Master thesis in computer science
15 credits, advanced level

Developer guidelines for e-commerce sites derived through benchmarks

Utvecklarens regler för utveckling av e-shop websidor hittade genom prestandamatning

Author: Sani Music
me@sanimusic.se

Exam: Master of computer science 60 credits
Subject: Computer science
Program: Computer Science
Handin: 2014-08-10

Supervisor: Edward Blurock
Examiner: Jan Persson
Abstract

E-commerce solutions are providing purchasing capabilities to users on products online, but Internet users are sensitive to delays. In-fact users see page load time as a website feature. Users are also sensitive to cognitive load time, which has to do with perceived aesthetics of a website and mental models which are created from previous interaction with websites. The scope of the thesis is that four commerce content management systems (CMSs) are compared in this thesis to find out which performs with the lowest page load time. These are Magento, Drupal with commerce, Wordpress with woocommerce and Joomla with Hikashop. The CMSs are all pushed to a very similar website design based on one universal theme, using the twitter bootstrap mobile first responsive framework. The design is based on previous research of users mental models for e-commerce websites. A fully functional fictional site Fastcommerce is developed on the four CMSs. The CMSs are filled with one category page of identical sample electronic products (12 items). The products have images and lorem ipsum text. The method of the thesis are experiments. The experiments are conducted to find the page load time on a local environment. The results show clear performance improvements on all four CMSs after applied optimizations. But some of the optimizations cause negative performance consequences.

Keywords: E-commerce, Benchmarking, Performance, Page load time, Guidelines, Front end, Drupal, Wordpress, Joomla, Magento, Theming, Content management systems
## Contents

1 Introduction .......................... 1
   1.1 Goals .................................. 2
   1.2 Research questions .................. 2
   1.3 Hypothesis .......................... 3
   1.4 Expected Results ...................... 3

2 Theoretical Background ............... 4
   2.1 Cognitive load time and aesthetics . . . . 4
      2.1.1 Flat design and Skeuomorphic design . . . 5
   2.2 Content Management Systems .......... 5
   2.3 Tools and technologies used in the thesis . 6
      2.3.1 CSS with Preprocessors .................. 6
      2.3.2 GRUNT and GULP our task runners .......... 6
      2.3.3 Firebug ................................ 6
   2.4 Page load time ...................... 7
      2.4.1 Onload time ......................... 7
      2.4.2 GET Requests ......................... 8
   2.5 Performance Considerations .......... 8
      2.5.1 Front end .......................... 9
      2.5.2 Back end .................... 9
      2.5.3 Twitter Bootstrap .................. 9
      2.5.4 Measuring performance ................. 10
   2.6 Conclusion ......................... 10

3 Literature Review ..................... 11
   3.1 Cognitive Load Time and Aesthetics .... 11
      3.1.1 Visit duration ....................... 11
      3.1.2 Placement of objects and site layout . . . 11
      3.1.3 Menu type .......................... 12
      3.1.4 Task and page complexity .......... 13
      3.1.5 Comparison of CMSS ............... 14
   3.2 Performance Optimization ............ 15
      3.2.1 Request overhead .................... 15
      3.2.2 Payload size ....................... 15
   3.3 Conclusion of the literature review ... 15

4 Method .................................. 17
   4.1 Motivation For Experiments .......... 17
      4.1.1 Risks with experiments .............. 17
   4.2 Constructing Experiments ............. 18
      4.2.1 Developing a standard theme .......... 18
5 Experiments (Benchmarks)

5.1 Definition ......................................................... 19
5.2 Planning .......................................................... 20
5.3 Operation .......................................................... 20
5.3.1 Creating a standard theme .................................. 21
5.3.2 Creating dummy data ....................................... 26
5.3.3 Gathering Baseline performance statistics out of the box installations standard themes .............................................. 26
5.3.4 Gathering baseline performance statistics with modules installed standard themes .................................................. 27
5.3.5 The Experiment Path with dummy data .................. 27
5.3.6 Gathering Baseline performance statistics with modules installed, dummy content with default themes .............................................. 31
5.3.7 Gathering Baseline performance statistics with modules installed, dummy content and universal themes .............................................. 32
5.3.8 Gathering performance statistics for the optimized installations .............................................. 33
5.3.9 Gathering performance statistics variation of experiments .............................................. 33

6 Result (Evaluation)

6.1 Front End Guidelines .............................................. 36
6.1.1 Default Aesthetics ............................................ 36
6.2 Improvement for the CMSs ...................................... 36
6.3 CMS Guidelines ..................................................... 38
6.4 Reflecting on the research questions and hypothesis .... 38

7 Conclusion and Future Work

7.1 Future Work ......................................................... 42

8 References ............................................................... 43
1 Introduction

Website content management systems (CMS) such as: Drupal, Joomla, Wordpress, Magento offer standard solutions for deployment of various classes of websites. These websites can be e-commerce, blogs, informative sites (interchangeably) or sites customized to accomplish specific purposes. The four CMSs chose in this thesis are popular in

Figure 1: CMS search popularity by Google trends

![Google Trends Chart]

source: Google trends between 2004 and 2014 May 29

Google searches, see figure[20] this motivates their choice for this thesis. The fact that Drupal and Joomla seem to be on a negative trend is only relative to Wordpress and Magento. The Y axis shows a relative popularity in which Wordpress has at a certain time set the 100% mark of search queries.

One important user experience when visiting websites is the time it takes for the page to load, called page load time (PLT). Research shows a correlation between website performance and success. With this in mind website developers and others involved need to be mindful of website performance.[28]

Research shows the time users are willing to wait for a page to load is limited. User patience is sensitive to what might seem as low amounts of time. Users lose patience in 2 seconds for information retrieval. This implies importance to page load time especially if the user has alternatives to the service provided.[32] After a single second the user starts thinking about other things. This can be very costly if they decide to ”purchase” something of an e-commerce site later on. “1.0 second is about the limit for the users flow of thought to stay uninterrupted, even though the user will notice the delay.”[15]

There are two kinds of time we must consider when dealing with the time aspect of website performance. The actual physical time the website needs to load (PLT) and something called cognitive load, which is user perceived waiting time. The cognitive load time is influenced by the amounts of time the user needs to understand the websites features. Such as page aesthetics of the page, navigation and structure. When designing websites we should attempt to keep the cognitive load low, in doing this, the
users feel it is easier to understand what to do and how to do it. \[33\]

In e-commerce the PLT is a key factor to achieve success along side ease to use. \[19\]

The page load time duration when loading websites is caused by many factors such as the server location and connection. In this thesis we discuss CMS factors affecting the page load time performance of E-commerce websites. We also take into consideration the cognitive load time for websites. The programming language PHP is a widely used sever-side-scripting language for websites. When developing with PHP certain content management systems (CMS) are frequently used such as: Drupal, Wordpress, Joomla. These are of big aid to the developer, since they allow back end functionality out of the box. This means the developer saves a lot of time in numerous common cases. The clients or users are often provided with standard and therefore intuitive content editors.

An analysis of performance in content management systems has been conducted in 2011\[37\]. CMSs chosen where Drupal, Wordpress and Joomla to study out of multiple reasons, some of which are: Highest installation count, Highest page ranks, User friendly environments.\[37\]

Industry standard software is more likely to have extensive support and documentation as opposed to less common software. When working with any semi advanced website a database is inevitable to be used. The MySQL (My Sequel) database is the worlds second most used relational database management system \[7\] and is therefore suitable to be used in this thesis.

1.1 Goals

This thesis aims to provide guidelines to improve the cognitive load time awareness and page load time performance of E-commerce web pages. Goals of the thesis:

- Suggest guidelines for theming which will inflict performance improvements for the E-commerce oriented CMSs: Drupal, Wordpress, Joomla, Magento.

- Suggest one of the CMSs to use in one specific E-commerce problem setting of this thesis and motivate it’s selection.

- Produce guidelines for developers to follow when developing for e-commerce sites with specific CMSs.

