WSU Libraries Accessibility Project

Project Report

WSU Libraries

WSU Libraries Accessibility Team

Trent Bultsma, Reagan Kelley and Marisa Loyd

TABLE OF CONTENTS

I. Introduction

In accordance with and in shared pursuit of WSU's research exchange mission, we would like to help create a space designed to preserve and share university scholarship [1]. Within a single and shared digital repository, not only do we want a limitless array of knowledge from articles, books, papers, and reports, but we want this digital media to be accessible to all. This starts, first, with these digital documents meeting the standards set out by W3C, an international community trying to bring public work together by providing concrete standards for websites and digital media.

Many digital works are brought to the research exchange repository lacking the initial accessibility standards for digital media set out by the international community, and has resulted in an unknown but copious amount of educational media with sub-optimal accessibility [3]. Our goal is to create an application that can take a pdf and create a modified version that does not change the comprehension or meaning of the work but heightens that document's accessibility to that of W3C standards. We wish to then streamline this process, with it not just assisting a single document, but that of an entire repository, to make the entire WSU research exchange significantly more accessible to all.

II. Background and Related Work

Through researching our project field, we have discovered a lack of automated solutions to the problem of document, and specifically pdf, accessibility. Adobe Acrobat does have some tools for accessibility but they require manual input. For example, when using Adobe Acrobat to make a pdf accessible, the user must first choose the accessibility option in the tools menu, then they have to click on the full check option which opens a pop up window for the user. After this, the user would have to go to the report options of the pop up window, select the page range, which accessibility options to search for and then choose to start checking the document. This manual input required by Adobe Acrobat is something that our team aims to bypass. Our goal is to create a process which automatically updates a pdf's accessibility based on what it lacks without requiring user input.

During our research into the problem of making pdfs accessible, we found that there is an open source repository, pdfminer.six, on GitHub that extracts data from pdf documents. Most specifically, this repository is able to parse, analyze and convert pdfs, extract content as text, html or images, extract tagged content, and extract images. Extracting information from pdfs is a necessary part of our project. However, our focus will be on the document's metadata, color contrast, tags, alternative text for images and reading order in addition to extracting the content and images in the document. Our system will also update these areas instead of simply extracting the information.

To complete this project, our team will need to learn how to extract information from pdfs, how to update the targeted accessibility features based on the requirements of the Web Content Accessibility Guidelines and how to create a new pdf with the updated accessibility features as well as the original information extracted from the pdf.

III. Project Overview

In our quest to bring more accessibility to the WSU research exchange, we have landed on a few key accessibility features to focus on, at least at the start, which can be expanded to

other things as our project progresses. These initial features include document metadata, color contrast, tagging, alternative text for images, and reading order. Our desire is to create a fully automated system of taking pdf documents and converting them into these more accessible versions.

Regarding document metadata, the goal is to include the following information. Title, author, subject, keywords, and document language. The title and author can most often be gathered from looking at the first page of the document for things like large or centered text or comparing groups of words with name databases. To determine the subject and keywords, we will need to implement some sort of data mining algorithms to find common words within the document in question that are uncommon among most documents.

The feature of tagging documents for software output is similar to html tagging. It involves marking things as a header, paragraph, or entries within a table, before converting the intermediary data gathered from the original pdf. Tagging helps define the reading order, which is another accessibility feature we aim to provide, especially in tables, as well as being used to define alt text for images.

For the issue of alternative text on images, we aim to tackle this with some sort of machine learning solution that can generate a description of the contents of images. It is not feasible for us to acquire the data required for a comprehensive image categorization program so we will need to resort to implementing a 3rd party solution. More research is required at this time to find a solution that will not be too costly to realistically implement, be that an open source database of categorized images or a cloud based ai to identify the images for us. This feature does seem like it could be the most time consuming and doesn't have the highest benefit so for now, we have it at a low priority.

With regards to the reading order of the inputted document, our software solution aims to correctly identify and mark the order in which to read text on the page. This involves selecting the heading first, then title, then body in order. For the body of the document, identifying columns and selecting the order of those or choosing images and descriptions in a certain order will be part of that process. The goal of specifying a reading order for our documents is to provide a better experience for people who use assistive screen readers.

