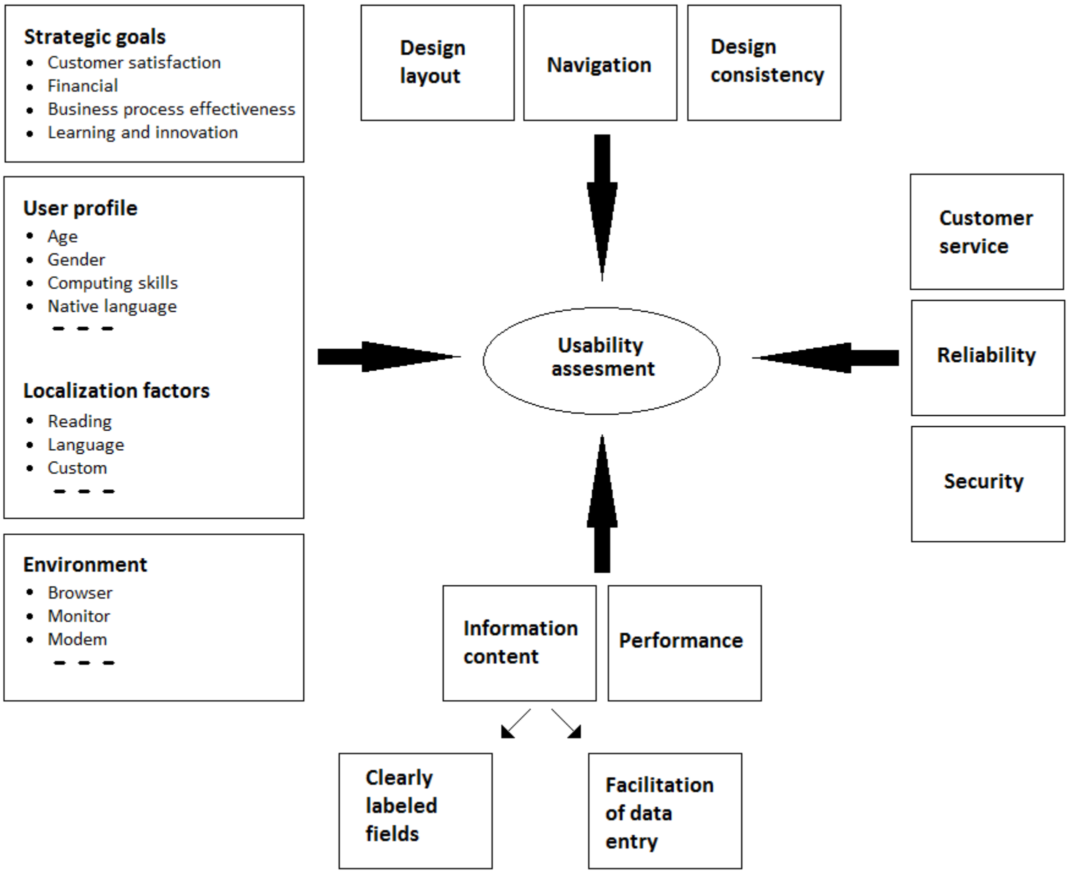
A lot of usability research and work is based on and expanded upon a book about usability engineering. While most of the real word examples are not about CMS usability, still a lot of this research, experience and practices is also related to CMS usability [Nei94].

The following usability factors are used as parts of usability assessment model, which attempts to describe what qualities are required to be considered to achieve highly usable software.

Design layout is the main visual area of usability. It is all the little and big things to make the site look nice. It defines the positioning of elements, page theme, color palette and the way elements are shown on the page, spacing, contrast, relative positioning and light. These characteristics affect the ease and comfort of use and the speed of navigation and task performance. [Jpw02]. Limited human information processing capabilities should be considered when designing user interfaces to reduce information overload [ZW14].

In chapter 2, introduction of the CMS, we found out that all these CMS use a design theme for producing websites. Through these the frontend of the websites can be customized to an extent that depends on the skills and understanding of the user. For a

Figure 4 – Usability assessment model [BM01]



content manager, choosing themes for new pages and using the content types provided by either the software, a developer or a third party is similar in complexity, though with Drupal the approach, in terms of how different pages and content types are managed and linked, is more complex. From developers' perspective, the customization is a whole different beast requiring frontend programming skills and a good understanding of the framework. On the developer admin UI side of things, based on the clarity and unambiguousness, WordPress had the most user-friendly experience to offer, Joomla and Plone came second and Drupal last.

Clear **navigation** schema is mandatory along with simple logic to prevent the user from feeling lost while surfing more complex sites. This determines the breadth and depth of navigation paths, and the different ways of navigation available to the user, like buttons, drop-down menus and links. Research in the areas of Usability and Design suggests that organization of content and the ease of navigation through a site are potentially key usability factors [Jpw02]. A group of researchers gathered minimal data through internet and email. Their analysis of the data indicate that navigation structure is a main concern when building websites with CMS and that the quality of links on the page is very important to site usability [GMM16]. Navigation within admin UI had little differences among these 4 CMS.