1.2 Research questions

We will focus with the research questions on page load time only because of the time scope of the thesis. The focus of the thesis will be on the following research questions:

- RQ1: Out of Magento and a E-commerce extended setup of Drupal, Wordpress and Joomla which is performing with the lowest page load time using the same custom optimized front end theme and dummy content.
• RQ2: What performance increase can be observed for the CMSs relative to their baseline performance and each other after applied optimization?

1.3 Hypothesis

• Magento will initially be slower in comparison to other CMSs because of e-commerce capabilities but after making all CMSs e-commerce capable the gap will be reduced.

• Significant performance increase can be observed in all of the CMSs relative to themselves by optimizing the CMSs by with extensions and by using one universal theme.

1.4 Expected Results

The method of approach (Experiments) will produce quantitative data which will through analysis confirm or discard the two stated hypotheses. The results will be of use for developers when deciding between the presented CMSs to use in their project because of how much effort they want to put into optimization, but also which CMS seems most suitable for projects in the scope reflecting these results. The results will be valid exclusively on software releases that are benchmarked, this is a natural limitation of software benchmarking. The result will present amount of times faster CMS relative to each other. Results will be in form of guidelines developers can follow for PLT and cognitive load.
2 Theoretical Background

In this chapter we will look into literature covering the theoretical background, which is then used in chapter 4 and 5 to formulate a research method and later apply it.

Website page load time can be decomposed into three stages:

1. Network
2. Back end
3. Front end

These three parts can be decomposed into detailed sub parts. In this thesis we will briefly discuss the network stage but will not go into depth with the stage. We will mainly focus on the back end and front end stages on a localhost. We consider the back end and front end as parts of a CMS where the core parts of the CMS is the back end. As figure [20] shows in the previous chapter we select four popular CMSs: Drupal, Wordpress, Joomla and Magento to discuss. Except for these three ”real” parts we must take into consideration the cognitive load time, that is to say the time for human viewers to interpret the site subjectively.

2.1 Cognitive load time and aesthetics

Cognitive load can be described as the time necessary for users to understand how to proceed in order to solve a problem.[51] ”Cognitive load: Refers to the total amount of cognitive activity imposed on working memory at an instance in time. The major factor that contributes to cognitive load is the number of items that need to be attended to.”[51]

Brken et al. [5] summarize we can measure cognitive load time by a subjective or objective approach [5]. For the subjective approach we can let users self report their investment in mental effort or stress level. The objective approach is brain activity, behavioural measures, outcome measures and dual task performance measure. Users don’t want to think about where to find features of a website. They have expectations of where to find features, if these are respected it increases satisfaction with the website. [3] Users have mental models of how websites ”should be” designed, which they formed over prolonged time elapses from previous interaction with websites. Cognitive load time is the subjective user perceived load time of the website.

"Positioning navigational areas at expected locations and using specific web objects consistently, helps users to remember them (Oulasvirta, 2004). Several studies show that people expect to find web objects such as search, home button, and navigation at specific locations (Bernard, 2001; Oulasvirta et al., 2005; Roth et al., 2010).”[12]
2.1.1 Flat design and Skeuomorphic design

When designing websites we can try to make them look realistic with shadows and such, this design "tactic" is referred to as skeuomorphic design and often involves emulating 3D real world objects. However the trend in website design currently is to use a concept called "Flat Design" [24], which discourages any use of shadows, gradients and in general emulation of real world 3D objects. The focus of flat design is simplicity and ease. Flat design often focuses on bright colors and very simple easy to understand icons and fonts. Free icons can be found on Flaticons website and free fonts can be found on The google fonts site. Colors play a major role when designing a flat design theme, there are selections of schemes such as the ones existing on the Adobe kuler website. Another interesting aspect of flat design is that it generally tends to reduce load time, especially on mobile devices. "Websites that use flat design also tend to have less data to load within a browser. This means load times can be faster, especially on mobile devices." [24]. Since flat design follows a minimalistic approach to visual presentation it means that it should keep an low cognitive load time. Here are some examples of flat design icons and skeumorphic icons. [24]

Figure 2: Skeumorphic icons to the left an flat icons to the right

source: Steve Johnson, The rise of flat design 2013. [24]

2.2 Content Management Systems

A CMS [4] [35] in web development context enables users with different privileges to view, edit, add and delete content on websites using a graphical user interface sometimes called "control panel" or "administration site". The word content means information in different forms such as: text, images, videos. The word management implies possibilities to add, edit, delete this information. The word system implies a composition of different parts such as a underlying database and graphical user interface working together. [4] [35] Drupal is free open source software developed and maintained by the community. The
CMS can be installed by downloading it from the website and putting it on a local or remote server. Theming, general guidelines and other resources are freely available at the official [Drupal site](#). E-commerce functionality can be achieved by using the free Drupal commerce extension [9].

The popular CMS [Wordpress](#) is used widely for blogs, informative sites and more. To achieve e-commerce functionality developers can install plugins such as: "Orilla Cart", "WP e-Commerce", "MarketPress", "Woo Commerce". Available at the official [Wordpress website](#). Wordpress has a large extension base because of it’s popularity. Some extensions are free while others cost.

Joomla [25] is another open source popular CMS with an special aim at being easy to install and use. Joomla can also be extended by installing extensions from its official website. For E-commerce purposes some of the most popular extensions are "virtueMart", "JoomShopping", "HikaShop".

Magento is different from the previously mentioned CMS because it is specially designed for E-commerce but also not all versions are free. Magento has one free open source version called Magento Community Edition with community based support. In order to acquire the installation files for Magento the user has to register on the [Magento website](#).

### 2.3 Tools and technologies used in the thesis

#### 2.3.1 CSS with Preprocessors

We can use CSS preprocessors in order to make CSS code much more maintainable and easier to develop with. We can choose for instance from the preprocessors STYLUS SASS and LESS. In this thesis we will choose LESS. LESS is compatible with CSS which means we can type regular CSS. But We can also use more complicated features such ass variables, nesting and mixins. The LESS file is always compiled into regular CSS.[16]

#### 2.3.2 GRUNT and GULP our task runners

Task runners automate repetitive work for us. [GRUNT](#) and [GULP](#) have large libraries of automation plug ins for different tasks. In tandem with LESS we will use the "grunt-contrib-less plugin". We can compile LESS files from the terminal by just typing "grunt -watch" from the root folder of the working project as long as we have installed GRUNT on the system and then installed project dependencies (npm install from terminal) and set up the project with the necessary files [21].

#### 2.3.3 Firebug

[Firebug](#) is a Firefox web developer add on which enables us to interact with inspect and edit HTML and CSS in the browser, Monitor network activity (page load time),
debug, execute, log JavaScript, manage cookies. We will primarily use this tool for tweaking CSS and monitoring network.

2.4 Page load time

The speed of light constrains the improvement that can be accomplished on any given page load time from a network (physical) perspective, it is well established that information cannot travel faster than the speed of light in a straight line.

\[ c = 299792458 m/s \]

Fortunately this speed is high [39], making round trip time (RTT) quick. Albeit not quick enough for fields such as stock trading, since such fields aspire instantaneous knowledge of events, however in general cases sufficiently quick. Depending on where the web server is located will have a impact on the entire load time (therefore user experience) of the website. The network improvements available to providers are in line with: building shorter distance cables between point a, b and improving capacity. Increasing bandwidth capacity does not necessarily mean faster page load time. Allocated bandwidth up to 5Mbps implies an decrease in page load time [28]. Contra intuitively it does not impact the PLT much if the user has 10Mbps or 100Mbps, the user will still experience nearly the same page load time. However downloading files is faster with a larger bandwidth but that situation is different partly because the connection only needs to be established once.

This means every one of these "network" improvements is in control of the Internet provider while none is (generally) in the hands of the developer. Developers can choose servers that are closer to the target audience to experience less RTT. The improvements developers can accomplish are therefore mainly in step 2 and 3, that is to say between server, client and on respective parts. This means there should be latency in any observation or experiment if the server is not local. Since the latency is profoundly harder to impact, it is more interesting to look at the back end and front end stage of the performance aspect.

