Software Requirements Specification

for

STV (Droop Quota)/Plurality Voting System

Version 0.4 approved

Prepared by Team 4

Sara Nelson (*nels8907*)
Brendan Ritchie (*ritch167*)
Yiwen Xu (*xu000515*)
Yifan Zhang (*zhan4372*)

CSCI 5801

21 February 2020

Table of Contents

Revision History

Name	**Date**	**Reason for Change**	**Version**
Droop	16 February 2020	SRS Draft Creation	0.0
Droop	18 February 2020	Add product functions	0.1
Droop	19 February 2020	Added use cases	0.2
Droop	20 February 2020	Updated use cases; Addition of Glossary, Product Perspective, User Classes and Characteristics, Operating System, User Documentation, Hardware Interfaces, Software Interfaces, Communication Interfaces,	0.3
Droop	21 February 2020	Editing Product Functions, Use Cases, SRS Formatting, Final Review	0.4

1. Introduction

1.1. Purpose

The purpose of this document is to present a detailed description of the STV/Plurality Voting System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system.

1.2. Document Conventions

This Document was created based on the IEEE template for System Requirement Specification Documents.

1.3. Intended Audience and Reading Suggestions

The materials of this document provide instructions and guidance for typical users of the STV/Plurality Voting System. These users will be election officials who need to tally the votes from an election to determine the winner(s) using the voting system. Furthermore, any programmers who might be interested in working on the STV/Plurality Voting System by further developing it or fixing existing buggs may find this document useful.

1.4. Product Scope

This software will be a voting system that has the option of either running an STV system or a plurality voting system. Users will be collecting ballots from elections and will be using this system to determine the winner of an election based on the desired system selected. The user will be able to input the number of seats that are open during the election, the file names that contain the ballots, and which algorithm to use. The outcome of the software will be a fair evaluation of the ballots to fill the number of seats available in the election.

More specifically, if the STV (Droop) option is selected, the software will provide a method for randomly shuffling the ballots and calculating the droop. It will avoid the early voter bias that can occur with using the Droop method for voting by the random shuffle.

1.5. References

1. IEEE Template for System Requirement Specification Documents: https://goo.gl/nsUFwy
2. System Context Diagram: https://ep.jhu.edu/about-us/news-and-media/systems-context-diagrams

2. Overall Description

2.1. Product Perspective

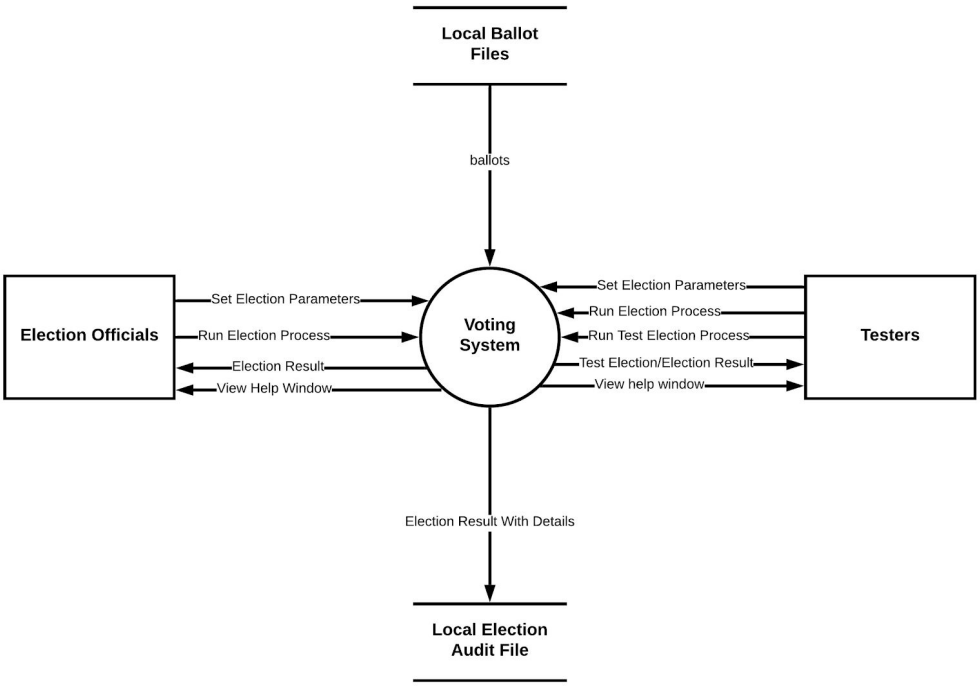

Figure 1 System Context Diagram

The STV/Plurality voting system is developed to sort through voting ballots and determine the winners of an election based on the selected voting method and number of seats that needed to be filled. The users need to upload the ballot files in the same directory as the program to enable the system to read the files. The users will also need to select the desired method of voting that matches the ballot formating. The user will be able to audit the election by viewing the audit report that will be automatically generated at the conclusion of an election. This file will show the ballots that were assigned to a candidate as the election progressed that the results of the election can be verified if necessary. All pertinent election data and results will be printed at the beginning of the audit report. A snapshot of the election will appear after the voting algorithm runs that displays the number of ballots, the number of seats, and the number of candidates, and the winner/losers with their percentages or order (depending on the election type).

