Arizona State University

Text File Analyzer Program

Group Project

Jordan Lewis, Ben Shu, Andrew Nguyen

CSE 360: Intro to Software Engineering

Debra Calliss

Table of Contents

Content **Page**

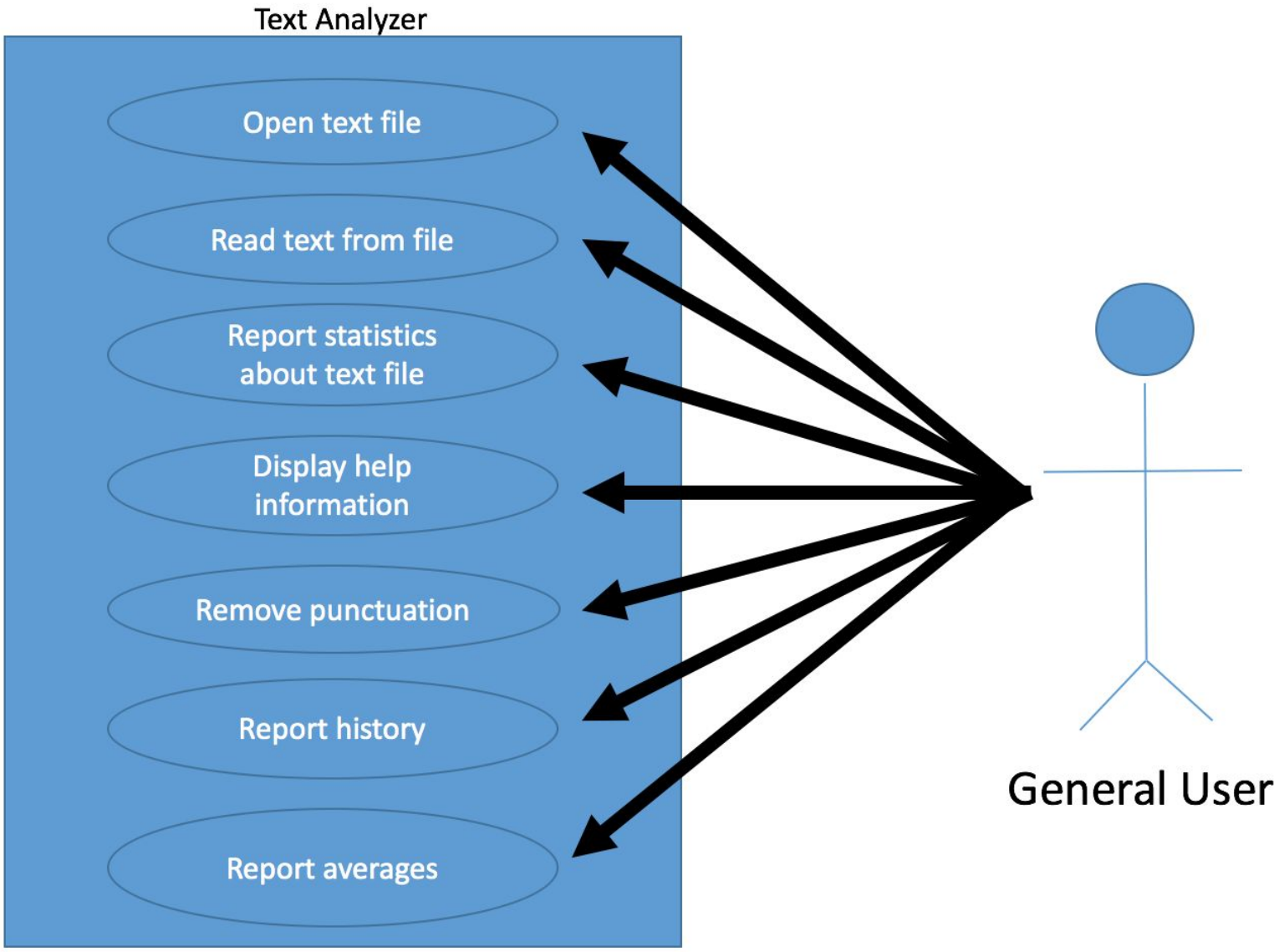
Revisions

1.0	Initial Documents	Jordan Lewis
1.1	Initial files with GUI	Andrew Nguyen
1.15	Updated GUI with some functionality	Andrew Nguyen, Jordan Lewis, Ben Shu
1.2	Stable prototype	Andrew Nguyen
1.25	Show history of files loaded	Andrew Nguyen
1.3	Shows number of spaces, words, average characters per line	Jordan Lewis
1.35	Added help page	Andrew Nguyen
1.4	Shows top 3 most common words, average word length	Ben

Overview

The program will be able to open files and validate file names, read text from a file and report the number of lines, blank lines, spaces, and words along with the average characters per line, word lengths, and most common words. This program will also display help information and remove punctuation and possession from analysis. It will be able to handle multiple files, including reporting history of all files processed (name and date), and the averages across all files as well.