Consistent design across the site is ideal so that there is a consistent location for each component and element between all similar pages and there is also a consistency of names and locations of same elements in different pages. This helps faster usage and with user familiarity. For us the perspective for this factor is the design of the admin UI/overlay and the restrictions the system places on the design. All CMSs managed to maintain good consistency across the admin portal.

Performance is a high priority attribute because, no matter how attractive your page looks if it doesn't feel responsive or snappy or if the wait times are long it will quickly turn away most users. The site performance is greatly affected by reliability factor as inaccessible service has a total performance of zero. Performance is discussed at length in chapter 5.

Information content is about timely and correct error messages, correct translations, terminology should relevant to the site's niche, placeholders for missing content. Most of these factors are not directly linked to CMS however error messages and placeholders can be configured to be active from the settings.

Reliability is about the trust that can be placed on the service. This factor is described in terms of downtime, crashes, and steady performance. The downtime of a service, or web sites in this scope, is basically the percentage that how often it is inaccessible in given time a period. Low amount of crashes and runtime errors mean that the service provides a bug-free experience so that everything executes as expected. A Steady performance signifies that the fluctuation in response times is minimal. This characteristic is more related to the hosting platform than the CMS so our focus is from the perspective of the publishing pipe. Plone was clearly ahead of the three other CMS in this thesis and those were similar in lack of sophisticated publish flow.

Security governs privacy and safety of users at any given time. All sensitive information should be secure and not accessible to users and unauthorized sources shouldn't be able to damage other users or the system or steal any non-public information. [BM01]

Joomla, Drupal and WordPress had similar strengths and weaknesses regarding web security and hacking in similar conditions [MJS14]. These 3 CMS were compared based on two different experiments to see which provides better web security. It was concluded that all of the CMS listed provide an effective basic security that prevents direct hacking of the site using SQL injection, cross site scripting, remote and local file inclusion, directory traversal, brute force attack, cookie poisoning or cross-site forgery. They found that if a hacking occurs the fault generally lies with vulnerabilities in plugins uncertified by the main developers of that CMS. Out of these three CMS WordPress has less sensitive files available to hackers than the other two, thus being the most secure of the three [PRP13]. With all the CMS, the community driven model itself improves the security factor for any stable community supported open source CMS a great deal.

Simplicity In her research, Karvonen discusses at length about the impact that simplistic beauty has towards improved usability and that there is indication that beauty through simplicity can be evaluated to some extent. With these points in mind we analyze the four CMS for their developer portal aesthetics [Kar00].

None of the CMS reviewed really aimed for simplicity due to the number of features they offer at each view. Starting from the admin dashboard, there is a lot of information to process.

4.3 User Productivity

From the perspective of a developer, CMS user productivity is closely tied to the speed and ease at which a user can produce content and functionalities to a website. Examples of such items are new pages, page templates and incorporating plugins to the site. Any software that is easy to learn allows new users working with it to quickly become sufficiently proficient to be called productive. In contrast, with a software where there is limited effort to smooth out this learning experience, it takes much longer to attain proficiency [Nei94, chapter 2.2]. Researchers found out that users that view a preview about how to perform tasks with a CMS helps them to recover from errors more easily while significantly improving their satisfaction and confidence while using that CMS, and these qualities are directly related to learnability [RK08]. A common way to measure this speed of progression is a learning curve. A commonly used principle with measuring the learning curve of a software: take a group of users unfamiliar with the system and measure how fast their productivity increases in terms of how many steps are taken towards launching a site or how many features are completed in given time [Nei94, chapter 2.2]. This kind of quality evaluation is usually done with a sample selection of test users who are picked to represent the targeted audience. [Nei94, chapter 6]

When we discuss about learning curve we tend to speak in terms of novices and experts. Expert focused systems have higher efficiency cap than that of its respective novice counterpart, but it is a lot faster to achieve reasonable productivity with software that has novice-friendly learning curve. Counterintuitively software can have best of both worlds with easy learning- and good expert tools. This is achieved with multiple interaction-styles, a novice starts learning with tools that have corresponding icons in the UI but there is also text input and shortcut hotkeys, called accelerators from now on, for more experienced users. Preferably a novice user can transition to a more productive expert interface in terms of efficiency. A novice shouldn't have to confront the expert interface and the expert should neither be hindered by the novice UI [Nei94, chapter 2.4]. A study examined the effects of information overload induced by time constraints, complex tasks and the amount of information needed to process. This overload negatively affects user satisfaction and perceived system usability, which together lead to reduced user performance. [ZW14]. Another point is that if some function is behind a special interaction style, a double-click in example, that same action should be available and visible in a menu somewhere. In trade-off scenarios with usability decisions, a win-win resolution should be sought out as it's usually possible to achieve a more efficient solution [Nei94, chapter 2.4].

