2016

# Active Analytics: Adapting Web Pages Automatically Based on Analytics Data

William R. Carle II
*University of North Florida*, n00431448@ospreys.unf.edu

# ACTIVE ANALYTICS: ADAPTING WEB PAGES AUTOMATICALLY BASED ON ANALYTICS DATA

by

William R. Carle II

A thesis submitted to the
School of Computing
in partial fulfillment of the requirements for the degree of

Master of Science in Computing and Information Sciences

UNIVERSITY OF NORTH FLORIDA
SCHOOL OF COMPUTING

April, 2016

The thesis "Active Analytics: Adapting Web Pages Automatically Based on Analytics Data" submitted by William R. Carle in partial fulfillment of the requirements for the degree of Master of Science in Computing and Information Sciences has been

Approved by the thesis committee: Date

Dr. Karthikeyan Umapathy
Thesis Advisor and Committee Chairperson

Dr. Ching-Hua Chuan

Dr. Sherif Elfayoumy

Accepted for the School of Computing:

Dr. Sherif Elfayoumy
Director of the School

Accepted for the College of Computing, Engineering, and Construction:

Dr. Mark Tumeo
Dean of the College

Accepted for the University:

Dr. John Kantner
Dean of the Graduate School

ACKNOWLEDGEMENT

I would like to extend a huge and sincere thank you to my thesis advisor Dr. Karthikeyan Umapathy who was an amazing guide and mentor through this long and difficult process. I would like to thank him for all of the advice and guidance as well as his constant enthusiasm and encouragement throughout the project. I would also like to thank my thesis committee members Dr. Ching-Hua Chuan and Dr. Sherif Elfayoumy for their valuable input and help throughout the process. I am incredibly grateful to the University of North Florida ITS department for their generous assistance and support in the implementation of this system, it would not have been possible without their cooperation. Special thanks to two of my former managers in ITS: A.J. Parise and Dmitriy Bond, as well as the rest of my former colleagues and friends who offered their support not only as I worked on my thesis, but throughout the entire master's program.

I would like to extend a special thanks to Jim Littleton for reviewing my thesis and ensuring it was up to standards. I would also like to thank Roger Eggen and the entire School of Computing for their support since I started my educational career at UNF back in 2006. Finally, I would like to extend my thanks and love to my amazing family and friends, especially my parents Robert and Donna Carle, for their constant encouragement throughout my long educational career. I wouldn't have made it this far without you all.

CONTENTS

List of Figures ..... ix

List of Tables ..... xi

Abstract ..... xii

Chapter 1 Introduction ..... 1

1.1 Problem Statement ..... 2

1.2 Contributions ..... 3

1.3 Plan ..... 5

1.4 Organization ..... 7

Chapter 2 Background and Literature Review ..... 8

2.1 Background ..... 8

2.1.1 Theory Background Topics ..... 8

2.1.2 Implementation Background Topics ..... 16

2.2 Related Work ..... 19

2.2.1 Web Usage Mining ..... 20

2.2.2 Web Usability ..... 24

Chapter 3 Research Methodology ..... 26

3.1 Design Science Research Methodology ..... 26

3.2 Design Science Research Guidelines ..... 26

3.2.1 Design as an Artifact ..... 27

3.2.2 Problem Relevance ..... 27

3.2.3 Design Evaluation ..... 27

3.2.4 Research Contributions ..... 28

3.2.5 Research Rigor ..... 28

3.2.6 Design as a Search Process ..... 29

3.2.7 Communication ..... 29

Chapter 4 Dynamic Analytics Framework ..... 30

4.1 Website Improvement Process ..... 30

4.2 Dynamic Analytics Framework Architecture ..... 32

4.2.1 Extracting Analytics Data ..... 33

4.2.2 Analytics Data Store ..... 37

4.2.4 Client Side Framework ..... 43

4.3 Technology ..... 45

4.3.1 Google App Engine Technology Stack ..... 45

4.3.2 Microsoft Technology Stack ..... 46

4.4 Budget ..... 47

4.4.1 Budget: Google App Engine Technology Stack ..... 47

4.4.2 Budget: Microsoft Technology Stack ..... 48

4.4.3 Technology Stack Choice ..... 49

Chapter 5 Real World Application ..... 50

5.1 Ranking Links ..... 51

5.2 Global and Context Based Suggestions ..... 53
5.3 Search Suggestions ..... 56
Chapter 6 Evaluation ..... 59
6.1 Evaluation Goals and Objective ..... 59
6.2 Testing Process ..... 60
6.3 Study Participants ..... 62
6.4 Institutional Review Board (IRB) Approval ..... 63
Chapter 7 Evaluation Results and Analysis ..... 64
7.1 Demographics ..... 65
7.2 Independent Samples t-Test Results ..... 69
7.2.1 Task Completion Times ..... 70
7.2.2 Task Navigation Steps ..... 80
7.3 Task Skips ..... 88
7.3.1 Chi-Square Test Results for Task Skips ..... 91
7.3.2 Task Skips Result Summary ..... 94
7.4 Effect Size ..... 95
7.3.1 Task Completion Times ..... 95
7.3.2 Task Navigation Steps ..... 96
7.3.3 Task Skips ..... 97
7.5 Survey Responses ..... 97
7.5.1 Experience Ratings ..... 98
Chapter 8 Future Improvements ..... 100

8.1 Scalability and Expandability ..... 100
8.2 Features and Improvements ..... 101
8.3 Site Implementation ..... 102
Chapter 9 Conclusion ..... 104
References ..... 106
Appendix A Navigation Tasks ..... 109
Appendix B User Experience Survey ..... 110
Demographic Questions: ..... 110
Task Specific Questions: ..... 110
User Experience Ratings: ..... 111
Appendix C IRB Documents ..... 112

FIGURES

Figure 1. Google Analytics Behavior Flow Report ..... 10

Figure 2. The Website Improvement Process ..... 31

Figure 3. Dynamic Analytics Framework Architecture ..... 33

Figure 4. Querying and Storing Analytics Data ..... 35

Figure 5. Data Store Schema ..... 39

Figure 6. Web Service Interfaces ..... 41

Figure 7. Web Service Sequence Diagram ..... 42

Figure 8. UNF Website Homepage ..... 51

Figure 9. Ranking Menu Links Example ..... 52

Figure 10. Popular Links Example ..... 54

Figure 11. Contextual Popular Links Example ..... 55

Figure 12. Search Suggestions Example ..... 56

Figure 13. Evaluation Process ..... 62

Figure 14. Survey - Internet Experience ..... 66

Figure 15. Survey - Age Group ..... 66

Figure 16. Survey - Browser ..... 67

Figure 17. Survey - English Primary Language ..... 67

Figure 18. Survey - Frequency of Visits to UNF.edu ..... 68

Figure 19. Survey - Class Standing ..... 68

Figure 20. Average Task Completion Time ..... 71

Figure 21. Average Task Navigation Steps ..... 81

Figure 22. Percent of Users Who Skipped Each Task ..... 89

Figure 23. Survey Rating Responses ..... 99

TABLES

Table 1. Select HHS Usability Guidelines ..... 15

Table 2. Analytics API Queries ..... 36

Table 3. Components ..... 45

Table 4. Google App Engine Technology Stack ..... 46

Table 5. Microsoft Technology Stack ..... 47

Table 6. Budget: Google App Engine Technology Stack ..... 48

Table 7. Budget: Microsoft Technology Stack ..... 49

Table 8. Normality Test for Task Navigation Times ..... 72

Table 9. Group Statistics for Task Navigation Times - Non-Normalized ..... 73

Table 10. Group Statistics for Task Navigation Times – Normalized ..... 74

Table 11. Task Completion Times t-Test Results ..... 75

Table 12. Group Statistics for Task Navigation Steps ..... 82

Table 13. Task Navigation Steps t-Test Results ..... 83

Table 14. Chi-Square Test for Task Skips ..... 91

Table 15. Effect Size for Task 2 Completion Time ..... 96

Table 16. Effect Size for Task 2 Navigation Steps ..... 96

Table 17. Effect Size for Task 7 Navigation Steps ..... 96

Table 18. Effect Size for Task Skips ..... 97

# ABSTRACT

Web designers are expected to perform the difficult task of adapting a site's design to fit changing usage trends. Web analytics tools give designers a window into website usage patterns, but they must be analyzed and applied to a website's user interface design manually. A framework for marrying live analytics data with user interface design could allow for interfaces that adapt dynamically to usage patterns, with little or no action from the designers. The goal of this research is to create a framework that utilizes web analytics data to automatically update and enhance web user interfaces. In this research, we present a solution for extracting analytics data via web services from Google Analytics and transforming them into reporting data that will inform user interface improvements. Once data are extracted and summarized, we expose the summarized reports via our own web services in a form that can be used by our client side User Interface (UI) framework. This client side framework will dynamically update the content and navigation on the page to reflect the data mined from the web usage reports. The resulting system will react to changing usage patterns of a website and update the user interface accordingly. We evaluated our framework by assigning navigation tasks to users on the UNF website and measuring the time it took them to complete those tasks, one group with our framework enabled, and one group using the original website. We found that the group that used the modified version of the site with our framework enabled was able to navigate the site more quickly and effectively.

# Chapter 1

# INTRODUCTION

Modern Web analytics tools are an incredibly valuable asset for any organization with a strong Web presence. These tools track user actions on a site offering insight into what users want from the website, and what they have trouble finding. Popular tools like Google Analytics (Google Analytics, 2014) are widely used by sites across the Internet, but the value they provide fluctuates greatly depending on how well the tracking data is analyzed and acted upon. Analytics tools are a valuable source of usability and behavioral data that is too often overlooked or not used to its full potential (Phippen, Sheppard, & Furnell, 2004). For analytics data to be properly utilized, an organization would need to keep a constant eye on site usage and user behavior statistics, and update the site design to reflect the changing needs of its users (Prom, 2011). Such constant vigilance and development is often not feasible for many organizations. Ideally, an automated system could keep track of trends discovered through analytics and adapt a site in real time, with little to no interaction from a developer. However, there isn't any such automated system that is efficient and effective in dynamically adapting the site's design to meet user needs using its web analytics data.

## 1.1 Problem Statement

For content driven websites, relevant navigation and placement of information should be the top priority to help drive as many people as possible to informational pages (Phippen et al., 2004). Too often however, a site is designed to the specifications of content owners rather than to the needs of actual visitors to the site. With many stakeholders involved in site design and variety of contents, it is sometimes challenging for a designer to argue for a site update that removes rarely used information and pushes useful information to the forefront. This can often lead to busy and difficult to use sites that don't take into account what visitors actually need from a site. Sometimes you need hard data to convince someone that the link to their very specific corner of a website isn't as important as some other navigation options. Web analytics tools gather data on usage patterns of a website. Data gathered by web analytics can help designers address the usability problems of a site and keep track of changing usage patterns. The problem with web analytics tools is that they require active monitoring of usage patterns and acting on them in a timely manner to keep the site user interface (UI) useful.

There has been extensive research in field of web usability, but further work still needs to be done to marry good web interface design with analytics data that tells designers what visitors to a site really want to see. Good practices in web usability should be paired with analytics data to ensure that a site is not only easy to use, but also surfaces the information that visitors are actually interested in. This is not a one-time process, it is a process that needs to be repeated as visitor's needs could change over time. The majority

of visitors may need to find information on certain topics during certain times of year, and a static site design cannot react to the changing needs of users. Unless site owners are constantly monitoring these trends and adapting site design to fit these needs, a site will quickly become less usable (Prom, 2011).

As a case study, we looked at the official website of the University of North Florida. There are nearly 70 different links on the homepage alone and the relevance of these links changes over time. We evaluated the current design of the homepage and various other high traffic pages on the site, including the library's homepage, and use these pages to test the effectiveness of our system. Content owners often have neither the manpower nor the web design expertise needed to keep up with these changing trends year round. The university website could potentially benefit from better analysis of user needs by using web analytics data to adapt the site over time as visitors' needs change. For example: many student users would not use a course registration link in the middle of a semester but would likely use that link during the registration windows. Trends like this need to be acted upon in a timely manner and ideally without involving actions from a web designer. By automatically detecting these trends, and taking action such as moving the registration link to the top of a list of menu items, or drawing attention to it through styling, users are likely to be able to find their intended destination quicker.

## 1.2 Contributions

The goal of this research is to develop a generic automated system that will monitor the

vast amount of data gathered from web analytics and adapt web pages in real time to reflect usage trends. This thesis seeks to create a system for automatically processing tracking data from services like Google Analytics and transforming that data into adaptations of a site's user interface. By modifying the user interface dynamically according to usage patterns, we will improve the usability of the site and surface information that is important and relevant to the current visitors.

In particular, we are aiming to improve the navigational elements of a user interface. Often, a large number of navigational elements are presented to a user with little emphasis on which elements are most important or most popular. The data on which navigational elements are most important are available to us through analytics. We use this wealth of navigational data to build an ever-adapting and predictive website navigation system.

There is plenty of research into how to apply analytics data to improve a site's usability, but most of the proposed solutions require human analysis and manual action (Prom, 2011). While there will always be a need for designers to work on improving the usability of a site, we believe some of this burden can be offloaded to an automated system. For example, if a designer sees that that a certain page on the website is experiencing consistently high traffic they would need to adapt the navigation on the site to make links to that page more prevalent. We believe we can automate these and other similar tasks with our framework. To test this hypothesis, we have implemented a

functioning real-world system with user interfaces that can adapt to the changing needs of users.

## 1.3 Plan

As a case study for this research, we have used the University of North Florida main website (UNF, 2014). The university has a single unified content management system that covers most of the web presence for the entire university. The sheer size in terms of content and navigation items on this site makes it a perfect candidate for the automation of user interface improvements. Since 2009 UNF has been gathering tracking data using Google Analytics totaling to over 9.6 million unique visitors and over 94 million page views. We used this wealth of data to develop an automated way of analyzing visitor trends and applying the lessons learned from the available research on web usability to develop a smart web site that adapts to changing user needs over time.

The first challenge in realizing this vision was gathering and acting upon a wealth of analytics data. We tapped into the analytics data gathered over the past 5 years on the university website and transformed that data into a format that can be queried and reported on in real time. Using the Google Analytics API (Google Analytics, 2014), we query past analytics data as well as recent trends in site usage and store that data in a simple local reporting data store.

Once we set up an interface to extract the usage data, we wrote a web service interface that can be called from a web client to expose the common usage patterns of the page the

user is on. The challenge in this module was reporting on and summarizing the usage data quickly so the client code could make adaptations to the user interface in time to serve the user's needs.

The final piece of this solution is a client framework that is able to query the reporting service and take action on the data provided. The challenge here was to make a user interface framework that is generic enough to apply to a wide range of site designs and navigation structures. The idea of this piece being that a web developer can utilize it to provide suggestions as to what a user may require on the current page. Based on the usage patterns of this page, it executes a set of rules to adapt the interface by increasing the visibility of frequently accessed content and navigation items.

To test this system we have implemented it on the university's main site. We created a mirrored version of the site that uses our system to make automated improvements to the user interface. With the mirrored version of the site in place, we tested the effectiveness of these changes by asking users to find certain popular content by navigating the site. We compared the average time it took users to find the requested content to determine the efficacy of the automated improvements. In addition to this quantitative analysis, we surveyed the users to obtain qualitative data on the automated user interface changes.

## 1.4 Organization

This thesis is divided into nine chapters. In the second chapter, we will an overview of web analytics and web usability concepts, and review the current state of the industry. We also performed a literature review, which analyzed the current state of the art research in web analytics and web usability. In that chapter, we summarized sources that relate to the goal we are attempting to accomplish, we focused on papers that offer insight on how to analyze web analytics data and how to create usable web interfaces. In the third chapter, we discuss the design science methodology (Hevner, March, Park, & Ram, 2004), and how we applied its guidelines to conduct our research. In the fourth chapter, we discuss our implementation of the automated analytics system; we present the architecture of our solution and discuss how we implemented the different pieces of the system. In the fifth chapter, we applied our fully realized system to a mirrored version of the UNF website, outline the process of implementing our system in a real-world scenario, and discuss the pitfalls and lessons we encountered along the way. In the sixth chapter, we evaluated the effectiveness of our system by subjecting the dynamic mirrored version of the UNF website to various user tests. We directly compared the current version of the site with the dynamic version to get a sense of effectiveness of our system. In the seventh chapter, we statistically analyzed the data we gathered in the evaluation process to determine if our changes were effective. In the eighth chapter, we discussed potential future improvements and other possible directions to take with our prototype and research. Finally, in chapter nine, we compiled our results and form a conclusion on the state of our research and its potential utility for organizations like UNF.

# Chapter 2

# BACKGROUND AND LITERATURE REVIEW

## 2.1 Background

In this chapter, we discuss various concepts relevant to our proposed dynamic analytics system. The two main concepts of our system are web analytics and web usability; we will discuss these two topics at length to provide an overview of the state of the industry. We also provide a brief overview of other relevant areas used in this research, which includes web services and data warehousing. Understanding concepts specified above is necessary to properly design our system.

### 2.1.1 Theory Background Topics

#### 2.1.1.2 Web Analytics

Web analytics is the measurement, collection, analysis, and reporting of Internet data for the purposes of understanding and optimizing a web page (Prom, 2011). The origins of web analytics can be traced back to the practice of web usage mining. Web usage mining involves analyzing web server logs that record every request made to a web server. The idea is that by mining server activity logs, reports could be generated about usage patterns on a site (Kumari, Praneeth, & Raju, 2014). There are various problems with this method of obtaining analytics data. Most of these problems revolve around the fact that web server logs keep track of every single request made to a server (Mican & Sitar-

Taut, 2009). Because every request is logged even requests that don't represent normal user actions, raw web log data can be inaccurate and must be properly filtered. For example, every request for page content is recorded separately including images, stylesheets, and script files. Recording of each individual request can result in a lot of noise in server logs, which can complicate reporting. Another problem with these logs is that all clients are logged equally including bots and search engine crawlers. Data from bots and crawlers are not relevant when trying to determine the behavior of humans on a website, and should be excluded (Mican & Sitar-Taut, 2009). In the end, data obtained from web usage mining is definitely useful, but a better solution is needed. This better solution had to be designed from the start with the intention of logging user activity specifically for reporting, and this is where web analytics comes in.

Analytics tools have been constantly evolving since the early days of web usage mining. Modern analytics tools offer a robust set of reporting tools that can help designers determine usage patterns on a website as well as provide other important data about the site. Some of these additional reports include information on how the user found the site, the geographic location of users, the devices used to view the site, and much more (Google Analytics, 2014). The current market landscape for web analytics tools is skewed in the direction of Google Analytics, one report form 2011 put Google's market share at 81% of all websites that use analytics tools (W3Techs, 2011). We will focus mainly on Google Analytics because of their dominance in the market, their wealth of features, and their lack of a service fee.

Google Analytics offers a wealth of reporting tools that surface information about almost every aspect of a site's user base. For our research, we will focus on a subset of these tools that surface mainly user behavioral data (Beasley, 2013). User behavioral data reports include user flow paths that show how users navigate a site, content drilldown reports that show the most popular pages on the site as a whole as well as on a given page, and traffic source reports that show how users reached the site (Google Analytics, 2014). Figure 1 shows Google Analytics a user behavioral flow report for UNF website.

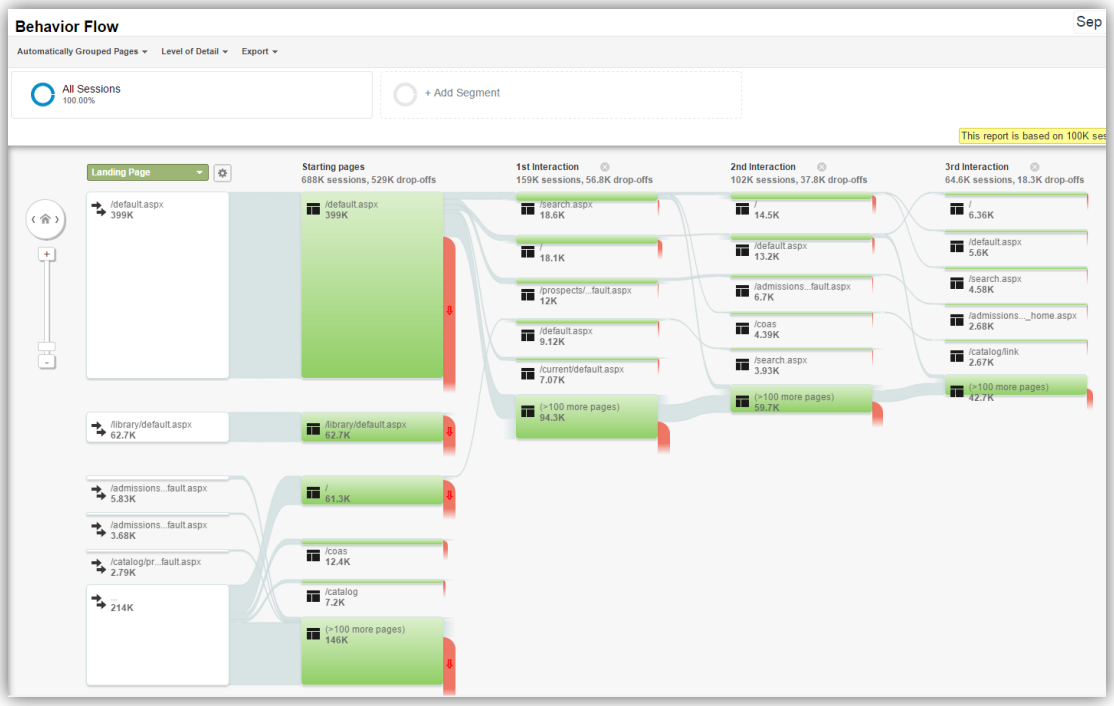


Figure 1. Google Analytics Behavior Flow Report

To extract this reporting data programmatically, we will utilize Google Analytics' extensive reporting API web services. These APIs are exposed as REST web services and offer a programmatic endpoint for most of the data exposed in the Google Analytics

UI. The API requires OAuth 2.0 authentication to query the data and has quota limits for each user, therefore we will likely need an intermediate layer that will query the API and cache the results for use on a high traffic website (Google Analytics, 2014).

