Software Requirements Specification

for

Voting Systems

Version 1.0 approved

Prepared by Isaiah Herr, Emma Barnes, Benjamin Keeney, and Moritz Lindner

University of Minnesota Twin-Cities

February 2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Creation	2/19/21	SRS document is created	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed description of the voting system being developed based off of the Instant Runoff Voting and Open Party Listing election processes. This document will provide information about the purpose of the system, features, the constraints the system must work on, and how the system will react to certain events. This document is intended for testers, developers of this system, and election officials that will be using this product.

1.2 Document Conventions

This document is created based off of the IEEE template for System Requirement Specification Documents.

1.3 Intended Audience and Reading Suggestions

The next chapter, chapter 2, of this document will provide in-depth information on the general system requirements and constraints of the system. Chapter 2 and 3 should be read by both developers and users for an overall understanding of the system, the system requirements, and how the system works. Chapter 4 should be read primarily by developers to understand various use cases of the system. Chapter 5 contains requirements this software will fulfill and Chapter 6 contains a glossary of terms used in this document.

1.4 Product Scope

Our group prides itself on providing efficient and easy to use software for our customers based on their needs. This system is a system that will take an election file based off of IRV and OPL to generate an audit file indicating winners of the election based on votes. This system quickly and efficiently calculates votes for each candidate, outputs the process to the user, and finalizes the top candidate in an audit file provided to the user.

1.5 References

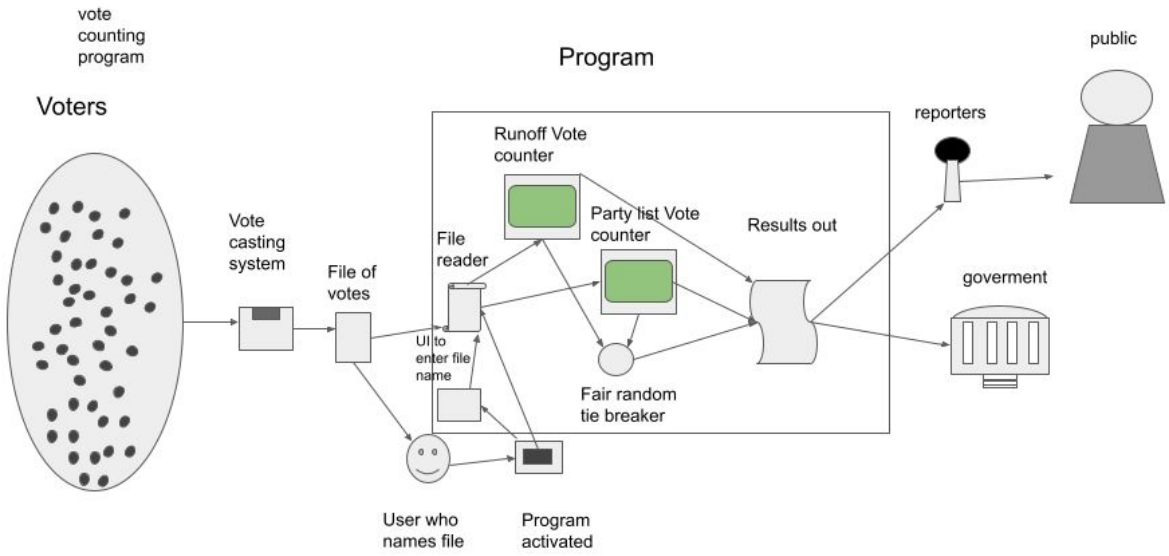
IEEE Template for System Requirement Specification Documents:

IEEE Software Requirements Specification Template

2. Overall Description

2.1 Product Perspective

This product is part of a class assessment on creating a system to count votes. It is primarily targeted towards election officials in order to efficiently count ballots and decide election winners. Presumably there are voters then a system that records the ballots in the file system the counting system can read. Then after our system declares the winners there is a parliament that will be filled appropriately. The larger system this is a part of is the parliament of a government. Below is a diagram how this system would work as a whole.