Drupal presents itself as an expert system with a steep learning curve but with a solid user productivity and customizability at the top of the function. Web development with Drupal feels more like you are creating software components than content for a website and brings the complexity of software development with it. Also, the power to customize how you build the site comes with a cost of requiring experience.

With Plone, their ideology about features and their extendibility is backwards to the other CMS, in that it hopes to provide much of the needed tools with the initial installation, but this actually works counterintuitively and sharpens the slope of the learning curve, as developers unexperienced in the system are faced with myriad of tools and options right from the start. We think that it also has an inherent soft cap for experience, where suddenly the developer needs to know a lot about the system to achieve results.

Learning to develop in the WordPress way takes a while to get into, but for a development where no customization through programming is needed, WordPress is a solid winner in pleasant learning curve and productivity. Although, when programmatic customization is needed, this learning curve becomes a lot steeper and even unfriendly.

Joomla again falls to the middle. Its extensible and the development cycle is like that of WordPress, but not quite as streamlined.

4.4 Content Structure

When features (or web content) accumulate to such extent that a tree-like menu hierarchy is needed to access them all, we arrive at the classical trade-off depth vs breadth. Modern interfaces are designed for breadth over depth to achieve simplicity and faster access times [Nei94 ch. 3.3]. This kind of menu structuring for content on the user side is supported in all these CMS.

With WordPress content is divided to multiple basic content types and the organization of content works fairly well for small needs, but the handling gets bloated with large quantities of data.

Joomla provides additional structuring and organizing options for content managing with customizable content categories and a summary view that shows a hierarchical tree breakdown of content.

Content structuring in Drupal offers a lot of flexibility with a lot of responsibility as it offloads the responsibility for content structuring to the developer. Basic content types that are needed are expected to be defined by the user. The default content structuring is rather ambiguous and multi-leveled for content managers to use as is.

Plone offers basic content types right off the bat and allows defining custom types with more ease than what the other CMS offer.

4.5 Results

WordPress shines at building a small and airy website, albeit a bit generic one, with few sophisticated add-in components, and the result would be usable and solid with a quick develop cycle. At larger content amounts the handling get somewhat boggled. Serious customization is similar in difficulty and usability as in Drupal.

With Joomla, the content is structured in a logical manner. It’s basically Templates that contain modules that can host components and all these types can be riddled with plugins that are event handlers in Joomla. Different content types like articles are handled as custom Joomla modules that are assigned to templates. Organization of these modules is up to the developer. Joomla has beginner friendly tools and help sections for each action, though going to help sections to solve a problem doesn’t advocate good usability. When starting a new site with Joomla, getting the site to work is not as unambiguous as it could be. Localization management in Joomla is a big focus.

Drupal is in the steep end of the CMS learning curve spectrum and getting it running and configured is one of the harder installation processes. It has the largest commitment required to start building websites and it’s not very beginner friendly compared to the other three CMS in this thesis. Drupal doesn’t come with a WYSIWYG editor, but it is available as a plugin. Basic customization requires editing the ambiguously organized Drupal core twig template files and changing the code in those. The other way is to access Drupal theme hooks and directly customize the site execution from there, but these are more complicated and offer less possibilities. With larger sites, the need to create custom content types and module is an upside while it’s less appealing for smaller sites as it comes with a lot of overhead.

Plone comes with the most functionality on install, while with other CMS the priority after getting the software running is usually to install some powerful plugins. Installing plugins in Plone is cumbersome and not beginner friendly. Plone incorporates the TinyMCE editor which has more options and tools available than the default editors present in the other three CMS. Plone also offers a way to implement custom content types more easily than with the other editors.

5 Performance

This chapter is about measuring and comparing the performance of websites built with a CMS. Analysis is done by comparing quantifiable attributes like wait times and website size on disk of each CMS. Then we find out if different CMS platforms perform as well as they claim by looking at sites made with said systems and comparing the results of our site-by-site performance tests to the ones claimed by the CMS itself or data presented by relevant papers.