The feature of color contrast will be focused on contrast between the text and background color. This will be done by converting all text to 100% black on a white background during the recreation of the pdf in our automated process to be described shortly.

Now with the different initial features defined, we will go over the broader process of bringing those features into reality. Firstly, we will automatically acquire data in the form of pdf documents from the research exchange repository through some means of data harvesting. Next, we will use that data as input to our software that will convert the pdf into a form that can be understood by our automation such as html or a json file. We will then do intermediary processing to ensure the features described above are present. Finally, we recreate the pdf from all that data, resulting in a document that has the same text and image content, but with a much more accessible layout and backend tags/data. This process containing the features described should help us reach our goal of providing more accessibility to users of the WSU research exchange.

IV. Client and Stakeholder Identification and Preferences

Our primary clients are Washington State University Libraries and Anath Jillepalli, our professor for CptS 421. Stakeholders in this project include Talea Anderson, our primary contact for the project, the employees of Washington State University Libraries, students, professors and researchers.

The stakeholders and clients of this project have a few distinct requirements and preferences. For instance, our client, Washington State University Libraries, requires that our team provide a way for employees to process and ensure that documents on Research Exchange, their online repository, meet the Web Content Accessibility Guidelines. The main preference of Washington State University Libraries is that we look at a single collection, identify problems in this collection and then provide a solution to fix the problems identified.

Our stakeholders have slightly different needs, however. Talea and other employees of Washington State University Libraries require that we not only provide a way to process and ensure the documents on Research Exchange meet the Web Content Accessibility Guidelines, but also that we explain our tools and processes to them and show them how to use them. Stakeholders such as students, professors and researchers require that documents on Research Exchange be accessible and readable. The prominent preference for stakeholders is that the documents on Research Exchange have proper tags, alternative text for images, correct metadata and a clear reading order, especially when using screen readers.

