Software Requirements Specification

for

a Straightforward Voting System

Version 1.0

Prepared by Amir Jalili (jalil014), Arun Sharma (sharm485), Nicholas Lovdahl (lovda015), Taylor O'Neill (oneil569)

Team 2

02/21/2020

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Amir Jalili, Arun Sharma, Nicholas Lovdahl, and Taylor O'Neill	2/21/20	Completion of the initial Software Requirements Specification document.	1.0

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detail of system and user requirements for the Straightforward Voting System (SVS). The Straightforward Voting System is meant to tally and determine the results of elections from provided ballots given as CSV files. SVS can operate both plurality and single transferable voting (using the Droop Quota) based elections. This system should assist election officials in determining the winners and losers of such elections in a manner that is quick, fair, and accountable.

1.2 Document Conventions

This document was created by following the IEEE format for System Requirement Specification documents. Currently, all system features have equal priority due to their interlocking nature; all those features listed in this document are necessary for running elections and as such cannot be partially present in a functioning product.

1.3 Intended Audience and Reading Suggestions

This document is primarily intended for the developers of the project team working on SVS. Its contents should inform as to what the project is and what the desires of the client are. However, the contents of this document should also be able to demonstrate to the client that their desires have been understood correctly and in their entirety. As such, this document outlines the functional and non-functional requirements for SVS, its interfaces, and the conditions under which it will be operating. The writing should be understandable to those outside of the development of SVS.

However, this does not imply that this document is directed for reading by the Election Officials in general, the primary users of SVS. This document contains details, such as the ability to disable the shuffling of ballots for testing purposes, whose use could compromise the integrity of real elections. Separate documentation should be provided to the users of SVS which contains only the details necessary to successfully operate the software. Separate documentation detailing the implementation of SVS should also be prepared and available to the clients.

Proceeding with this document, it is recommended that it be read from start to finish during a first read. This will provide a lense through which to understand the intent behind SVS before moving on to its precise specifications. Future readings, especially those by developers, may place greater emphasis on particular sections of this document for consultation on particular features or requirements. The Table of Contents may be referenced to locate these sections.

1.4 Product Scope

SVS is a software that allows election officials to conduct elections based on either plurality voting or a single transferable voting system using the Droop quota. During elections, ballots are collected indicating a voter's selection of candidate(s). SVS will then be given these ballots, the number of seats available, and the type of election, and determine the winners and losers. SVS will also produce an audit file in addition to these results. This audit file will provide a record of SVS's action in determining the results of the election. SVS will ease the burden on election officials by automating the tallying of ballots. It will also aid in removing human error from the election process, increase the speed at which election results arrive, and provide for a reliable and transparent election process.

1.5 References

This section lists resources referenced by this document. The titles of these resources are given first, along with details to be used to locate them.

- IEEE Template for System Requirement Specification Documents: This template was developed by Karl Wiegers in 1999.
- Single transferable vote: An article in Encyclopaedia Britannica. This article was created by the editors of Encyclopaedia Britannica, published by Encyclopaedia Britannica, Inc. on April 21, 2017, and accessed on February 21, 2020. This article can be accessed with the following url: https://www.britannica.com/topic/single-transferable-vote.

2. Overall Description

2.1 Product Perspective

Elections involve the tallying of numerous ballots before determining the results of an election. This process can be both time-consuming and prone to human error, especially in elections where a large number of ballots must be accounted for. These problems can be compounded for more complicated systems of voting, such as in a single transferable vote election where determining the winners and losers of an election comes as the result of several passes of distributing and redistributing ballots amongst candidates. However, these processes can be automated, and doing so helps election officials to alleviate these problems - this is the purpose of SVS.

SVS is a stand-alone product capable of running two different types of election: plurality, and single transferable vote with the Droop quota. Users interact with SVS through a graphical user interface, select all the CSV files containing information from the ballots, enter some additional information about the election (i.e. the type of election and number of seats being voted on), and then run SVS. SVS carries out the tallying of ballots and returns the results of the election: the winners and losers, statistics about the election, and an audit file to account for the steps taken by SVS to arrive at these results. This will provide for a speedy, accountable election process which is less prone to human error.