2.2. Product Functions

- Set voting algorithm: All users can choose plurality or STV algorithm, through a prompt by the system.
- Set number of seats: All users can set the number of seats, through a prompt by the system.
- Choose ballot files: All users can choose which ballot files the election will be based on by entering their file names through a prompt by the system.
- Open help window: All users can open the help window via a text command.

- Turn off shuffle option: Programmers and testers can turn off the shuffle option through a prompt by the system.
- Run voting algorithm: All users can run the voting algorithm based on the selected algorithm and number of seats.
- Display parameters of the election prior to running the election: The system can display the parameters of the election, including the type of voting algorithm, the ballot file names, and the number of seats to all users.
- Display election results summary: The system can display results of the election, including the number of ballots, number of seats, number of candidates, and winners and the losers with pertinent information such as percentage of votes received for plurality or the order of the winners and losers for STV.
- Cancel prior to voting algorithm running: All users can prevent the voting algorithm from running before an election starts through a text command when the system asks them to confirm the election parameters.
- Display audit file name and location: The system can display the name and the file path of the election audit file of last election to all users after the voting algorithm has finished running.
- Run test of voting algorithm: Programmers and testers can run the test of voting algorithm process based on chosen test files, the selected algorithm, number of seats, and the status of the shuffle option.
- Invalid input handler: If users enter invalid parameters for the election or unrecognizable commands, the system will require users to reenter.

2.3. User Classes and Characteristics

The reader of this document is expected to be familiar with the internet and have a background knowledge of the use of simple menu systems with selection options. There will be inputs that are needed to run the algorithms and the method selection. In addition, there will be a help window available via a text command.

The election official will be expected to be able to use text commands to run the election through the user interface.

The testers will be expected to be familiar with Linux terminal commands to test the software system as well as the C++ coding language.

2.4. Operating Environment

The STV/Plurality Voting System will be running on a Linux operating system.

2.5. Design and Implementation Constraints

Multiple runs of the software voting algorithm will not be allowed for the given set of ballots. This is due to the possibility of a results change due to the shuffle of the ballots. If the algorithm is run multiple times, the results could possibly change and affect the integrity of the election.

The STV/ Plurality Voting System is not verified to run on any operating system other than a Linux OS.

The system will be implemented in C++ coding language and all source files and a Makefile will be provided. The program is expected to be executed from the command prompt.

2.6. User Documentation

The user will be provided with a help window for any questions regarding the usage of the software program. The user will also have access to this document as guidance when using the system.

The testers will be provided with a documentation of how to run tests in this system.

2.7. Assumptions and Dependencies

It is assumed that there are no erroneous ballots being presented to the system; that they were all removed prior by a third party. It is also assumed that all the files will be stored in the same directory as the program files. It is assumed that the first line of each ballot will contain the names of the candidates and each subsequent line will be the voter's ballot. If Droop is selected for the voting algorithm, it will be assumed that at least half of the candidates are ranked.

3. External Interface Requirements

3.1. User Interfaces

There will be a text based user interface.

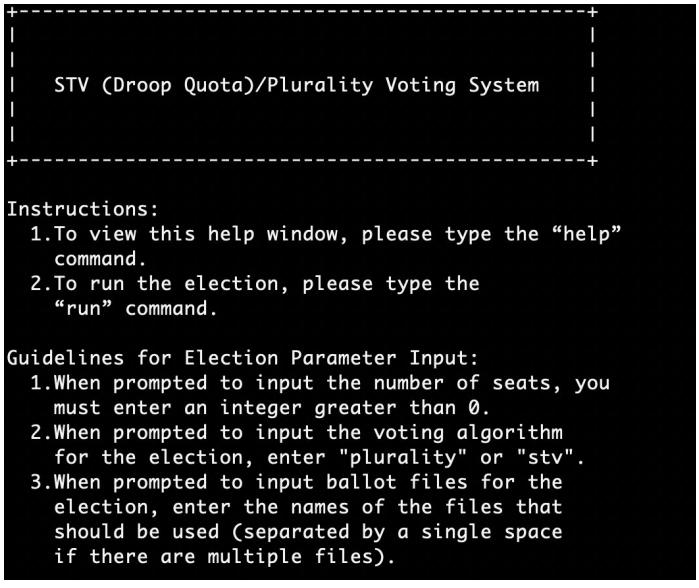

Figure 2: Help Window

3.2. Hardware Interfaces

There will be no embedded systems (hardware interfaces) in the STV/Plurality Voting system.