V. System Requirements Specification

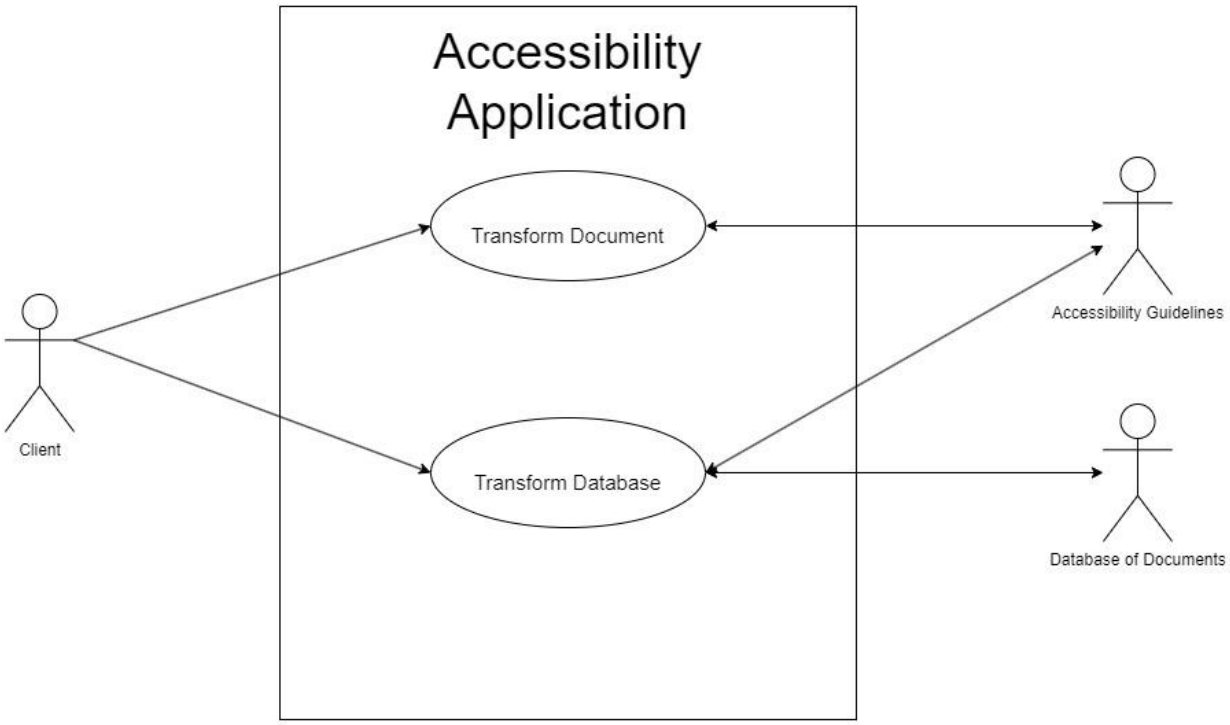
V.1. Use Cases

Individual Documents

The user starts with a new document, void of prior accessibility checks and filters, and when the user desires to enter it into the database they are then queried by the software, if they would like to transform the document into an accessible version.

The Database

The user desires to transform not just a single document but a segment or all of the database into accessible format. Given desired parameters for database querying, the user will click a button and let it run through the database, taking in documents, and transforming each one of them into an accessible version.



V.2. Functional Requirements

V.2.1. Download documents

Obtain Documents for Analysis: We need to be able to access the backend database that contains the many documents for accessibility transformation.
Source: Our client, the WSU libraries, would appreciate the ability to use this accessibility translation software for many documents quickly, not just on a single document. This will require dynamic document retrieval from the database for later translation.
Priority: Priority Level 0: Essential and Required functionality

V.2.2. Exporting documents

Export Documents to PDF: The software must be able to export documents as to pdfs after all intermediary processing is done. This happens at the end of the process.
Source: WSU research exchange users are our primary stakeholders for this functionality. Creating a readable document for them is important, one that is visually appealing in addition to being accessible, which is the point of the project, is important. Ideally, the outputted document should be equal or better in visual quality to the input document gathered from the research exchange repository.
Priority: Priority Level 0: Essential and Required functionality

V.2.3. Document metadata

Configuring Metadata from Documents: The application needs to be able to translate all documents into a readable, programmable, and transformable format. This means transforming

the data from its visual representation into a text and numerical description that describes the nature of the document, its metadata. No matter the content of the document, this procrustean format will allow quick identification of accessibility errors and later correction.
Source: It is important for the people using our software at the WSU libraries that the speed of this process, the moduluzaration of metadata, accessibility detection and correction, are expedited. If the user wishes to transform an entire database, we need to ensure that our software is rigid, correct, and fast; and this can be ensured by a normalized metadata format.
Priority: Priority Level 0: Essential and Required functionality

V.2.4. Document tags

Providing Tags for Document Content: Tagging documents involves marking content with labels such as a header, paragraph, row entries within a table, etc. Tagging helps define the reading order, which is another accessibility feature we aim to provide, especially in tables, as well as being used to define alt text for images.
Source: The presence of tags within a document like a PDF allows for clear identification of document elements, which is a mandated accessibility requirement by W3C; a requirement from our client, Talea, at the WSU libraries. Tags are also important to users of the WSU library research exchange because they allow for things like screen readers to function properly.
Priority: Priority Level 0: Essential and Required functionality

V.2.5. Reading order

Document Reading Order: Our application needs to be able to specify a reading order within the PDF output documents for use with screen readers.
Source: Users of the WSU research exchange who are reading through the documents on the repository need to have a specified reading order to use screen readers effectively. This functionality will satisfy that requirement.
Priority: Priority Level 0: Essential and Required functionality

V.2.6. Color contrast

Sufficient Color Contrast: Outputted documents should follow a standard of color contrast that is easy to make out words on the page. This requirement is specifically all black text on an all white background for the most effective contrast.
Source: Color contrast is important to users of the WSU research exchange, especially those who have impaired vision or experience difficulty deciphering text with a similar color to the background. The color contrast functionality of our software also fulfills a W3C requirement, which has been requested by our client, Talea, at the WSU libraries.
Priority: Priority Level 0: Essential and Required functionality