Reports like the one above offer a window into user behavior on a site. From the report above we can see the top initial landing pages where users entered the website. We can also see the most common paths taken once on that page. We can see that most users started on the UNF homepage, and from there performed a search landing them on the "search.aspx" page. Going down the chain of user interactions we see that some of the most popular destinations for users starting on the homepage are: Admissions, COAS (College of Arts and Sciences), Catalog, Library, etc. These links should be featured more prominently on the homepage, especially pages like COAS or Admissions, which despite being available straight on the homepage, often took users multiple interactions to find. This report represents a one month snapshot of time and may only represent user needs for this specific period of time. Because of this, reports like these need to be re-evaluated multiple times a year to adapt to changing user needs. For our purposes, the data presented in this visual graph are also available in raw format from the Google Analytics API, we will discuss our process for extracting this data later in section 4.2.1.

#### 2.1.1.2 Web Personalization

Analyzing usage data and adapting a user interface is a concept that has been around for a long time. Many different sites utilize user browsing patterns to determine additional products or information a site visitor may be interested in. A good deal of research has

been done exploring the idea of web personalization. Usage patterns of individual users are analyzed and categorized into profiles that seek to predict their future behavior (Mobasher, Cooley, & Srivastava, 2000). The idea of this process being that users with similar needs and tastes would browse in similar patterns and additional products and information could be recommended to them. Most modern shopping sites utilize this type of analysis, for example Amazon.com has a recommendation feature that gives users suggestions based on the activity of users with similar shopping patterns. Where we will differentiate ourselves from these well-developed practices is that we seek to facilitate overall site improvement rather than personalization for individual users. Rather than personalize a site based on similar users' behavior we aim to improve the overall usability of a site based on global usage trends, using these trends to dictate the layout of a site. Personalization has a very important place in web design, and it can be useful for many different applications. We will learn what we can from established research on personalization, but we do not want to personalize websites for specific users. Our goal is to improve usability for all visitors to a site by analyzing usage trends on the site as a whole.

#### 2.1.1.3 Web Usability

Web design is a complex task with many different facets to consider especially as it relates to web usability. It can be difficult to develop a site that takes into account all the possible areas of usability. In an effort to document the different types of usability concerns and provide a sort of checklist for web developers, various web usability standards have been developed. Many organizations have created their own sets of

usability standards including International Organization for Standardization (ISO) 9241, ISO 25010, and ISO 1340 (Bevan, 2005). These standards cover various aspects of web applications and their development including: design process and evaluation, optimizing the user experience, accessibility, page layout, navigation, and many others. The different sections contain specific suggestions that a developer should apply to their site. For example in the "designing page layout" section of some usability guidelines it may suggest establishing a level of importance for the content, or placing important elements in the top center of the page. For navigation they may suggest providing feedback on the user's current location or keeping main navigation links always visible (Herring & Prichard, 2012). These are just a few examples of the many usability standards offered by various organizations. For our purposes we must ensure that, as we adjust page navigation and structure based on analytic data, we adhere to these usability standards and adapt the UI to more effectively implement the suggestions they provide. Adherence to web usability guidelines has been proven to positively effect a user's perception of a site, and it is in the best interest of web developers to be familiar with, and apply these guidelines to their work (Bevan, 2005).

One of the better sets of usability guidelines found in our research was the one created by the U.S. Department of Health and Human Services (U.S. Dept. of Health and Human Services, 2006). These guidelines put in simple terms the consideration that web designers need to take into account when designing a usable website. There are over 200 guidelines in the HHS document, each with a detailed description, example, and importance rating. For the sake of brevity we will not include the full listing of

guidelines here, but instead will include some of the guidelines most relevant to our research in Table 1 below.

| # | Guideline | HHS Comments |
|---|---|---|
| **5:2** | Show All Major Options on the Homepage | Users should not be required to click down to the second or third level to discover the full breadth of options on a Web site. Be selective about what is placed on the homepage, and make sure the options and links presented there are the most important ones on the site. |
| **5:7** | Limit Homepage Length | Any element on the homepage that must immediately attract the attention of users should be placed 'above the fold'. Information that cannot be seen in the first screenful may be missed altogether - this can negatively impact the effectiveness of the Web site. If users conclude that what they see on the visible portion of the page is not of interest, they may not bother scrolling to see the rest of the page. |
| **6:2** | Place Important Items Consistently | Put important, clickable items in the same locations, and closer to the top of the page, where their location can be better estimated. |
| **6:5** | Establish Level of Importance | The page layout should help users find and use the most important information. Important information should appear higher on the page so users can locate it quickly. The least used information should appear toward the bottom of the page. Information should be presented in the order that is most useful to users. |
| **7:2** | Differentiate and Group Navigation Elements | Clearly differentiate navigation elements from one another, but group and place them in a consistent and easy to find place on each page. |
| **7:11** | Use 'Glosses' to Assist Navigation | 'Glosses' are short phrases of information that pop up when a user places his or her mouse pointer over a link. A 'gloss' provides a preview of the type of information that will be found behind a link. Users prefer the preview information to be located close to the link, but not placed such that it gets in the way of reading the link. A gloss can be created by defining the Title attribute for a link. However, designers should not rely on the 'gloss' to compensate for poorly labeled links. |

| # | Guideline | HHS Comments |
|---|---|---|
| **9:5** | Highlight Critical Data | Visually distinguish (i.e., highlight) important page items that require user attention, particularly when those items are displayed infrequently. |
| **10:2** | Link to Related Content | Users expect designers to know their Web sites well enough to provide a full list of options to related content. |
| **10:5** | Repeat Important Links | Establishing more than one way to access the same information can help some users find what they need. When certain information is critical to the success of the Web site, provide more than one link to the information. Different users may try different ways to find information, depending on their own interpretations of a problem and the layout of a page. Some users find important links easily when they have a certain label, while others may recognize the link best with an alternative name. |
| **11:4** | Ensure Visual Consistency | Visual consistency is the consistent use of design elements such as typography, layout, colors, icons, navigation, images, and backgrounds. While users can overcome certain inconsistencies (e.g., entry fields, pushbuttons), consistent interfaces can reduce errors and task completion times. It can also reduce learning curves, and increase user satisfaction. |
| **11:11** | Highlighting Information | One study found that participants were able to complete tasks faster when the interface contained either color-coding or a form of ranking, but not both. The presence of both seemed to present too much information, and reduced the performance advantage by about half. |
| **12:2** | Place Important Items at Top of the List | Experienced users usually look first at the top item in a menu or list, and almost always look at one of the top three items before looking at those farther down the list. Research indicates that users tend to stop scanning a list as soon as they see something relevant, thus illustrating the reason to place important items at the beginning of lists. |
| **12:4** | Display Related Items in Lists | A well-organized list format tends to facilitate rapid and accurate scanning. One study indicated that users scan vertical lists more rapidly than horizontal lists. Scanning a horizontal list takes users twenty percent longer than scanning a vertical list. |

Table 1. Select HHS Usability Guidelines

### 2.1.2 Implementation Background Topics

#### 2.1.2.1 Web Services

The goal of a web service is to expose a programmatic interface for transmitting data or performing actions over the Internet. Web services are called by software systems to integrate data and functionality across a network. We plan on utilizing web services for two of the main components of our solution. For our solution, we will exclusively be using REST web services. REST stands for Representational State Transfer, and is characterized by stateless service endpoints that explicitly use the HTTP methods such as GET and POST (Fielding, 2000). REST web services are services to manipulate XML (or other data formats) representations of web resources using a uniform set of stateless operations (Booth et al., 2004). REST web services are designed to be simple and adhere closely to the basic HTTP protocol. As a result of this all persistence and state management must be handled by the application.

Authentication and authorization for REST services are usually handled through the use of authentication tokens. Most REST web services offer some form of authentication using temporary authentication tokens or permanent application key tokens. Temporary tokens are often used for client side applications, and involve some authentication process with the service provider, usually OAUTH, which will provide a token that will last a limited amount of time before that authentication process must be repeated. Permanent tokens are pre-shared tokens that are often associated with a specific

developer account, and are designed for server side applications that will connect directly to the REST services using this secret token (Booth et al., 2004).

Google Analytics uses REST web services to expose the reporting data, in order to extract this data we will need to authenticate to their services and extract this data. Google analytics uses permanent pre-shared application tokens, which require minimal setup (Google Analytics, 2014). In addition to extracting analytics data via web services we will need to create REST endpoints to expose summarized and pre-computed data to our client-side framework. These services will be open, and will not use authentication tokens because these endpoints need to be exposed directly to anonymous clients. We will provide more details on the design of these web services in section 4.2.4.

#### 2.1.2.2 Data Warehousing

Below, we provide brief discussion on data warehousing as it relates to our proposed system. We need the ability to query summarized reporting data on every page load, we were not able to rely solely on the Google Analytics API for this, as this would greatly increase the time it takes our page to fully load. We also need to pre-compute and store summarized usage statistics to further increase speed. For this task, we use some well-established data warehousing techniques (Fasel & Zumstein, 2009).

A data warehouse is a subject-oriented, integrated, time-varying, non-volatile collection of data in support of a decision making process (Inmon, Strauss, & Neushloss, 2010). In other words, it is a way to store data about certain subjects as they change over time.

This is a good fit for the kind of data we are attempting to gather and analyze. In our case the subjects are the web pages being visited by users. We need to analyze how traffic to and from these pages changed over time. Because we are using a warehousing database in a real time manner, we will need to develop a warehouse that can respond quickly while still providing the subject-oriented time-variant strengths of a traditional warehouse.

The first process that needs to take place when developing a data warehouse is the design of the schema. A simple data warehouse schema, known as a star schema, includes two types of data: facts and dimensions. Facts are the central object of a star schema and contain the summarized data from snapshots of time. The dimension tables radiating off of the fact tables provide the detailed information about the objects represented in that snapshot of time in the fact table. This schema allows for historical record of statistics over time by querying for summarized data (facts) based on different attributes of business data (dimensions) such as dates, product names, etc. (Inmon et al., 2010). Because we are planning on using NoSQL database technology that isn't as heavily designed around relationships, we will be flattening this idea of a star schema, while also retaining some of its core features. We will discuss our specific implementation details in section 4.2.2.

Another important aspect of data warehousing is the Extract Transform Load (ETL) process. This involves pulling data from a transactional data source, transforming it into a format more suited for reporting purposes, and loading it into the warehouse. The ETL

process maps the schema of the transactional database to the schema of the warehouse dimension tables (Inmon et al., 2010). It also performs data summarization tasks to store statistics about dimensions in the fact tables. We will be performing a continuous ETL process based on pages a user is requesting. We will be mapping the data pulled from the Google Analytics APIs to the documents in our data store when pages are requested for the first time by a client. As part of this process, we will be making multiple calls to the Google Analytics API and combining the data from multiple queries into single facts about page navigation trends. We will give a detailed description of this process in sections 4.2.1 and 4.2.2.

## 2.2 Related Work

From our research into web analytics and its application to web user interface design, we found that it was a well explored topic with research dating back to the early days of the Web. We found that the techniques of web usage mining and its applications to site personalization have been around for a long time and are relatively well explored. The more recent trend of using web based analytics tools such as Google Analytics to improve web usability is also a well-represented topic. We did, however, find a gap in the published literature relating to improving site usability based on analytics data in an automated fashion. We chose to focus our research on taking the knowledge from published sources about improving usability based on analytics data and finding a way to apply those methods in an automated way. In this literature review, we present some of the most useful sources we found relating to this topic and will discuss how we plan to use the existing research to develop our solution.

### 2.2.1 Web Usage Mining

The idea of gathering website usage data for use in improving site design began with the concept of web usage mining. Web usage mining involves analyzing web server logs and drawing conclusions about usage patterns from these logs. Traditional data mining techniques such as loading the data into analysis cubes in star and snowflake schemas and reporting on that data are used to track individual users and find overall trends of usage on the site. In Büchner's paper on web usage mining for marketing purposes, he outlines a process for creating a generic reporting cube for analytical data (Büchner & Mulvenna, 1998). This paper offers some insights on how to organize and report on web usage data. With so much data constantly flowing in from high traffic websites these reporting techniques could prove useful for our research. This paper focuses on using web usage mining and reporting for ecommerce purposes to help drive product strategy for companies, which is not the primary focus of our research and it may not entirely apply (Büchner & Mulvenna, 1998). The paper used web log data from an online retailer to perform its analysis. Because their primary focus was retail applications, the research doesn't entirely apply to our goal of improving usability and finding informational data rather than products. The paper also devotes a good deal of time discussing the extraction of web log data, which is irrelevant for our purposes as we are using web analytics data. What we can take from this paper is some insight into how to architect a data store based around web usage data (Büchner & Mulvenna, 1998).

Another common application of web usage mining is user personalization. From the web usage data mined from server logs it is possible to extract profiles of user activity and

match other anonymous users to those profiles. In the paper by Mobasher et al. the idea of mining user profiles is presented (Mobasher et al., 2000). Based on user navigation patterns, they form profiles of user activity and attempt to match live user activity to these profiles. If a user's activity fits one of their mined profiles they then automatically offer the user suggestions of other pages or products they may be interested in. This approach to automatically guiding a user based on analytics data is somewhat similar to our proposed process. The way they generate user profiles and determine other pages a user might be interested in could be very useful in the implementation of our solution. Where we believe they fall short is in the area of updating the user interface. This paper does not go into concrete ways of improving user experience, it is more focused on matching users to profiles and suggesting links. The paper is also based on data mined from web logs, which can be unreliable and misleading as compared to modern web analytics tools due to the nature of data collected in web logs (Mican & Sitar-Taut, 2009). With our research, we plan to expand on the ideas in this paper and focus less on matching users to profiles and instead making general user interface improvements based on overall site trends (Mobasher et al., 2000).

There are some inherent problems with any web usage monitoring system that must be overcome if any useful data is to be mined. The paper by Mican et al. covers some of the difficulties that must be considered when mining usage data (Mican & Sitar-Taut, 2009). Mining data from web server usage logs was the standard way of finding out what your users were looking at on your site until web analytics came along. The problems identified by this paper about this kind of data mining include things like search engine

bots, content requests that are part of a different overall page request, differentiating between content pages and navigational pages, and various other problems. Although these problems were addressed in relation to web usage mining rather than web analytics, we believe they provide good insight into some of the problems we may face when mining web analytics data. These problems must be taken into account when analyzing analytical data, especially when that data will be used for automatic changes to a user interface. For our research, we plan to use some of the insights presented in this paper to evaluate whether the analytics data we are mining are legitimate user behaviors. This paper does not draw any significant conclusions about content pages versus navigational pages which will also need to be a consideration in our final design so we will need to do our own research in that area (Mican & Sitar-Taut, 2009).

There is extensive research into web usage mining as it applies to selling products. The data mined from user activity can be applied to other ends rather than just trying to recommend more products and services. The paper by Kumari et al. explores the potential of using web usage mining and user profile analysis to improve the structure and content of a website and track how user interests change over time (Kumari et al., 2014). This constant analysis of changing trends over time is a key tenant of our research, which is why this paper is useful for our purposes. This paper also takes this analysis a step further and does not only analyze usage patterns but also analyzes the content that the users are viewing. By analyzing the content of a page that a user ends up on, they draw conclusions based on analysis of that content to find other pieces of content that may be related semantically to the content the user found. This paper focuses mainly

on web usage mining and is also concerned with generating user profiles, which is not the direction we want to take with our research. Although this paper does not apply specifically to the ideas we are pursuing it does present some very interesting points especially related to content driven websites rather than product driven websites (Kumari et al., 2014).

Analyzing data from web metrics is a complex task; the data is overwhelming and the potential pitfalls are abundant. The paper by Weischedel et al. performs an extensive case study on the use of web metrics (Weischedel & Huizingh, 2006). The papers seeks to find the limitations of analyzing web metrics and finding the alternative data sources that help supplement this data. The paper draws some interesting conclusions on web metrics analysis including the idea of gathering queries made from particular pages and using that data to determine what information should be included on that page. It also champions the usefulness of customer opinion data to supplement hard log data to gather some qualitative information that may help improve the design of a site. This paper focuses mainly on clickstream data obtained from server logs, which can be unreliable and lead to incorrect conclusions. Because we plan to use web analytics as opposed to log based web usage mining many of the conclusions reached in this paper do not apply. Despite the limitations of this paper it does offer some interesting conclusions about applying knowledge gained from usage data into concrete site improvements (Weischedel & Huizingh, 2006).

### 2.2.2 Web Usability

Gathering and analyzing usage data is only half of the problem we address in this thesis. These metrics on user behavior are useless without the concrete design improvements that follow them. There are several sets of usability guidelines that attempt to address the design considerations of a site. The paper by Lai et al. analyzes one of the industry standard sets of guidelines, the Microsoft Usability Guidelines (MUG) by applying the Repertory Grid Technique which is a qualitative evaluation methodology used heavily in marketing research (Lai, Xu, & Tan, 2009). This paper offers some valuable insight into what users are looking for in a web page in their own words and categorizes them into actionable areas based on the MUG. One of the important points presented in this research was the emphasis on relevance on a site, the idea that content on any given page is relevant to the core users of that page. This is one of the core ideas of our research, by mining data from analytics as to what other pages are most useful to other users of this page is backed up by these updated usability guidelines. This paper offers some suggestions on how to improve usability of a site such as increasing icon size for important elements, but it doesn't go very far in suggesting user interface improvements, we will have to draw these conclusions from other areas of our research (Lai et al., 2009).

For some concrete ideas on improving a website's usability, we looked at other sources, specifically a paper by Webster et al. entitled "Enhancing the Design of Web Navigation Systems" (Webster & Ahuja, 2006). This paper tackles the topic of navigation usability. It looks at the global navigation of a site and emphasizes concepts like a sense of where you are, and where your next click will lead you, and what content you will find at that

link. This concept is important for our work, some indication of what other users found after following a certain path could lead to subsequent users finding relevant information quicker. This paper examines the idea of global navigation, a common navigation element across the whole site that shows your current location in the site, to reduce the perceived disorientation of users on a site. The paper compares three different versions of a site, by asking participants to find specific information on the site. The findings of the paper suggest that simple local navigation systems often behaved better than global navigation, perhaps because users were presented with fewer choices and less of an information overload. For our system, we took into account some of the lessons learned in this research to help us adapt navigation systems using analytics data (Webster & Ahuja, 2006).

# Chapter 3

# RESEARCH METHODOLOGY

## 3.1 Design Science Research Methodology

For this thesis, we used the Design Science Research Methodology. Design science research involves the creation of new knowledge through the design of novel or innovative artifacts and the analysis of the use and/or performance of those artifacts along with refection and abstraction (Vaishnavi & Kuechler, 2013). The real point of the design science research methodology is the idea that design is research, and the act of designing an artifact is a valid method of conducting research. What the design science methodology stresses over the typical design process is the idea of knowledge contribution. A design project should have a strong focus on contributing knowledge to the field and sharing the results.

## 3.2 Design Science Research Guidelines

Design science research sets forth various guidelines that provide a framework for executing the design process. These are not strictly enforced guidelines for the design process, rather they are simply aspects to consider during the design process (Peffers, Tuunanen, Rothenberger, & Chatterjee, 2007). Below is a listing of those guidelines and how we plan to meet them for our research.

### 3.2.1 Design as an Artifact

The purpose of this guideline is to ensure that the research works towards producing a viable artifact in the form of a construct, a model, a method or an instantiation. For our research, we produced a working example of our dynamic analytics framework. This represents our physical working artifact that we tested and evaluated. We outline this artifact in detail in chapter four.

### 3.2.2 Problem Relevance

The point of this guideline is to define a specific research problem that is relevant to the business problems of the real world and justify the value of a solution to that problem. The problem this thesis addresses is the degrading usability of websites over time, as user needs change, and the large amount of manual work that must be done to maintain a site's usefulness. There is a need for an automated way to utilize the web analytics data that is already being gathered on many sites to keep a site up to date and reflective of current user needs.

### 3.2.3 Design Evaluation

The goal of the evaluation guideline is to examine the effectiveness of the finished artifact on the problem. We used the live UNF website, and the alternate version of the site discussed in the previous activity, to perform A/B testing with site users. The users were asked to accomplish a task on one version of the site. We compared quantitative measurements such as time to accomplish the task, as well as qualitative measurements in

the form of user surveys for the two versions of the site to determine if our artifact is effective in solving the problem.

### 3.2.4 Research Contributions

The research contributions guideline states that the research should provide contributions in the areas of the design artifact, design foundations, or methodologies. The objective of this research is to create a generic and reusable platform for querying web analytics data, analyzing usage patterns, and using that data to adapt the user interface of a site. This artifact contributes to the growing field of adaptive analytics and serves as an example of how to dynamically use analytic data to adapt sites. This system is generic enough to apply to any site while also allowing levels of developer customization to fit an organization's individual needs. The resulting site adapts dynamically to traffic patterns and lead users to their destination more quickly. We believe that this research contributes significant insights into the field of dynamic analytics especially in areas outside of E-Commerce.

### 3.2.5 Research Rigor

The purpose of this guideline is to ensure that the decisions made when implementing an artifact are well informed and represent the best possible solution to the problem. Decisions made in the development of the artifact should be justified and backed up by research. For our research, we back up every decision with specific research and exhaustive analysis. We justify each of our decisions according to the best information

available to us. Essentially our research rigor is derived from the effective use of the existing knowledge base.