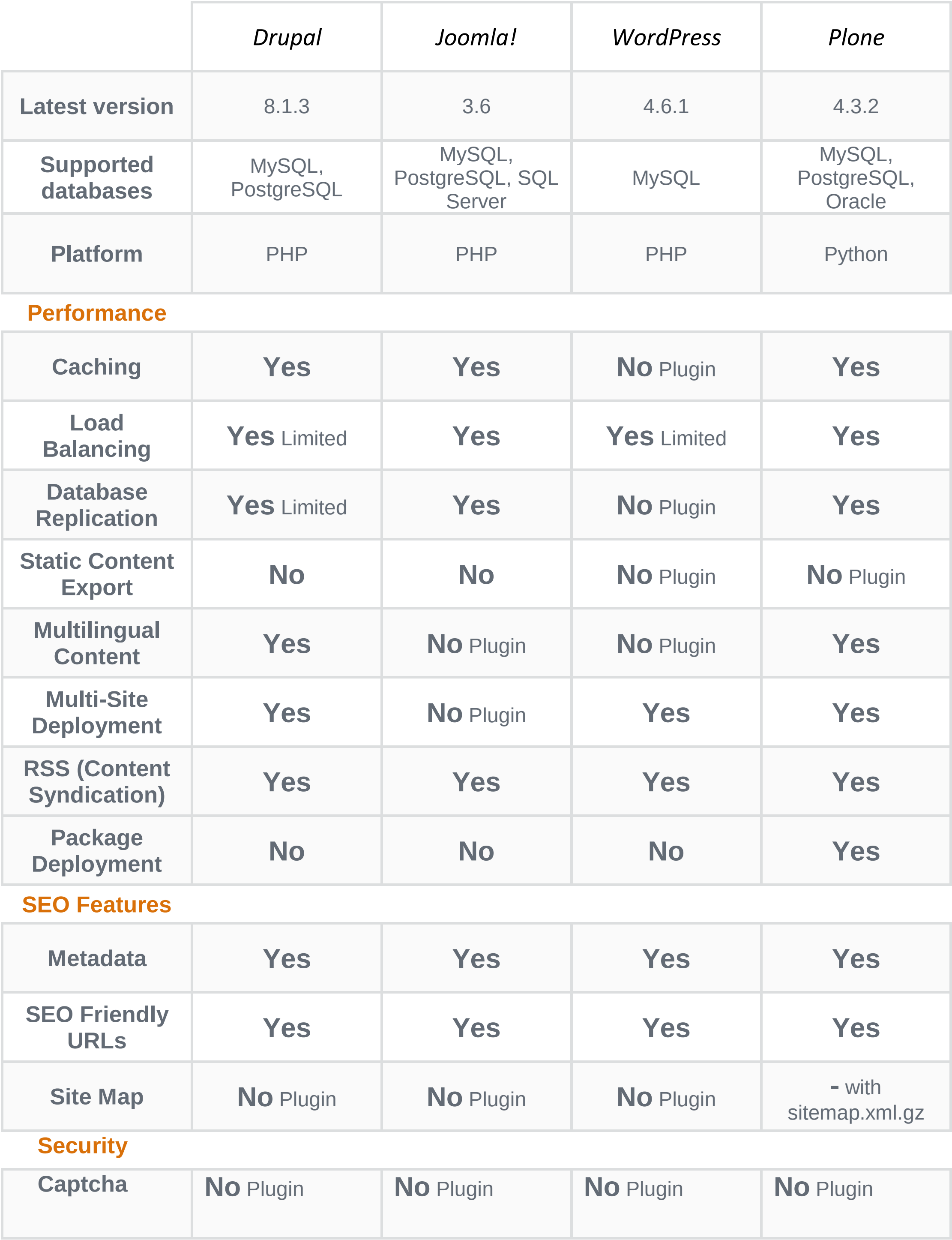
5.1 Performance features

Performance related feature comparison chart of all four CMS (*table 3)*. The first part of the table presents specification data on the version number of the program of which the data is derived from, the databases each CMS supports and the programming language it is based on.

It is clearly visible from the table that almost all features not shipped with the software is found as a plugin. Plone seems to be most flexible as the CMS which ships with most features without need for plugins.

Plone again has the most features without need for plugins. WordPress ships with least performance features, which can be fixed with plugins. Overall again majority of these performance features either come with the box or can be installed via plugin.

Table 2 - Performance feature table [Soc17]



	Drupal	*Joomla!*	*WordPress*	*Plone*
Latest version	8.1.3	3.6	4.6.1	4.3.2
Supported databases	MySQL, PostgreSQL	MySQL, PostgreSQL, SQL Server	MySQL	MySQL, PostgreSQL, Oracle
Platform	PHP	PHP	PHP	Python
Performance				
Caching	**Yes**	**Yes**	**No** Plugin	**Yes**
Load Balancing	**Yes** Limited	**Yes**	**Yes** Limited	**Yes**
Database Replication	**Yes** Limited	**Yes**	**No** Plugin	**Yes**
Static Content Export	**No**	**No**	**No** Plugin	**No** Plugin
Multilingual Content	**Yes**	**No** Plugin	**No** Plugin	**Yes**
Multi-Site Deployment	**Yes**	**No** Plugin	**Yes**	**Yes**
RSS (Content Syndication)	**Yes**	**Yes**	**Yes**	**Yes**
Package Deployment	**No**	**No**	**No**	**Yes**
SEO Features				
Metadata	**Yes**	**Yes**	**Yes**	**Yes**
SEO Friendly URLs	**Yes**	**Yes**	**Yes**	**Yes**
Site Map	**No** Plugin	**No** Plugin	**No** Plugin	**-** with sitemap.xml.gz
Security				
Captcha	**No** Plugin	**No** Plugin	**No** Plugin	**No** Plugin

5.2 Measures

Page performance measurements.

Following are the parameters associated with page performance and load.