2.2 Product Functions

This program will need to read votes and the type of election off a comma separated values (CSV) file. It will then need to be able to count the votes in accordance with the given voting system. It will need to do this for 2 possible types of voting Instant runoff voting and party list voting. The product will then produce an audit file that showcases the final results and the process of how those results were determined. In addition, the product can also create a media file if needed by the user.

2.3 User Classes and Characteristics

The two types of users of our system are the election officials who input the voting file and various outside testers. These groups are not mutually exclusive and it is likely the voters and politicians they elect will care about and be impacted by this system. This system won't be used often to input election votes but when it is it will likely get a large input and it is very important it

returns accurate results. It is likely various people will run numerous tests on this system that have varying amounts of input. Well it is not as important that the system give accurate results in this circumstance it is still very important as any failures found here would degrade voter confidence and have to be fixed before the next election.

2.4 Operating Environment

The program will operate mainly on Ubuntu CSE machines at the University of Minnesota Twin-Cities. These machines will be on a 64 bit operating system and must have either Java 7 or Java 8 installed. In general, this program can be run on Windows 10, Ubuntu, Ubuntu 16.04 or higher, and Mac OS. It is likely the machines will have at least 16GB ram and a 3GHz processor.

2.5 Design and Implementation Constraints

As an assignment we are limited by timing because this is due 2/19/2020. We are all also limited by the fact that each of us has other responsibilities and that we can't meet in person due to covid. We are also restricted to only using Java as a programming language and the waterfall methodology. We are not allowed to go outside of those things for this project.

2.6 User Documentation

We will leave instructions for users on how to activate our software and how to get it to read files in a text file. This will include where to put the files and should include the files format and how to properly execute the program.

2.7 Assumptions and Dependencies

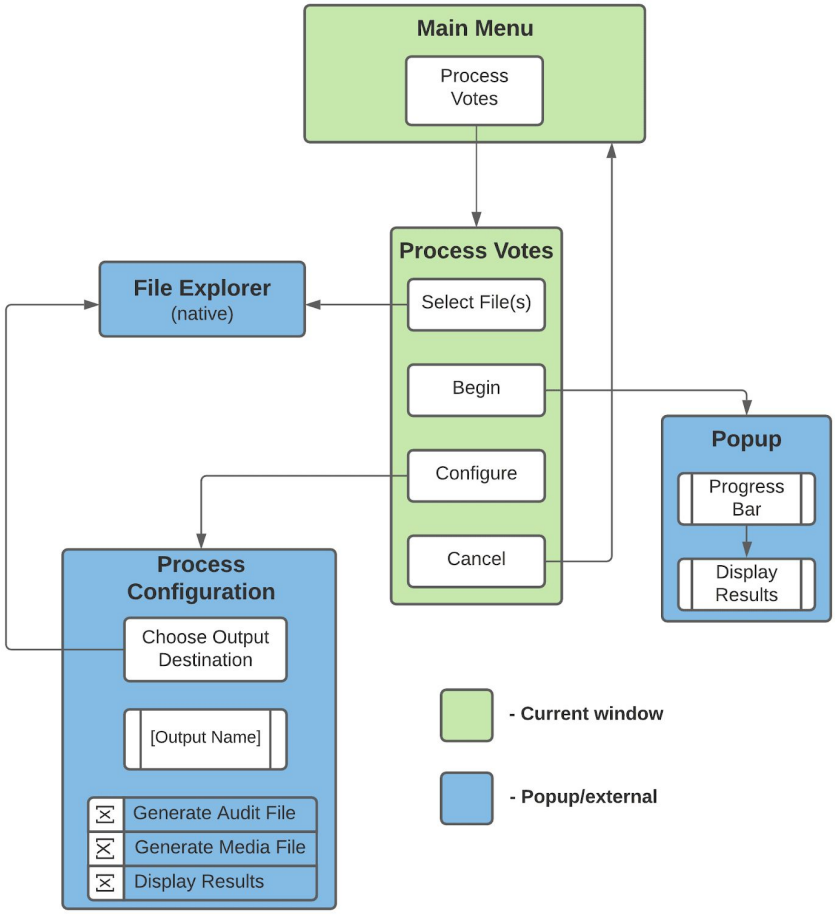
We can assume that this project will be graded based on not only how well the code runs and is documented but also how well the code is written. We can also assume that for this type of software making sure the code is well written and documented is more important than adding bells and whistles

This system also assumes that there are no errors in the CSV file and that everything is correctly formatted. This system also assumes that the machines being used are all up to date CSE lab machines as specified in a previous section. Lastly, this system also assumes that only one file will be used at a time on the system. This means that there will never be more than one file being used on the system. If more than one file is inputted, the program will throw an error to the user.