2.2 Product Functions

- Reading ballots compile in CSV files: Ballots are given to SVS in the form of a number of CSV files. SVS reads these files to determine the results of an election.
- Running Plurality Elections: Users can run plurality elections after providing the number of seats available and the files that contain the ballots. SVS will determine the winners and losers of the election.
- Running Single Transferable Vote Elections with the Droop Quota: Users can run single transferable vote with Droop Quota elections by providing the number of seats available and the files that contain the ballots. SVS will determine the winners and losers of the election.
- Election Statistics: SVS will display statistics related to an election to the users. These statistics will be relevant to the type of election being run.
- Audit Files: SVS will create a file detailing each step SVS takes when determining the outcome of an election. This file may be read to verify that SVS has taken the correct steps.
- Deterministic Testing: For the purposes of testing and evaluation, SVS provides an option to disable the shuffling of ballots before they are tallied. This allows for deterministic testing. This can be invoked using a flag to disable shuffling as a command line argument.

2.3 User Classes and Characteristics

SVS has two anticipated groups of users: Election Officials and Testers. These user groups have differing knowledge about the functionalities of SVS - Testers will be aware of the option to disable the shuffling of ballots through the command prompt, but Election Officials will not, for example. Testers are likely to be involved in the development of SVS, or have complete access to its documentation. They may use this knowledge to verify that SVS is operating according to its requirements through tests and the examination of its implementation.

Election Officials, however, are the primary users of SVS. Election Officials will not have access to all of SVS's documentation (just the documentation relevant to them) and do not necessarily have any particular technical expertise or familiarity with SVS or even computer systems. Election Officials will have access to the system features related to running an election and these features must be simple enough to be used in these cases. Since Election Officials will be the class of users operating SVS in real-life elections, the quality and ease-of-use of their use cases are the most important.

2.4 Operating Environment

SVS is a Java program. This means that SVS will be implemented using the Java programming language and will run on a machine with an implementation of the Java Virtual Machine. This provides a level of abstraction that will allow SVS to run on many different systems, although these systems must have certain abilities. SVS must be run in an environment that has the ability to store and load files, an graphical interface that can support windows, the ability for users to interact with these windows through a keyboard and mouse, and reasonable amount of memory and processor speed to run the Java Virtual Machine and SVS. The requirements to run the Java Virtual Machine vary, but are available on most common operating systems (Windows, OSX, Linux).

2.5 Design and Implementation Constraints

The client will be responsible for ensuring that SVS is run in an environment satisfying the requirements in the previous section (Section 2.4, Operating Environment). Most, if not all personal computers will fulfil these requirements. Even relatively cheap devices often have at least 2GB of memory and a processor running at 1GHz or faster. However, the Java Runtime Environment may need to be setup on these devices if it is not already installed. The latest version of the Java Runtime Environment should be installed on any system on which SVS is to be run.

An additional constraint related to the Java Virtual Machine is that SVS will make use of the high-quality random number generation. Most implementations of the Java Virtual Machine use information generated by the operating system for this random number generation. The Java Virtual Machine will block SVS if it is made to wait for this information. This may result in SVS requiring additional time to determine the results of elections. Generally, this should not be a problem on devices like personal computers, but the client should be aware of this potential issue and should ensure that SVS is not run on any device which is likely to encounter this issue. The client will be responsible for making this determination.

2.6 User Documentation

Two sets of documentation should be prepared, one for each user group; this should allow users to more easily access documentation which is relevant to them. Testors should have access to the full documentation of SVS. Election Officials, however, are only concerned with operating SVS in a limited fashion (e.g. they should not know how to disable ballot shuffling). The documentation for Election Officials should be relatively simple to understand and should not assume that Election Officials have any particular technical expertise or knowledge of buzzwords.

In addition to this prepared documentation, SVS will be able to provide a help window. While running SVS, all users will have access to this window, which will provide them with some instructions on how to understand and successfully operate SVS.