### 3.2.6 Design as a Search Process

This guideline states that the design process for an artifact should be a search process to find the best possible solution to the problem. For our research, we have done extensive searching into the field of analytics and used various existing tools to help us architect our solution. We are not starting development of our system from scratch, we took the state of the art technologies available and expanded on them with our own ideas. We continued to evaluate alternative options developing our solution keeping in mind that we are always searching for a better solution to the problem.

### 3.2.7 Communication

The final guideline of the design science research methodology is communication. The problem and its importance as well as the resulting artifact and its effectiveness should be conveyed to relevant audiences. In adherence to this guideline, this thesis will be defended in a public forum and the resulting research will be published along with the source code of the resulting artifact.

# Chapter 4

# DYNAMIC ANALYTICS FRAMEWORK

In this chapter, we discuss our development plan for our system and justify the decisions we made during each step of the design process. We first discuss the architecture of the system which includes components for the extraction of data from Google Analytics and the client side framework that adapts the website. We present different options available to us in terms of frameworks and technologies, and justify our final choice based on the features of that technology and our ability to learn and implement that technology in a timely manner. Finally, we discuss the economic feasibility of our complete system. We must address the costs required to develop and host each of the components involved in the system. Where possible, we used open source technologies to avoid large costs, although there was some cost associated with server hosting.

## 4.1 Website Improvement Process

The current process for updating the design of a website based on analytics data is a primarily manual process involving multiple stakeholders. The basic process involves a web team that can consist of many different specialized individuals including developers, designers, marketing personnel, content creators, etc., generating reports from analytics tools and using those reports to make decisions about website design (Weischedel & Huizingh, 2006). The developers and designers then go to work updating the underlying code, the design, and the content of the website. Those changes are published to the live

site and the cycle continues again as analytics data are gathered on the new design. This process needs to be repeated often to maintain the usability of a site as user needs change. Figure 2 provides diagrammatic overview of the website improvement process.

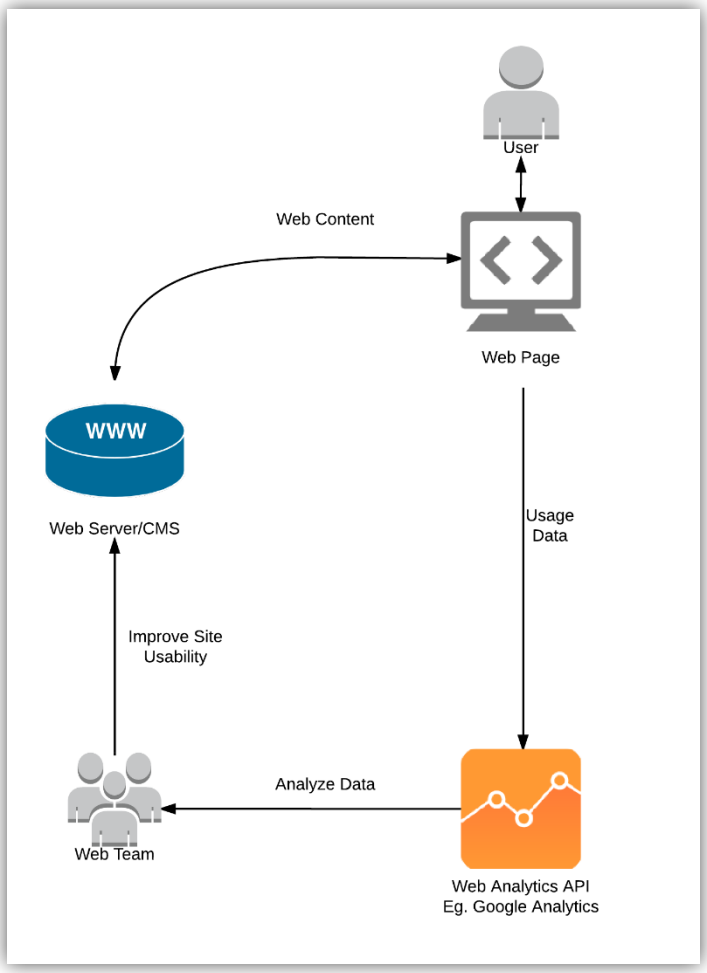


Figure 2. The Website Improvement Process

## 4.2 Dynamic Analytics Framework Architecture

While the process outlined above cannot be completely be replaced by automated processes, we believe that our system is able to take over some of the smaller, more data driven decisions. This frees up the web team and allows them to focus on the more sweeping and important interface changes. The system reads and analyzes analytics data and applies the lessons learned from this data to site improvements, much in the same way that a web team does. To do this, the system needs to consist of four major components. Firstly it needs a mechanism for extracting data from Google Analytics reporting APIs. We then store that data in a way that it can be quickly extracted and used for interface updates. We use a service layer that exposes that reporting data to the client framework, which is responsible for updating the interface. Each of these layers is discussed in more detail later in this chapter. Figure 3 provides a quick overview of how each of these pieces fit together.

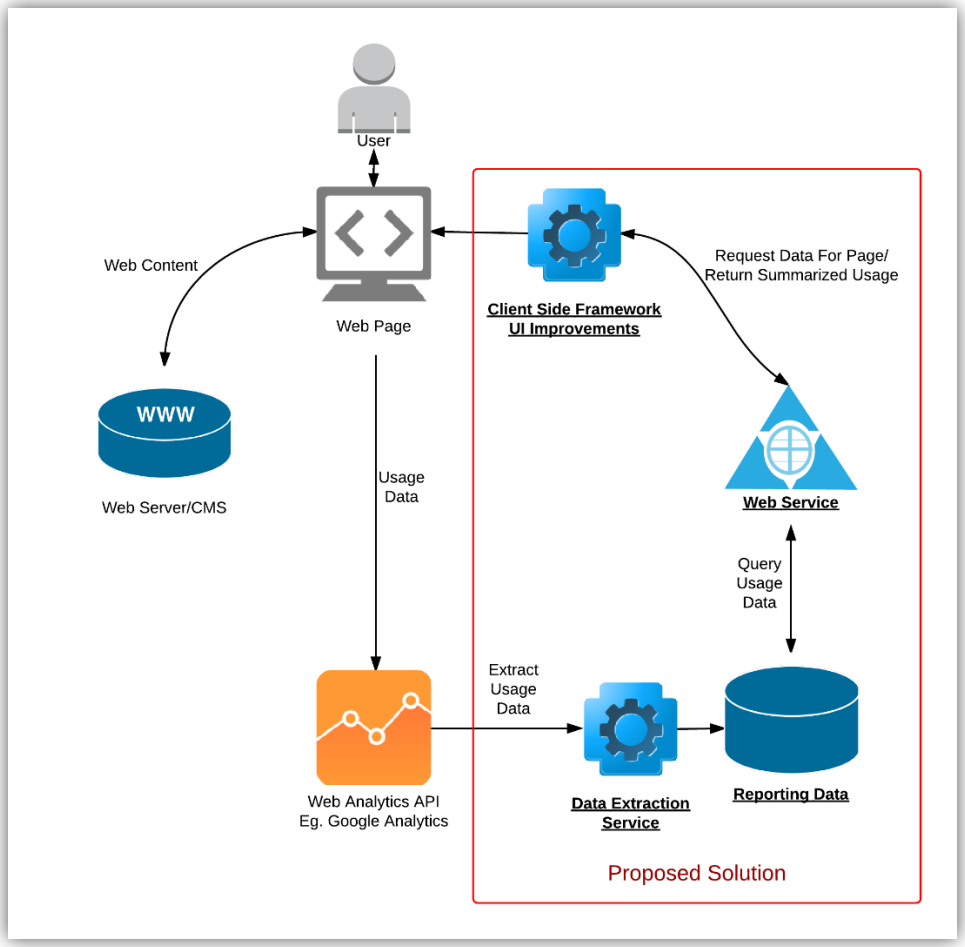


Figure 3. Dynamic Analytics Framework Architecture

### 4.2.1 Extracting Analytics Data

The first task our system must accomplish is extracting live analytics data from our web analytics provider. We have chosen to use Google Analytics for our system because of its overwhelming market share and comprehensive feature set. Google currently enjoys an 81% market share in the field of web analytics as of 2011 (W3Techs, 2011). Google also offers a feature rich reporting API that allows us to extract the analytics data and use it for our own purposes. These APIs are REST web service based and were able to easily

tap into them with a simple server-side REST client. The Google Analytics reporting API has two drawbacks that prevent us from using it in real time to query reporting data. Firstly, because it is a free service Google imposes rate limits on its reporting API to prevent overuse. Secondly, because the API is mainly intended for reporting it does not provide the kind of speed necessary for us to use it to update an interface in real time (Google Analytics, 2014). Because of these limitations we must extract the reporting data, transform it into a format more fit to our purposes and store it ourselves.

To facilitate this, our system receives incoming requests and first queries our database to see if we have already cached the reporting data for that request. If data for that page is not found, the Data Extraction Service (See Figure 3) will be executed to extract the data from the reporting API. This process is a separate module of the Web Services application we will discuss later in section 4.2.3. This module is responsible for asynchronously updating the data store with the data gathered from the analytics API. When data on a page is not available in our data store, or the data gathered previously is expired, the service application creates a new threaded task to update that data then return to the client. We store this reporting data with time stamps so we can enforce an absolute expiration time for these reports. This allows us to keep a history of activity over time while also obtaining data on new trends. If the data for a given page request is not found or if it is past its expiration, the middleware will fetch fresh data from the Google Analytics API, store it in our data store, then return it to the browser. Figure 4 provides flow chart representation of this process.

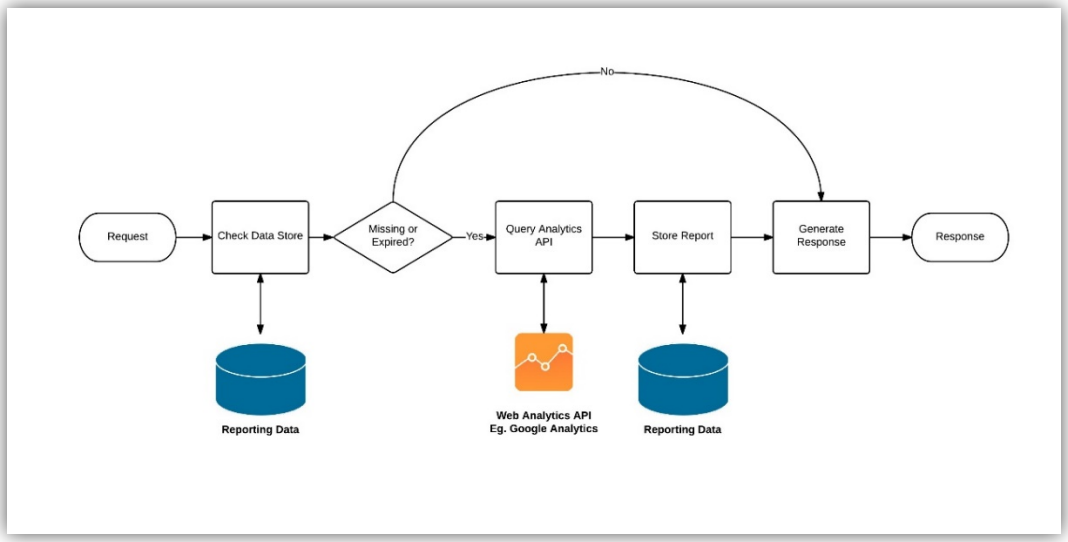


Figure 4. Querying and Storing Analytics Data

The data we extract from the Google Analytics API will need to be transformed and mapped to our database structure. The Google Analytics API provides all of its data via a single REST endpoint. To that endpoint we pass a set of dimensions and metrics, which will determine the data we get back. Dimensions represent attributes of single items such as pages (title, path, etc.), whereas metrics represent computed statistics about those pages (views, time on page, etc.). Table 2 shows how we query the reporting API data and how those dimensions and metrics are mapped to our database, PageSnapshot, schema. The query to extract GlobalTrend data is very similar; we simply remove the filter parameter. More information on the database schema is outlined in section 4.2.2.

| **Building PageSnapshot** | | | |
|---|---|---|---|
| | | | |
| **Property** | **Type** | | **Mapped To** |
| **PageURL** | String | | Provided |
| **DateRetrieved** | Date | | Current Date |
| **PrevPages** | Page[] | | Navigation Query |
| **NextPages** | Page[] | | Navigation Query |
| **CommonDestinations** | Page[] | | Navigation Query |
| **Searches** | Search[] | | Search Query |
| **Navigation Query** | Returns: Page[] | | |
| | | | |
| **Dimensions** | **Metrics** | **Filter** | **Sort** |
| **pagePath**<br><br>**OR**<br>**exitPagePath** | pageviews | prevPage = PageURL<br>OR<br>nextPage = PageURL<br>OR<br>pagePath = PageURL | pageviews Desc |
| **pageTitle** | avgTimeOnPage<br>exitRate | | |
| | | | |
| **Result Column** | **Mapping** | | |
| pagePath | Page.PageURL | | |
| pageTitle | Page.PageTitle | | |
| Pageviews | Page.Hits | | |
| avgTimeOnPage | Page.AvgTimeOnPage | | |
| exitRate | Page.ExitRate | | |
| | | | |
| **Search Query** | Returns: Search[] | | |
| | | | |
| **Dimensions** | **Metrics** | **Filter** | **Sort** |
| **searchKeyword** | searchResultViews | prevPage = PageURL | searchResultViews Desc |
| **exitPagePath** | | | |
| | | | |
| **Result Column** | **Mapping** | | |
| searchKeyword | Search.Keyword | | |
| exitPagePath | Search.Destination | | |
| searchResultViews | Search.Hits | | |

Table 2. Analytics API Queries

### 4.2.2 Analytics Data Store

The next component of our system is the analytics data store, which is used to store the reporting data we queried from the Google APIs. Our data store shares many characteristics with data warehouses. Data warehouses are subject oriented, time variant, and nonvolatile stores of summarized reporting data (Inmon et al., 2010). Our data store incorporates all of these properties. The data structure of our database is based on subjects such as the summarized analytics data of a given webpage and the pages users navigated to next. These are stored as a single document in our database (see below for a detailed data design.) We also store our data in a time variant and nonvolatile way. We are interested in analytics data in snapshots of time. Because of this, we store the analytics data pulled from Google Analytics with time stamps to indicate when it was pulled from the API; this allows us to look back on changes in traffic patterns over time. Although we are following many of the concepts of traditional data warehousing, we are not constraining ourselves to typical data warehouse design.

We are designing our database with facts and dimensions, just like a traditional data warehouse star schema. Because NoSQL relies less on relationships between documents we flattened out the facts and dimensions of our star schema into a single document. For example, our PageSnapshot object (See Figure 5 below) represents a fact. That fact contains summarized data about a specific web page at a specific time. The time data was retrieved, data about the page itself and its related pages are all dimensions that can be used to query information about that fact. As demonstrated below in Figure 5, the

facts and their dimensions are stored in the same document, which is more consistent with NoSQL document based data design.

For our data store we use a NoSQL database (Pokorny, 2013). NoSQL data stores provide a few key advantages we are interested in. NoSQL offers schema less design allowing us to easily expand our data models to add new functionality. As we discover new important metrics about user patterns and expand our framework, we will need to expand the data model and add additional summarized statistics. NoSQL gives us the ability to do this on the fly without completely redesigning our database schema. This gave us a good deal of flexibility during the design process. Most NoSQL implementations are also very horizontally scalable, meaning we can easily scale our single database to account for increased traffic. This means that even with our relatively limited resources and funds are able to create a scalable database that could be applied to a very popular website like the UNF website. By simply requesting additional instances of our data store we can rapidly increase the performance of our framework. Finally, NoSQL databases offer extremely quick reads and writes across multiple instances with an "Eventual Consistency," meaning we can very quickly perform writes to the data store and eventually get consistency with other users on different instances. Because we are not writing a purely transactional system we are not necessarily concerned with the immediate consistency between queries offered by traditional SQL databases, and as a result we are able to take advantage of the performance gains afforded by having multiple independent instances of our data store (Pokorny, 2013). We discuss our specific choice of NoSQL technology in section 4.3.

The design of our data store (see Figure 5) uses the concept of documents (Pokorny, 2013). We define document types for the different features of our framework. Below are the data definitions of our documents in UML format. Essentially they are documents containing key value pairs with sub documents containing their own key value pairs. These nested documents are stored together rather than in traditional in related tables.

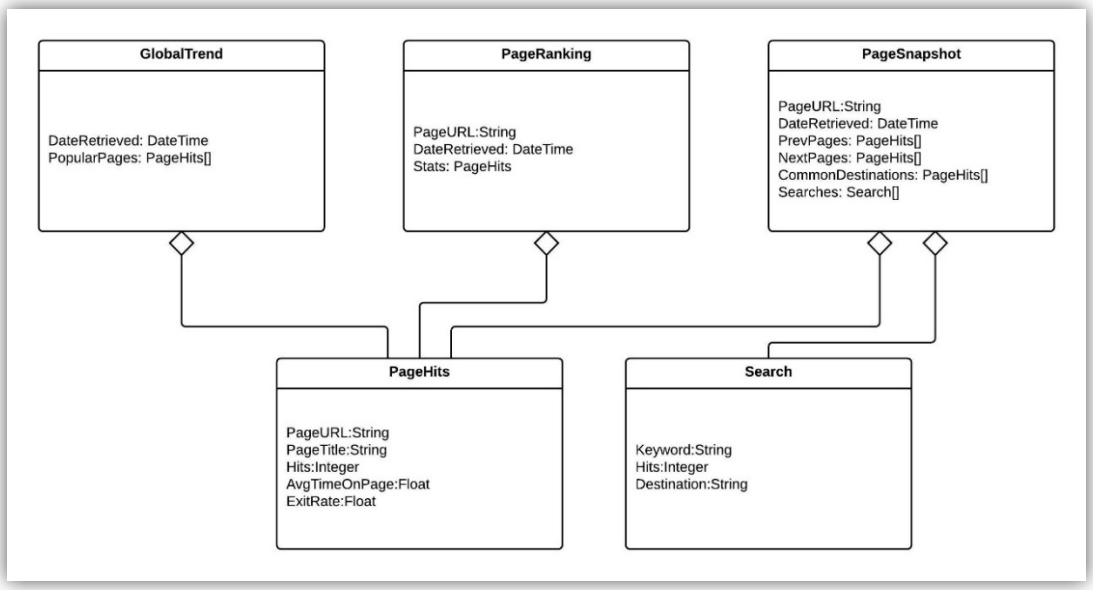


Figure 5. Data Store Schema

We have three main document types: GlobalTrends, PageRanking, and PageSnapshots. The GlobalTrends documents contain information about the most popular content on the site overall. This information is queried from various endpoints of the Google API and consolidated in a single document. These documents have time stamps of when they are retrieved allowing us to set an expiration time for this data as well as track changing usage trends over time. The PageRanking document contains data about links and their

popularity so they can be ranked against each other. The PageSnapshot documents contains information about specific pages a user is visiting. It contains information about the pages that are often navigated to next, which pages are often linked to the current page, and the common end destinations when navigating through this page. These documents also contain information about searches performed from this page and where those users eventually ended up. All these data are collated from various queries to the Google API and stored in this format to maximize retrieval speed. The sub-documents of PageHits and Search contain the raw data about page hits and search queries and are contained within their parent documents. We place indexes on the DateRetrieved and PageURL properties to improve performance of select queries on these properties.

Because our data is stored in a time variant way, we will need to come up with a set of parameters as to when we will refresh the data in our warehouse. To come up with these parameters we looked at research about how to best report on analytics. Because we are, in a way, reporting on website usage (instead of human readable reports, we used this same data for UI adjustments), we used established research about how to best report on this data (Gonçalves & Ramasco, 2008). The standard way web analytics are analyzed can vary based on the purpose of the report, but we settled on a fairly standard way of looking at this data. We gather summarized data about the previous week of activity and we refresh this data once per day. This will give us a good picture of popular items over a given week, while not favoring too heavily anomalous spikes that happen during a given day. This should result in website that is refreshed daily with the latest popular

content, but does not react too drastically to sudden spikes in traffic (Phippen et al., 2004).

### 4.2.3 Web Service Layer

Once the analytics data has been gathered and stored in our database, we will need to expose that data to client browsers. We use a web service layer that serves as the endpoint for queries on page analytics data. We expose this data via REST web services that return the summarized data from our data store in JSON format. We have three endpoints, one that returns a snapshot navigation summary of the paths in and out of a given page "Snapshot," one that returns global popularity rankings for a list of links "Ranking," and one that returns a list of globally popular links for the site as a whole "Popular." Figure 6 provides UML class representation of the REST web service.

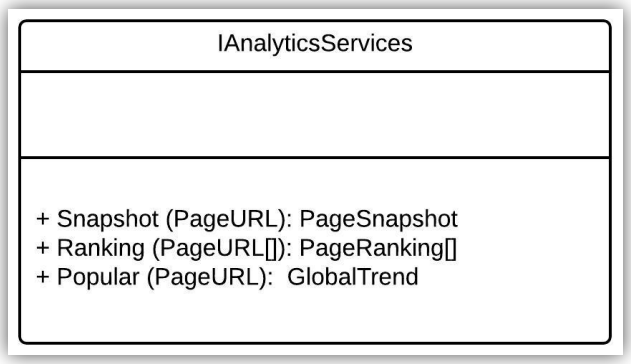


Figure 6. Web Service Interfaces

The web service application is responsible for determining where data is pulled from. The logic for determining whether the cache and the database are up to date exists in the web service layer. In addition, the web service layer is responsible for creating another threaded task to update the data store when it is discovered to be out of date. Figure 7 provides a UML sequence diagram of the data flow logic that determines where the analytics data is pulled from when the web services are called.