2.4.1 Onload time

The on load time describes the time progressed for all the resources to be downloaded for the presentation of a website. That is to say, the images, HTML, CSS, javascript. The onload event is supported by all the major browsers [53]. This is to the exclusion of for instance Facebook AJAX calls which might take longer to load but are not directly crucial to the presentation of site content. The onload event is fired after all resources, including HTML, CSS and images are loaded and rendered. If one image takes long to retrieve and render the onload event is fired only after the process is done. However users might be able to read text that is loaded before the "slow" image [8].
2.4.2 GET Requests

The GET request method \[52\] is one of the possible communication methods of the Hypertext Transfer Protocol and is used for communication between client and server. The request fetches data from a resource. In the most basic website consisting of only one HTML site the GET request might be the only request from the client to the server. This means the entire HTML page would be transferred in that request as a response from the server to the client request.

```html
<!DOCTYPE html>
<html>
<head>
  <title>Testing GET</title>
</head>
<body>
  <p>This is information</p>
</body>
</html>
```

2.5 Performance Considerations

Performance is a feature. Larger fractions of Internet users are now connected with mobile devices, some are using 3G/4G networks and some are using wifi. The users using 3G/4G networks face a initial unavoidable penalty we have to accept, they belong to the network stage. Speed is important for user experience of the service delivered because of these three arguments \[18\] :

1. better user engagement
2. better user retention
3. higher conversion

The first argument means users experience the website as a better service when visiting it. The second argument tells us users are more likely to be using the site more then once. The third argument can be illustrated with e-commerce where more customers purchase items when visiting the site.

Load time is a feature of the web application, as such the first argument corresponds to the experience of users when visiting the site. The second argument is in relation to keeping users on the site. The third argument can be viewed from an e-commerce sites perspective. It is necessary to avoid latency traps, to illustrate one point, spriting images is time consuming for development but it is very efficient for avoiding latency (but only for the ”old” http standard). Spriting images means taking for example a lot of social media icons and putting them in a single image instead of multiple small one,
this way only one image is sent thus it seems latency is avoided. However the problem is more complex due to mobile devices being limited in memory size and CPU. [18]

Offloading any part of a website to a framework can benefit load time for mobile devices. Experiments show mobile devices are initially 3x slower than desktop counterparts. Computation makes a significant portion of the load time since mobile devices have less CPU power. Offloading to cloud services is one alternative of reduced CPU necessity locally on the device. Research has identified three opportunities by offloading [55]:

1. reduced round trips
2. reduced time in computation
3. consistency with other mobile constrains.

2.5.1 Front end

Front end events are such that occur during and after the website has been received and processed by the web browser. Front end is the part the user can interact with. 80 percent of the time for a page to load occurs in the front end stage [46]. These 80 percent are the biggest fraction of the entire load time. There is less information available on improvement of the front end fraction of the load time. Front end performance improvements are easier in comparison to back end. The front end edits can include rearranging parts of a HTML document or configuring the server while the back end could include activities such as: rewriting a C++ engine which could take months for big sites. Since such a big fraction of the response time is due to front end, it follows that much larger improvements can be made per percent improved. [46]

2.5.2 Back end

Events that occur on the web server (for instance Apache, IIS, nginx) and database or "in" the CMS are what is considered back end [45]. This is a general term for database load time, server load.

Virtual denormalization can cause performance improvement. By denormalizing the database experiment show optimization ratio of 74.2%. [1]

2.5.3 Twitter Bootstrap

A mobile first, free, open source, framework which is out of the box responsive to the visiting screen size. Comes with a minimum version for smaller file sizes. Can be easily themed by over writing desired CSS in a separate file and then loading it into the HTML file after loading in the bootstrap CSS.
2.5.4 Measuring performance

Depending on what software is used will affect the outcome of the result. If we use Firefox to measure performance will produce different results then if we used Internet explorer. Benchmarking techniques such as Apache benchmark and Siege can be used on the server. Improving client download and rendering by looking into browser rendering content and tools such as: firebug, page speed and yahoo’s Yslow. These tools help in optimizing web pages by providing developers with visual and textual data. Variable caching can be done with tools as memcached \[36\]. In order to know how many times performance has been improved we need to measure ”base line performance” \[44\].

2.6 Conclusion

All of the discussed CMSs are written in PHP. Joomla, Drupal, Wordpress are open source and Magento has a community open source edition. The CMSs have many free extensions available on their official sites. Magento does not need any extensions to provide e-commerce functionality. This means that any experiment conducted will reflect the extensions applied on Drupal, Wordpres and Joomla as well as the interior of the systems. This means that the way E-commerce extensions are programmed will affect outcome significantly. Flat Design is currently popular and also seems to have a low cognitive load time since it is minimalistic. Much material can be downloaded on the internet with open source licenses. The Bootstrap framework will make websites mobile responsive.

Since 80 percent of PLT is on the client device it follows that a larger proportion of improvements can be made on the front end part of website projects. If we assume a 10% improvement on both the front end and back end part of a project, and assume the front end to be a fraction of 80% then it follows that:

\[
80 \cdot 0.1 = 8 \rightarrow 8\% \text{ overall improvement} \\
\]

while

\[
20 \cdot 0.1 = 2 \rightarrow 2\% \text{ overall improvement} \\
\]

This indicates that much more can be done on the front-end relative to the back-end.

Internet speed only increases PLT to a certain level (10Mps) then it evens out. Tools frequently used to observe front end performance are Firebug, Yslow. Base line performance needs to be observed in order to know the amount of improvement that has been done.
3 Literature Review

This chapter discusses the ongoing research and results of this with relation to the thesis research questions. The chapter focuses on the cognitive load time, page load time and website performance optimization.

3.1 Cognitive Load Time and Aesthetics

3.1.1 Visit duration

General websites consider task completion time to be better if lower. This is because it intuitively feels like the cognitive load time is lower and this should mean that user satisfaction will be higher. However in E-commerce time saving is not of equal priority, since extra customer time might increase exposure to more information. This does not mean that it is acceptable if customers find tasks hard, rather it is acceptable with longer visit durations relative to other non-commerce type of websites. [20]

3.1.2 Placement of objects and site layout

Humans seem to be sensitive to symmetry and we associate it to the aesthetic "beauty" of a website. The male gender seems to be more sensitive to symmetric positioning in a study of 60 people in 2010 (30 males). But this might be due to the sample size of the population. The vertical symmetry of a site seems to be praised by the male gender and appreciated by the sample population. The male gender on the sample population seems to find asymmetrical sites less beautifully [49].

108 screen shots of real sites where shown in a study during 2013 as stimuli to users, well familiar with websites. The e-commerce sites categorized sites contained a shopping cart, navigation, login and search. 36 images were of the e-commerce type. The participants eye movement was recorded and analyzed. The experiment was conducted on 40 students with 20/20 eye vision. The results of the study show in alignment with the hypothesis that the more typically the website object was located the faster the subjects would find it. The difference in cognitive load time ranged from 300ms to 2s. For web shops sites login area and search field seem to be sensitive to non typical positioning, participants where less affected by logos, navigation bars and shopping carts. “On online shop pages, the login area (r= .81 - .87) and search field (r=.83 - .86) seem to be most affected by location” [42]. This means users have standard expectations on appearances of web objects and on e-commerce sites they are sensitive to login area and search field are the most affected, however the same study shows that navigation and shopping cart are also sensitive to positioning. Maps showing eye tracking movement back up these statements in previous research. For users to quickly visually orient them selves means they will quickly find what they are looking for.

So where do we place these objects according to user mental models? A study conducted in 2010 on 516 users whom where asked to position objects for three different types of sites, these types where e-commerce, news and company sites. Figure 3 shows the
result for the online shopping sites. We can clearly see that the shopping specific

Figure 3: Mental Positioning of objects for e-commerce sites on a sample population of 516 (334 women, 179 men, 3 unspecified)

source: Roth et al. 2010 Mental models for web objects: Where do users expect to find the most frequent objects in online shops, news portals, and company web pages?

objects such as cart and login are expected predominantly to be in the top right corner of the page. If advertisement area does exist it shuld be placed to the right leaving the main area unclutered. The navigation should be to the left.