2.7 Assumptions and Dependencies

It is assumed that there are no errors (e.g. incorrect formatting, an incorrect number of rankings, duplicate rankings, different numbers of candidates in the files) in the CSV files with the ballots. Ballots for plurality elections must have only one candidate selected. Ballots for a single transferable vote election must have at least half of the candidates ranked. It is also assumed that the end of line signals in the file follow the format used by Windows. Ballots will be preprocessed before delivery and use by SVS and no error handling will be performed. Any mistakes with the ballots or their formatting may result in SVS crashing or operating incorrectly.

It is assumed that SVS will receive more than one-hundred thousand ballots. Additional ballots may require additional time for SVS to determine the results of an election outside of its timing requirements. It is also not guaranteed that SVS will not fail or make the correct determination in such a scenario.

SVS is a stand-alone product and will not use any components developed in any previous projects. SVS will, however, require the Java Runtime Environment to be installed as mentioned in Section 2.4, Operating Environment.

3. External Interface Requirements

3.1 User Interfaces

SVS will provide a simple GUI with the following features in its main window:

- A button to launch a file chooser to select the CSV files containing the ballots.
- A drop down menu to select the type of election to run.
- An input field to provide the number of seats available.
- A button to determine the results of the election.
- A section where statistics from the election are displayed. This will include winners and losers along with other statistics relevant to the type of election run.

In addition, SVS will provide a help window. This window can be launched from a menu option in the main window. The help window will have information for the user to understand and successfully operate SVS.

3.2 Hardware Interfaces

SVS is a stand-alone software product and will contain no interface with particular hardware devices. Since SVS is a Java program running on the Java Virtual Machine, any interface with the hardware running SVS will be abstracted.

3.3 Software Interfaces

SVS does not interface with any other software products, excepting the Java Virtual Machine on which it runs. The Java Virtual machine abstracts SVS's interfaces with the system running it. For example, the interfacing performed with the operating system to read the ballot files is carried out by the Java Virtual Machine on behalf of SVS.

3.4 Communications Interfaces

There are no communication requirements associated with SVS. The ballot files which SVS is to use must be transferred to the system running it outside of the scope of SVS; SVS itself can be run without any access to the Internet or other systems.

4. System Features

4.1 Disable Ballot Shuffle

Name	Disable Ballot Shuffling
ID	UC_01
Description	The shuffling of ballots will be disabled so that the testers can verify that the Droop Quota algorithm functions correctly.
Actors	Users in the Tester User Group.
Organizational Benefits	This feature will allow the tester group to verify the Droop Quota algorithm, ensuring that the system works properly before it is used by election officials
Frequency of Use	This use case will be invoked as many times as the Tester feels is necessary. It is estimated that this will be a series of at least a dozen invocations for throughout testing.
Triggers	The tester launches SVS through a command line or terminal with “-ds” as an argument.
Preconditions	SVS has been launched with the “-ds” flag.
Postconditions	SVS will launch, and the algorithm to run a single transferable vote with the Droop quota will not shuffle the ballots if that election type is selected.
Main Course	1. The tester adds the text “-ds” as an argument to the command line call that runs the SVS. 2. The tester executes the command. Ballots will not be shuffled for the duration of the program.
Alternate Courses	N/A
Exceptions	N/A

4.2 Load Ballot Files

Name	Load Ballot Files
ID	UC_02
Description	The user is prompted to select which CSV files will be used in the system's voting algorithm.
Actors	All users.
Organizational Benefits	This feature allows election officials to easily run the voting algorithm on particular ballots
Frequency of Use	This use case will be invoked each time a user changes the selection of files that SVS will use to determine the results of an election.
Triggers	The user pushes the button to pick the files for SVS to read.
Preconditions	N/A
Postconditions	SVS reads the ballots into memory for later use when an election is run.
Main Course	1. The system prompts the user to select the CSV files to load through the file chooser (see AC1). 2. The system loads the files selected by the user into memory.
Alternate Courses	AC1: User quits their attempt to load ballot files: 1. The system does nothing.
Exceptions	N/A