3.3. Software Interfaces

The STV/Plurality Voting System will need to be able to pull in Excel-created CSV files (Windows comma separated files) which will contain the ballot information for an election.

3.4. Communication Interfaces

The STV/Plurality Voting System does not need to be connected to the internet. All ballot files are sent to qualified election officials through methods outside this system.

4. System Features

4.1. User Enters Number of Seats

Name	User Enters Number of Seats
ID	UC_01
Description	The user inputs the number of seats that need to be filled in this election.
Actors	Election Official, Programmer
Organizational Benefits	Allows the voting algorithm to be able to fill the proper number of seats for the election.
Frequency of Use	Once
Triggers	The system prompts the user to input the number of seats for the election.
Preconditions	● The system has not prompted the user for the number of seats yet ● There is currently no value stored in the system for the number of seats to be filled in the election
Postconditions	● A value greater than 0 is stored in the system for the number of seats to be filled in the election
Main Course	1. The user types the number of seats that should be filled in the election 2. The user hits the ENTER key 3. The system processes the input (see EX1) 4. The input value for the number of seats is validated and stored in the system

Alternate Courses	None
Exceptions	EX1: Invalid User Input 1. The system displays in the interface that an invalid value was entered and that an integer greater than 0 should be entered 2. Return to Step 1 of the Main Course

4.2. User Selects Voting Algorithm

Name	User Selects Voting Algorithm
ID	UC_02
Description	The user indicates which voting algorithm that should be used for this election.
Actors	Election Official, Programmer
Organizational Benefits	Allows the system to know which voting algorithm to run so that the election results are tallied in the appropriate manner.
Frequency of Use	Once
Triggers	The system prompts the user to select the voting algorithm that should be used for the election.
Preconditions	● The system has not prompted the user to select the voting algorithm that should be used yet ● There is currently no value stored in the system for the voting algorithm that should be used for the election
Postconditions	● A value corresponding to the plurality algorithm or a value corresponding to the STV algorithm is stored in the system for the voting algorithm that should be used
Main Course	1. The user types the value corresponding to Plurality or the value corresponding to STV to indicate what voting algorithm should be used for the election 2. The user hits the ENTER key 3. The system processes the input (see EX1) 4. The input value for the voting algorithm choice is validated and stored in the system
Alternate Courses	None
Exceptions	EX1: Invalid User Input

	1. The system displays in the interface that an invalid value was entered and that the value corresponding to Plurality or the value corresponding to STV should be entered 2. Return to Step 1 of the Main Course

4.3. User Enters Ballot Files

Name	User Enters Ballot Files
ID	UC_03
Description	The user inputs the file paths for the ballot CSV files that hold the ballots to be tallied for this election and adds them to the list of files that the system needs to process.
Actors	Election Official, Programmer
Organizational Benefits	Allows the user to indicate which ballot files should be processed by the voting algorithm so that the election results are valid. Also allows for ballots to be distributed across multiple files.
Frequency of Use	Once
Triggers	The system prompts the user to input the ballot CSV files that should be tallied in the election.
Preconditions	● The system has not prompted the user to enter the file paths of ballot CSV files that should be tallied in the election yet ● There are currently no values stored in the system for valid file paths of ballot CSV files that should be tallied in the election
Postconditions	● At least one value for a valid file path of a ballot CSV file that should be tallied in the election is stored in the system ● No invalid file paths are stored in the system as files that should be tallied in the election ● No non CSV files are stored in the system as files that should be tallied in the election
Main Course	1. The user types the names of file paths of ballot CSV files that should be tallied in the election (separated by a single space character) 2. The user hits the ENTER key 3. The system processes the input file names (see EX1) 4. The input values for the file paths of ballot CSV files are validated and added to the list of ballot CSV files that need to be tallied in the election stored in the system

Alternate Courses	None
Exceptions	EX1: Invalid User Input 1. The system displays in the interface that invalid file paths were entered 2. The system displays in the interface which file paths that were entered were invalid 3. Return to Step 1 of the Main Course