3.1.3 Menu type

A important aspect of navigation in e-commerce is the menu showing products and categories. A study in 2011 on a sample population of 120 shows that the best suited
type of menu is a vertical non dynamic menu. “Vertical menu. All group titles and navigation items were always visible.” Figure 4 shows the vertical menu layouts in the middle of the three. This is the preferable menu type according to the study.

3.1.4 Task and page complexity

When it comes to task complexity such as a user viewing a product etc. The size of a site (screen length) is directly related to site complexity. If scrolling is necessary to find information it is consider more complicated in regard to cognitive load. For users visiting the site looking to buy a specific item, there should definitely be a search box on the e-commerce site and the search results should not present many advertising words, pictures, hyper links unrelated to the result of the query. For users who are not visiting the site with a specific item targeted, it seems according to the findings that moderate complexity is preferable. That is to say if a user navigates at random to say a category...
then he should be presented with moderate complexity. The findings indicate this can lead to longer sessions on the site and more rich exploration of products. [54]

Figure 5: Medium complexity is preferable for visitors not looking for a specific product

source: Qiuzhen Wang et al. 2014 An eye-tracking study of website complexity from cognitive load perspective. [54]

In a study conducted in 2012[50] a sample population of 80 was asked to judge the relationship between aesthetics and usability. The study found that it is more important for an e-commerce website to be usable then aesthetically pleasing. Good aesthetics do not affect users perception of usability but usability does affect user perception of post use aesthetics. The study shows that usability is to prioritize before aesthetics.

3.1.5 Comparison of CMSs

A study [38] was conducted comparing the three CMSs: Joomla, Drupal and Wordpress, features the systems provide and discuss them in relation to other CMSs on the market. The three CMS discussed have much higher installation count in comparison to other CMSs on the market. The three selected CMSs selected by the survey where the largest market share holders. 70 percent of the 4000 participants of the surveys use Joomla, Drupal or Wordpress. The authors discussed material printed about CMSs and noted that the three systems have the highest count available material in comparison to other CMS. Once the authors established the dominance of Drupal, Joomla and Wordpress they started comparing them to each other. They conducted a popularity comparison in which they observed Joomla got the highest Google search query volume. Drupal got second place, Wordpress got sixth place. When comparing budget for the three CMSs they notice Drupal is most expensive but this is due to the fact that Drupal often comes with more functionality. When comparing features they note performance wise Joomla scores the lowest while Drupal and Wordpress have almost the same score. The authors also discussed why Drupal and Wordpress scored highest on social bookmarking. Joomla scored the highest in a popularity comparison. The conclusion of the study is that Joomla, Wordpress and Drupal where the most efficient CMSs. Joomla is selected as the best CMS for Novice users in content management. Drupal is the best for flexibility and Wordpress is preferable for adding add-ons and has a high documentation support.
The three CMSs discussed Drupal, Joomla and Wordpress have been subjected to experiments over CMSs performance such as page load time and page size for a range of categorically equivalent web pages. The experiments carried out are specific for software versions described in one table. The conclusion of the experiments is that Drupal should be chosen for informative websites because of small page sizes. However if one is to have multi object site then Joomla is the choice recommended. [37]

3.2 Performance Optimization

3.2.1 Request overhead

In order for the client to fetch data from the server it must ask the server for the data. The client browser does so by sending a header request. This header request contains information. The information takes up bandwidth. When a HTTP request is made, relative cookies have to be sent with the request. Users do not have equal download and upload internet connections. This means an delay will be caused when ever user needs to send data back to the server. this can be countered with minimizing request size and by serving static content from a domain without cookies. [17]

3.2.2 Payload size

Http Compression response is one of the techniques suggested for bandwidth performance improvement, the idea is simply to send smaller files to save network bandwidth. Conducted experiments verify impact on performance. The results of the experiments give on average improvements of 77%, that is to say the page download time was reduced by 77% on average after using the GZIP compression mechanism proposed. The study contains a Case-Study using AJAX which is described as "a real money saver concerning network bandwidth" this is because the nature of the asynchronous technique is for the page not to reload. When using AJAX techniques performance is improved up to 98.2% in comparison to the traditional site not using AJAX. [45]

3.3 Conclusion of the literature review

Cognitive load time seems to be affected by the location of web objects. Users have mental models how websites work, these mental models must be respected if users are to perceive websites as easy to understand. Users have clear mental models of where they expect e-commerce functions to be present. We also have access of a sample of these positions which we should follow when designing the layout of e-commerce sites. Users seem to appreciate symmetry as beautiful (especially men). The menu should be vertical and unfolded, that is to say non dynamic for the categories and products. There should be a brand name which also functions as a home button on the top left corner. Functionality seems to be more valued then aesthetics and it also affects the
perception of the aesthetics afterwards, aesthetics do not seem to influence the perception of functionality.

Many methods are available for page load time performance optimization such as: Cache optimization, Payload size, Parallel requests, Request overhead improvements. It seems clear that these are not exclusive but rather work in tandem with each other.
4 Method

The method must reflect the research questions. If it does not, it will automatically fail in capability to produce results of value. The method of choice should have an high validity, since this would enable other research to verify or falsify results. That is to say results should be repeatable. Since the question is of such a nature that it can be done in an empiric manner, it becomes preferable with a quantitative approach.

To answer the research questions experiments is selected. Experiments have an high value in terms of validity but can critically be considered expensive to set up and configure. The results produced by the experiments will only be valid on the software on which the experiments are conducted.

The Literature review in the previous chapter was done until the experiments are finished, by searching different scientific sources with keywords. Once an interesting article has been found citations are looked at, key articles are the ones with many citations to them. Literature needs to be relatively new in this context (preferably submitted after 2008). However literature on psychology of users can be older (for instance 1999). Once the experiments are finished the literature review will also stop being done.

4.1 Motivation For Experiments

With experiments we get control over the variables effecting the outcome, we can prove disprove relationships and reason to cause effect events. Experiments give high validity and can easily be reproduced to verify or discard the results if all the necessary steps are documented and accessible. Experiments are conducted in certain steps. Experiment is selected as the method because of the high control the method allows. This means that any future research can discard or verify the results from this research. If they have the same variables. The data produced by the experiments will be used for answering both research questions. The individual tests of different CMSs will produce data in terms of time. This data can be divided to see what an speed up has been achieved.

4.1.1 Risks with experiments

Experiments produce a case of reality and don’t reflect other scenarios but those simulated. However they are reflecting them with a high repeatability. Once an hypothesis exists we want to discard it’s null statement (the opposite of what we consider to be truthful). This reasoning is strong because any statement must be true or false, or left to theory. We must however be careful about errors in what we perceive from the experiment. There are type I and type II errors to consider and be careful of. The first type is when we reject a true statement(Null hypothesis) and the second is when we accept a false statement(Hypothesis). The numbers should feel intuitive since it
covers all four combinations of a initial statement (with binary values), if we consider the opposites to the errors as hits. [33]

4.2 Constructing Experiments

In order to gather necessary data we need to conduct series of experiments. The data can then later be analyzed by simple arithmetic mathematics and presented for instance with bar charts. All of the different experiments include installing new configurations of the CMSs. This way we can run the experiments at the same time even tough preparations might have taken a while.

4.2.1 Developing a standard theme

In order to follow the aesthetic guidelines we should develop an standard theme which follows the aesthetics from the literature review and technical background chapter.
5 Experiments (Benchmarks)

5.1 Definition

The purpose is to find results from which we can derive results. During the experiments all results from the Firebug tool networking section are noted in tables. These results are presented in tables 4, 7, 6, 5 and in figures 14, 15, 16. The objective of these experiments is to answer the research questions.