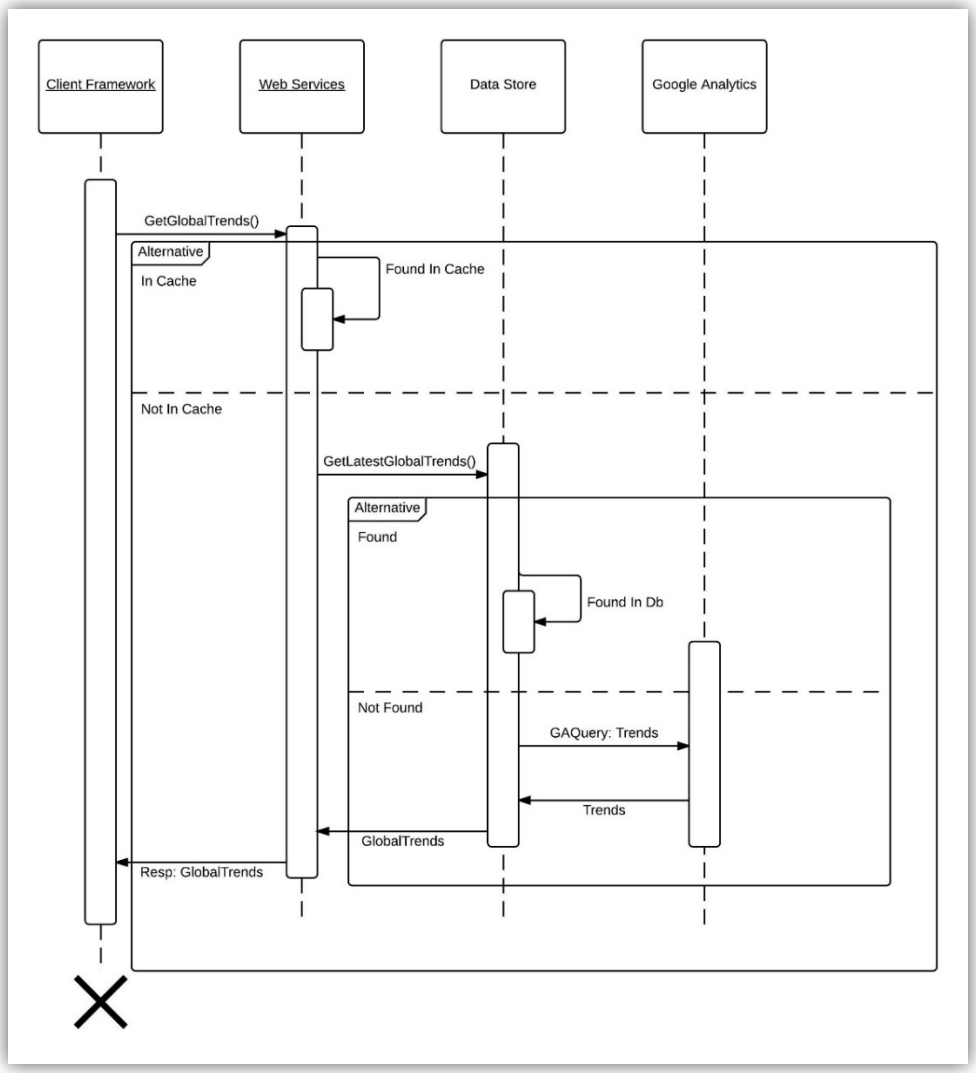


Figure 7. Web Service Sequence Diagram

The most important challenge for these web services is speed. These services need to return the requested usage data as quickly as possible so the load time the user experiences when visiting our modified site is as small as possible. To accomplish this we heavily utilize distributed caching technologies. Section 4.3 provides a detailed description of the specific technologies.

### 4.2.4 Client Side Framework

The final component of our system is the client-side framework that adapts the user interface of the web page. The primary function of this component is the adaptation of the user interface based on the data retrieved from the web services. The goal of this component is to change the interface in subtle ways that surface more popular navigation options, while not changing the interface in a way that disorients returning users. To do this, we considered all the lessons we learned about web usability, which we outlined in the background and literature review chapter.

The client-side framework needs to be highly customizable for web developers. We want to surface the analytics statistics we gather in such a way that developers can define behaviors based on data returned. Developers are able to subscribe to certain events in the client side framework that ranks navigation options on a page and allow developers to assign different styles to navigation options of different popularity. The client side framework also has functions that return popular and trending topics on the site as a whole (global trends,) which allows developers to create sections of a page that always

display the most important links on a site, and dynamically updates as those popular links change.

The way pages are ranked is also customizable in the client side framework. Developers can choose which metrics determine the popularity of links. These metrics include: page hits, time spent on the page, exit rate, and in the case of search auto complete, common destination page count. This allows developers implementing our framework to determine which metrics are most important for ranking links on their site. For example, if a site provides large information pages that users spend a lot of time on, then average time spent on a page will be more important than the raw number of hits on the page. Each call to our web services can be customized in this way to best fit the site on which it is being implemented.

We also utilize the virtually industry standard jQuery framework to assist with Document Object Model (DOM) manipulation tasks. The DOM is an interface for dynamically accessing and modifying the content and structure of HTML documents via JavaScript (W3C DOM Interest Group, 2005). To update the user interface of a website programmatically, we need to manipulate the DOM by adding styles and HTML elements. jQuery is used industry wide for client side user interface design, Additionally it greatly speeds up development efforts for the client side framework over vanilla JavaScript (jquery.com).

## 4.3 Technology

While researching different technologies to build our system, we found many different options that each offered their own unique advantages. In the end, we settled on a technology stack that allowed us to easily integrate all the modules of our application while also providing high performance scalability. In order to fulfil the requirements presented in Table 3, we analyzed two different possible technology stack options, as outlined below.

| | |
|---|---|
| **Server** | Host web application |
| **Web Application Language** | Application logic<br>Handle REST API requests |
| **Cache Technology** | Store frequently accessed reporting data |
| **Database** | Long term data storage<br>Historical reporting data |

Table 3. Components

### 4.3.1 Google App Engine Technology Stack

Our first technology stack choice is the Google App Engine platform (Google App Engine, 2014). This platform offers a high performance in-memory caching strategy backed up by a NoSQL database infrastructure based on Google's own BigTable technology (Google App Engine, 2014). This allows for automatic caching of frequently accessed data without additional programming effort integrating cache and database technologies. It also offers full-featured web application hosting for our REST web services and data extraction service layer using the Python language. We are already familiar with this technology stack, so the learning curve was not steep. App Engine also

promises to be highly scalable if we need to subject the system to heavy load (Google App Engine, 2014). The reason we considered the App Engine technology stack is that it offers all the components we require in a single integrated stack. With minimal integration work were able to satisfy all of our technology requirements. The pricing model for App Engine is also reasonable, and we discuss the details later in section 4.5. Table 4 provides summary of Google App Engine technology stack.

| | |
|---|---|
| **Server** | Google Cloud Platform (cloud.google.com/appengine/) |
| **Web Application** | App Engine Python Runtime Environment |
| **Cache** | Google NDB Datastore |
| **Database** | Google NDB Datastore |

Table 4. Google App Engine Technology Stack

4.3.2 Microsoft Technology Stack

As an alternative option, we have also chosen another technology stack that could satisfy the same technological requirements as the App Engine stack. We are also very familiar with the Microsoft .NET technology stack and we can use a collection of other tools to produce the same environment that is packaged together with Google's App Engine. The integration effort for this technology stack would be significantly higher than the app engine stack. The Microsoft .NET MVC framework allows for the development of REST web services and the creation of services for extracting data from the analytics API. The Redis cache server allows for in-memory storage of frequently accessed reporting data, and the mongoDB database allows for more permanent storage of historical reporting data. The difficulty of this technology stack pertains to the

integration effort between the components. There are frameworks available for integrating these different technologies, but the integration and installation efforts would be significantly higher than the Google App Engine stack, which comes pre-installed and integrated out of the box. The pricing for this stack would likely be higher than the App Engine stack and would take more effort to integrate each piece. Table 5 provides a summary of Microsoft technology stack.

| | |
|---|---|
| **Server** | Microsoft Windows Server 2013 running on Amazon EC2 Web Services (aws.amazon.com/ec2) |
| **Web Application** | Microsoft .Net MVC4 Web API (asp.net/web-api) |
| **Cache** | Redis Cache Server (redis.io) |
| **Database** | mongoDb (mongodb.com) |

Table 5. Microsoft Technology Stack

## 4.4 Budget

Below we outline two separate budgets for each of the technology stacks identified for the development of our system. All the development tools and machines we used are either already owned or free and open source.

### 4.4.1 Budget: Google App Engine Technology Stack

Google App Engine charges by the hour per running application instance, for outgoing network traffic, file system and database storage, and read/write operations on the database. Table 5 provides summary of the potential cost of running our application in a production system with live traffic. Because we are performing a much more limited test

involving a relatively small number of users we were able get by using only a single free instance, meaning our total hosting costs were $0.

| | |
|---|---|
| **Google Analytics**<br>50,000 requests/day | Free |
| **Google App Engine**<br>5 Instances 150 instance hours<br>500 MB outgoing network traffic<br>500 MB file system storage | $7.79/mo |
| **Google NDB Datastore**<br>5GB stored data<br>100,000 read & 100,000 write operations | $1.02/mo |
| **Total** | $8.81/mo |

Table 6. Budget: Google App Engine Technology Stack

4.4.2 Budget: Microsoft Technology Stack

For the Microsoft technology stack we would have utilized mostly open source and free technologies. For application hosting we would have used Amazon's EC2 dedicated hosting platform which charges for running instances only. We would have created a server instance and only paid for it while the server was running, keeping costs low. Table 7 outlines the costs for this strategy estimating 150 running instance hours.

| | |
|---|---|
| **Google Analytics**<br>50,000 requests/day | Free |
| **Amazon Web Services EC2**<br>Windows Server 2013 Large Instance | $0.329/hour (running instances only) |
| **.NET MVC 4** | Free |
| **Redis Cache Server** | Free (Open Source) |
| **mongoDB** | Free (Open Source) |
| **Total (150 hours)** | $49.35 |

Table 7. Budget: Microsoft Technology Stack

### 4.4.3 Technology Stack Choice

After analyzing the two technology stack options outlined above, we decided to utilize the Google App Engine stack. The App Engine stack offers tighter integration between different components of our framework, all the components mentioned above are integrated out of the box and built into the App Engine API. In contrast, the Microsoft technology stack would have required installation and integration of the different open source components needed to develop our full solution. In addition to the extra integration efforts needed for the Microsoft stack, the cost of running the servers was also a factor in our decision. Because the Microsoft stack requires a dedicated virtual server, as opposed to a shared application hosting environment, the cost to run our solution on that stack would have been significantly higher. The development efforts in terms of application logic for either was comparable, as we have experience developing with each these technology stacks. Although both options fit our needs, we believe that the App Engine stack made development easier and resulted in a better architected solution.

# Chapter 5

# REAL-WORLD APPLICATION

As part of the development process of our system, we applied the framework we developed to the University of North Florida website (UNF, 2014). We have full access to the UNF Google Analytics data, thanks to cooperation with the ITS department. We have also received approval from ITS to use this data for our research. We produced an alternate version of the UNF website that utilizes our framework, and applied analytics based user interface changes to the site for the end user testing we describe in chapter 6.

We implemented our framework on multiple pages of the UNF website. To implement our framework on the live site we injected our script into the website by creating a simple proxy. We wrote the script to apply analytics based improvements to some of the common navigation elements present across the site in addition to some page specific improvements made to the homepage and some other high traffic pages. Our goal is to make the site easier for users to navigate by surfacing the links most commonly used on each of these pages. The UNF homepage alone has over 70 links to other pages (Figure 8), we believe we can draw attention to the most important links on this page based on current user trends. Our ultimate goal is to offer users with the navigation options they are looking for without having to use the search feature on the page.

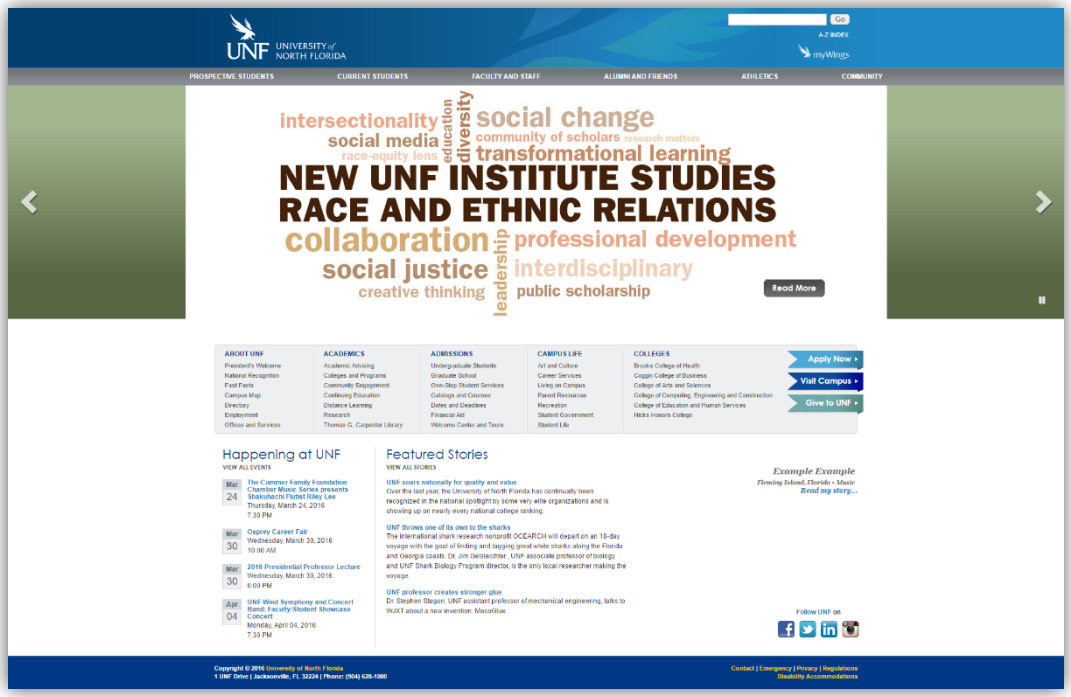


Figure 8. UNF Website Homepage

The changes we made to the site can be categorized in three different ways. We visually distinguished popular links so they are more prevalent on the page, provided additional popular links in context menus and other places, and improved the search suggestions on all pages. The following sections provide more details on these improvements, including JavaScript code snippets and how they were implemented.

## 5.1 Ranking Links

Firstly, we visually distinguished popular links on pages based on analytics data. Using our client side framework, a developer simply needs to point to a set of links on the page by selecting their DOM elements (W3C DOM Interest Group, 2005) and our framework

will crunch the numbers and apply styling based on their relative popularity. Figure 9 below shows an example of this ranking.


Figure 9. Ranking Menu Links Example

We applied two different style tweaks based on popularity of the links. The more popular the link is (how many users visited that page next, i.e., after the current page) the darker the background color and the larger the text. This is a very simple example of drawing user's eyes to the most popular links by visually distinguishing them from the less popular ones (U.S. Dept. of Health and Human Services, 2006). Our framework gives designers the power to control how popular links are visually distinguished from less popular links. We simply provide the ranking values and let the designer decide the range of Cascading Style Sheet (CSS) values (Bos, 2015). The style values can apply to any numerical CSS style property, including colors, sizes, margins, etc. The result in Figure 9 above can be created with the following small snippet of code which applies both color and font size ranking to all the links in the menu:

```
$("#UNFbignav li a")
        //Rank with color range
        .ActiveAnalytics("rankstyle", $.extend({
            rank: {
                rangeStart: "#EEEFF0",
                rangeEnd: "#86C3FF",
                rankBy: "hits",
                style: "background-color",
                distribution:"even"
            }
        }, settings))
        //Rank with font size
        .ActiveAnalytics("rankstyle", $.extend({
            rank: {
                rangeStart: 11,
                rangeEnd: 13,
                rankBy: "hits",
                style: "font-size",
                unit: "px"
            }
        }, settings));
```

## 5.2 Global and Context Based Suggestions

In addition to ranking existing links on pages with styling, so as to not disrupt users who visit the site frequently by moving or changing links (Bevan, 2005), we also added dynamic elements to the page that will change over time. These elements change based on popularity, giving users contextually and seasonally appropriate links based on changing site usage patterns. We have been careful to mark these navigation elements as "Popular Links" to make users aware they can look to these navigation elements to give them the currently most popular pages, but not necessarily rely on them to be exactly the same on each visit.

We have two different implementations of these popular link navigation elements. Firstly we have popular link menu blocks. These menu blocks give users the most popular links visited on the site as a whole. For example in Figure 10 below, we provide a list of links most popular on the entire site.

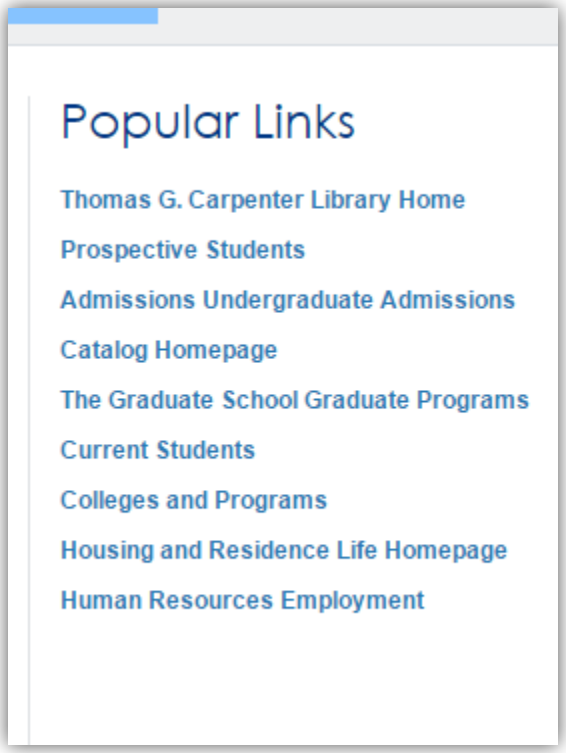


Figure 10. Popular Links Example

The above popular links module can be implemented with the following code:

```
$("#aaPopular").ActiveAnalytics("popular-global",
        $.extend({wrap: $("<h3>")}, settings));
```

We also have a second popular link navigation module, which provides users with context based suggestions on where to navigate next. This module is more complex than the simple menu block above. This module creates menus with popular links based on a navigation element the user is currently hovering over with their mouse. The idea of these menus is to give users suggestions based on the link they are about to click on. This allows our framework to make a more educated guess on where the user is trying to navigate. A great example of this is the top menu on the UNF homepage (Figure 8.) For

example, if a user hovers over the “Current Students” link, we provide that user with suggestions based on where previous users navigated after they reached the “Current Students” page, thereby potentially bypassing the navigation step of clicking through to that page. Figure 11 below gives an example of how these suggestions look.


Figure 11. Contextual Popular Links Example

Figure 11 shows that the hover suggestions given to the user display the most popular links clicked on from the “Current Students” page. This gives users quicker access to links current students specifically may be interested in. The example above can be applied to an entire menu with the following snippet of code:

```
//Create hover suggestions
$(".audience a").ActiveAnalytics("hover", settings);
```

## 5.3 Search Suggestions

Finally, search suggestions have been improved by replacing the standard static search suggestions currently on the UNF website with a dynamic list of links based on popularity. On the current site there is a single list of links used on all pages for search suggestions. Our framework taps into search data from Google Analytics and provides the user with more useful suggestions. We are able to pull from our analytics data and determine what keywords a user searched for on any given page. We can then determine where on the site those users ended up, giving us the ability to skip the search altogether and jump the user directly to the results. This gives us the ability to determine what information on a webpage is popular, but hard enough to find that users must search for it. Figure 12 shows an example of these dynamically driven search suggestions.

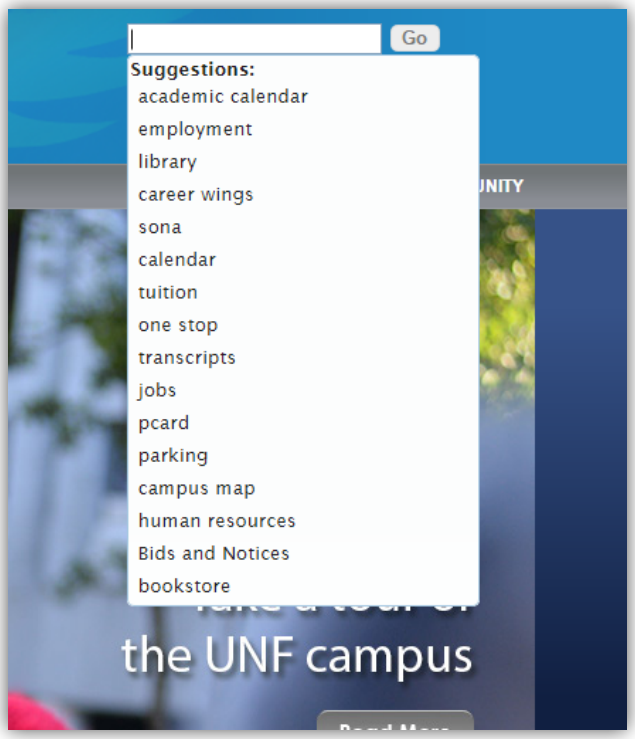


Figure 12. Search Suggestions Example

These suggestions represent the most popular recent queries on this particular page and take users directly to the most likely result. Each of these search suggestions leads users directly to the destination page for that query as determined by popularity. Search suggestions are a complicated area of study all on their own, with plentiful academic research on how to best generate them. We can't possibly tackle this vast topic in detail on top of all our other work so we are relying on some existing research while providing some tweaks on that well established research. Our goal is to take popular search queries from a given page, rank them in popularity based on the destination pages other users reached with that query, then provide those queries and destinations to end users as suggestions. We rank search suggestions for a given page based on how many people made that query, and how many of those users made it to the same destination (Santos, Macdonald, & Ounis, 2013). If multiple users made the same query and ended up on the same resulting page time after time, we can assume that the information users are searching for is on that destination page and jump them directly there. We believe this is a potentially very exciting avenue of research. Based on our relatively limited testing of this feature, it seems that it could provide users with an incredibly fast avenue directly to the information they are looking for. The search suggestions in the above example can be implemented on a site with the following simple code snippet:

```
//Populate search suggestions
$("#box").ActiveAnalytics("search", settings);
```

The above examples demonstrate that applied our research into analytics based navigation improvement was applied in various ways. We applied what we learned in our research into web usability to develop a site that dynamically adapts its navigation based on changing usage patterns. We have also attempted to make implementation of

our framework as simple as possible as evidenced by the code snippets provided with each example. We believe that the various analytics based improvements to site navigation will allow users to navigate the UNF website more efficiently, and find what they are looking for quicker. In order to evaluate the effectiveness of these changes, we conducted tests with real users comparing our updated version of the site with the original site. An in-depth description of these evaluation methods can be found in the next chapter.