4.3 Select Election Type

Name	Select Election Type
ID	UC_03
Description	The user is prompted to select which election type or voting algorithm (Plurality or Droop Quota) they want to run.
Actors	All users.
Organizational Benefits	This feature allows the user to choose between plurality or Droop Quota.
Frequency of Use	This use case will be invoked after the ballot file successfully loaded and every time when election officials need to run a new voting algorithm.
Triggers	The user clicks with the drop down menu for the type of election.
Preconditions	N/A
Postconditions	The voting algorithm that the user would like to run is selected.
Main Course	1. The system prompts the user to select an election type from a drop down menu (see AC1). 2. The system changes the election type to the user's selection. The user interface is adjusted to match the election type chosen.
Alternate Courses	AC1: User quits their attempt to select an election type: 1. The system does nothing. The previously chosen election type is kept.
Exceptions	N/A

4.4 Enter Number of Seats to Fill

Name	Enter Number of Seats to Fill
ID	UC_04
Description	The election official enters the amount of seats that need to be filled in the election.
Actors	All users.
Organizational Benefits	This feature will ensure that the system correctly processes the results of the election.
Frequency of Use	One time per election.
Triggers	When the system is ran, it automatically prompts the user to enter how many seats need to be filled in the election
Preconditions	The system is running, but no information has been entered yet.
Postconditions	The system knows how many seats need to be filled in the election.
Main Course	1. The user types in the number of seats that need to be filled for the given election. 2. The user hits the 'Enter' key.
Alternate Courses	N/A
Exceptions	N/A

4.5 Run Plurality Election

The use case for running an election that uses the plurality algorithm is displayed in the table below. The specific steps that the plurality algorithm will follow are:

1. A count will be maintained for each candidate. This count is initialized to be zero and will indicate the number of votes the candidate has received.
2. All of the ballots are read through once. A candidate's count is incremented if they are selected in a ballot.
3. After all of the votes have been counted, the candidate with the largest vote count will be declared a winner. This step repeats while for each available seat, picking the candidate with the next most votes. If there is a tie in a vote count for a given seat, the winner will be chosen at random.

Name	Run Plurality Election
ID	UC_05
Description	The user prompts the system to begin running a plurality election on the ballots submitted to the system
Actors	All users.
Organizational Benefits	The results of the plurality election are quickly processed by the system.
Frequency of Use	One time per election that was designated to use the plurality algorithm.
Triggers	The system will automatically prompt the user to run the election after the user has finished selecting which files hold the ballot results.
Preconditions	The user selected the plurality algorithm to process the election results.
Postconditions	Two seperate lists of elected and non-elected candidates, with pertinent results, will be displayed on the screen.
Main Course	1. The user clicks on the “Run Election” button. 2. The system computes the results of the election based off of the rules of the plurality algorithm.
Alternate Courses	N/A
Exceptions	N/A

4.6 Run Droop Quota Election

The use case for running an election that uses the Droop Quota algorithm is displayed in the table below. The specific steps that the Droop Quota algorithm will follow are:

1. The ballots will be shuffled first.
2. The formula shown below is used to calculate the Droop quota:

$$Droop\ Quota\ = \lfloor \frac{Number\ of\ ballots}{Number\ of\ Seats + 1} \rfloor + 1$$

3. The shuffled ballots will be selected one-at-a-time. Each selected ballot will be placed in a pile that corresponds to the candidate that was ranked as the top choice on the ballot.
4. If a candidate's pile reaches the Droop Quota, they will automatically be declared elected. The ballots in that candidates pile will be removed from the process. The name of the candidate will be placed in the first location of the “Elected List”. As the process continues, any selected ballot that has a candidate's name that is already on the “Elected List” as the

top choice will instead be placed in the pile of the next ranked candidate that is not on the "Elected List".

5. After the first round of dispersing the ballots that were in the first candidates pile, the candidate with the smallest pile will be removed from the election and placed on the "Non-Elected List". The ballots in the removed candidates pile will be redistributed. In the case of a tie, the candidate that received a ballot in their pile last will be eliminated.
6. This process will repeat until all of the ballots have been processed. After the algorithm completes, the "Elected List" and "Non-Elected List" will be displayed to the user.
7. An audit report will be created for the user and stored in a text file. The audit report will show how the ballots were assigned to different candidates as the process continued. The top of the report will state the type of election, the number of seats to be filled, the number of candidates, the winners, and the losers.