Table 1: The environment of the experiments

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Vendor</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>Ubuntu</td>
<td>Canonical</td>
<td>14.04</td>
</tr>
<tr>
<td>Webserver</td>
<td>Apache HTTP server</td>
<td>The Apache Software Foundation</td>
<td>2.4.7</td>
</tr>
<tr>
<td>Database</td>
<td>Mysql community server</td>
<td>Oracle Corporation</td>
<td>5.5.37</td>
</tr>
<tr>
<td>Scripting lang</td>
<td>PHP</td>
<td>The PHP Group</td>
<td>5.4.12</td>
</tr>
<tr>
<td>Browser</td>
<td>Firefox</td>
<td>Mozilla</td>
<td>29.0</td>
</tr>
<tr>
<td>Browser Extension</td>
<td>Firebug</td>
<td>The Firebug team</td>
<td>1.12.8</td>
</tr>
<tr>
<td>Browser Extension</td>
<td>Yslow</td>
<td>Open source</td>
<td>3.1.8</td>
</tr>
<tr>
<td>Hardware CPU</td>
<td>i7 3630QM</td>
<td>Intel</td>
<td>2.4GHZ</td>
</tr>
<tr>
<td>Hardware RAM</td>
<td>DDR3 1600 MHz</td>
<td>Samsung</td>
<td>8GB</td>
</tr>
</tbody>
</table>

Software and hardware specifications are shown in table 1 for observing and conducting the experiments. The software is installed on the computer running the experiments.

Table 2: Content management systems

<table>
<thead>
<tr>
<th>Name</th>
<th>Extensions</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Magento Community Edition</td>
<td></td>
<td>1.9.1.0</td>
</tr>
<tr>
<td>Drupal</td>
<td>Commerce</td>
<td>7.28 and 7.x-1.9</td>
</tr>
<tr>
<td>Wordpress</td>
<td>Woo commerce</td>
<td>3.9.1 and 2.1.9</td>
</tr>
<tr>
<td>Joomla</td>
<td>HikShop</td>
<td>3.3 and 2.3.1</td>
</tr>
</tbody>
</table>

The Drupal commerce module requires installing additional modules to work. These are listed in table 3.
Table 3: Commerce requires these modules

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Addressfield</td>
<td>7.x-1.0-beta5</td>
</tr>
<tr>
<td>Chaos tool suite</td>
<td>7.x-1.4</td>
</tr>
<tr>
<td>Entity</td>
<td>7.x-1.5</td>
</tr>
<tr>
<td>Rules</td>
<td>7.x-2.7</td>
</tr>
<tr>
<td>Views</td>
<td>7.x-3.8</td>
</tr>
</tbody>
</table>

Table 2 shows which CMSs will be used in these experiments.

5.2 Planning

The standard installation of Magento offers E-commerce functions out of the box. However to actually do the experiments with the other CMSs we must first make Drupal, Joomla and Wordpress E-commerce capable by installing necessary extensions for each CMS.

We need dummy data and a universal theme, which is designed using HTML, CSS and JavaScript. This theme will be used on the different CMSs, each CMS needs to be separately themed into the universal theme.

Dummy content data in media and text format was generated, the pictures are downloaded from Flickr while the text is lorem ipsum dummy text.

An realistic amount of pages need to be presented to emulate a running E-commerce site.

We need to collect baseline data, that is to say data which we will show the amount of speed up due to the optimization.

The independent variable in the experiments is time and it is central to all the experiments. Time will be measured with the Firefox extension, Firebug.

It is interesting to see the time penalty extensions add to the default installations of Wordpress, Drupal and Joomla with the rest of the systems unaltered in any way.

5.3 Operation

During the Experiments it is important to keep the same surrounding settings since these might affect the load on the web browser and apache web server. Every experiment on the same CMS will produce some variations in PLT, Onload event fired time and the GET request (see table 8) because of the CPU load on the apache server and web browser. These loads are best treated if all experiments are isolated in the best way possible, that is to say the computer should exclusively be used for experiments while the experiments are conducted. During the experiments the screen size is set to 1920x1080, this plays part because other some of the windows will adapt for mobile screen sizes depending on their size.
5.3.1 Creating a standard theme

We can select out of the Adobe kuler selection a color scheme for the site. The colors will be central to the site. We select the Moxy theme because of it’s popularity in jul 2014.

Figure 6: The popular Moxy theme from Adobe Kuler

- Color Dark Blue Hex: 2C3E50
- Color Red Hex: E74C3C
- Color Dark White Hex: ECF0F1
- Color Blue Hex: 3498DB
- Color Dark Grey Hex: 646464
The layout in figure 7 is derived from the literature review for the site theme. The layout matches the users mental mode as described in the literature review. Users should experience a low cognitive load time with the proposed theme. This layout does not meet any necessary mental model for non-desktop devices but it will resize automatically with the bootstrap framework.

For the approximately common theme in the thesis we use the flat font "Lato" available from Google Fonts and icons are used from the site Flaticon.com. All of the material used in the theme will be consistent with flat design principle, shadows, gradients and such are avoided. Focus is put on the colors from the Moxy scheme found on adobe kuler.

From the proposed layout a standard theme is created. Figure 16 shows the landing page of the standard theme. The images on the universal look are taken from flickr.com, published under creative commons liscense. The content is lorem ipsum
dummy text. For the shop page we create a vertical menu on the left side of the store.

Figure 8: The first page of the suggested theme

The landing page of the standard theme.
Figure 9: The Shop Page

The shop product view page of the standard theme.
Using the CSS framework bootstrap an universal theme is developed for the different CMSs see figure [16][9] and [10]. This theme is then specially created for each of the CMS systems. The standard theme should is based on a witter bootstrap and is then pushed into all different CMSs. This way all variates of the theme will be the same size. This will push the benchmarks closer. This way Drupal 7 will also be mobile responsive. There are open source base themes for bootstrap in respect to the different CMSs.

- Bootstrap for Wordpress with Roots
- The Bootstrap Blocks theme for Drupal 7
• The H5BPTB bootstrap theme for Joomla
• The Magento Boilerplate Bootstrap theme for Magento.

With these start points we now develop an equivalent look for all of the CMSs based on the universal theme.

5.3.2 Creating dummy data

Exact equal dummy data is generated. The text and images inserted into products need to be same size and content this way the data is equally large for all products on the different databases. We create a simple category page in the shop section with 12 laptop products on 4 rows. The information of the products is lorem ipsum text with images published under the creative commons license at the site flickr.com. The titles and prices are identical on all sites. Two additional empty categories are created but not filled with data, these are tablets and mobile phones.

5.3.3 Gathering Baseline performance statistics out of the box installations standard themes

The CMS software is installed with the default themes and the default modules. No modules are added. Everything is local. The software versions are available in table 2, 3 and 1. Firebug is used to observe the time necessary for the different pages to load the PLT is collected. Table 4 and 5 gathers only baseline performance then this statistics will be used to derive results. These baseline experiments are necessary to study the penalty for adding modules to Wordpress, Drupal and Joomla in order to make them e-commerce capable.

Table 4: Singlerun baseline PLT No Modules Installed No Theming No Content Cache disabled

<table>
<thead>
<tr>
<th>CMS</th>
<th>PLT</th>
<th>Onload event fired</th>
<th>Operation ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drupal</td>
<td>248ms</td>
<td>341ms</td>
<td>1a</td>
</tr>
<tr>
<td>Wordpress</td>
<td>161ms</td>
<td>334ms</td>
<td>2a</td>
</tr>
<tr>
<td>Joomla</td>
<td>145ms</td>
<td>330ms</td>
<td>3a</td>
</tr>
<tr>
<td>Magento</td>
<td>739ms</td>
<td>769ms</td>
<td>4a</td>
</tr>
</tbody>
</table>

Table 5: Single run baseline PLT No Modules Installed No Theming No Content cache enabled

<table>
<thead>
<tr>
<th>CMS</th>
<th>PLT</th>
<th>Onload event fired</th>
<th>Operation ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drupal</td>
<td>84ms</td>
<td>279ms</td>
<td>1b</td>
</tr>
<tr>
<td>Wordpress</td>
<td>100ms</td>
<td>244ms</td>
<td>2b</td>
</tr>
<tr>
<td>Joomla</td>
<td>41ms</td>
<td>349ms</td>
<td>3b</td>
</tr>
<tr>
<td>Magento</td>
<td>129ms</td>
<td>640ms</td>
<td>4b</td>
</tr>
</tbody>
</table>
Table 5 shows PLT with cache enabled for the default installation with the default themes. During the experiment if the browser is refreshed it gives different values this is due to the background processes on the apache server and Firefox browser.

