Software Requirements Specification

for

PageRank Network Analysis of Website Usability

Version 1.0 approved

Prepared by Andrew Braunagel, Dominic Mathiang

SUNY Oswego,

August 30, 2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Dominic Mahiang	8/30/21	Using the template (1.1, 1.2, 1.3, 1.4)	1.0
Andrew Braunagel	8/31/21	Additional information (1.1, 1.2, 1.4, 2.1, 2.2)	1.1
Andrew Braunagel	9/1/21	Added info (1.5, 2.3, 2.4)	1.2

Dominic Mathiang	9/1/21	Added info (1.5, 2.1, 2.6, 2.7)	1.3
Andrew Braunagel	11/27/21	Edited previous entries, and added additional information to sections 1 through 5	1.4

1. Introduction

1.1 Purpose

The purpose of this project is to analyze and create visualizations of the page structure and PageRank of the websites for several 4-year universities in the State University of New York system. With this information, we hope to be able to make predictions about the usability/likeability of the websites.

1.2 Document Conventions

- n/A - (not available) used for sections that we determined do not need to be filled out as part of the product documentation
- f/I - (future information) used for sections that we predict will be filled with more information as it becomes available
- TBD - (to be determined) used for sections that are deemed important, but have not yet been populated
- ~~Strikeout - used for information that may be deleted in the future, but could still be useful~~

1.3 Intended Audience and Reading Suggestions

- The intended audience for this document are university administrators and web developers. The project will help them determine ways in which they can improve upon their university's website for increased usability and structuring.

1.4 Product Scope

- Our team will come up with a product that will crawl a user-provided web URL to automatically gather all the pages and links between pages on the given website.
- The goal is to provide a clear visual representation of the given website's page structure.
- Offer a quantitative analysis of the presence of hub pages throughout the website via a random walk
- Offer a qualitative analysis of how well a website aligns with an objective rating system to determine where on the 'good/bad website' scale it falls.

- Offer suggestions on how a developer can improve their websites.

1.5 References

- Beautiful soup: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- urllib: https://docs.python.org/3/library/urllib.html
- Networkx: https://networkx.org/documentation/stable/index.html
- Gephi - The Open Graph Viz Platform: https://gephi.org/
- Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart: "Exploring network structure, dynamics, and function using NetworkX", in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA the USA), pp. 11–15, Aug 2008

2. Overall Description

2.1 Product Perspective

Our product will utilize a python web crawler to identify all of the individual pages and links between pages associated with the website the product is pointed at. Using networkX we will configure a .gexf file in order to use Gephi to visualize the site's structure/network and use PageRank to rank the presence of hub pages on the site. By using the visualizations of the website's overall structure and the results from the random walk, we hope to provide useful information that webmasters and developers can use to make decisions about their site.

2.2 Product Functions

- Web crawler - crawl the given URL for a list of all pages
- Page Structure Visualization - use networkX and Gephi to visualize page structure
- Random Walk - analyze pages in order for PageRank through random walk of data from web crawler.

2.3 User Classes and Characteristics

- Can be used by Web Developers to analyze their sites to help visualize the page network and make improvements to the site.

2.4 Operating Environment

- The software will be developed, tested, and utilized on personal laptops, running the Windows 10 operating system. Ideally, we will use a wired connection; however, much work will be done over wifi.
- The web crawler is programmed using the python programming language and requires the urllib and BeautifulSoup libraries to be installed.
- Gephi software will need to be installed on the computer in order to create the visualization of the page network.

2.5 Design and Implementation Constraints

- F/I
- Integrating the python web crawler with networkx in order to visualize the page structure.

2.6 User Documentation

- Urllib: https://docs.python.org/3/library/urllib.html
- BeautifulSoup: https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- Networkx: https://networkx.org/documentation/stable/index.html
- Gephi - The Open Graph Viz Platform: https://gephi.org/
- Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart, "Exploring network structure, dynamics, and function using NetworkX", in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), (Pasadena, CA USA), pp. 11–15, Aug 2008

2.7 Assumptions and Dependencies

Dependencies:

- Our web crawler is programmed using python programming language and therefore we felt installing the urllib and BeautifulSoup libraries to the crawler code would enhance the desired result. Urllib will need to be passed a User-Agent value to access most .edu websites.
- By using BeautifulSoup, we will parse the HTML from page source code that can be used to extract URL and link data in a more readable manner.