Use Case Diagram of Text Analyzer



User Stories

- As a user, I want to open a text file so I can read the text file.
- As a user, I want to read a text file so I can analyze the text in the file.
- As a user, I want to report statistics from a text file so I can gain extensive information about the text in the file.
- As a user, I want to display help information so I can learn how to use the text analyzer.
- As a user, I want to remove punctuation so I can remove unnecessary punctuation..
- As a user, I want to report file history so I can see the past files that were loaded in a session.
- As a user, I want to report averages so I can see the statistics of all the past files that have been loaded collectively.

How to Test:

Test Case	**Description**	**How to test**
Number of lines	Returns the number of lines the text file has	Make a test file with a certain number of lines, and use it in the program. The JUnit tests will look for the exact number of expected lines from a test text file.
Number of blank lines	Returns the number of blank lines the text file has	Make a test file with a certain number of blank lines, and use it in the program. The JUnit tests will look for the exact number of expected blank lines from a test text file.

Number of spaces	Returns the number of spaces the text file has	Make a test file with a certain number of spaces, and use it in the program. The JUnit tests will look for the exact number of expected spaces from a test text file.
Number of words	Returns the number of words the text file has	Make a test file with a certain number of words, and use it in the program. The JUnit tests will look for the exact number of expected words from a test text file.
Average chars per line	Returns the average number of characters per line	Make a test file with multiple lines that can have characters, and put it in the program. The JUnit tests will look for the exact number of the average characters per line from a test text file.
Average word length	Returns the average number of the length of the words	Make a test file that can have multiple words, and use it in the program. The JUnit tests will look for the exact number of the length of the words from a test text file.

Most common words	Returns a list of words in order from most common to least common with the number of times used	Make a test file with a certain number of words, and use it in the program. The JUnit tests will look for the most common words in order from a test text file.
Average of all above statistics for all processed files	Returns a list of all the above statistics' averages such as average	Make a test file with a certain number of lines, and use it in the program. The JUnit tests will look for the exact number of the average of all above statistics for all processed files from a test text file.

General test plan: Make a test case (text file) that we will manually perform the operations on (Ex: count the number of lines). We will then put our test file in our program and see if we get the same answers.

Goals for Deliverables/ Tasks and Assignments

Goals for Deliverable 1:

1. Create GUI with minimal functionality - The deliverable must at least be able to load a text file. Button and GUI elements, layouts, and basic navigation of the application will be will be showcased.
2. Create feature to count number of lines - The feature does not have to be implemented into the GUI, but at least a prototype of the feature could be used in a console application and ready to test with JUnit.

Tasks and Assignments for Deliverable 1:

- Algorithm to count the number blank of lines - Jordan
- GUI - Andrew
- Load files - Ben

Goals for Deliverable 2:

1. Count number of spaces
2. Count number of words
3. Calculate average number of lines
4. Calculate average characters per line
5. Calculate average word length
6. Calculate most common words
7. Calculate average of all statistics for all processed files
8. Make a help page

Tasks and Assignments for Deliverable 2:

- Count number of spaces - Jordan
- Count number of words - Jordan
- Calculate average number of lines - Jordan
- Calculate average characters per line - Jordan
- Calculate average word length - Ben
- Calculate most common words - Ben
- Make a help page - Andrew
- File history list with date - Andrew
- Branch approval and administrator -Andrew

Tasks and assignments for Deliverable 3:

- Calculate average number of lines across all used files - Jordan
- Calculate average number of blank lines across all used files - Jordan
- Calculate average number of spaces across all used files - Jordan
- Calculate average characters per line across all used files - Andrew
- Calculate average word length across all used files - Andrew
- Calculate most common words across all used files - Andrew
- JUnit testing - Ben

Class Diagram