### 5.3.4 Gathering baseline performance statistics with modules installed standard themes

The CMS software is installed with the default themes. Necessary modules to make Wordpress, Joomla and Drupal e-commerce capable are installed from table 2 and 3. These local installations are now e-commerce capable and we have baseline data from the previous two experiments. Firebug is used again to observe the PLT. The results are from a single run of experiments. The reason for conducting experiments without any extensions is to gather baseline data. These experiments are also single run experiments.

Table 6: Single run baseline PLT Modules Installed No Theming No Content cache Disabled

<table>
<thead>
<tr>
<th>CMS</th>
<th>PLT</th>
<th>Onload event fired</th>
<th>Operation ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drupal</td>
<td>372ms</td>
<td>415ms</td>
<td>5a</td>
</tr>
<tr>
<td>Wordpress</td>
<td>280ms</td>
<td>525ms</td>
<td>6a</td>
</tr>
<tr>
<td>Joomla</td>
<td>260ms</td>
<td>971ms</td>
<td>7a</td>
</tr>
<tr>
<td>Magento</td>
<td>795ms</td>
<td>887ms</td>
<td>8a</td>
</tr>
</tbody>
</table>

Table 7: Single run baseline PLT Modules Installed No Theming No Content cache enabled

<table>
<thead>
<tr>
<th>CMS</th>
<th>PLT</th>
<th>Onload event fired</th>
<th>Operation ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drupal</td>
<td>187ms</td>
<td>539ms</td>
<td>5b</td>
</tr>
<tr>
<td>Wordpress</td>
<td>232ms</td>
<td>557ms</td>
<td>6b</td>
</tr>
<tr>
<td>Joomla</td>
<td>152ms</td>
<td>801ms</td>
<td>7b</td>
</tr>
<tr>
<td>Magento</td>
<td>135ms</td>
<td>679ms</td>
<td>8b</td>
</tr>
</tbody>
</table>

### 5.3.5 The Experiment Path with dummy data

In the experiments with dummy data all CMS’s responded to the following simulated user requests.

**First**  Page Shop with laptops as category

**Second**  Product view with Vaio2 as product

**Third**  Checkout with Vaio2 in Cart
The pages loaded almost identical themes based on the previously described universal theme. The following three pictures show how the Magento pages looked during the experiments.

Figure 11: This image shows the Magento product grid view with category laptops.
Figure 12: This image shows the magento view of the product "vaio 2".
and finally the checkout page

Figure 13: This image shows the magento view of the checkout page.
5.3.6 Gathering Baseline performance statistics with modules installed, dummy content with default themes

We first measured the time the default themes took to retrieve the above pages. No cache means disabling both browser and CMS cache. Figure 14 is produced. The data in this table can now be used to understand the impact of the universal theme. The dummy content is present in figure 14. This means it is exactly the same except that it lacks the universal theme.

Figure 14: This data shows experiment output of the experiments with standard themes, the time is in milliseconds.

<table>
<thead>
<tr>
<th>Module</th>
<th>Theme</th>
<th>Operation</th>
<th>Page Load Time</th>
<th>Onload Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drupal With Cache</td>
<td>Product Display</td>
<td>GET Request (For Page)</td>
<td>263</td>
<td>250</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page Load Time</td>
<td>263</td>
<td>68</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Onload Time</td>
<td>489</td>
<td>250</td>
</tr>
<tr>
<td>Drupal Without Cache</td>
<td>Product Display</td>
<td>GET Request (For Page)</td>
<td>493</td>
<td>303</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page Load Time</td>
<td>527</td>
<td>625</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Onload Time</td>
<td>759</td>
<td>564</td>
</tr>
<tr>
<td>Joomla With Cache</td>
<td>Product Display</td>
<td>GET Request (For Page)</td>
<td>134</td>
<td>209</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page Load Time</td>
<td>349</td>
<td>649</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Onload Time</td>
<td>541</td>
<td>872</td>
</tr>
<tr>
<td>Joomla Without Cache</td>
<td>Product Display</td>
<td>GET Request (For Page)</td>
<td>220</td>
<td>175</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page Load Time</td>
<td>762</td>
<td>688</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Onload Time</td>
<td>800</td>
<td>636</td>
</tr>
<tr>
<td>Magento With Cache</td>
<td>Product Display</td>
<td>GET Request (For Page)</td>
<td>316</td>
<td>348</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page Load Time</td>
<td>316</td>
<td>776</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Onload Time</td>
<td>692</td>
<td>761</td>
</tr>
<tr>
<td>Magento Without Cache</td>
<td>Product Display</td>
<td>GET Request (For Page)</td>
<td>605</td>
<td>670</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page Load Time</td>
<td>806</td>
<td>1360</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Onload Time</td>
<td>1270</td>
<td>1230</td>
</tr>
<tr>
<td>Wordpress With Cache</td>
<td>Product Display</td>
<td>GET Request (For Page)</td>
<td>278</td>
<td>221</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page Load Time</td>
<td>278</td>
<td>678</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Onload Time</td>
<td>635</td>
<td>616</td>
</tr>
<tr>
<td>Wordpress Without Cache</td>
<td>Product Display</td>
<td>GET Request (For Page)</td>
<td>293</td>
<td>246</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Page Load Time</td>
<td>801</td>
<td>685</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Onload Time</td>
<td>1000</td>
<td>806</td>
</tr>
</tbody>
</table>
5.3.7 Gathering Baseline performance statistics with modules installed, dummy content and universal themes

This is the result of the universal theme experiments with dummy content. No cache means disabling both browser and CMS cache. The table in figure [15] is intended to be used for two purposes, firstly we can take the difference from the figure [14] to see the impact of the push to the universal theme. Secondly we can use these results as a baseline for our optimized results.

Figure 15: This data shows singlerun experiment output of experiments with the universal theme the time is in milliseconds