- Page Load Time (PLT), usually measured in milliseconds (ms). This measure indicates that how long it takes to load the page from the first request to the last.
- Page Size (PS), this measures the total size of the page in kilo bytes (KB). As this value gets larger the PLT also increases accordingly.
- Total Requests measures the number of requests send to the server to load the page. This measure affects PLT directly because of the latency in the round-trip time of the request to the server and back and the time it takes to handle each request.
- Total cascading style sheet (CSS) files: Number of CSS files used by CMS to make a page. Basically, more CSS files means more requests must be fulfilled. This can be enhanced by packing all the CSS to a single file.
- Total java script (JS) files: Number of JS files used by CMS to make a page. This like the CSS files increases the number of requests and can be enhanced via packing assets.
- Total individual assets like images also add to the number of requests needed.
- PLT after caching: when page load first time some of its content store in cache memory so when we load that page again only rest of the data which is not in cache will load from the server so it decreases load time.
- PS after caching: As system cache used to decrease the PLT it also reduces PS as mention above. [PRD12]
- Request time: The time to first byte to arrive from the server after a client requests a page.

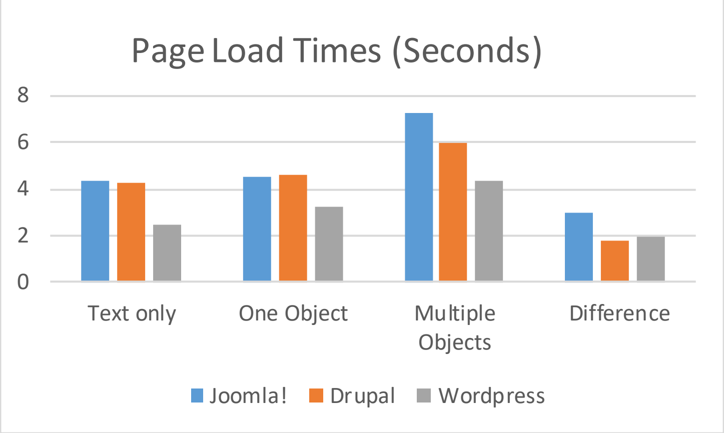
5.3 Simple website performance

In 2011 a group [PRD12] compared the performance of three CMS systems, namely Joomla, Drupal and WordPress. They hosted simple web pages of all these systems on a local and a remote server and compared the performance statistics. The tests were made by comparing three scenarios: In the first the websites only had informational text content. In the case of one object these pages had a calendar plugin installed. In the third the pages had multiple plugins added to them. The following table (table 3) shows the CMS versions and the plug-ins used.

Table 3: [PRD12]

CMS	Version	Calendar Plug-ins	Clock Plug-ins	Gallery Plug-ins
Joomla!	1.6.4	Minicalendar	Jmtimenow	Simplespotlight
Drupal	6	Calendar Block	Timeblock	Cycle Plugin
Wordpress	3.1.3	In Build	Local Time Clock	Js Banner Rotate

Table 4: [PRD12]

<table>
<tr><td></td><td colspan="4"> </td></tr>
<tr><td></td><td>Text only</td><td>One Object</td><td>Multiple Objects</td><td>Difference</td></tr>
<tr><td>Joomla!</td><td>4.35</td><td>4.51</td><td>7.3</td><td>2.95</td></tr>
<tr><td>Drupal</td><td>4.26</td><td>4.58</td><td>6.01</td><td>1.75</td></tr>
<tr><td>Wordpress</td><td>2.45</td><td>3.23</td><td>4.39</td><td>1.94</td></tr>
</table>

From the table (table 4) we can say that WordPress carries less overhead then the other two. We can also note how increased complexity also increases load times sharply for Joomla so we can say that it doesn’t scale well with added complexity to the site.

In the next table (table 5) we can see the load times from the second time a user comes to a web page and loads part of the data from the cache. In this scenario Joomla doesn’t seem to scale well when compared to Drupal and WordPress. WordPress seems to be a bit faster though it also benefits the least from caching. Still when comprising the results shown in table (table 6) to determine the overall impact of caching we find out that Joomla reaps the largest benefits from caching, but even when cached it couldn’t match the others in loading speeds.

Table 5: [PRD12]