V.2.7. Alternative text on images

Image Alternative Text: The feature of adding alternative text to images would be nice to have if possible, but would probably require a significant amount of effort for a functionality that isn't the most important. It would result in having text embedded in image data in the PDF that describes what is the content of the image.
Source: Alternative text is a feature that would be appreciated by users of the WSU research exchange, specifically those who use screen readers, who have difficulty getting the full depth of

information from an image. It is also a requirement specified by the W3C standard, which has been requested by our client, Talea, with the WSU libraries.
Priority: Priority Level 2: Extra feature or stretch goal

V.2.8. Uploading documents to the repository

Document Uploading: One potential functionality for our software is uploading PDF documents back to the WSU research exchange repository after accessibility conversion takes place. Our client, Talea, has mentioned that this feature may not be possible to implement so we will look into it if we have extra time later.
Source: This feature would be helpful for employees at the WSU libraries to reduce their manual time to upload documents after accessibility conversion has been completed.
Priority: Priority Level 2: Extra feature or stretch goal

V.3. Non-Functional Requirements

V.3.1. Must create pdfs that are accessible

Create Accessible PDFs: PDFs created must follow the accessibility guidelines outlined in the Web Content Accessibility guidelines with a focus on metadata, tags, reading order, color contrast and alternative text for images.

V.3.2. Must work with the WSU research exchange repository

Compatibility with Research Exchange: The system must be able to export documents from Research Exchange, WSU's repository, so it has to be compatible with Research Exchange repository.

V.3.3. Able to run autonomously overnight

Run Autonomously: The system shall be autonomously able to create accessible PDFs while left to run overnight.

V.3.4. Files less than 2 Gb

Produce Documents Under 2 GB: Documents produced by the system must be under 2 giga-bytes in size as per storage requirements set by WSU Libraries.

V.3.5. Works on windows machines

Run on Windows: WSU Libraries primarily uses PCs which run the Windows operating system, so the system will be compatible with Windows to ensure our client is able to use it to the fullest extent.

V.3.6. Response time not important/relevant

No Relevant Response Time: Talea has asked us to create a system that the WSU Libraries faculty can run overnight. Thus, response time is not extremely relevant.

V.3.7. For product delivery, send the executable file

Send Executable File: System shall be sent to WSU Libraries in the form of an executable file upon completion of coding and testing.

VI. System Evolution

Some of the fundamental assumptions of our project include the following. One assumption we have made is that we are using the WSU research exchange to get our pdf data. We do not anticipate this changing as we are creating a specialized application for the libraries specifically, but it is an assumption our project is based on. Another assumption is that the program will be run on windows machines. The library has mostly or all windows machines so this will also probably not change, but it is an assumption we will be basing off of to do windows based development. Also, we are assuming that we are unable to upload pdfs to the research exchange, which is why we are holding off on that feature. One last assumption we are basing our project off is that we are unable to work with the PDFs to make them more accessible without recreating them. This is why we are harvesting the data and then recreating the pdf. We did look into the adobe API but it seemed inconvenient and less functional than what is possible with recreating it.

Some points of risk in our project include the following. There is a risk of overwriting data on the repository if we do ever implement uploading, also with the possibility of uploading improper data that loses information, so we will need to be careful about double checking our outputs until we are confident in the quality. Another risk is that the PDF code libraries we are planning to use may not have all needed functionality. From our research, it does seem that the functionality is there, but when we start developing our application it may turn out that some things are not included.

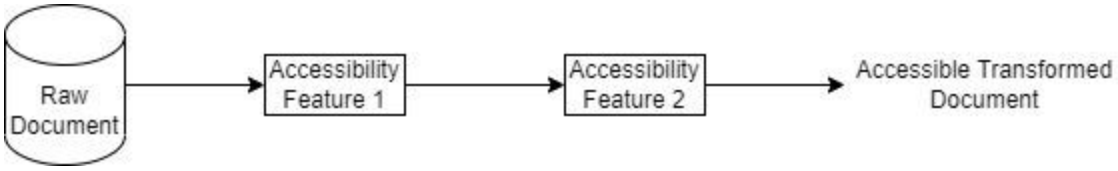
VII. System Overview

General description of the functionality and design of the project (intro to next couple sections). This should be easy to write using the above documentation we have already made. <Marisa>