<table>
<thead>
<tr>
<th>Drupal With Cache</th>
<th>Product</th>
<th>Display</th>
<th>View Product (vaio 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (For Page)</td>
<td>282</td>
<td>86</td>
<td>145</td>
<td></td>
</tr>
<tr>
<td>Page Load Time</td>
<td>282</td>
<td>86</td>
<td>145</td>
<td></td>
</tr>
<tr>
<td>Onload Time</td>
<td>811</td>
<td>488</td>
<td>570</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Drupal Without Cache</th>
<th>Product</th>
<th>Display</th>
<th>View Product (vaio 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (For Page)</td>
<td>507</td>
<td>312</td>
<td>382</td>
<td></td>
</tr>
<tr>
<td>Page Load Time</td>
<td>625</td>
<td>560</td>
<td>906</td>
<td></td>
</tr>
<tr>
<td>Onload Time</td>
<td>1260</td>
<td>861</td>
<td>860</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Joomla With Cache</th>
<th>Product</th>
<th>Display</th>
<th>View Product (vaio 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (For Page)</td>
<td>193</td>
<td>178</td>
<td>198</td>
<td></td>
</tr>
<tr>
<td>Page Load Time</td>
<td>193</td>
<td>178</td>
<td>198</td>
<td></td>
</tr>
<tr>
<td>Onload Time</td>
<td>683</td>
<td>624</td>
<td>628</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Joomla Without Cache</th>
<th>Product</th>
<th>Display</th>
<th>View Product (vaio 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (For Page)</td>
<td>249</td>
<td>197</td>
<td>219</td>
<td></td>
</tr>
<tr>
<td>Page Load Time</td>
<td>1040</td>
<td>928</td>
<td>1050</td>
<td></td>
</tr>
<tr>
<td>Onload Time</td>
<td>1000</td>
<td>878</td>
<td>1010</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Magento With Cache</th>
<th>Product</th>
<th>Display</th>
<th>View Product (vaio 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (For Page)</td>
<td>296</td>
<td>318</td>
<td>384</td>
<td></td>
</tr>
<tr>
<td>Page Load Time</td>
<td>501</td>
<td>539</td>
<td>553</td>
<td></td>
</tr>
<tr>
<td>Onload Time</td>
<td>753</td>
<td>723</td>
<td>860</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Magento Without Cache</th>
<th>Product</th>
<th>Display</th>
<th>View Product (vaio 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (For Page)</td>
<td>601</td>
<td>629</td>
<td>729</td>
<td></td>
</tr>
<tr>
<td>Page Load Time</td>
<td>1170</td>
<td>1540</td>
<td>1510</td>
<td></td>
</tr>
<tr>
<td>Onload Time</td>
<td>1450</td>
<td>1490</td>
<td>1450</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Wordpress With Cache</th>
<th>Product</th>
<th>Display</th>
<th>View Product (vaio 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (For Page)</td>
<td>265</td>
<td>188</td>
<td>218</td>
<td></td>
</tr>
<tr>
<td>Page Load Time</td>
<td>265</td>
<td>188</td>
<td>218</td>
<td></td>
</tr>
<tr>
<td>Onload Time</td>
<td>696</td>
<td>649</td>
<td>614</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Wordpress Without Cache</th>
<th>Product</th>
<th>Display</th>
<th>View Product (vaio 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (For Page)</td>
<td>278</td>
<td>185</td>
<td>193</td>
<td></td>
</tr>
<tr>
<td>Page Load Time</td>
<td>581</td>
<td>533</td>
<td>1140</td>
<td></td>
</tr>
<tr>
<td>Onload Time</td>
<td>975</td>
<td>887</td>
<td>1160</td>
<td></td>
</tr>
</tbody>
</table>
5.3.8 Gathering performance statistics for the optimized installations

Figure [16] presents the result of the experiments with the optimizations for the various CMSs. This data is intended to be used with the table in figure [15] in order to see the impact on the various measured data is positive or negative.

Figure 16: This data shows single run experiment output of experiments with the optimized CMSs, using the dummy data and universal theme.

<table>
<thead>
<tr>
<th></th>
<th>Product/Display</th>
<th>View Product (via 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drupal With Cache</td>
<td>GET Request (for Page)</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Page Load Time</td>
<td>18</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Onload Time</td>
<td>518</td>
<td>454</td>
</tr>
<tr>
<td>Drupal Without Cache</td>
<td>GET Request (for Page)</td>
<td>337</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td>Page Load Time</td>
<td>931</td>
<td>372</td>
</tr>
<tr>
<td></td>
<td>Onload Time</td>
<td>1040</td>
<td>707</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Joomla With Cache (JCH PLUGINS)</th>
<th>Product/Display</th>
<th>View Product (via 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (for Page)</td>
<td>54</td>
<td>53</td>
<td>211</td>
</tr>
<tr>
<td>Page Load Time</td>
<td>54</td>
<td>53</td>
<td>850</td>
</tr>
<tr>
<td>Onload Time</td>
<td>540</td>
<td>554</td>
<td>817</td>
</tr>
<tr>
<td>Joomla Without Cache</td>
<td>GET Request (for Page)</td>
<td>71</td>
<td>83</td>
</tr>
<tr>
<td>Page Load Time</td>
<td>452</td>
<td>402</td>
<td>729</td>
</tr>
<tr>
<td>Onload Time</td>
<td>666</td>
<td>488</td>
<td>991</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Magento With Cache</th>
<th>Product/Display</th>
<th>View Product (via 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (for Page)</td>
<td>214</td>
<td>257</td>
<td>359</td>
</tr>
<tr>
<td>Page Load Time</td>
<td>356</td>
<td>420</td>
<td>508</td>
</tr>
<tr>
<td>Onload Time</td>
<td>534</td>
<td>524</td>
<td>706</td>
</tr>
<tr>
<td>Magento Without Cache</td>
<td>GET Request (for Page)</td>
<td>452</td>
<td>486</td>
</tr>
<tr>
<td>Page Load Time</td>
<td>597</td>
<td>922</td>
<td>716</td>
</tr>
<tr>
<td>Onload Time</td>
<td>723</td>
<td>806</td>
<td>866</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Wordpress With Cache</th>
<th>Product/Display</th>
<th>View Product (via 2)</th>
<th>Checkout</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET Request (for Page)</td>
<td>118</td>
<td>170</td>
<td>155</td>
</tr>
<tr>
<td>Page Load Time</td>
<td>186</td>
<td>170</td>
<td>155</td>
</tr>
<tr>
<td>Onload Time</td>
<td>567</td>
<td>562</td>
<td>522</td>
</tr>
<tr>
<td>Wordpress Without Cache</td>
<td>GET Request (for Page)</td>
<td>215</td>
<td>174</td>
</tr>
<tr>
<td>Page Load Time</td>
<td>598</td>
<td>700</td>
<td>1120</td>
</tr>
<tr>
<td>Onload Time</td>
<td>798</td>
<td>754</td>
<td>976</td>
</tr>
</tbody>
</table>

5.3.9 Gathering performance statistics variation of experiments

Since the output data of the experiments is based on single runs we need to have some variation data. However we will only redo one experiment setup to gather this data.
Table 8: Experiment variations cache enabled Magento optimized product vaio2 view

<table>
<thead>
<tr>
<th>GET</th>
<th>Page Load Time</th>
<th>Onload Time</th>
<th>Operation ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>251ms</td>
<td>410ms</td>
<td>530ms</td>
<td>Variation1</td>
</tr>
<tr>
<td>252ms</td>
<td>431ms</td>
<td>594ms</td>
<td>Variation2</td>
</tr>
<tr>
<td>260ms</td>
<td>418ms</td>
<td>507ms</td>
<td>Variation3</td>
</tr>
<tr>
<td>250ms</td>
<td>418ms</td>
<td>514ms</td>
<td>Variation4</td>
</tr>
<tr>
<td>266ms</td>
<td>495ms</td>
<td>503ms</td>
<td>Variation5</td>
</tr>
<tr>
<td>248ms</td>
<td>406ms</td>
<td>574ms</td>
<td>Variation6</td>
</tr>
<tr>
<td>260ms</td>
<td>405ms</td>
<td>583ms</td>
<td>Variation7</td>
</tr>
<tr>
<td>248ms</td>
<td>427ms</td>
<td>526ms</td>
<td>Variation8</td>
</tr>
<tr>
<td>260ms</td>
<td>499ms</td>
<td>545ms</td>
<td>Variation9</td>
</tr>
<tr>
<td>252ms</td>
<td>439ms</td>
<td>505ms</td>
<td>Variation10</td>
</tr>
<tr>
<td>254.7ms</td>
<td>434.8</td>
<td>538.1ms</td>
<td>Average</td>
</tr>
<tr>
<td>18ms</td>
<td>94ms</td>
<td>89ms</td>
<td>Highest - lowest</td>
</tr>
</tbody>
</table>

From table 8, we can see that the difference between the single run experiment (Fig 16) of the optimized magento installation and average value is 2.3ms. The lowest time for the GET request was 248ms and the highest was 266ms this makes a range of 18ms. With this in mind we can calculate a error of 2.3ms in the GET request for the product view page of the vaio2.

The difference between the average PLT and the optimized Magento installation is 14.8ms and the difference in on load event time is 14.1ms. The PLT ranged from 405ms to 499ms. That is to say it had difference of 94ms.

The onload event was fired on average after 538ms. The difference between this onload event time and the time from the figure 16 is 14ms. The values range from 505 to 594. The difference between highest and lowest value is 89ms.

34
6 Result (Evaluation)

There is clearly a correlation between added modules, plugins and extensions to Drupal Joomla and Wordpress and their individual PLT. The penalty for adding modules to Joomla, Drupal and wordpress is clearly visible inf figure 17 and 18 when taking the time difference with and without the extensions and cache.