- We will use networkX and Gephi for creating diagrams and visualizations of the web page's structure
- Random walk is dependent on the data gathered from the webcrawler

Assumptions:

- website will be available for fetch requests
- data remains accessible in the format that we assume
- the website stays active long enough for web crawlers to harvest information
- Internet connection will hold up for entirely of process

3. External Interface Requirements

3.1 User Interfaces

- Website will display the visualizations of each website, along with some statistics information and the results of the random walk.

3.2 Hardware Interfaces

- N/A

3.3 Software Interfaces

- We will primarily use the **python** programming language to develop our application
- **Urllib** is a python module that will be used by the web crawler in order to *request* and *open* the HTML contents of a provided URL
- The **BeautifulSoup** library allows us to pull data from within the HTML contents pulled by **urllib** by given HTML tags. We will use this library to parse the HTML for link tags (<a….>....) and pull the 'href' string.
- For the visualization of the page structure, we will use **NetworkX**, a python package that can be used for the creation, manipulation, and study of the structure and dynamics of complex networks.
- **Gephi** will be used to visualize and edit the graph created by **networkx**

3.4 Communications Interfaces

- ❖ HTTP standards for web crawler using urllib
  - We use the urllib module for python to fetch data (URLs) from the websites.

4. System Features

4.1 WebCrawler

4.1.1 Description and Priority

The webcrawler sends HTTP get requests to the user-provided URL, and parses the page contents for all of the links pointing to other pages within the website and adds their url to a list. The crawler will crawl through every url on the website, and output a .gexf file and a text file containing a list of all the links throughout the site in SourcePage,DestinationPage format. This is a high priority component because it gathers the data needed to analyze and visualize the website's structure. Without it, there would be nothing.

4.1.2 Stimulus/Response Sequences

- ❖ user runs web crawler python file, and enters the URL for the homepage of the website.
- ❖ The program sends HTTP get request to URL
- ❖ The program parses the contents of the request and pulls all of the href tags from html links
- ❖ The program formats the href text and appends it to a list
- ❖ The program repeats the previous three steps until it reaches the end of the crawl list
- ❖ The program creates a .gexf file that can be used for gephi
- ❖ The program creates a text file that can be used by the random walk

4.2 Gephi

4.2.1 Description and Priority

Gephi is a software that is used to create graphs and other visualizations. It is a high priority component as it allows us to create the visualizations of the websites' structures and provides statistical analysis that is helpful in determining the usefulness of websites.

4.2.2 Stimulus/Response Sequences

- ❖ User opens gexf file in gephi
- ❖ For first visualization,

- ➢ Select Yifan Hu Proportional Layout
- ➢ Run layout until the nodes spread out far enough to display structure of site
- ➢ Edit the appearance of nodes and edges to user's liking

❖ For second visualization,

- ➢ Run gephi statistical analysis for modularity
- ➢ Select Force Atlas 2 layout
- ➢ Run layout until the nodes spread out far enough to display structure
- ➢ Set color of nodes based on modularity classification

4.3 Random Walk

4.2.1 Description and Priority

The Random Walk randomly steps through the website by traveling through randomly selected links from each page. It keeps track of how many times it hits each hub and calculates the percentage of times each hub is hit to determine the probability of the user hitting that hub if they randomly clicked around. This is a medium priority component as it is not crucial to the outcome of the project; however, it does provide interesting information that can be of value to the website administrators.

4.2.2 Stimulus/Response Sequences

- ❖ User runs the Random Walk python file and inputs the domain of the website to walk
- ❖ The program opens the output file from the web crawler for that domain
- ❖ The Program organizes the data into a dictionary that is easily used by the program
- ❖ From the current page, the program will randomly select an outward pointing link and travel to that page.
- ❖ The program splits the url to determine the hub, and increases the count by one for that hub.
- ❖ Repeat the previous 2 steps as many times as directed by the user.
- ❖ The program calculates the percentage of how often it reached each hub page and outputs the information in a csv file that can easily be converted to an html table for display on the project website.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- Suitable internet connection

5.2 Safety Requirements

- N/A

5.3 Security Requirements

- N/A

5.4 Software Quality Attributes

- N/A

5.5 Business Rules

- N/A