Name	Run Single Transferable Vote with Droop Quota Election
ID	UC_06
Description	The user prompts the system to begin running a Droop Quota election on the ballots submitted to the system
Actors	All users.
Organizational Benefits	The results of the Droop Quota election are quickly processed by the system.
Frequency of Use	One time per election that was designated to use the Droop Quota algorithm.
Triggers	The system will automatically prompt the user to run the election after the user has finished selecting which files hold the ballot results.
Preconditions	The user selected the Droop Quota algorithm to process the election results.
Postconditions	Two seperate lists of elected and non-elected candidates, with pertinent results, will be displayed on the screen. A text file will also be generated containing the audit report.
Main Course	1. The user clicks on the "Run Election" button. 2. The system computes the results of the election based off of the rules of the Droop Quota algorithm.
Alternate Courses	N/A
Exceptions	N/A

4.7 Open Help Window

Name	Open Help Window
ID	UC_07
Description	The user opens up the help window.
Actors	All users.
Organizational Benefits	Being able to access a help window provides users with additional information for them to understand how to use SVS. This will allow users who are confused or have questions to successfully operate SVS.
Frequency of Use	This use case may be triggered an arbitrary number of times while SVS is running.
Triggers	The user selects a button / menu option to display the help window.
Preconditions	The help window is not already being displayed.
Postconditions	The help window is displayed.
Main Course	1. The system checks whether the help window is already being displayed, if this is the case the system does nothing. 2. The system displays the help window to the user.
Alternate Courses	N/A
Exceptions	N/A

5. Other Nonfunctional Requirements

5.1 Performance Requirements

SVS must be capable of determining the results of elections in less than five minutes.

5.2 Safety Requirements

In order to preserve the integrity of elections, SVS must not write to any of the ballot files that it reads. No alterations to these files should occur, even if SVS crashes while reading them or determining the results of an election.

5.3 Security Requirements

There are no security requirements for SVS. The ballot files are sent to it through means outside of its scope by Election Officials in a secure manner.

5.4 Software Quality Attributes

Correctness and usability are the primary concerns for SVS. Without these properties, SVS may return incorrect results for elections, damaging trust in such events and elected officials. SVS must also be able to be used by Election Officials effectively with only 15 minutes of training. If Election Officials cannot understand how to use this software product this could also lead to incorrect operation.

5.5 Business Rules

Since the election officials are the only users running the program, there will not be any business rules or roles implemented within the system. Any laws or regulations, local, state, or federal, related to conducting elections are the responsibility of the client to adhere to.

6. Other Requirements

No other requirements apply to SVS. All those requirements are detailed in other sections of this document. Consult the Table of Contents to locate particular sections.

Appendix A: Glossary

This section contains a list of terms used in the document which may require further explanation.

- SRS: The acronym for “Software Requirements and Specifications”, a document which lays out the requirements and specifications that a software product should conform to. In the context of this project, that refers to this document.
- SVS: The acronym for “Straightforward Voting System”, the name of the product being developed and the subject of this document.
- CSV files: CSV is an acronym for “Comma Separated Values”. CSV files are a file type describing a text file which contains a number of values separated by commas. Such files are used to store and organize data, such as values in a spreadsheet, for example.
- Plurality election: A plurality election is a simple voting system where each voter chooses a single candidate on their ballot and the candidates with the most votes win. Should a tie occur in this system, it would be broken by randomly selecting a tied candidate as a winner.
- Single transferable vote election: A single transferable vote election is a voting system in which voters rank candidates based on their preferences. These ballots can then be used to determine the winners and losers based on the collective preferences of the voters. For more information on this system, consult the Encyclopaedia Britannica article on “Single transferable vote” listed in the References section.

- Droop Quota: The Droop Quota is a quota used in some single transferable vote elections to determine the threshold of votes at which point a candidate can be declared a winner. Any votes after a candidate has been declared a winner are then provided to the voter's next ranked choice (if they have one).

Appendix B: To Be Determined List

Initial elicitation with the client has been completed; there are no further questions related to SVS at the present time of writing this document.