# Chapter 6

# EVALUATION

As we mentioned in the previous chapter, we have built an alternate version of the UNF website and tested the two versions of the site with different users. We measured the time it took users to complete certain tasks, and gathered qualitative responses about their experience navigating the site. In this chapter, we describe in detail how we evaluated these two designs.

## 6.1 Evaluation Goals and Objective

The goal for the evaluation stage of this research is to prove that our Active Analytics system has improved the usability of our example site. To do this we set up an A/B experiment evaluating the two versions of the UNF site with different users and compared certain statistics about their usage. We will describe this experiment in detail in section 6.2. The different versions of the site served as our independent variable. The participants were given a fixed set of tasks to accomplish. They were given either the updated version of the site using our framework or the current version of the site to accomplish those tasks. Participants were requested to complete seven different navigational tasks. See Appendix A for task descriptions presented to participants. Multiple dependent variables were measured about the usage patterns of users on the two different versions of the site. Firstly, we measured the time to complete each task starting from the time the users were presented with a landing page to the time they reached the

destination page. Secondly, we measured the number of clicks it took users to get to that destination. Finally, we requested that users complete a short survey on their experience, See Appendix B for the full survey instrument.

There are some extraneous variables that also needed to be taken into account. First, the population we chose for our evaluation could potentially skew our results. Factors like user's familiarity with the existing site and user's overall computer literacy need to be accounted for. To mitigate the impact of these potentially confounding variables we assembled a diverse sample size of both technical and non-technical users, as well as users that have varying levels of familiarity with the existing site. We present the demographic information of our participants in the next chapter.

## 6.2 Testing Process

Our testing process was a simple A/B testing approach (Kohavi, Longbotham, Sommerfield, & Henne, 2009) to evaluate the new design of the site alongside the original design. A/B testing is a popular method of evaluating alternate designs of a user interface. Normally A/B testing assigns random users to different versions of the same interface. Statistics about decision time, conversion rate, and user satisfaction, are compared between the two designs and a decision is made based on the success of one interface over another (Kohavi et al., 2009).

We followed the same overall idea, but used a less programmatic, and more manual approach. Because we cannot place our alternate design on the live UNF website we

were not be able to gather large amounts of statistics with automated A/B testing. We instead presented the updated interface to users individually, and asked them to complete some simple tasks, such as: “navigate to the course registration page.” For a full listing of all the tasks we asked users to perform see Appendix B. To some users, we simply presented the current live UNF website. We then asked them to perform the navigation tasks mentioned above, these were our control users. To other users we presented the updated version of the site using our framework and asked them to perform the same set of tasks as the control users. We also asked the participants to complete a short survey about their experience navigating the site when the testing was complete. See Appendix B for the full survey. We randomly assigned users to each version of the site and asked them to perform the same set of tasks. Users performed multiple tasks on their assigned version of the site to make up for our relatively small sample size.

Because users will become familiar with the version of the site they first use, we were not able to ask the same user to perform the task on both versions of the site. Once users were assigned to a version of the site, they completed all of their tasks on that version, either with the framework enabled or on the original unaltered site. Figure 13 below illustrates our evaluation process.

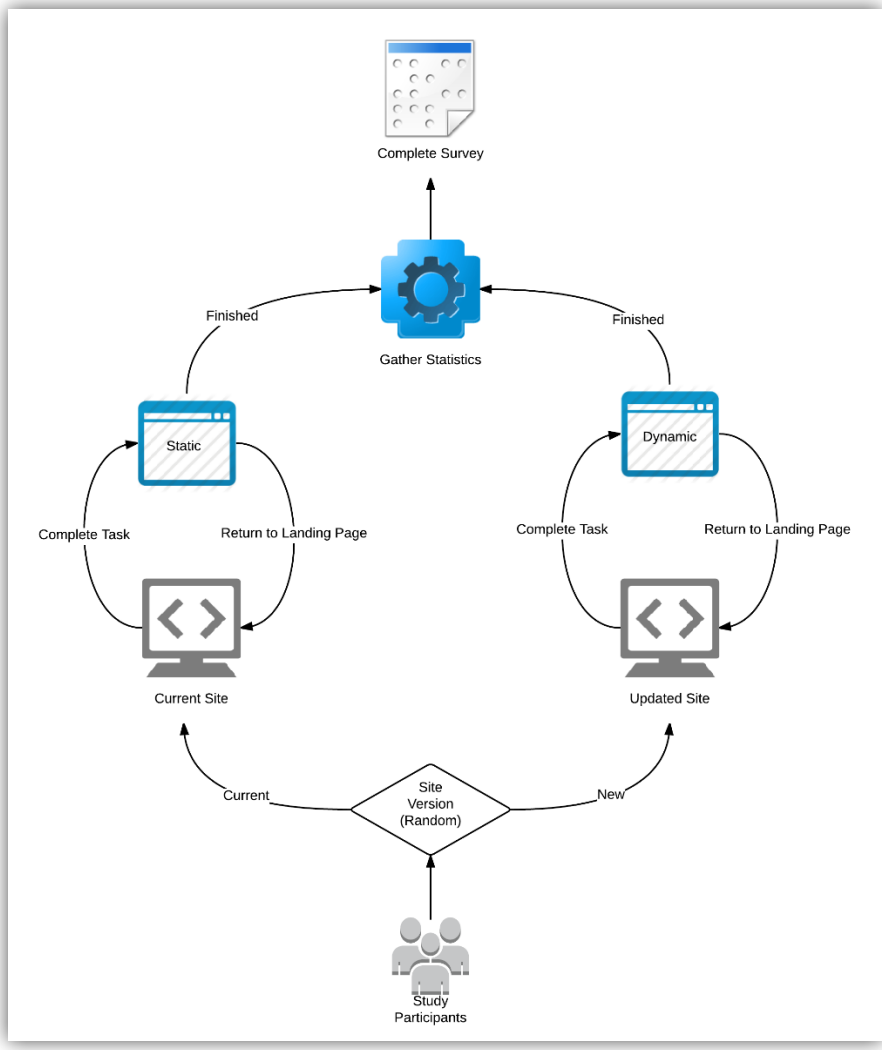


Figure 13. Evaluation Process

## 6.3 Study Participants

We asked for volunteers for our testing process and did not offer monetary compensation. We asked mainly UNF students to participate. Because users are volunteering their time, we kept the testing process short and only took up around 10-15 minutes of the user's time to decrease the chance that a user would quit before finishing. Because of this, we

chose tasks that did not require the user to locate obscure information and had relatively short critical paths from the UNF homepage. We created an automated testing process that guided the users through a set of small tasks and record the time along the way. At the end of the process the users were asked to provide feedback on the interface in the form of a short survey. We sent requests to faculty members within the School of Computing and some other select faculty in other colleges to request the participation of their students in the study. The professors were asked to place a link to the study on their class Blackboard page, and notify the students that they can participate in an optional study that had no effect on their grade.

## 6.4 Institutional Review Board (IRB) Approval

Because our study utilized human subjects as a part of the testing, we submitted the project to the UNF Institutional Review Board (IRB). Our study presented no risk to participants and therefore qualified for expedited IRB review. The study was approved on September 2nd 2015. The IRB reference number for this project is 784254-1 (see Appendix C).

# Chapter 7

# EVALUATION RESULTS AND ANALYSIS

For our study, we collected both quantitative and qualitative data, and we handled each differently. We gathered the following quantitative data by recording participant activity on the site: time to complete tasks, number of steps to reach a destination, and the number of times a task was skipped. We recorded this data for each version of the site separately and compared the two groups against each other. We used an independent samples t-Test (Salkind, 2010) to determine if the averages for navigation times, and navigation steps where significantly different between the two sides of the A/B Test. We provide more information on the t-Tests we performed in section 7.2. In addition to the t-Tests performed on completion time and navigation step data, we also performed a chi-square analysis on the proportion of users who were unable to complete each task on the two version of the site. The results of the chi-square test are included in section 7.3. Finally, we performed effect size calculations to determine the practical significance for each of our quantitative measurements in section 7.4.

For the qualitative survey data, we included the mean rating of each version of the site for the qualitative survey questions, as well as the demographics information of the participants. Because the free text fields in the survey were optional there weren't a significant number of useful comments to perform any meaningful analysis on them.

The goal of this evaluation is to validate our hypothesis, that our updated analytics-driven version of the site helped users complete the assigned tasks faster, and provided a more user-friendly experience than the original version of the site. We will now cover each of these metrics in detail.

## 7.1 Demographics

We asked the participants to provide some basic information about themselves and their familiarity with the UNF website and the Internet in general. We wanted to get an idea of how varied our sample was. Based on the survey responses we found that most of the participants were in the 18-25 age group and identified themselves as experienced with the Internet. We wanted to survey participants with various levels of familiarity with the UNF website, and according to survey results we were able to get a wide sampling of participants that use the UNF website at varying levels of frequency. Figures 14 through 18 provide an overview of the summarized results of the demographic questions we asked in the survey. The figures show all the possible answers for each of the questions and the total number of users that selected each answer. We were able to get a wide range of participants with different class standings and different levels of familiarity with the UNF website.

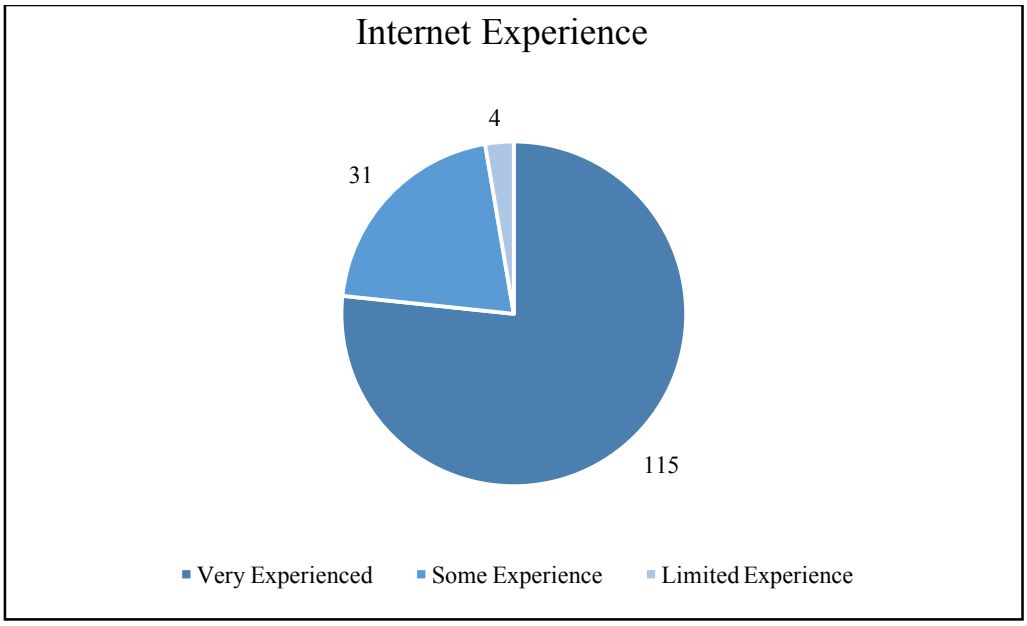


Figure 14. Survey - Internet Experience

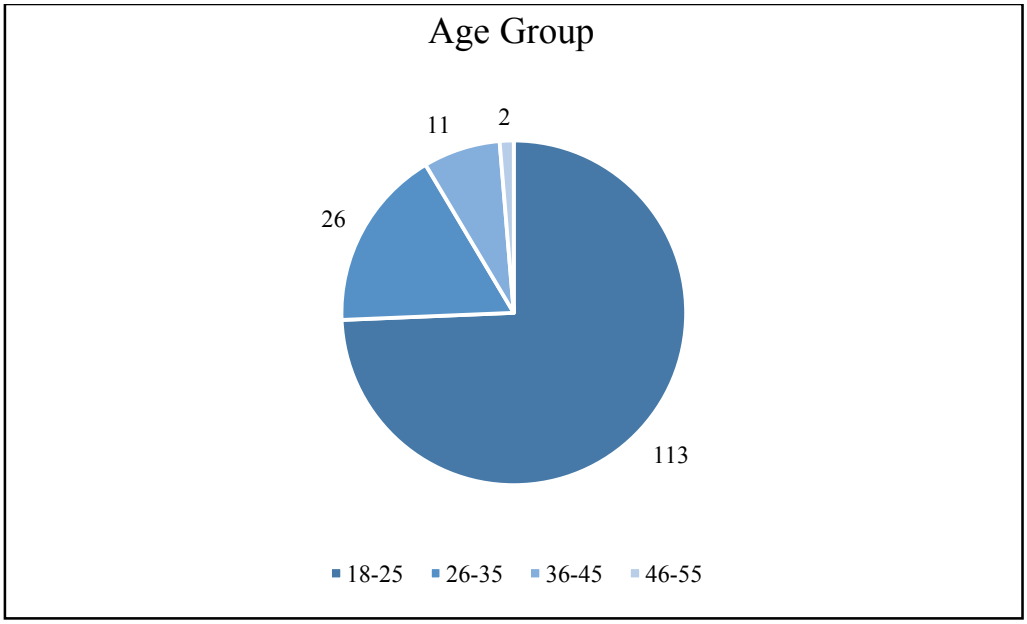


Figure 15. Survey - Age Group

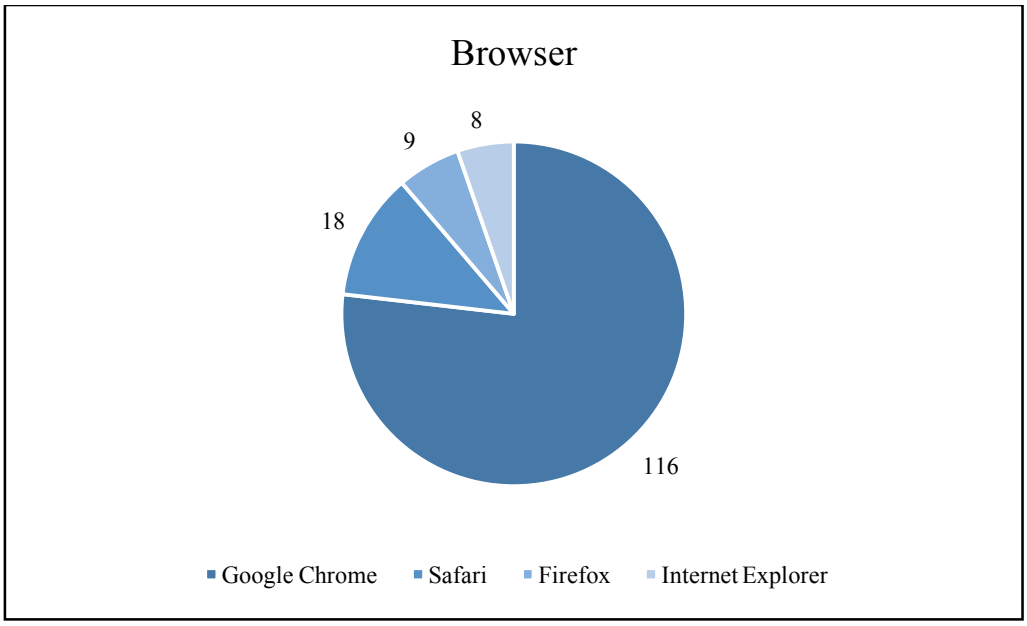


Figure 16. Survey - Browser

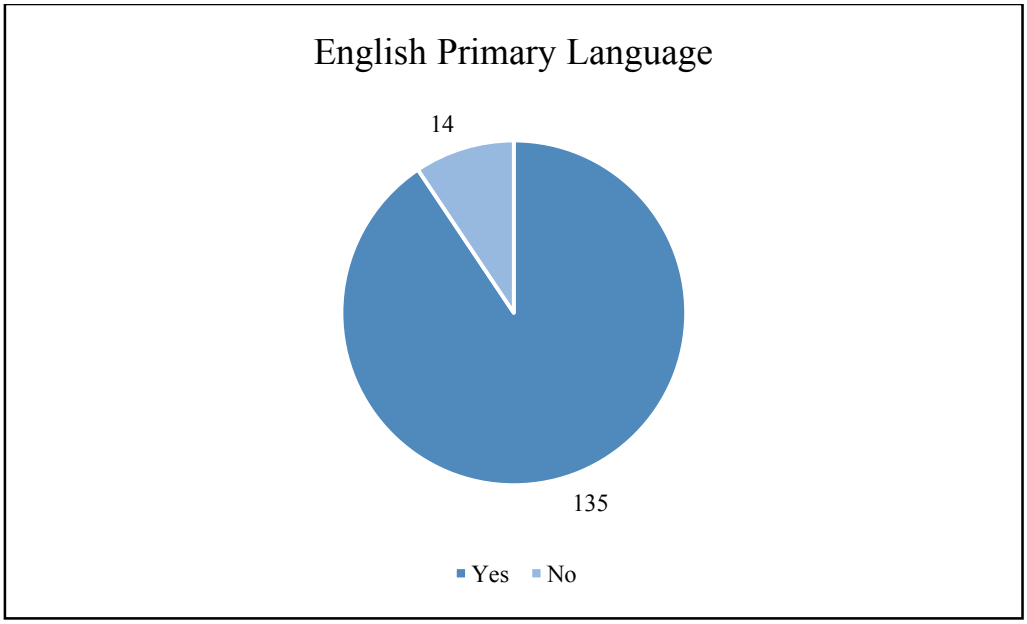


Figure 17. Survey - English Primary Language

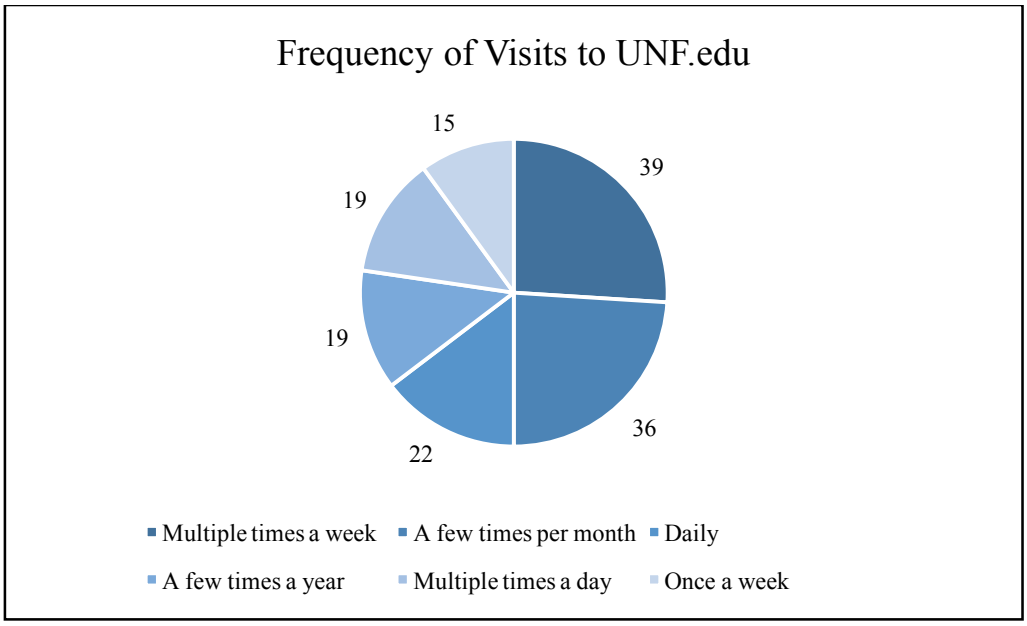


Figure 18. Survey - Frequency of Visits to UNF.edu

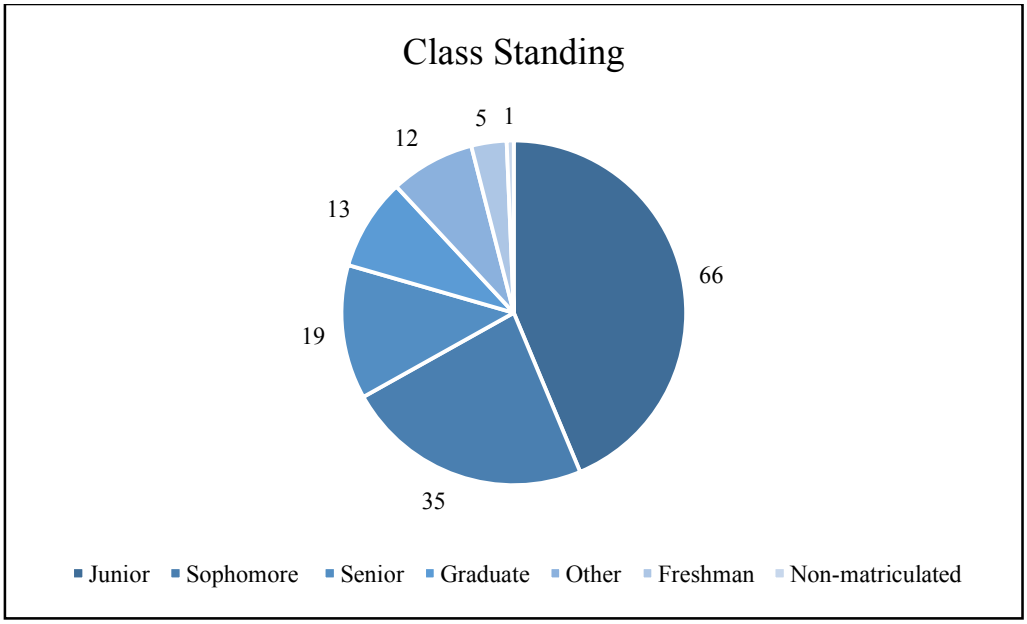


Figure 19. Survey - Class Standing

## 7.2 Independent Samples t-Test Results

For each of our measured metrics we compared the two groups of users and determined if the difference in averages between the two groups was statistically significant using an Independent Samples t-Test (Salkind, 2010). There are three assumptions that must be met for an independent samples t-Test to produce accurate results (Salkind, 2010):

- Assumption 1: The data for the two groups in the study must be independent observations. Each observation cannot be predictive of another observation in the study. For our study, each participant was randomly assigned a version of the website and asked to complete all tasks on that version. No user was able to participate in the study more than once, and the participants never interacted with each other as part of the study.
- Assumption 2: The second assumption is the equality of variance in each of the populations. For each of the metrics in the study we used Levene's test for equal variances (Salkind, 2010) to determine if this assumption is met. If the results of Levene's test are not significant ($p > 0.05$) then the assumption holds true. If this is the case we refer to the "Equal variances assumed" value in the t-Test table. If the results of Levene's test are significant, we instead retrieve the t-Test value from the "Equal variances not assumed" row of the result table.
- Assumption 3: The final assumption states that the sample must be drawn from a population that follows a normal distribution. We tested all of our measured metrics for normality using the Shapiro-Wilk test for normality (Salkind, 2010). For those results that did not follow a normal distribution we applied a transformation to fit that data to a normal distribution. For each of the metrics

below we will state the normality of the data and what transformations we had to use to fit the data to a normal distribution. It is worth noting that this assumption can be violated for reasonably large sample sizes (N > 30) as long as the departure from normality is not too severe (Salkind, 2010).