Figure 17: Comparison cache disabled experiments constructed by taking the difference between table 4 and table 6. The Y axis represents time difference in milliseconds.

![Figure 17: Comparison cache disabled experiments](image)

Figure 18: Comparison cache enabled experiments constructed by taking the difference between table 5 and table 7. The Y axis represents time difference in milliseconds.

![Figure 18: Comparison cache enabled experiments](image)

Figure 17 shows a relatively large time difference in PLT between the first set of experiments with no cache enabled. The first experiments had only the default instal-
lation the second had modules. By taking the difference we get bar chart from figure [17]. Which shows that all CMSs increased in PLT and onload event. The reason magento increased is coincidence and is related to background load to the apache web server and Firefox web browser. We can see in figure [18] similar results but with larger varieties for the PLT and generally the onload event fired.

6.1 Front End Guidelines

6.1.1 Default Aesthetics

Figure 19: The different CMSs installed showing their front page.

![Figure 19: The different CMSs installed showing their front page.](image)

Figure [19] shows the 4 standard themes on the landing pages for the 4 CMSs. The themes are provided with the CMSs out of the box. All of the themes are mobile responsive except for the standard Drupal bartik theme. The experiment output in table [7] shows that Magento outperforms the other CMSs with cache enabled. In this thesis a standard theme was developed following previous research for lower cognitive load time. The 4 CMSs shown in figure [20] have an almost identical theme. This is the result of the cognitive load time consideration.

6.2 Improvement for the CMSs

Figure [23] Shows the improvements of the CMSs after optimization has been applied.
Figure 20: The different CMSs installed showing their product view page with the universal theme.

Figure 21: Improvements of the CMSs after optimization for the different steps of the experiments
This table is constructed by taking the difference off 16 and 15. By constructing this table we can now reflect if our optimization has caused any improvements to the CMSs. There are some negative values in the table of figure 21. This is because not all of the optimization worked as intended. However the overall results indicate clear performance improvement in all CMSs.

6.3 CMS Guidelines

The general performance improvement that can be applied to all four CMSs is the minimization of the universal theme. The smaller the theme is the faster the pages load. This has been taken into account with GULP and GRUNT plugins that minimize CSS and JavaScript. The pictures uploaded to the theme should be minimized as well. Magento has been improved in various ways. The first is by enabling the "Enable Flat Categories and Products" setting. The second is by enabling the "Merge CSS and JS Files" setting. Further we have enabled compilation. CMS specific speed up guidelines goes here Drupal has been improved by disabling all unused modules and by enabling the performance settings of internal cache "Cache pages for anonymous users" and "Cache blocks" as well as "Aggregate and compress CSS files.". Wordpress has been improved by installing the wordpress security fence module which improves performance by cache optimization. As well as disabling one unnecessary plugin in the installation folder. Joomla has been improved in several ways. The Gzip compression capability of the CMS have been activated as well as the system cache plugin. We have also added an plugin called JCH which improves Cache.

6.4 Reflecting on the research questions and hypothesis

Our two research questions can now be answered with the statistics produced from the experiments. In order to answer the first research question (RQ1) we use the results presented in figure 16. According to the output of the experiments we can see that Drupal with cache enabled, outperforms all other CMS's under all three different page visits in respect to the onload time, page load time and the GET request. When it comes to the GET request time Drupal drastically outperforms the other CMSs(when cache is enabled). When it comes to the onload time it is not a big difference but Drupal is still number one. Drupal does not perform equally well with cache disabled. When cache is disabled Joomla seems to load majority of the requests the quickest, however Drupal is still with cache disabled outperforming Joomla in some situations. That is to say, Drupal performs quickest after optimization when cache is enabled and Joomla performs quickest in majority of the cases when cache is disabled.

To answer the second research question we can use the results obtained from figure 21 and construct diagrams based of these see 22 and 23. We can see that in the majority of cases the experiments produced positive values.
Magento has improved overall very positively with cache disabled and enabled after optimization. As can be seen from figure 22 and 23, the optimization has been exclusively positive and has not had any negative impact on the CMS installation.

We can also see that Joomla took longer in terms of page load time with cache enabled.
abled after optimization on the checkout page especially. It seems the optimization heavily slows down the page load time on the checkout page with cache enabled. When cache is disabled almost all the three pages seem to have improved in all three load times.

Wordpress seems to have increase in page load time when cache is disabled after the optimization is applied but the GET request is loaded quicker and the onload has exclusively improved. The wordpress optimization has had only positive effect when cache is enabled. The checkout page has been drastically improved in respect to page load time.

Drupal has improved very well with cache disabled on all pages with respect to all times measured. Drupal has exclusively improved except for the onload time for the checkout page with cache enabled.

With this data we can confirm the first hypothesis. However the second hypothesis is partly faulty. When it comes to the experiments conducted with cache enabled for Joomla checkout, page load time has drastically suffered. The Drupal onload time with cache enabled for the checkout page has not improved. Also The Wordpress sites page load time for the product display and product view when cache is disabled has not improved. That is to say, the first hypothesis is confirmed while the second hypothesis is valid in the majority of the experiments but clearly not in all of them see figure 22 23.
7 Conclusion and Future Work

In this thesis we have made 3 CMSs e-commerce capable namely, Drupal with the plugin commerce, Joomla with the plugin Hikashop and Wordpress with the plugin woocommerce. We have also compared the CMS Magento in this thesis which is e-commerce capable by default. We made all 4 CMSs use a common theme based on the framework Bootstrap. The individual push to Bootstrap was made using different popular Github projects that offered the CMSs Bootstrap themes. The universal theme layout and aesthetics is based on keeping a low cognitive load by following rules in previous studies. Optimization has been performed on all CMSs and data was collected by experiments to answer the research questions. Except for using the same universal theme the CMSs used the same dummy content with fake products and text. We have produced guidelines for optimization of the CMSs and theming of the CMSs. The experiments focused on measuring three parameters, namely, page load time, the time necessary to respond to the GET request and when the onload event is fired. All of the experiments are based on single run experiments. There is also a special experiment to produce the average difference in executing the same experiment. All of the installations benchmarked are fully functional fictional e-commerce sites. The fictional site is called Fast commerce.

The experiments have shown in some cases a penalty when pushing from the standard theme to a special universal theme constructed for this thesis. The theme used in this thesis is based of the twitter Bootstrap mobile first framework and the color scheme named Moxy on Adobe Kuler. Thus making all themes on the CMSs responsive to a variety of screen size present across different devices.

Experiments have shown there is a clear penalty for installing modules to make Joomla, Wordpress and Drupal e-commerce capable both with cache enabled and disabled. The highest penalty observed is for Joomla and is approximately 600ms for the onload event to be fired. After optimization of the thesis experiments show that improvements can clearly be made with and without cache for different experiments on all the CMSs. However some negative results where obtained. Magento has uniformly shown positive trends in the GET request time, page load time and onload event fired time.

By installing optimization modules and configuring the CMSs we have performed optimizations. We have minified the development output when using the JavaScript automation task runners GRUNT and GULP in the theme development. In order to speed up the CMSs we have tried to disable any and all unnecessary extensions when possible. We have enabled any cache improvement and file minimized built into the CMSs. Merging and compression has been enabled when ever possible.

Drupal has according to the experiment output become the fastest of the CMSs after optimization. The most positive optimization was accomplished with Magento and
the optimization showed an improvement of 792ms in page load time. The most negative consequence of the optimization was with the Joomla CMS, which took 658ms extra to load. The correlation of optimization is strong and the e-commerce systems become generally faster in comparison to their baseline performance after optimization. This is in accordance with the hypothesis that CMSs will see improvement in the measured parameters.

7.1 Future Work

More benchmarks should be conducted on information of various sizes for instance on databases of increasing sizes. This will create interesting results on the reactions of the CMSs to increasing database sizes. The experiments should not be single run, the experiments should rather be executed many times and the average value should be presented.
8 References