VIII. Architecture Design

VIII.1. Overview

The architectural model we have selected is the pipe and filter model. The pipe and filter model involves a series of transformations done on some data like an assembly line to create an overall larger transformation [6]. We have selected this model for our project because there are many different transformations our data needs to go through, those being accessibility features such as adding metadata or alt text, to produce one overall transformation of making the document more accessible. Each of our functional requirements should map to a subsystem, where each subsystem is a transformation or filter in the overall pipeline of our model.



The components are going to map to each of the accessibility guidelines (each feature will be a component like one for alt text images, one for metadata, one for tags, etc.)

Overall architecture of software and rationale

Need a UML component diagram
Need to describe each layer/component in our architecture and its responsibilities
<Trent>

VIII.2. Subsystem Decomposition

How the system is composed into subsystems
Identify dependencies between subsystems
Diagram for each subsytem
<Trent half and Reagan half>

VII.2.1 [Subsystem Name] (each of these will map to a functional requirement)

Include the following sub-sections for each subsystem.

a) Description

Describe the subsystem and identify its responsibilities.

b) Concepts and Algorithms Generated

Discuss the concepts, algorithms or solutions generated and considered for this subsystem. Report the selected solution and explain the solution selection process. Include any special considerations and/or trade-offs considered for the solution approach you have chosen.

c) Interface Description
Provide a description of the subsystem interface. Explain the provided services in detail and give the names of the required services.

Services Provided:

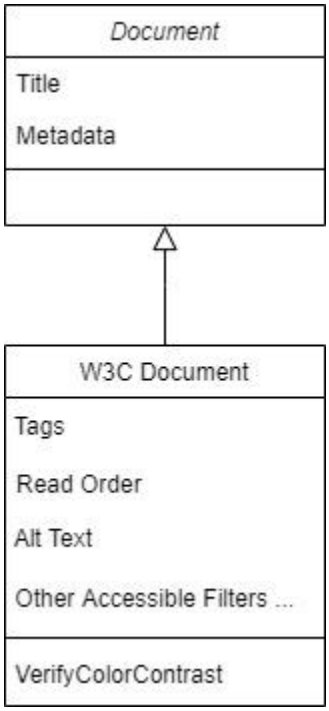
Service name
Service provided to: [list the receiving subsystems here]
Description: [Describe what the service is and what it does. Provide its input and output values. Briefly describe the major functions that the service provides.]

Services Required:
Names of the required services and the subsystems that provide them.

IX. Data Design

We will need to transform a PDF formatted file into a programmable object, and to this, we will first translate the PDF data into JSON format. We will then parse this data and represent the JSON attribute pairs as a dictionary. To manage and manipulate each PDFs metadata and corresponding JSON dictionary we will need to use a data structure.

We can object-oriented programming to represent each respective document that goes through our software. Regardless of if the document meets the accessibility guidelines of W3C, we can substantiate each PDF in a document class. Before this object goes through our assembly line filters, it will need to be transformed into a child class, a W3C Document, which inherits the values of the document but wrapped with accessibility checks. Our filter components that check each accessibility guideline, respectively, will verify that the metadata contains all the accessibility changes it needs, and these checks can be easily managed through a W3C Document's class member variables and functions.



X. User Interface Design

The user interface design for this project is going to be very basic. Our client, WSU Libraries, has requested the ability to run the program overnight and pause in the morning leaving very little interaction between the program and the user. However, the interaction between the user and the program is very important especially when determining which documents on the Research Exchange website to transform into accessible PDFs.

The program can either be used to transform individual documents, selected by the user, into accessible PDFs or it can be used to transform a segment of documents from the Research Exchange website into accessible PDFs. The initial user interface will provide two buttons, one to transform individual documents and one to transform a segment of documents into accessible PDFs.