### 7.2.1 Task Completion Times

For each task we measured the time it took each user to make it from the homepage to the destination page. Our hypothesis for this metric is that our modified version of the site allows users to complete the tasks more quickly than the original version of the site. The null hypothesis we would like to disprove is that it takes users the same amount of time no matter which version of the site they use. Figure 20 contains the average times it took users to complete each task separated by whether they were given the original UNF site or the modified version of the site using our framework.

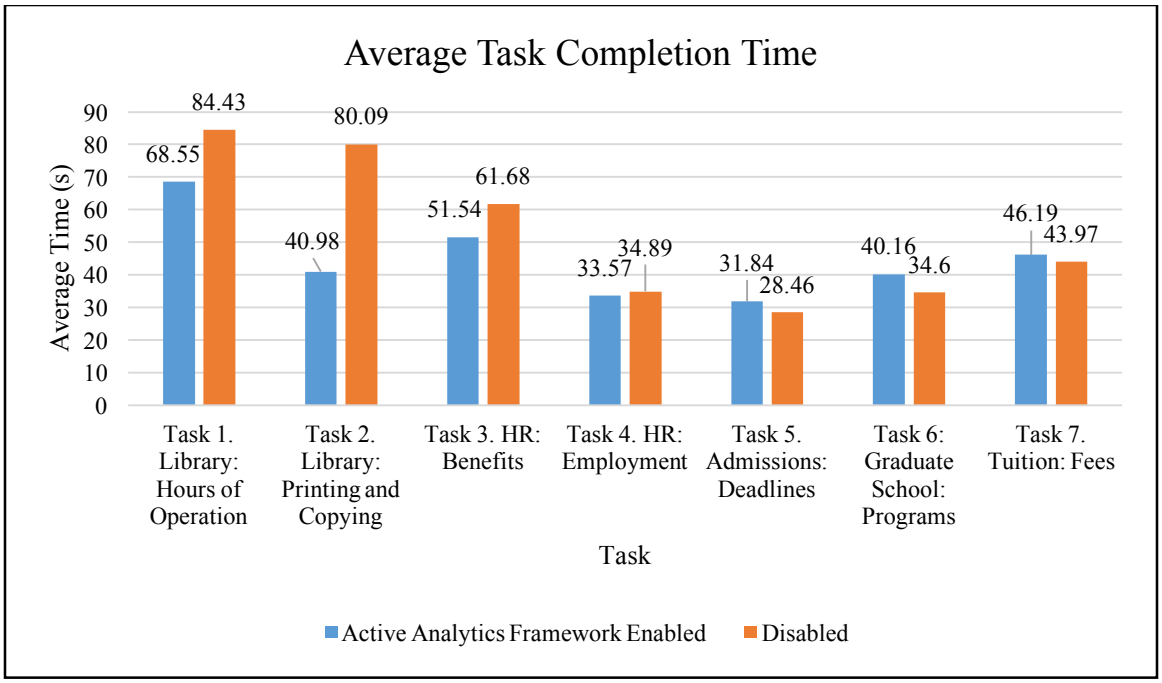


Figure 20. Average Task Completion Time

The graph above shows that four of the seven tasks were completed in a shorter average time when the framework was enabled. The final three tasks were completed in a shorter time on the non-modified version of the site. For each of these tasks we performed an independent samples t-Test to determine if the difference in the two average times were statistically significant enough to disprove the null hypothesis that navigating the two versions of the site result in the same average completion times.

7.2.1.1 Normalizing Measured Data

As stated in the t-Test assumptions above, the data being analyzed via independent samples t-Test must fit a normal distribution. In order to satisfy this assumption we first had to transform the data, as it did not fit a normal distribution. We used either a log or square root transformation on the navigation times data to fit it to a normal distribution. Table 8 shows which transformation was used for each task to normalize the data. After performing the transformations, the Shapiro-Wilk test was used to determine if the transformed data fit to a normal distribution. In Table 8, we have included the results of the Shapiro-Wilk test on the transformed data. The Sig. column contains the result of the Shapiro-Wilk test. For each of the tasks the value was above 0.05, indicating the results do not significantly deviate from the normal distribution.

| | Framework | Transformation | Statistic | df | Sig |
|---|---|---|---|---|---|
| Task 1 | enabled | SQRT | 0.965 | 46 | 0.174 |
| | disabled | | 0.972 | 67 | 0.137 |
| Task 2 | enabled | LOG | 0.975 | 11 | 0.931 |
| | disabled | | 0.986 | 62 | 0.710 |
| Task 3 | enabled | LOG | 0.978 | 41 | 0.613 |
| | disabled | | 0.983 | 67 | 0.506 |
| Task 4 | enabled | LOG | 0.983 | 45 | 0.737 |
| | disabled | | 0.913 | 66 | 0.190 |
| Task 5 | enabled | LOG | 0.982 | 51 | 0.633 |
| | disabled | | 0.963 | 73 | 0.290 |
| Task 6 | enabled | LOG | .983 | 52 | 0.654 |
| | disabled | | .979 | 71 | 0.267 |
| Task 7 | enabled | LOG | .985 | 50 | 0.764 |
| | disabled | | .980 | 66 | 0.366 |

Table 8. Normality Test for Task Navigation Times

In tables 9 and 10 we include the group statistics for the task navigation times measured during the study. Table 9 includes the group statistics prior to normalization. Table 10 includes group statistics for the normalized navigation time data, the normalized data set used for our t-Test analysis below. As show in tables 9 and 10, after the values were normalized the skewness and kurtosis values were all much closer to 0 indicating a normal distribution. Only task 5 has a kurtosis value slightly larger than the acceptable value of between -2 and 2.

| | Framework | N | Mean | Std. Deviation | Std. Error Mean | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| Task 1 | enabled | 67 | 68.5496 | 48.12934 | 5.87993 | 1.838 | 6.366 |
| | disabled | 46 | 84.4317 | 60.48084 | 8.91741 | 1.376 | 2.806 |
| Task 2 | enabled | 62 | 37.9040 | 31.26270 | 3.97037 | 2.928 | 11.772 |
| | disabled | 11 | 80.0908 | 58.39325 | 17.60623 | 2.143 | 5.268 |
| Task 3 | enabled | 69 | 51.5377 | 45.81503 | 5.51548 | 2.068 | 5.029 |
| | disabled | 41 | 61.6815 | 50.66503 | 7.91255 | 1.472 | 1.975 |
| Task 4 | enabled | 68 | 33.5680 | 43.77866 | 5.30894 | 2.819 | 3.448 |
| | disabled | 45 | 34.8911 | 36.74714 | 5.47794 | 1.961 | 8.323 |
| Task 5 | enabled | 75 | 31.8356 | 33.47481 | 3.86534 | 5.133 | 32.563 |
| | disabled | 51 | 28.4576 | 25.51808 | 3.57325 | 2.621 | 9.069 |
| Task 6 | enabled | 73 | 40.1574 | 35.90881 | 4.20281 | 2.082 | 4.234 |
| | disabled | 52 | 34.6024 | 22.98864 | 3.18795 | 1.774 | 4.414 |
| Task 7 | enabled | 68 | 46.1865 | 32.41155 | 3.93048 | 1.372 | 2.257 |
| | disabled | 50 | 43.9714 | 32.81269 | 4.64041 | 1.784 | 4.067 |

Table 9. Group Statistics for Task Navigation Times - Non-Normalized

| | Framework | N | Mean | Std. Deviation | Std. Error Mean | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| Task 1 | enabled | 67 | 7.7660 | 2.89193 | 0.35331 | 0.103 | 0.360 |
| | disabled | 46 | 8.6220 | 3.21196 | 0.47358 | 1.436 | 0.015 |
| Task 2 | enabled | 62 | 1.4784 | 0.28775 | 0.03654 | 0.307 | 0.224 |
| | disabled | 11 | 1.8242 | 0.26416 | 0.07965 | 0.561 | 0.905 |
| Task 3 | enabled | 67 | 1.5997 | 0.32630 | 0.03986 | 0.272 | -0.461 |
| | disabled | 41 | 1.6489 | 0.36968 | 0.05773 | -0.189 | -0.586 |
| Task 4 | enabled | 66 | 1.3402 | 0.37691 | 0.04639 | 0.961 | 0.267 |
| | disabled | 45 | 1.3417 | 0.43229 | 0.06444 | -0.039 | -0.251 |
| Task 5 | enabled | 73 | 1.4143 | 0.27074 | 0.03169 | 0.626 | 2.299 |
| | disabled | 51 | 1.3266 | 0.33274 | 0.04659 | 0.138 | -0.203 |
| Task 6 | enabled | 71 | 1.4948 | 0.31854 | 0.03780 | 0.250 | 0.130 |
| | disabled | 52 | 1.4559 | 0.27501 | 0.03814 | -0.119 | 0.071 |
| Task 7 | enabled | 66 | 1.5009 | 0.32282 | 0.03974 | 0.217 | 0.151 |
| | disabled | 50 | 1.4597 | 0.28655 | 0.04052 | -0.138 | -0.235 |

Table 10. Group Statistics for Task Navigation Times – Normalized

7.2.1.2 Task Completion Times t-Test Results

After the results were normalized indicating that we satisfied the third assumption of the independent samples t-Test and can begin our analysis. Below in Table 11 we include the t-Test results for the average navigation times measured for each task. We will now analyze the results of each task individually. We will first determine if we are to assume equal variances as described in assumption 2 above using Levene's test, and choose the appropriate result column in the t-Test results listing in Table 11 below. We will then use the result of the t-Test to determine if the results of the difference in measured navigation times with the framework enabled and disabled are significantly different to within the 95% confidence interval.

| | Levene's Test for Equality of Variances | | t-test for Equality of Means | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 95% Confidence Interval | |
| | F | Sig. | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference | Lower | Upper |
| **Task 1** | | | | | | | | | |
| Eq. var. assumed | 0.523 | 0.471 | -1.477 | 111 | 0.142 | -0.85599 | 0.57937 | -2.00406 | 0.29207 |
| Eq. var. not assumed | | | -1.449 | 90.018 | 0.151 | -0.85599 | 0.59085 | -2.02981 | 0.31783 |
| **Task 2** | | | | | | | | | |
| Eq. var. assumed | 0.300 | 0.585 | -3.714 | 71 | 0.000 | -0.34575 | 0.09310 | -.53138 | -0.16013 |
| Eq. var. not assumed | | | -3.946 | 14.548 | 0.001 | -0.34575 | 0.08763 | -.53304 | -0.15846 |
| **Task 3** | | | | | | | | | |
| Eq. var. assumed | 0.705 | 0.403 | -0.722 | 106 | 0.472 | -0.04918 | 0.06807 | -.18414 | 0.08578 |
| Eq. var. not assumed | | | -0.701 | 76.670 | 0.485 | -0.04918 | 0.07016 | -.18890 | 0.09053 |
| **Task 4** | | | | | | | | | |
| Eq. var. assumed | 0.751 | 0.388 | -0.019 | 109 | 0.985 | -0.00148 | 0.07737 | -0.15482 | 0.15185 |
| Eq. var. not assumed | | | -0.019 | 85.824 | 0.985 | -0.00148 | 0.07941 | -0.15934 | 0.15638 |
| **Task 5** | | | | | | | | | |
| Eq. var. assumed | 3.253 | 0.074 | 1.615 | 122 | 0.109 | 0.08773 | 0.05433 | -0.01982 | 0.19529 |
| Eq. var. not assumed | | | 1.557 | 93.116 | 0.123 | 0.08773 | 0.05635 | -0.02416 | 0.19963 |
| **Task 6** | | | | | | | | | |
| Eq. var. assumed | 1.264 | 0.263 | 0.709 | 121 | 0.480 | 0.03894 | 0.05493 | -0.06981 | 0.14770 |
| Eq. var. not assumed | | | 0.725 | 117.683 | 0.470 | 0.03894 | 0.05370 | -0.06740 | 0.1452 |
| **Task 7** | | | | | | | | | |
| Eq. var. assumed | 0.406 | 0.525 | 0.713 | 114 | 0.477 | 0.04116 | 0.05770 | -0.07314 | 0.1554 |
| Eq. var. not assumed | | | 0.725 | 111.100 | 0.470 | 0.04116 | 0.05676 | -0.07130 | 0.1536 |

Table 11. Task Completion Times t-Test Results

Task 1. Library: Hours of Operation

Task 1 asked users to navigate to the library hours of operation page. The average time it took users to complete this task was smaller when the framework was enabled. On

average it took users roughly 16 seconds less to navigate to the destination page with our framework enabled. It can be observed that for Task 1, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average times measured for the two groups is not considered significant.

Task 2. Library: Printing and Copying

Task 2 asked users to navigate to the library printing and copying page. The average time it took users to complete this task was much smaller when the framework was enabled. On average it took users roughly 42 seconds less to navigate to the destination page with our framework enabled. It can be observed that for Task 2, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two falls within the 95% confidence interval ($p < 0.05$). Therefore, the difference between the average times measured for the two groups is considered significantly different in favor of the modified site with our framework enabled.

Task 3. HR: Benefits

Task 3 asked users to navigate to the human resources benefits page. The average time it took users to complete this task was smaller when the framework was enabled. On average it took users roughly 10 seconds less to navigate to the destination page with our framework enabled. It can be observed that for Task 3, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average times measured for the two groups is not considered significant.

Task 4. HR: Employment

Task 4 asked users to navigate to the human resources employment page. The average time it took users to complete this task was slightly smaller when the framework was enabled. On average it took users roughly 1 second less to navigate to the destination page with our framework enabled. It can be observed that for Task 4, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average times measured for the two groups is not considered significant.

Task 5. Admissions: Deadlines

Task 5 asked users to navigate to the admission deadlines page. The average time it took users to complete this task was slightly larger when the framework was enabled. On average it took users roughly 3 seconds longer to navigate to the destination page with our framework enabled. It can be observed that for Task 5, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average times measured for the two groups is not considered significant.

Task 6. Graduate School: Programs

Task 6 asked users to navigate to the graduate school programs page. The average time it took users to complete this task was slightly larger when the framework was enabled. On average it took users roughly 6 seconds longer to navigate to the destination page with our framework enabled. It can be observed that for Task 6, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average times measured for the two groups is not considered significant.

Task 7. Tuition: Fees

Task 7 asked users to navigate to the controller's office tuition and fees page. The average time it took users to complete this task was slightly larger when the framework was enabled. On average it took users roughly 2 seconds longer to navigate to the destination page with our framework enabled. It can be observed that for Task 7, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average times measured for the two groups is not considered significant.

7.2.1.3. Task Completion Times Result Summary

In summary, we found that of the seven average task completion times, only the results of Task 2 (library printing and copying page) was significantly different enough to fall into the 95% confidence interval. The average navigation time for Task 2 showed a significantly lower completion time when the framework was enabled. As you can also see in the group statistics table (Table 9), the number of results in the framework enabled group were much higher than the number of results in the disabled group. This is due to the number of times this task was skipped. If the task is skipped by the user, we could not include those task times in our analysis. One likely reason for the lack of convincing evidence for our framework using the navigation time measurement is that after a certain amount of time spent on a task a user is much more likely to skip a task. If we did not

allow users to skip tasks we may have seen more convincing results in average navigation times. Because we did give the user an option to skip however, we had far more skips on the unmodified version of the site, possibly keeping navigation times similar between the two versions of the site. We will address this discrepancy further in section 7.3, when we discuss the number of times each task was skipped. In addition to the problem of task skips, the small sample size may have kept this metric from being as convincing as we would have liked. An A/B test like this would be best served to a much larger set of users, so we could be more confident that our framework significantly improves user experience. A future test of this framework on a live production site would be ideal, but was not feasible for us at this time. We were able to disprove the null hypothesis for Task 2 with a significant level of confidence, indicating for that specific task, our framework significantly reduced the time it took users to navigate to the task destination page.

### 7.2.2 Task Navigation Steps

For each task we also measured the number of navigation steps it took users to make it from the homepage to the destination page. Our hypothesis for this metric was that our modified version of the site will allow users to complete the tasks in fewer navigation steps than the original version of the site. The null hypothesis we would like to disprove is that it took users the same number of navigation steps no matter which version of the site they used. In Figure 21 we show the average number of navigation steps it took users to complete each task separated by whether or not they were given the original UNF site or the modified version of the site using our framework.

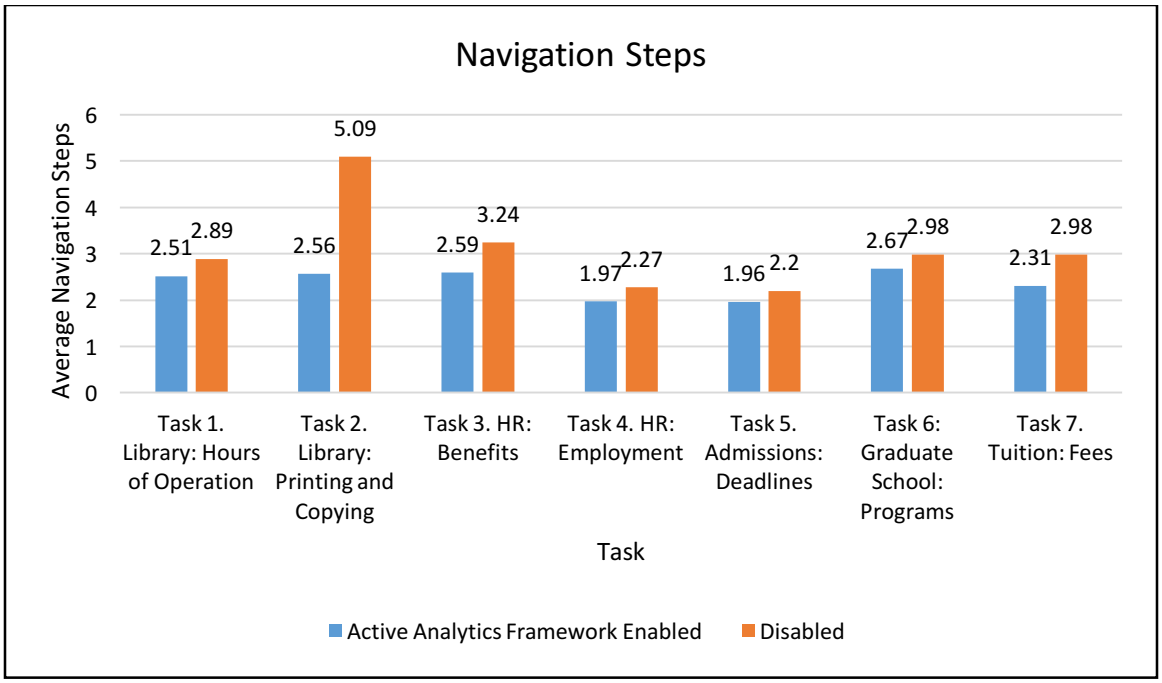


Figure 21. Average Task Navigation Steps

The graph above demonstrates that the average navigation steps it took a participant to complete the task was smaller when our framework was enabled for all seven tasks. For each of these tasks we performed an independent samples t-Test to determine if the difference in the number of navigation steps was statistically significant enough to disprove the null hypothesis that navigating the two versions of the site result in the same average number of navigation steps.