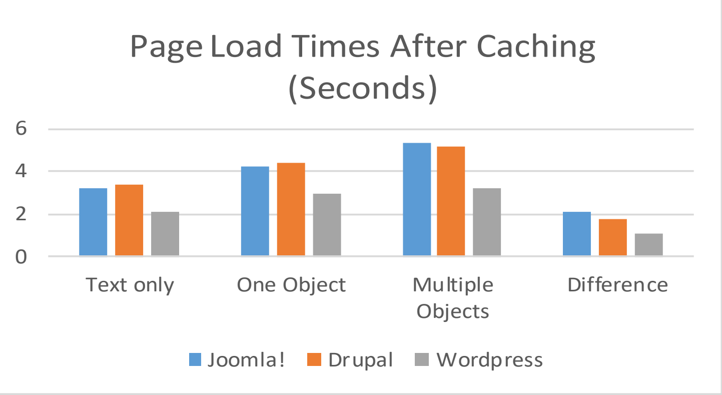
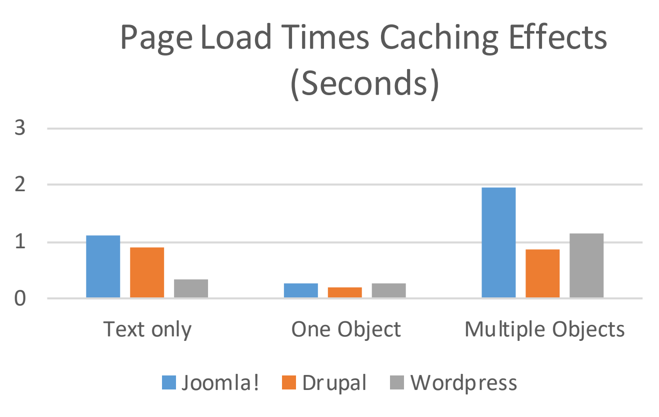
<table>
<tr><td></td><td colspan="4"> </td></tr>
<tr><td></td><td>Text only</td><td>One Object</td><td>Multiple Objects</td><td>Difference</td></tr>
<tr><td>Joomla!</td><td>3.25</td><td>4.26</td><td>5.33</td><td>2.08</td></tr>
<tr><td>Drupal</td><td>3.35</td><td>4.4</td><td>5.13</td><td>1.78</td></tr>
<tr><td>Wordpress</td><td>2.13</td><td>2.96</td><td>3.26</td><td>1.1</td></tr>
</table>

Table 6: [PRD12]

<table>
<tr><td></td><td colspan="3"> </td></tr>
<tr><td></td><td>Text only</td><td>One Object</td><td>Multiple Objects</td></tr>
<tr><td>Joomla!</td><td>1.1</td><td>0.25</td><td>1.97</td></tr>
<tr><td>Drupal</td><td>0.91</td><td>0.18</td><td>0.88</td></tr>
<tr><td>Wordpress</td><td>0.32</td><td>0.27</td><td>1.13</td></tr>
</table>

In conclusion WordPress seems to be the fastest loading CMS of these three candidates. Joomla is affected most by caching content where it shines and beats the loading times of Drupal but still losing the first place to WordPress by a wide margin. But arguably the most important case is the multiple objects scenario as from what I have seen most websites have multiple objects and almost none have only text content and in such light the fastest CMS is WordPress, then Drupal and then Joomla. When we combine the data from all these tables we find that on first visits Joomla trends towards rapidly increasing tardiness in loading times when site complexity increases. With subsequent visits Joomla closes most of this gap [PRD12].

5.4 Live websites performance

In this chapter, we show statistics from my measurements about visits to Finnish websites which are built with Drupal, WordPress, and Joomla as my queries yielded zero Plone websites residing in Finland. This research is done to measure the speed, response time and page size of the CMS. The reference site where these top websites at the time of this test were listed is [Bur17].

The computer used and network performance statistic at test time:

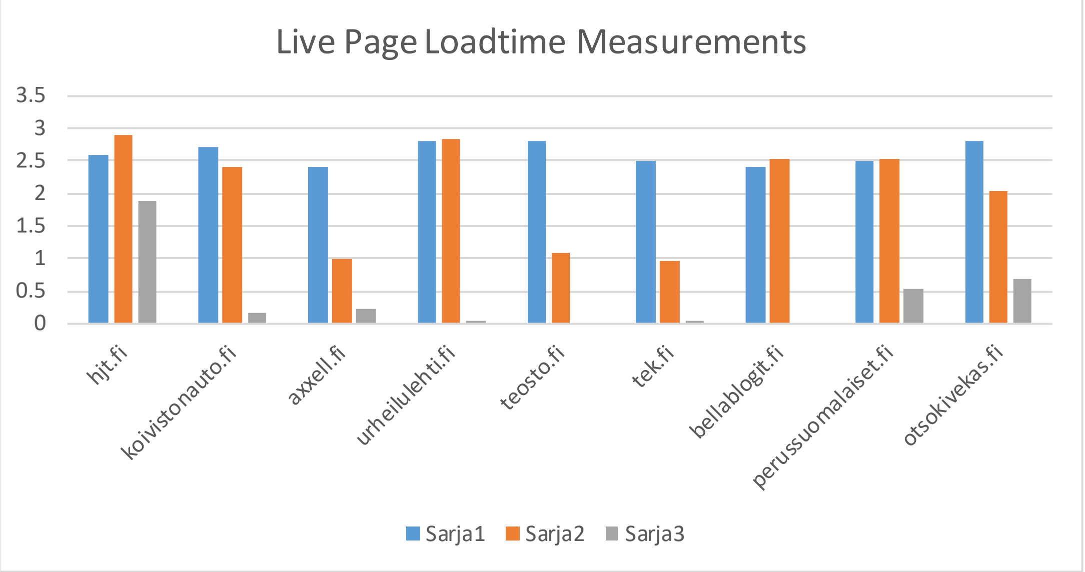
- Windows 10 64-bit
- Intel core i7-4770k 3.5 Ghz
- Ram 8 Gb
- Network: ping 17 ms, upload 32.80 Mbps, Download 54.13 Mbps