In the case of transforming individual documents into accessible PDFs, the user interface will ask the user for the document and ask if the user is sure they want to transform the document into an accessible PDF. If the user selects the 'yes' option, a new and accessible PDF will be created for the user. If the user selects the 'no' option, the user will be prompted for another document to transform.

In the case of transforming a segment of documents, the user interface will include a start button and a pause button. The user will press the start button to start the program. After selecting the start button, the user will be prompted to provide parameters, such as which section of the Research Exchange website to start in or which document to start on, and select the run button which will then pull PDFs from the Research Exchange website based on the user's parameters and create a new, accessible PDF, from the data collected with correct reading order, metadata, tags, color contrast and alternative text for images. The program will do this without user input, so during this time there is no interaction between the user and the program. The user interface will display a pause button and the program will run until the user selects the pause button to stop the program. After the program is paused, the user will be able to either start the program again, starting from where it left off, or go back to the main screen to choose between transforming individual documents and transforming a segment of documents.

After the project is completed, WSU Libraries will be given a flash drive containing an executable file which they can run on their computers. This executable file will be the program. They will not need to install the program, simply transfer the executable file from the flash drive to their computer and open the file to start the program. The user can open the executable file in several ways including double clicking the file after transferring it to the new computer, selecting the file from the start menu, or right clicking on the program and selecting the run option [7]. Upon opening the program for the first time the user's computer will likely ask for permission from the user to run the program. After the program is opened the user will be able to select the start button to begin pulling PDFs from the Research Exchange website and creating new, accessible PDFs which they can upload to the Research Exchange website in place of the old PDFs pulled from the site.

**The information in this section should be accompanied with proper images showing how exactly you vision the interface to be like (for example mock-ups).

XI. Glossary

Accessibility - When websites, web tools, and software are properly designed and coded, it allows for people with disabilities to use them. W3C, World Wide Web Consortium, provides standards or expectations on how digital media should be presented to ensure those with disabilities can still gain full advantage and understanding of the material. Accessibility in digital media is how well the given software is in accordance with W3C accessibility standards.

Data mining - A process of discovering and analyzing patterns within large data sets. This is done through machine learning, statistics, and data collected through database systems.

Executable file - A program that can be run with a set of instructions or options to make it do something on a computer.

JSON - JavaScript Object Notation; it is a type of file that specifies an object's attributes.

Metadata - Data that provides information about data. This allows us to retrieve descriptive information about a file without looking at the content.

XII. References

[1] "Create and verify PDF accessibility (acrobat pro)," *Create and verify PDF accessibility, Acrobat Pro*. [Online]. Available: https://helpx.adobe.com/acrobat/using/create-verify-pdf-accessibility.html. [Accessed: 20-Sep-2022].

[2] Pdfminer, "Pdfminer/pdfminer.six: Community maintained fork of pdfminer - we fathom PDF," *GitHub*. [Online]. Available: https://github.com/pdfminer/pdfminer.six. [Accessed: 20-Sep-2022].

[3] *Rex.libraries.wsu.edu*. [Online]. Available: https://rex.libraries.wsu.edu/esploro/. [Accessed: 20-Sep-2022].

[4] W. C. W. A. I. (WAI), "Introduction to web accessibility," *Web Accessibility Initiative (WAI)*. [Online]. Available: https://www.w3.org/WAI/fundamentals/accessibility-intro/. [Accessed: 20-Sep-2022].

[5] "Working with content streams¶," *Working with content streams - pikepdf 6.0.2.dev9+g25f0537 documentation*. [Online]. Available: https://pikepdf.readthedocs.io/en/latest/topics/content_streams.html#extracting-text-from-pdfs. [Accessed: 20-Sep-2022].

[6] S. Hasan. "Pipe and Filter Architecture," *Medium.com,* Available: https://syedhasan010.medium.com/pipe-and-filter-architecture-bd7babdb908 [Accessed 5-Oct-2022].

[7] B. Stockton, "What is an executable file & how to create one," *Help Desk Geek*, 13-Sep-2020. [Online]. Available: https://helpdeskgeek.com/how-to/what-is-an-executable-file-how-to-create-one/. [Accessed: 05-Oct-2022].