### 7.2.2.1 Task Navigation Steps t-Test Results

The measured results for this portion of the study did not strictly follow a normal distribution as required by assumption 3 above. The results were skewed toward smaller numbers of navigation steps, because of this the distribution was weighted heavier

towards fewer (2-3) navigation steps. Because the data was restricted to a relatively small set of discreet values, standard transformations (like log and square root transformations) don't help to normalize the data. The t-Test results in this case, however, can still be useful, as assumption 3 states above: with reasonably large sample sizes (N > 30), the data does not need to strictly adhere to a normal distribution. Table 12 contains the group statistics data on the measured navigation steps for each task. Table 13 below contains the detailed results of the t-Tests performed on each task. We will now analyze the t-Test results of each task in detail.

| | Framework | N | Mean | Std. Deviation | Std. Error Mean |
|---|---|---|---|---|---|
| Task 1 | enabled | 67 | 2.5075 | 1.29537 | .15825 |
| | disabled | 46 | 2.8913 | 1.28631 | .18966 |
| Task 2 | enabled | 63 | 2.5556 | 1.36521 | .17200 |
| | disabled | 11 | 5.0909 | 2.98176 | .89904 |
| Task 3 | enabled | 69 | 2.5942 | 2.35970 | .28407 |
| | disabled | 41 | 3.2439 | 2.09500 | .32718 |
| Task 4 | enabled | 68 | 1.9706 | 1.85255 | .22466 |
| | disabled | 45 | 2.2667 | 1.68415 | .25106 |
| Task 5 | enabled | 75 | 1.9600 | 1.21299 | .14006 |
| | disabled | 51 | 2.1961 | 1.45629 | .20392 |
| Task 6 | enabled | 73 | 2.6712 | 1.49122 | .17453 |
| | disabled | 52 | 2.9808 | 1.26010 | .17474 |
| Task 7 | enabled | 68 | 2.3088 | 1.62313 | .19683 |
| | disabled | 50 | 2.9800 | 1.33233 | .18842 |

Table 12. Group Statistics for Task Navigation Steps

| | Levene's Test for Equality of Variances | | t-test for Equality of Means | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | 95% Confidence Interval | |
| | F | Sig. | t | df | Sig. (2-tailed) | Mean Difference | Std. Error Difference | Lower | Upper |
| **Task 1** | | | | | | | | | |
| Eq. var. assumed | 0.091 | 0.763 | -1.552 | 111 | 0.124 | -0.38384 | 0.24733 | -0.87395 | 0.10627 |
| Eq. var. not assumed | | | -1.554 | 97.314 | 0.123 | -0.38384 | 0.24701 | -0.87407 | 0.10638 |
| **Task 2** | | | | | | | | | |
| Eq. var. assumed | 10.114 | 0.002 | -4.604 | 72 | 0.000 | -2.53535 | 0.55067 | -3.63310 | -1.43761 |
| Eq. var. not assumed | | | -2.770 | 10.743 | 0.019 | -2.53535 | 0.91534 | -4.55590 | -0.51481 |
| **Task 3** | | | | | | | | | |
| Eq. var. assumed | 0.272 | 0.603 | -1.454 | 108 | 0.149 | -0.64970 | 0.44668 | -1.53511 | 0.23571 |
| Eq. var. not assumed | | | -1.499 | 92.213 | 0.137 | -0.64970 | 0.43330 | -1.51024 | 0.21084 |
| **Task 4** | | | | | | | | | |
| Eq. var. assumed | 1.184 | 0.279 | -0.862 | 111 | 0.391 | -0.29608 | 0.34354 | -0.97682 | 0.38466 |
| Eq. var. not assumed | | | -0.879 | 100.400 | 0.382 | -0.29608 | 0.33690 | -0.96444 | 0.37229 |
| **Task 5** | | | | | | | | | |
| Eq. var. assumed | 7.966 | 0.006 | -0.988 | 124 | 0.325 | -0.23608 | 0.23894 | -0.70902 | 0.23686 |
| Eq. var. not assumed | | | -0.954 | 94.147 | 0.342 | -0.23608 | 0.24739 | -0.72727 | 0.25511 |
| **Task 6** | | | | | | | | | |
| Eq. var. assumed | 1.398 | 0.239 | -1.218 | 123 | 0.225 | -0.30954 | 0.25406 | -0.81242 | 0.19335 |
| Eq. var. not assumed | | | -1.253 | 119.365 | 0.213 | -0.30954 | 0.24698 | -0.79856 | 0.17949 |
| **Task 7** | | | | | | | | | |
| Eq. var. assumed | 2.437 | 0.121 | -2.390 | 116 | 0.018 | -0.67118 | 0.28078 | -1.22729 | -0.1150 |
| Eq. var. not assumed | | | -2.463 | 114.540 | 0.015 | -0.67118 | 0.27248 | -1.21093 | -0.1314 |

Table 13. Task Navigation Steps t-Test Results

Task 1. Library: Hours of Operation

Task 1 asked users to navigate to the library hours of operation page. The average number of steps it took users to complete this task was smaller when the framework was enabled. On average it took users 0.38 fewer steps to navigate to the destination page with our framework enabled. It can be observed that for Task 1, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average number of steps recorded for the two groups is not considered significant.

Task 2. Library: Printing and Copying

Task 2 asked users to navigate to the library printing and copying information page. The average number of steps it took users to complete this task was smaller when the framework was enabled. On average it took users 2.53 fewer steps to navigate to the destination page with our framework enabled. It can be observed that for Task 2, Levene's test is significant ($p < 0.05$), therefore we used data from the "Equal variances not assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples falls within the 95% confidence interval ($p < 0.05$). Therefore, the difference between the average number of steps to complete the task measured for the two groups is considered significantly different in favor of the modified site with our framework enabled.

Task 3. HR: Benefits

Task 3 asked users to navigate to the human resources benefits page. The average number of steps it took users to complete this task was smaller when the framework was enabled. On average it took users 0.65 fewer steps to navigate to the destination page with our framework enabled. It can be observed that for Task 3, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average number of steps recorded for the two groups is not considered significant.

Task 4. HR: Employment

Task 4 asked users to navigate to the human resources employment page. The average number of steps it took users to complete this task was smaller when the framework was enabled. On average it took users 0.30 fewer steps to navigate to the destination page with our framework enabled. It can be observed that for Task 4, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average number of steps recorded for the two groups is not considered significant.

Task 5. Admissions: Deadlines

Task 5 asked users to navigate to the admissions deadlines page. The average number of steps it took users to complete this task was smaller when the framework was enabled. On average it took users 0.24 fewer steps to navigate to the destination page with our framework enabled. It can be observed that for Task 5, Levene's test is significant ($p < 0.05$), therefore we used data from the "Equal variances not assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average number of steps recorded for the two groups is not considered significant.

Task 6. Graduate School: Programs

Task 6 asked users to navigate to the graduate school programs of study page. The average number of steps it took users to complete this task was smaller when the framework was enabled. On average it took users 0.31 fewer steps to navigate to the destination page with our framework enabled. It can be observed that for Task 6, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the average number of steps recorded for the two groups is not considered significant.

Task 7. Tuition: Fees

Task 7 asked users to navigate to the controller's tuition and fees page. The average number of steps it took users to complete this task was smaller when the framework was enabled. On average it took users 0.67 fewer steps to navigate to the destination page with our framework enabled. It can be observed that for Task 7, Levene's test is not significant ($p > 0.05$), therefore we used data from the "Equal variances assumed" row for analysis. It can also be observed in the "Sig" column of the t-Test results, the difference between the two samples falls within the 95% confidence interval ($p < 0.05$). Therefore, the difference between the average number of steps to complete the task measured for the two groups is considered significantly different in favor of the modified site with our framework enabled.

7.2.2.2 Task Navigation Steps Result Summary

The second measured metric, navigation steps taken to reach a destination, came out more clearly in favor of our modified version of the site. For every single task, it took on average fewer navigation steps to reach the destination page of that task. For only two of the tasks, however, was the difference in average times significant enough to fall within the 95% confidence interval. For these two tasks (2 and 7) we were able to disprove the null hypothesis: that navigating the two versions of the site result in the same average number of navigation steps. We were able to conclude that our framework significantly reduced the number of navigation steps it took participants to complete tasks 2 and 7. Just as in the previous metric of navigation times, navigation steps measured were also

affected by the significantly larger number of task skips on the original version of the site. Every time a task was skipped, we had to discard the measured navigation steps. It seems likely that after a user reached a certain number of navigation steps the user became frustrated or lost and skipped the task. Just as with the navigation time metric, the small sample size and ability to skip tasks may have skewed our results. A much better test of our framework would have been on a live production site with real user traffic. Despite these confounding factors, it seems that the modified version of the site using our framework presented users with the link to the page they were trying to reach sooner than the unmodified version of the site. We believe this to be a convincing finding in favor of our framework.

## 7.3 Task Skips

The final recorded measurement we analyzed is the total number of skips recorded for each version of the site. With each task assigned to the participants, they were given the option to skip the task completely if they felt lost or frustrated. Because of this ability to skip tasks, we had differing numbers of responses for the timed tasks and navigation steps above. Our hypothesis for this metric was that our modified version of the site will cause the participants less frustration and therefore skip fewer tasks than the original version of the site. The null hypothesis we would like to disprove is that participants were as likely to skip a task no matter which version of the site they used. As evidenced by the charts below, users who were given the unmodified version of the site skipped tasks more often. Because of this, we have more data about tasks with the framework enabled; simply because users were able to complete the tasks more often. This proved to

be the most convincing metric in favor of the modified version of the website. The results show that there were far more task skips on the unmodified version of the site as opposed to our enhanced version. Figure 22 shows the percentage of users that skipped each task. For example; with the framework disabled, 83.6% of the users who attempted Task 2 skipped the task before they were able to complete it.

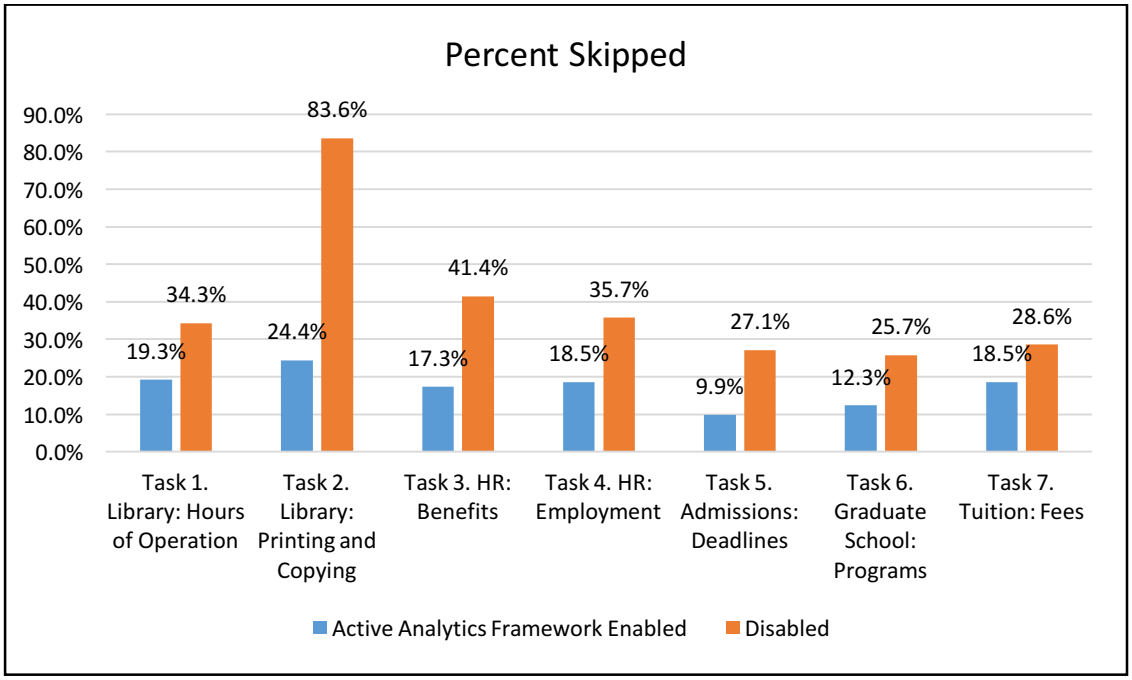


Figure 22. Percent of Users Who Skipped Each Task

Clearly, there were far more task skips on the unmodified version of the site. To test these results and determine if the difference between the two versions of the site were significantly different we performed a chi-square test on the task skip data for each task. The chi-square test is a test of the statistical significance of a relation between two ordinal variables (Salkind, 2010). In our study, we compare the two cases of our independent variable, whether or not our framework was enabled, and the dependent

variable of whether or not the user completed the task before skipping. We would like to determine if the differences in task skip frequency shown above in Figure 22 can be considered statistically significant. The chi-square test requires the following six assumptions about the data must be satisfied in order to perform the analysis (Salkind, 2010):

- Assumption 1: Chi-square is most appropriate for normal ordinal variables. Our variable is simply whether or not the user skipped the task they were presented. This is an ordinal variable with two possible values.
- Assumption 2: The sample must be randomly drawn from the population. Our sample was drawn from a sampling of UNF students and they were randomly assigned to either test group.
- Assumption 3: The data must be reported in raw frequencies. We analyzed our data simply as a record with a value of skipped or not skipped, not in the form of percentages.
- Assumption 4: Measured variables must be independent of each other. As we stated above the two groups were completely independent, and no task requires the completion of a previous task.
- Assumption 5: Values and categories on independent and dependent variables must be mutually exclusive and exhaustive. In our study, each user was presented with a given task only once, and we simply recorded the binary value of whether or not they skipped that task.

- Assumption 6: Observed frequencies cannot be too small. We analyzed our complete data set excluding no measurements, and our sample size was sufficiently large with over 100 observations for each task.

### 7.3.1 Chi-Square Test Results for Task Skips

Table 14 shows the total number of skips versus completions for each task with the framework both enabled and disabled. It also shows the resulting value of the Chi-Square test for statistical significance along with the associated probability of error in the "Sig" column.

| | Skipped | N | | Total | Pearson Chi-Square | |
|---|---|---|---|---|---|---|
| | | Enabled | Disabled | | Value | Sig. |
| Task 1 | Yes | 16 | 24 | 40 | 4.430 | 0.035 |
| | No | 67 | 46 | 113 | | |
| Task 2 | Yes | 20 | 56 | 76 | 51.697 | 0.000 |
| | No | 62 | 11 | 73 | | |
| Task 3 | Yes | 14 | 29 | 43 | 10.748 | 0.001 |
| | No | 67 | 41 | 108 | | |
| Task 4 | Yes | 15 | 25 | 40 | 5.702 | 0.017 |
| | No | 66 | 45 | 111 | | |
| Task 5 | Yes | 8 | 18 | 26 | 6.608 | 0.010 |
| | No | 73 | 52 | 125 | | |
| Task 6 | Yes | 10 | 71 | 28 | 4.443 | 0.035 |
| | No | 18 | 52 | 123 | | |
| Task 7 | Yes | 15 | 20 | 35 | 2.131 | 0.144 |
| | No | 66 | 50 | 116 | | |

Table 14. Chi-Square Test for Task Skips

Task 1. Library: Hours of Operation

With our framework enabled, 19.3% of the participants who attempted Task 1 were unable to complete it, as compared to 34.3% of participants when our framework was disabled. It can be observed that for Task 1 in the "Sig" column of the chi-square results, the difference between the two samples falls within the 95% confidence interval ($p < 0.05$). Therefore, the difference between the percentages of users who were able to complete the task without skipping it is significantly higher with our framework enabled.

Task 2. Library: Printing and Copying

With our framework enabled, 24.4% of the participants who attempted Task 2 were unable to complete it, as compared to 83.6% of participants when our framework was disabled. It can be observed that for Task 2 in the "Sig" column of the chi-square results, the difference between the two samples falls within the 95% confidence interval ($p < 0.05$). Therefore, the difference between the percentages of users who were able to complete the task without skipping it is significantly higher with our framework enabled.

Task 3. HR: Benefits

With our framework enabled, 17.3% of the participants who attempted Task 3 were unable to complete it, as compared to 41.4% of participants when our framework was disabled. It can be observed that for Task 3 in the "Sig" column of the chi-square results, the difference between the two samples falls within the 95% confidence interval ($p <$

0.05). Therefore, the difference between the percentages of users who were able to complete the task without skipping it is significantly higher with our framework enabled.

Task 4. HR: Employment

With our framework enabled, 18.5% of the participants who attempted Task 4 were unable to complete it, as compared to 35.7% of participants when our framework was disabled. It can be observed that for Task 4 in the "Sig" column of the chi-square results, the difference between the two samples falls within the 95% confidence interval ($p < 0.05$). Therefore, the difference between the percentages of users who were able to complete the task without skipping it is significantly higher with our framework enabled.

Task 5. Admissions: Deadlines

With our framework enabled, 9.9% of the participants who attempted Task 5 were unable to complete it, as compared to 27.1% of participants when our framework was disabled. It can be observed that for Task 5 in the "Sig" column of the chi-square results, the difference between the two samples falls within the 95% confidence interval ($p < 0.05$). Therefore, the difference between the percentages of users who were able to complete the task without skipping it is significantly higher with our framework enabled.

Task 6. Graduate School: Programs

With our framework enabled, 12.3% of the participants who attempted Task 6 were unable to complete it, as compared to 25.7% of participants when our framework was

disabled. It can be observed that for Task 6 in the "Sig" column of the chi-square results, the difference between the two samples falls within the 95% confidence interval ($p < 0.05$). Therefore, the difference between the percentages of users who were able to complete the task without skipping it is significantly higher with our framework enabled.

Task 7. Tuition: Fees

With our framework enabled, 18.5% of the participants who attempted Task 7 were unable to complete it, as compared to 28.6% of participants when our framework was disabled. It can be observed that for Task 7 in the "Sig" column of the chi-square results, the difference between the two samples does not fall within the 95% confidence interval ($p > 0.05$). Therefore, the difference between the percentages of users who were able to complete the task without skipping it is not significantly higher than with our framework enabled.

### 7.3.2 Task Skips Result Summary

The final and most convincing measured metric of our study is the number of times users skipped a task after feeling lost or frustrated. We found that users using our modified version of the site skipped far fewer tasks than users using the unmodified version of the site. For six out of our seven tasks, we were able to disprove the null hypothesis and conclude that our framework significantly reduced the number of tasks skipped by study participants. Only the number of skips recorded in Task 7 was not significantly different between the two versions of the site. We believe this is some of the strongest evidence in

favor of our framework. The completion rate on each of the tasks shows that our framework directed users to their desired pages before users felt frustrated and skipped the task completely.

## 7.4 Effect Size

Effect size is a measure of how practically significant the results of a research study are. Statistical significance ensures that a result is not due to random chance. In order to determine the level of difference between two results, and the practical significance of those results, we used the effect size calculation. To calculate the effect size of our study we use the Cohen's d-statistic (Salkind, 2010). According to this statistical calculation, the resulting value will determine whether the outcome is practically smaller or larger than typical effect. Cohen's test categorizes results into three levels of effect size. A smaller than typical effect size ($d < 0.5$), a typical effect size ($0.5 \leq d < 0.8$), and a larger than typical effect size ($d \geq 0.8$). The tables below show the results of the Cohen's d-Effect Size calculation for each of our statistically significant results

### 7.3.1 Task Completion Times

Table 15 shows that the effect size for Task 2 navigation times fell into the larger than typical effect size range ($d \geq 0.8$). From this we can conclude that our framework resulted in a large practical improvement in how long it took users to navigate the site specifically for Task 2 of the survey.

| Task 2 | N | Mean | Std. Deviation | Effect Size (d) |
|---|---|---|---|---|
| Enabled | 62 | 37.9040 | 31.26270 | 0.901 |
| Disabled | 11 | 80.0908 | 58.39325 | |

Table 15. Effect Size for Task 2 Completion Time

### 7.3.2 Task Navigation Steps

For the navigation steps metric, we calculated the effect size of both of our statistically significant task results. Table 16 shows that for Task 2, the effect size was well within the category of larger than typical effect size ($d \geq 0.8$). Also, Table 17 shows that for Task 7, the effect size was just below the typical effect size and technically fell with the smaller than typical effect size category ($d < 0.5$). We can conclude from these results that our framework provided large practical improvement in the number of navigation steps it took user to complete Task 2, and provided a smaller practical improvement for Task 7.

| Task 2 | N | Mean | Std. Deviation | Effect Size (d) |
|---|---|---|---|---|
| Enabled | 62 | 2.5556 | 31.26270 | 1.093 |
| Disabled | 11 | 5.0909 | 58.39325 | |

Table 16. Effect Size for Task 2 Navigation Steps

| Task 7 | N | Mean | Std. Deviation | Effect Size (d) |
|---|---|---|---|---|
| Enabled | 68 | 2.3088 | 1.62313 | 0.452 |
| Disabled | 50 | 2.9800 | 1.33233 | |

Table 17. Effect Size for Task 7 Navigation Steps

### 7.3.3 Task Skips

For task skips, we used a different statistic to measure effect size. To measure the effect size of our chi-square test we used the Cramer's V-statistic (Salkind, 2010). Like the Cohen's d-test above, the Cramer's V-test splits effect size into three different categories: A smaller than typical effect size ($V < 0.30$), a typical effect size ($0.3 \leq V < 0.5$), and a larger than typical effect size ($V > 0.80$). Table 18 shows the Cramer's V-test for each of our statistically significant tasks. The Cramer's V-test calculations show that our framework provided a large practical improvement over the unmodified version of the site for Task 2. Also, for Tasks 1, 3, 4, 5, and 6 our framework provided a small practical improvement over the unmodified version of the site.

| Task | Chi-Square Value | Cramer's V | Effect Size |
|---|---|---|---|
| **1** | 4.430 | 0.170 | Small |
| **2** | 51.697 | 0.589 | Large |
| **3** | 10.748 | 0.267 | Small |
| **4** | 5.702 | 0.194 | Small |
| **5** | 6.608 | 0.209 | Small |
| **6** | 4.443 | 0.172 | Small |

Table 18. Effect Size for Task Skips

## 7.5 Survey Responses

In addition to recording user actions as they navigated around the site, we also asked the users to complete a survey about their experience. We asked study participants some basic demographic information about themselves and asked them to rate their experience

navigating the site. In the following section we present the summarized results of the survey.