4.4. User Runs Voting Algorithm

Name	User Runs Voting Algorithm
ID	UC_04
Description	The user tells the system to run the voting algorithm which runs the election, displays the results summary, and generates and saves an audit file for the election.
Actors	Election Official, Programmer
Organizational Benefits	Allows for a fast tallying of the ballots for an election in what would otherwise be a slow and tedious process that could be prone to human error or the accidental introduction of bias.
Frequency of Use	Once per election
Triggers	The user indicates they want to run the voting algorithm via a text command.
Preconditions	None
Postconditions	● Election results summary displayed in the interface to the user ● Audit file for the election generated and saved in the top level of the program ● User notified in the interface of the generation and location of the saved audit file for the election
Main Course	1. The system processes the user input command (see EX1) 2. The system confirms the run text command and begins the running process 3. The system prompts the user for the number of seats to be filled in the election (see UC_01) 4. The system prompts the user to select the voting algorithm to use for the election (see UC_02) 5. The system prompts the user to input the file paths of the ballot CSV files which contain the ballots that must be tallied in the election (see UC_03) 6. The system displays to the user the election parameters it has entered for the election 7. The system prompts the user to confirm the election parameters 8. The system processes the user input (see EX2)

	9. The parameters are confirmed and the system runs the indicated voting algorithm with the specified parameters and concurrently begins generating the election audit file 10. When the algorithm finishes running the audit file is saved in the same directory as the program executable 11. The system notifies the user of the generation and location of the saved audit file for the election 12. The system displays the election results summary
Alternate Courses	ALT1: The User Displays the Help Window 1. The user can display the help window at any time during steps 3-5 (see UC_06) 2. If this happens, the system will display the help information in the interface 3. The system will then resume where it left off in the running process
Exceptions	EX1: Invalid User Input 1. The system notifies the user that it does not recognize the text command given 2. The system continues to display whatever was displayed in the interface at the time of the invalid command EX2: User Cancels Election 1. User realizes election parameters are incorrect and does not confirm them 2. The system does not run the election, forgets the election parameters previously set, and restarts back at the initial startup screen

4.5. Programmer Runs Test of Voting Algorithm

Name	User Runs Voting Algorithm
ID	UC_05
Description	The user tells the system to run the voting algorithm which runs the election, displays the results summary, and generates and saves an audit file for the election.
Actors	Programmer
Organizational Benefits	Allows for testing to be done on the system and ensure that the system is calibrated properly
Frequency of Use	Once per election
Triggers	The programmer indicates they want to run a voting algorithm test via a text command.

Preconditions	None
Postconditions	● Election results summary displayed in the interface to the user ● Audit file for the election generated and saved in the top level of the program ● User notified in the interface of the generation and location of the saved audit file for the election
Main Course	1. The system processes the programmer input command (see EX1) 2. The system confirms the run text command and begins the running process 3. The system prompts the programmer for the number of seats to be filled in the election (see UC_01) 4. The system prompts the programmer to select the voting algorithm to use for the election (see UC_02) 5. The system prompts the programmer to input the file paths of the ballot CSV files which contain the ballots that must be tallied in the election (see UC_03) 6. The system asks the programmer if they would like to turn off the shuffle ballots option 7. The programmer indicates their choice via a text command 8. The system processes the programmer input command (see EX2) 9. The system displays to the user the election parameters it has entered for the election 10. The system prompts the user to confirm the election parameters 11. The system processes the user input (see EX3) 12. The parameters are confirmed and the system runs the indicated voting algorithm with the specified parameters and concurrently begins generating the election audit file 13. When the algorithm finishes running the audit file is saved in the same directory as the program executable 14. The system notifies the user of the generation and location of the saved audit file for the election 15. The system displays the election results summary
Alternate Courses	ALT1: The Programmer Displays the Help Window 1. The programmer can display the help window at any time during steps 3-7 (see UC_06) 2. If this happens, the system will display the help information in the interface 3. The system will then resume where it left off in the running process
Exceptions	EX1: Invalid User Input 1. The system notifies the user that it does not recognize the text command given 2. The system continues to display whatever was displayed in the interface at the time of the invalid command EX2: Invalid User Input

	1. The system notifies the programmer that it does not recognize the text command given 2. The system resumes again from step 6 of the main course EX3: User Cancels Election 1. User realizes election parameters are incorrect and does not confirm them 2. The system does not run the election, forgets the election parameters previously set, and restarts back at the initial startup screen

4.6. User Displays Help Window

Name	User Displays Help Window
ID	UC_06
Description	The user opens up the help window which contains information on how to use and interact with the different elements of the system.
Actors	Election Official, Programmer
Organizational Benefits	Gives the user a guide to how to use the system so that if they forget how to do something help is only a click away.
Frequency of Use	As many times as the user wants prior while the voting algorithm is not running.
Triggers	The user indicates they want to display the help window via a text command.
Preconditions	● The system is has not run the voting algorithm yet
Postconditions	● The help window is displayed in the interface
Main Course	1. The system processes the user input command (see EX1) 2. The system displays the help window information in the interface
Alternate Courses	None
Exceptions	EX1: Invalid User Input 1. The system notifies the user that it does not recognize the text command given 2. The system continues to display whatever was displayed in the interface at the time of the invalid command

5. Other Nonfunctional Requirements

5.1. Performance Requirements

Any election should process a maximum of 100,000 ballots in under 5 minutes.