Measurement results are displayed in table 7. All the measurements are taken from websites hosted in Finland to eliminate as much interference as possible caused by difference in request round-trip times to the webserver caused by varying distances from the queried server. The location was found with address location finder. The sites used will be similar in size to achieve maximum consistency among publishing systems and websites. Measurements are averaged out from 10 page loads and all measurements made within a thirty-minute time frame to minimize fluctuations in the download bandwidth.

It should also be noted that load time is highly dependent on how the content packing is handled as the total latency of content requests is multiplied by every individual file that needs to be downloaded. Table (table 7) doesn’t take a stand on this aspect.

We can determine from this rough data that all CMS are capable of comparably fast load times, though overall WordPress seems to have a slight edge over the others. Speeds are very close when we remove the request time factor, denoted in the Request column in the table. Request time means the time it takes for the first byte to arrive after the page request is sent by a client. which is anyway mostly due to a server that is either bad or under a heavy load or further away from the request location.

Table 7



Platform	Site	Size (MB)	Load time (s)	Request (s)
Joomla	hjt.fi	2.6	2.901	1.9
	koivistonauto.fi	2.7	2.402	0.15
	axxell.fi	2.4	0.995	0.21
Drupal	urheilulehti.fi	2.8	2.853	0.04
	teosto.fi	2.8	1.075	0
	tek.fi	2.5	0.969	0.02
WordPress	bellablogit.fi	2.4	2.54	0
	perussuomalaiset.fi	2.5	2.536	0.52
	otsokivekas.fi	2.8	2.049	0.7

5.5 Results

These tests didn’t take dissimilarities, in publishing formats and asset packaging, into account but results give some direction as to which CMS can achieve the fastest loading sites. Feature-wise Plone had the most coverage either from the get-go or installed as plugins but the other CMS were rather close behind so no clear winner can be found here.

The load time research only yielded results for Joomla, WordPress and Drupal as Plone websites fitting the test criteria were not found. The test results indicate that the first-time visit loading times to Joomla sites were higher than with the rest, but with subsequent loads this gap is reduced as caching kicks in. The other notion was that WordPress first time loading was trending towards being the fastest but with cached loading in the mix on subsequent visits Drupal took the lead.

6 Conclusions

We compared Drupal, WordPress, Joomla and Plone from two sides, Usability and Performance and from two perspectives, the developer's and the end site's view point. For testing purposes, we used websites built with three of these CMS and organized a live-website loading time experiment in addition to any literature sources about CMS website testing. We did this research to answer our RQ1, that was about which CMS was best suited for what task. We came to the conclusion, that the results are as expected and the findings suggest using WordPress, if the task at hand stays manageable. While the themes of WordPress cater to the customization needs when only slight coding is needed, they fare worse after that point. For larger and more complex sites, where components can be reused in multiple places so that the additional overhead is justified, Drupal is the right tool for the job.

On the other hand, as our RQ2, we tried to find a golden goose, namely a CMS that would shine in both: In the usability sector, concerning developers that build sites and in the performance sector, by measuring, who has the strongest performance statistics.

First, usability ranks these four CMS in the following order: WordPress, Joomla, Plone and Drupal. It's arguable whether Plone or Drupal is more usable as both have their weaknesses. Drupal isn't beginner friendly and Plone offers good tools initially but extending it with plugins or customizing it isn't as usable as it could. Differing from the others, WordPress has put extra effort in how it presents itself to users and has very usable extension support. Joomla falls in the middle-ground with no glaring weaknesses but no shining edges either.

Second, the main differences in features are that Plone has the most included initially, Drupal comes close behind with performance features, while WordPress and Joomla depend on third party plugins to enable similar functions. Though, it should be noted that in the other CMS than Plone there is a lot of effort to make sure that installing new plugins is very convenient, and this dulls the advantage that Plone has in this sector.

Third, Performances across the CMSs Drupal, WordPress and Joomla were somewhat similar. Plone doesn't appear in the comparisons as no suitable data could be gathered. Results of our own research, where we compared a heavily narrowed down group of websites to find comparable real-life data about the loading times of webpages built with these CMS, and literature research yield similar statistics. Joomla was surprisingly found to be on the slower end of the CMS loading times. With WordPress, first time page visits load faster than for the others both in literature research and in the measurement test conducted by us, but on subsequent loads this gap is interestingly reduced and Drupal clearly scales better than WordPress when the number of objects increase. This scalability, when site complexity increases shows surprisingly a clear ranking where Drupal scales better than the others, WordPress comes second and Joomla last.