### 7.5.1 Experience Ratings

At the end of the study we asked users to rank their experience by answering a series of questions with a 1-5 rating based on how much they agreed with the presented statement (See Appendix B.) These questions were the same, no matter which version of the site they received. The results of this ranking where not very conclusive, as their average responses were similar and did not vary much between the two versions of the site. Figure 23 shows a chart of the survey responses. Responses for each of the questions were very similar and did not vary much between the two versions of the site. We can't draw any meaningful conclusions from user responses considering our small sample size. Future studies should make the questions less open ended to hopefully draw out more meaningful responses from participants.

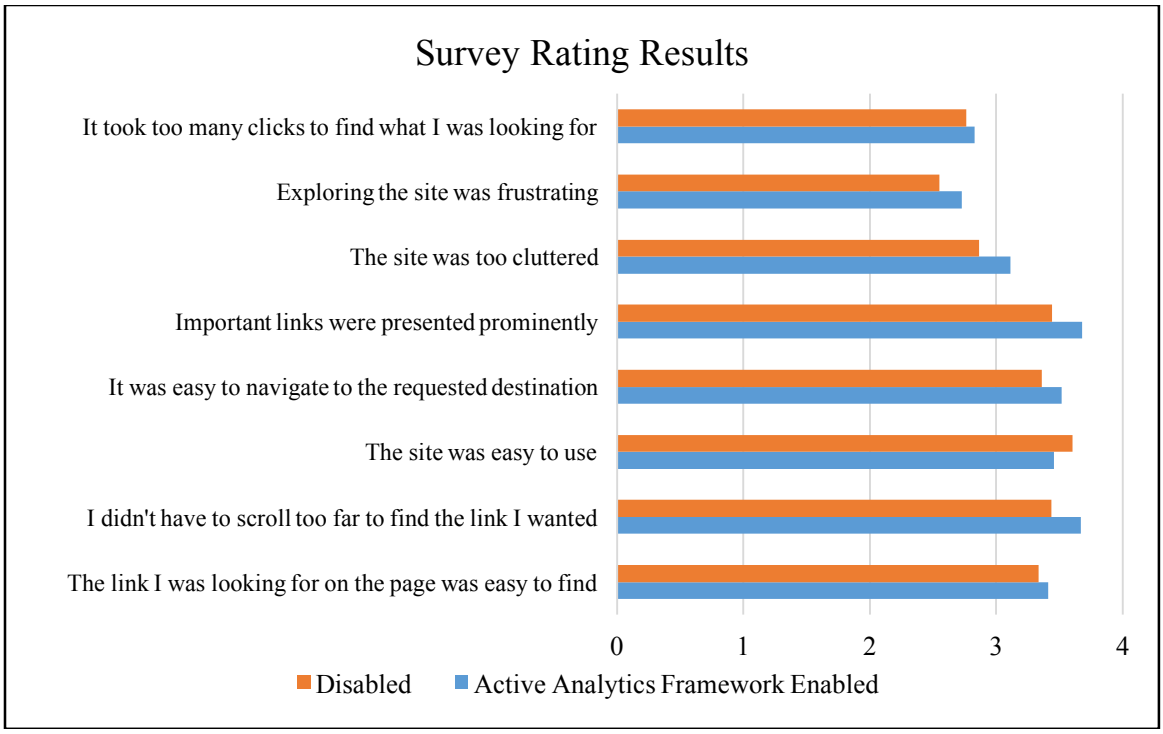


Figure 23. Survey Rating Responses

# Chapter 8

# FUTURE IMPROVEMENTS

We believe the ideas presented in this research will have the potential to solve real world problems. We were fairly ambitious in the breadth of our research, and because of this we couldn't possibly have given due attention to all potential applications for our framework. We believe that with some additional work this framework could be expanded to a more complete and useful system. In this chapter we will present possible directions to extend this research.

## 8.1 Scalability and Expandability

The working system described in chapters 4 and 5 was simply a proof of concept to test our theories and serve as a jumping off point for a production system. That being said, we made a concerted effort to make the system performant and scalable. The system was built on a platform that can be scaled horizontally, adding additional server instances that could handle large amounts of load. We did not test performance or attempt to scale for larger sites as part of our research due to time and financial constraints. We do believe however, that the framework we built has the potential for scaling and supporting heavy loads.

In addition to scaling for a single site, we would also like to make the entire solution more expandable to apply to multiple sites by connecting to additional analytics accounts.

For our purposes, we designed a point solution applying specifically to the UNF website and the UNF Google Analytics account. We believe the design of our framework however, is generic enough to work with additional sites and potentially different analytics providers. With some additional work on improving how generic our API is implemented, we could potentially serve data on multiple sites and from multiple analytics providers at once. With some additional time and resources, we believe we could have made our framework more generic and expandable to other sites.

## 8.2 Features and Improvements

In addition to simply improving the existing feature set through performance and scalability enhancements, we believe there is potential for adding more features. Firstly, we believe the improved search suggestions we added to the site have great potential if given additional effort. Some promising initial testing shows that these search suggestions, which jump the user directly to the most popular result for common queries, can improve the speed at which users find the information they are looking for. Search suggestions are a very complicated field of study, and we couldn't give that aspect of the framework the time it deserved.

Another piece of the implementation we didn't have time to expand upon to the degree we would have liked was the time variant nature of our database design. We designed our framework to query and store data from different snapshots of time. We even created a tool that allowed us to view the site at different snapshots of time in order to see how site improvements based on analytics data would change over time. There is some

exciting potential for predictive analytics given this feature. For example, we could potentially look back to the previous year's analytics to get a sense of what might become popular in the near future and serve that content up more prominently just as it is needed. We didn't have time to implement features like this, but the framework is in place and the data is there. With some additional work in this area, we believe the site could be made even more responsive to user needs. For example, imagine that final exams are coming up for the university, looking back at trends from the previous year, the system could determine that at this time last year there was a spike in traffic to the exam schedule page. We could detect this trend and present links to that page more prominently even before we observe that trend emerge again this year.

## 8.3 Site Implementation

As discussed in chapter 5, in order to test our framework we applied some of our ideas to the existing UNF website. We essentially retrofitted an existing site to incorporate our analytics based site improvements. Because of this, our abilities to update the site were relatively limited, and we couldn't design a site from the ground up to adapt to changing usage trends. Ideally, when designing a site you would take this analytics framework into consideration from the start, designing parts of the site to specifically take advantage of analytics data. A more interesting exercise would be designing a site from the start using our framework.

Another side effect of implementing our framework on a live working site without effecting the production site directly, was that we had to proxy the site through our own

server, causing some performance problems. Because we had to proxy the site through our own server in order to inject our own scripts and markup, the site did not perform as quickly as it would under normal circumstances. A true test of our framework would be to implement it directly on a website without the need to proxy through another server. Obviously, we couldn't do this on the live UNF site, but an actual A/B test on the live site could offer some very valuable insights into our ideas.

Overall, we were happy with what we were able to implement, and we were fortunate enough to be able to use real production data from the UNF analytics account. We believe this framework has very promising real world application potential, and we hope to continue with our research in the future.

# Chapter 9

# CONCLUSION

The goal of this research was to investigate the possibility of using web analytics data to reduce the time and level of effort required it takes to find information on a website. We created a working system to test our framework and solicited the help of UNF students to navigate the site and measure the effectiveness of the framework. This system used analytics data already being gathered on the website to adapt pages in real time without the need for any custom re-working of any backend code. What we found in our investigation was that in many cases our modified version of the UNF website, using live analytics data to modify the user interface in real time, performed significantly better than the unmodified current version of the site. We believe that our framework can offer benefits in terms of usability to websites that already have a great wealth of analytics data, but don't necessarily have the resources to build custom dynamic pages from scratch.

We believe there is merit to the idea of an adapting website that changes automatically based on the analytics data that is already being gathered. We have shown that, by using our framework and doing some basic implementation on a site, we can significantly improve a user's navigation experience. The intent of this framework is to make it easy for developers to tap into the wealth of analytics data that many sites have already been gathering for years. By constantly sampling this data and using it to direct users to

popular content as that list of popular content changes, a site can remain fresh and useful with little to no intervention from site designers and developers.

The system we designed and tested here is a proof of concept that was able to back up our hypothesis on dynamically adapting websites. We believe there is great promise in this concept and think there is a potential for it to be implemented and tested on live websites. We have shown that a site can be improved and adapted in real-time using the data already being gathered by web analytics tools. Based on the successes of our proof of concept, we suggest that further research and development be done to extend these concepts. Adaptive websites no longer have to be custom solutions requiring large development teams. Analytical tools already in place on many websites, with their large wealth of data, can be put to work to build modern adaptive websites quickly and with a limited development effort.

# REFERENCES


Beasley, M. (2013). *Practical web analytics for user experience : How analytics can help you understand your users*. Amsterdam: Morgan Kaufmann, an imprint of Elsevier.

Bevan, N. (2005). Guidelines and standards for web usability. *Proceedings of HCI International 2005, Lawrence Erlbaum,* Las Vegas, Nevada.

Booth, D., et al. (2004). *Web services architecture - W3C working group note.* Retrieved 03/20, 2016, from http://www.w3.org/TR/ws-arch/

Bos, B. (2015). *Cascading style sheets.* Retrieved 03/20, 2016, from http://www.w3.org/Style/CSS/Overview.en.html

Büchner, A. G., & Mulvenna, M. D. (1998). Discovering internet marketing intelligence through online analytical web usage mining. *SIGMOD Rec., 27*(4), 54-61.

Fasel, D., & Zumstein, D. (2009). A fuzzy data warehouse approach for web analytics. In M. Lytras, et al. (Eds.), (pp. 276-285) Springer Berlin Heidelberg.

Fielding, R. T. (2000). Architectural styles and the design of network-based software architectures. University of California, Irvine).

Gonçalves, B., & Ramasco, J. J. (2008). Human dynamics revealed through web analytics. *Physical Review E, 78*(2), 026123.

Google Analytics. (2014). *Reporting developer guides.* Retrieved 03/20, 2016, from https://developers.google.com/analytics/devguides/reporting

Herring, M., & Prichard, J. (2012). The effect of web usability on user's web experience. *Proceedings for the Northeast Region Decision Sciences Institute (NEDSI),* , 207-215.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Q., 28*(1), 75-105.

Inmon, W. H., Strauss, D., & Neushloss, G. (2010). *DW 2.0: The architecture for the next generation of data warehousing: The architecture for the next generation of data warehousing* Elsevier Science.

Kohavi, R., Longbotham, R., Sommerfield, D., & Henne, R. (2009). Controlled experiments on the web: Survey and practical guide. *Data Mining and Knowledge Discovery, 18*(1), 140-181.

Kumari, G. V., Praneeth, P. 2., & Raju, V. P. (2014). An application of web usage mining framework for mining dynamic web sites. *International Journal of Advanced Research in Computer Science, 5*(2), 91-93.

Lai, L. T., Xu, Y., & Tan, F. B. (2009). Attributes of web site usability: A study of web users with the repertory grid technique. *International Journal of Electronic Commerce, 13*(4), 97-126.

Mican, D., & Sitar-Taut, D. (2009). Preprocessing and Content/Navigational pages identification as premises for an extended web usage mining model development. *Informatica Economica, 13*(4), 168-179.

Mobasher, B., Cooley, R., & Srivastava, J. (2000). Automatic personalization based on web usage mining. *Commun.ACM, 43*(8), 142-151.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems, 24*(3), 45-77.

Phippen, A., Sheppard, L., & Furnell, S. (2004). A practical evaluation of web analytics. *Internet Research, 14*(4), 284-293.

Pokorny, J. (2013). NoSQL databases: A step to database scalability in web environment. *International Journal of Web Information Systems, 9*(1), 69-82.

Prom, C. (2011). Using web analytics to improve online access to archival resources. *American Archivist, 74*(1), 158-184.

Salkind, N. J. (2010). t test, independent samples. *Encyclopedia of research design* (pp. 1551). Thousand Oaks, Calif: SAGE Publications, Inc.

Santos, R., Macdonald, C., & Ounis, I. (2013). Learning to rank query suggestions for adhoc and diversity search. *Information Retrieval Journal, 16*(4), 429.

U.S. Dept. of Health and Human Services. (2006). *The research-based web design & usability guidelines.* Retrieved 03/20, 2016, from http://guidelines.usability.gov/

UNF. (2014). *University of north florida.* Retrieved 03/20, 2016, from http://www.unf.edu

Vaishnavi, V., & Kuechler, B. (2013). *Design science research in information systems.* Retrieved 03/20, 2016, from http://desrist.org/desrist/

W3C DOM Interest Group. (2005). *Document object model (DOM)*. Retrieved 03/20, 2016, from http://www.w3.org/DOM/

W3Techs. (2011). *Usage statistics and market share of google analytics for websites.* Retrieved 03/20, 2016, from http://w3techs.com/technologies/details/ta-googleanalytics/all/all

Webster, J., & Ahuja, J. S. (2006). Enhancing the design of web navigation systems: The influence of user disorientation on engagement and performance. *MIS Quarterly, 30*(3), 661-678.

Weischedel, B., & Huizingh, E. K. R. E. (2006). *Website optimization with web metrics: A case study*. Fredericton, New Brunswick, Canada: ACM.

# APPENDIX A

## NAVIGATION TASKS

1. A new student Alex, recently transferred from UCF and he is in your class. He wants to know hours when UNF Library will be open for this semester. Help Alex by navigating to UNF library page that displays its operation hours. Please navigate to the Library "Hours of Operation" page (the page with a full calendar on it.)

2. Alex wants has some questions on printing and copying at library. Help Alex by navigating to UNF library page that displays printing and copying information. Please navigate to the UNF library "Printing and Copying Information" page.

3. Alex is interested in working for UNF and has some questions on benefits offered to UNF employees. Help Alex by navigating to human resource page that displays benefits information. Please navigate to the Human Resources "Benefits" page.

4. Alex is interested in learning about employment opportunities at UNF. Help Alex by navigating to human resource page that displays employment information. Please navigate to the Human Resources "Employment" page. You begin to wonder if Alex has ever seen a computer before.

5. Alex mentions that his cousin Zack is also considering applying for UNF. Alex would like to know information regarding application deadlines. Help Alex by navigating to UNF admissions page that displays deadlines information. Hopefully he won't need help applying too. Please navigate to the UNF Admissions "Deadlines" page.

6. Alex mentions that Zack would be interested in graduate programs. Alex wants to obtain information on available graduate programs at UNF. Help Alex by navigating to graduate school page that displays available graduate programs at UNF. Please navigate to the Graduate School's "Graduate Programs" page.

7. Alex would like to obtain information on tuition and fees for UNF students. Really Alex? Help Alex by navigating to controller page that displays tuition and fees details. Please navigate to the "Tuition" page with the breakdown of tuition and fees for students.

APPENDIX B

USER EXPERIENCE SURVEY

Demographic Questions:

Age

- 18-25
- 26-35
- 36-45
- 46-55
- 56-65
- 65+

How experienced are you in using the internet?

- Very Experienced
- Some Experience
- Limited Experience
- No Experience

Which browser did you use to view the site?

- Internet Explorer
- Google Chrome
- Safari
- Firefox

Is English your primary language?

- Yes
- No

Task Specific Questions:

Have you visited the UNF website before?

- Yes
- No

If yes how often do you visit the UNF website?

- A few times a year
- A few times per month
- Once a week
- Multiple times a week
- Daily
- Multiple times a day

Were you able to complete all the tasks?

- Yes
- No

If not, why were you not able to complete the tasks?

Did you get lost at any point while trying to complete a task?

- Yes
- No

If yes please describe what happened

Were you frustrated at any point when trying to complete a task?

- Yes
- No

If yes please describe what caused the frustration.

User Experience Ratings:

Please rate the following statements from 1 (strongly disagree) to 5 (strongly agree):

| | **Strongly Agree (5)** | **Agree (4)** | **Neither Agree nor Disagree (3)** | **Disagree (2)** | **Strongly Disagree (1)** |
|---|---|---|---|---|---|
| The link I was looking for on the page was easy to find | | | | | |
| I didn't have to scroll too far to find the link I wanted | | | | | |
| The site was easy to use | | | | | |
| It was easy to navigate to the requested destination | | | | | |
| Important links were presented prominently | | | | | |
| The site was too cluttered | | | | | |
| Exploring the site was frustrating | | | | | |
| It took too many clicks to find what I was looking for | | | | | |

# APPENDIX C

# IRB DOCUMENTS

## IRB Approval Letter

UNF
UNIVERSITY of NORTH FLORIDA.

Office of Research and Sponsored Programs
1 UNF Drive
Jacksonville, FL 32224-2665
904-620-2455 FAX 904-620-2457
Equal Opportunity/Equal Access/Affirmative Action Institution

**MEMORANDUM**

UNF IRB Number: 784254-1
Approval Date: 9-02-2015
Expiration Date: 9-02-2016
Processed on behalf of UNF's IRB KLC

**DATE**: September 2, 2015

**TO**: Dr. Karthikeyan Umapathy
Computing

**FROM:** Dr. Jennifer Wesely, Chairperson
On behalf of the UNF Institutional Review Board

**RE**: Review of New Project by the UNF Institutional Review Board IRB#784254-1:
"Active Analytics: Adapting Web Pages Automatically based on Analytics Data"

---

This is to advise you that your project, "Active Analytics: Adapting Web Pages Automatically based on Analytics Data" underwent "Expedited" (Category 7) review on behalf of the UNF Institutional Review Board. Your reviewer recommended approval without modifications.

A waiver of *signed* informed consent was requested and approved for this research based on the criteria in 45 CFR 46.117(c). You may use the electronic consent procedures for participants as outlined in your approved documents. Only the approved versions of the consent information should be used during this research. All participants must receive a stamped and dated copy of the approved informed consent document when possible.

Please note that your reviewer identified a typographical error in your survey document (i.e., item #3 "Alex wants has some questions ..." should be "Alex has some questions..."). Additionally you may consider using the words "non degree seeking" instead of "non-matriculated" in your demographics question #2 for ease of student understanding. Although these very minor changes will be allowable without resubmitting an updated version of your survey document, any additional changes will need to be requested via an amendment.

This approval applies to your project in the form and content as submitted to the IRB for review. Any variations or modifications to the approved procedures or documents (aside from the minor changes identified above) must be cleared with the IRB prior to implementing such changes. *For example*, if you plan to make changes to your stamped and dated informed consent form, it will be necessary to submit a copy of the revised form via an amendment so that it can be reviewed and approved prior to use. Once approved, a new stamp and date will be included on the revised consent form so that it can be used. To submit an amendment, please complete an Amendment Request Document and submit it along with any updated documents affected by the changes via a new package in IRBNet. Any unanticipated problems involving risk and any occurrence of serious harm to subjects and others shall be reported promptly to the IRB within 3 business days.

**Your study has been approved for a period of 12 months as of 9/02/2015**. If you would like your project to continue for more than one year, you will be required to provide a completed Status Report and other continuing review documentation to the UNF IRB prior to **8/02/2016.** An extension will be necessary if your study will be continuing past the 1-year anniversary of the approval date. *We ask that you submit your status report and other continuing review information 30 days before the expiration date as noted above to allow time for review and processing.* When you are ready to close your project, please complete a Closing Report Form. Please note that it will be necessary to create a new package in IRBNet in order to submit amendments, status reports, or closing reports in the future. All applicable records relating to this research shall be retained for at least 3 years after completion of the research. Data containing protected health information are to be retained for 6 years.

**CITI Course Completion Reports are valid for 3 years**. Your completion report is valid through 7/12/2018 and Mr. Carle's completion report is valid through 5/31/2018. The CITI training for renewal will become available 90 days before the current CITI training expires. Please renew your CITI training when necessary and ensure that all key personnel maintain current CITI training. Individuals can access CITI by following this link: http://www.citiprogram.org/. Should you have questions regarding your project or any other IRB issues, please contact the research integrity unit of the Office of Research and Sponsored Programs by emailing IRB@unf.edu or calling (904) 620-2455.

This letter has been electronically signed in accordance with all applicable regulations, and a copy is retained within UNF's records. All records shall be accessible for inspection and copying by authorized representatives of the department or agency at reasonable times and in a reasonable manner. A copy of this approval may also be sent to the dean and/or chair of your department.

# VITA

William Carle has a Bachelor of Science degree from the University of North Florida in Computer and Information Sciences and expects to receive a Master of Science in Software Engineering from the University of North Florida in spring 2016. William is currently employed as a Senior Software Engineer at CBS Interactive in San Francisco and previously as a Senior Applications Systems Analyst in the University of North Florida ITS department. William has been working as a full time software engineer for over 5 years.

William specializes in web development and has a strong interest in developing easy to use and intelligently architected web applications. William has professional experience developing within the Microsoft .NET technology stack, the open source LAMP technology stack (Linux, Apache, Python, PHP, MySQL,) and with common web technologies like JavaScript and CSS. In addition to William's work experience he also regularly works on personal and charitable programming projects utilizing multiple different technologies. For more information you can visit William's portfolio website here: http://www.willcarle.com/portfolio.