5.2. Safety Requirements

Limited file access permissions will lead to failures of generating voting algorithm results.

5.3. Security Requirements

STV (Droop Quota)/Plurality Voting System and its ballotes files are sent to qualified election officials so there are no security requirements in the system.

5.4. Software Quality Attributes

STV (Droop Quota)/Plurality Voting System provides the users with both a typical voting feature and a ranked choice voting feature. Due to its well designed and easy to use interface it can be used by both experts and typical users. However, users must already have a basic knowledge of the two voting algorithms before using it.

5.5. Business Rules

Only programmers/testers have the privilege to turn off the Shuffle Ballots option and test the system.

6. Other Requirements

6.1. File Requirement

1. CSV file format requirement: the appropriate file will be a comma separated values (CSV) file saved as a Windows Comma Separated file (determines the end of the line signal we will need for our own testing) where each row is separated by a newline. The first line of the file will provide you with the names of the candidates and each subsequent line will be a voter's ballot.

2. File path requirement: the users need to upload the ballots in the same directory as the program to enable the programs to read the files. When the algorithm finishes running, the audit file is saved in the same directory as the program executable.

6.2. User Interface Training Requirement

Training for an election official to learn the user interface should take less than 15 minutes.

Appendix A: Glossary

Term	Definition
Election Official	This is the person that will be running the election
Programmer	This is the person who is testing the software to verify that it is running correctly
Plurality	Plurality voting is an electoral system in which each voter is allowed to vote for only one candidate, and the candidate who polls the most among their counterparts (a plurality) is elected.
STV	The single transferable vote (STV) is a proportional voting system designed to achieve or closely approach proportional representation through voters ranking candidates in multi-seat organizations or constituencies (voting districts).
Droop Quota	The Droop quota is the quota most commonly used in elections held under the single transferable vote (STV) system. It is also sometimes used in elections held under the largest remainder method of party-list proportional representation (list PR). In an STV election the quota is the minimum number of votes a candidate must receive in order to be elected. Any votes a candidate receives above the quota are transferred to another candidate.

Appendix B: Droop Quota Algorithm

The Droop Quota algorithm is as follows:

- Step 1: Shuffle the ballots.
- Step 2: Calculate the Droop Quota:

$$\left[\left(\frac{numberOfBallots}{numberOfSeats+1}\right)\right] + 1$$

- Step 3: Distribute the ballots one-at-a-time into piles for the candidates. (The piles are ordered by the first vote received, not alphabetically.) If any candidate has reached the quota (for example, 22 ballots), they are immediately declared elected, those ballots are removed permanently from the process, and the elected candidate's name is recorded (in the order elected, not alphabetically). Subsequently, any ballot that follows in the distribution with their name listed goes to the next name on the list.
- Step 4: After the first round of distributing the candidates, the candidate with the fewest number of votes is permanently eliminated and their ballots are redistributed. (In the case of a tie, the candidate who was last to receive their first ballot is eliminated.) As in the first round, if any candidate reaches quota they are immediately declared elected, etc. There may be multiple rounds repeating this process.
- Step 5: You will continue this process until all ballots have been processed. You will have two lists, the elected and the non-elected. These lists will be ordered based on when they were elected and when they were placed on the non-elected list. It is possible for a "winner" of a seat to be on the non-elected

list if they did not reach quota. The candidates placed on the non-elected list last will be higher up on the election list itself. You must provide both lists to the user.

- Step 6: The user wants a report that acts as an audit for the election. This report should show the ballots that were assigned to a candidate as the election progressed. The report does not print to the screen but should be in a text file. All pertinent election data must be included in the report at the top of the report (e.g. type of election, number of seats, number of candidates, winners, losers, etc).

Appendix C: Plurality Voting Algorithm

The plurality algorithm should work as follows:

- Each ballot will be assigned to a candidate and a count maintained for each candidate.
- The candidate(s) with the greatest number of votes will be declared the winner(s).
- If there is a tie for a given seat, you will randomly select the winner of that seat.