In conclusion, we have to state that no golden gooses this time. From the basis of this thesis we can say, that when developing websites with need for regular content updates, it’s crucial to select the right CMS for the job. Following papers can use this thesis as a foundation for researching other aspects of these CMS or expand these results and research to other prominent CMSs in the field.

7 References

[BM01] Becker, SA., and FE. Mottay. *A Global Perspective on Website Usability*. In *IEEE Software* (IEEE), 2001, Volume: 18, Issue: 1, pages 54-61.

[Nei94] Neilsen, J. *Usability Engineering*. Academic Press Inc, 1993.

[Hol05] Holzinger, A. *Usability engineering methods for developers*. In *Communications of the ACM - Interaction design and children*, ACM, 2005, Vol 48 issue 1, pages 71-74.

[Kar00] Karvonen, K. *The beauty of simplicity*. In *CUU '00 Proceedings on the 2000 conference on Universal Usability,* ACM, 2000, pages 85-90.

[GMM16] Gilani, S., Majeed, A., Muzammal. M., Rehman, A.U., Zaheer, H., Jan, Z. *A navigationl evaluation moel for contonet management.* In *The Nucleus*. 2016, vol 53 No 2, pages 82-88.

[RK08] Redding, E. and Karreman, J. *Watch out for the preview: The effects of a preview on the usability of a Content Management System and on the users' confidence level.* In *Professional Communication Conference, IPCC 2008 IEEE International.* 2008

[Pol12] Polgar, J. *Do You Need a Content Management System?* Information Science Reference, 2012.

[PRD12] Patel, SK., Rathod, VR. ja Dean, SP. *Joomla, Drupal and WordPress - A Statistical Comparison of Open Source*. in *Trendz in Information Sciences and Computing (TISC), 2011 3rd International Conference,* IEEE, 2012.

[ZW14] Zou, J. ja Webster, J. *Information overload in using content management systems: causes and consequences.* in *PACIS 2014 Proceedings, AISeL, 2014, article 235.*

[XiYu10] Xiang Cao and Yu Wenhua, *Using Content Management System Joomla! to Build a Website for Research Institute Needs* in Management and Service Science (MASS), 2010 International Conference, IEEE, 24-26 Aug, 2010.

[MJS14] Mirdha, A., Jain, A., Shah, K., *Comparative Analysis of Open Source Content Management Systems.* In Computational Intelligence and Computing Research (ICCIC), 2014 IEEE International Conference, IEEE, 18-20 Dec. 2014

[PRP13] Patel, K. Savan, Rathod V. R., Prajapati B. Jigna, *Comparative Analysis of Web Security in Open Source Content Management System.* In Intelligent Systems and Signal Processing (ISSP), 2013 International Conference on 1-2 March 2013, IEE

[OBS08] Ozkaya, I., Bass, L., Sangwan , R. S., Nord, R. L, *Making Practical Use of Quality Attribute Information.* In IEEE Software, Volume: 25, Issue: 2, March-April 2008, IEEE

[Jpw02] Palmer, J., W., *Web site usability, design, and performance metrics*, in information systems research, Volume: 13, Issue: 2, Pages 151-167, June 2002, Institute for Operations Research and the Management Sciences, Linthicum.

7.1 Web sources

[Plo17] Plone Foundation, *www.plone.org*. [23.1.2017]

[Wor17] WordPress.com. *www.wordpress.org*. [23.1.2017]

[Dru17] Drupal.org. *www.drupal.org*. [23.1.2017]

[Ope17] Open Source Matters, Inc. Joomla. *www.joomla.org*. [23.1.201]

[Che17] CheckPageRank.net, *http://checkpagerank.net*. [23.1.201]

[Soc17] Socialcompare, *http://socialcompare.com/en/comparison/popular-content-management-system-cms-comparison-table*.

[Spe17] Speedtest, http://www.speedtest.net. [15.5.2017]

[Bur17] http://top-websites.burtronix.co.za/. [25.5.2017]

[Ipf17] IP location finder of website hosts, *http://www.ipfingerprints.com/*. [25.5.2017]

The reference date of following test websites is 5.6.2017.

[Hjt17] Helsingin Jääkiekkotuomarit HJT ry, *http://www.hjt.fi/*

[Axx17] Axxell Utbildning Ab, *https://www.axxell.fi/*

[Koi17] Koiviston Auto -konserni, *www.koivistonauto.fi/*

[A-l17] A-lehdet Oy, *www.urheilulehti.fi/*

[Teo17] Teosto, *www.teosto.fi/*

[TEK17] Tekniikan Akateemiset TEK, *https://www.tek.fi/fi*

[Bab17] Babler Oy, *bellablogit.fi*

[Per17] Perussuomalaiset Rp, *https://www.perussuomalaiset.fi/*

[Kiv17] Otso Kivekäs, *otsokivekas.fi*