3. External Interface Requirements

3.1 User Interfaces

3.1.1 Screen State Flowchart



3.2 Hardware Interfaces

The operating system must be 64 bit. It is recommended to have at least 16GB RAM, and a processor speed of 3.0GHz. USB 2.0 port or higher for importing election and audit files.

All media, read or written, will be located on the device's HDD/SDD. Reading operations will be performed on election files and audit files, writing operations will produce audit files and media files.

3.3 Software Interfaces

This software requires that Java 7 or Java 8 is installed on the device. To run the software, it must operate on Windows 10, Ubuntu, Ubuntu 16.04 or higher, or Mac OS X.

All inputs to the program will be CSV files and all outputs will be text. Inputs include election files from the user. Outputs include audit files and media files. Election files include information about the election and candidates to be parsed and analyzed to determine the outcome. Audit files include all information about a previously analyzed election. Media files will be produced in addition to the audit file. It will contain basic information about the results of the election such as candidates, percentages, and other relevant information.

3.4 Communications Interfaces

A secure internet connection is required to import files relevant to the system, if not importing through an IO port.

4. System Features

Please review the use cases document included with the Software Requirements Specifications in order to view all the use cases associated with this system. The document is titled UseCases_Team19.pdf.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The Voting System must be able to count 100,000 ballots within 8 minutes.

5.2 Safety Requirements

It is always important to keep a secondary copy of the file being used somewhere safe. It is recommended for users to simply copy the file from their USB/storage device or create a copy of the file on their machine and store the original files somewhere else safely. This is to prevent total loss of data in case loss, damage, or corruption of the CSV file could occur. This is important as the system will only read in files in the same directory of the system.

5.3 Security Requirements

The voting system does not have any security requirements as it is assumed that if a user uses the program, they already have authorization. As a result, any kind of user with access can use this system assuming that format of the file and the information inside the file is correct.

5.4 Software Quality Attributes

This system is designed to be very easy to use with minimal effort from the user. It will take information either from a file given by the user or from user input. It can be used with basic knowledge as long as the user gives the program a file of the correct format and information. It is, however, assumed that the user has basic knowledge of how an IRV or OPL election works in order to properly understand the output of audit files and media files.

5.5 Business Rules

In the case of a tie during an election, a coin toss will occur to randomly choose between two candidates.

6. Other Requirements

Appendix A: Glossary

Term	Definition
Users	Any person that will be using the system
Developer	Any person that is working on developing and creating the system
Testers	Any person that will be using the system to test for functionality and bugs.
Machine, Device	Refers to the computer that the system will be run on
Election File	File provided by relevant legal body that contains voting type, candidates, and ballots for the election
Audit File	File produced by this software that contains exhaustive information gathered by analyzing an Election File
Media File	Refined/processed form of an Audit File more suitable for direct viewing
System/Software/Program	The product for voting systems that is being developed and documented
Instant Runoff Election/Voting (IRV)	An election in which an individual ranks candidates from their favorite to their least favorite. If a candidate is eliminated, and is an individual's favorite, their ballot will be distributed to their next favorite candidate and so on. The candidate with the most votes is elected.
Open Party Listing (OPL)	An election in which an allocation formula is used. A quota is determined by taking the total number of ballots divided by the total number of seats. If a candidate or party reaches that quota, they are given a seat. The party/candidate with the largest remainders will be given any seats leftover.
Comma Separated Values (CSV)	A delimited text file that uses commas to separate values. It is commonly used in programs such as Microsoft Excel

Awarded seat	A seat given to a official as a result of the election process