Software Requirements Specification

for

Voting System

Version 1.0 approved

Prepared by Team 3 (Hailin Archer - deak0007, Bryan Baker - bake1358, Colin Kluegel - klue0037, Josh Spitzer-Resnick - spitz123)

CSCI 5801 - Software Engineering I, Spring 2020

University of Minnesota

February 21, 2020

Table of Contents

1. Introduction **1**
1.1 Purpose 1
1.2 Document Conventions 1
1.3 Intended Audience and Reading Suggestions 1
1.4 Product Scope 1
1.5 References 2

2. Overall Description **2**
2.1 Product Perspective 2
2.2 Product Functions 2
2.3 User Classes and Characteristics 3
2.4 Operating Environment 3
2.5 Design and Implementation Constraints 4
2.6 User Documentation 4
2.7 Assumptions and Dependencies 4

3. External Interface Requirements **4**
3.1 User Interfaces 4
3.2 Hardware Interfaces 6
3.3 Software Interfaces 6
3.4 Communications Interfaces 6

4. System Features **7**

5. Other Nonfunctional Requirements **7**
5.1 Performance Requirements 7
5.2 Safety Requirements 7
5.3 Security Requirements 7
5.4 Software Quality Attributes 7
5.5 Business Rules 7

6. Other Requirements **9**

Appendix A: Glossary **9**

Appendix B: Analysis Models **10**

Appendix C: To Be Determined List **11**

Revision History

Name	Date	Reason For Changes	Version
Archer, Baker, Kluegel, Spitzer-Resnick	2/21/2020	Final version	1.0
Archer, Baker, Kluegel, Spitzer-Resnick	2/20/2020	Secondary revision	0.2
Archer, Baker, Kluegel, Spitzer-Resnick	2/18/2020	Preliminary revision	0.1
Spitzer-Resnick	2/17/2020	UC_3, UC_7, 1.5, 2.4, 3.1, 5.3, Appendix A	0.07
Kluegel	2/16/2020	1.4, 2.3, 2.7, 3.4, 5.2, 6	0.06
Baker	2/16/2020	1.1, 1.2, 1.3, 2.2, 2.6, 3.3, 5.1, 5.5	0.05
Archer	2/16/2020	2.1, 2.5, 3.2, 5.4, Appendix B	0.04
Baker	2/15/2020	UC_1, UC_5	0.03
Kluegel	2/15/2020	UC_2, UC_6	0.02
Archer	2/14/2020	UC_4, UC_8	0.02
Archer	2/13/2020	Cover page	0.01

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the *Voting System (VS) Version 1.0*. It will explain the purpose and features of the software, the interfaces of the software, what the software will do and the constraints under which it must operate. This document is intended for users of the software and also potential developers.

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 intended audience includes:

- Election officials who want to use the VS to run an election with supplied ballot information.
- Election officials and system testers who want to test and calibrate the VS.
- Programmers who are interested in working on the project by further developing it or fixing existing bugs.

1.4 Product Scope

The software system being developed is a voting system to be used in local elections. The system will be designed to automate the counting of ballots to simplify the running of elections. The main feature of the software will be to run two types of elections, a plurality voting election and a single transferable voting (STV) election.

In addition to its primary purpose of running an election the software will need to provide some additional features. The software needs to display detailed information about the election results, that is, it should display the number of ballots, the number of seats, the number of candidates and the winner /winners of the election. The software will also need to create a detailed report that will act as an audit for the election. The report will be saved as a text file and show details about how ballots were assigned to candidates as the election progressed.

The software will also require a diagnostic mode. The diagnostic mode will be entered using a command line option. The diagnostic mode is required to support an option to disable ballot shuffling so the system can be calibrated. The diagnostic mode will also have options for developers to use to debug the software.

To aid the user of the voting system a help window will be provided that will give the user information about how to run the program.

1.5 References

"Project 1 - Waterfall Methodology: Software Requirements Specification (SRS) Document for Voting System", Watters, Feb 2020:
https://canvas.umn.edu/courses/158173/assignments/1012061?module_item_id=3578775

"UseCases_Team3", Archer, Baker, Kluegel, and Spitzer-Resnick, Feb 21, 2020:
https://docs.google.com/document/d/1Rn_KQWHovoyd1n85b0jzUSr-Kml9FoitqQ3spQduIxk/edit

"IEEE Software Requirements Specification Document Template", Wiegers, 1999:
https://canvas.umn.edu/courses/158173/pages/software-requirements-specification-supporting-documents?module_item_id=3541620

2. Overall Description

2.1 Product Perspective

Voting system is a new system capable of performing both plurality voting and an STV system using the Droop quota. It is a stand alone program. Fig 1 shows major components of this system and how they interact with each other.

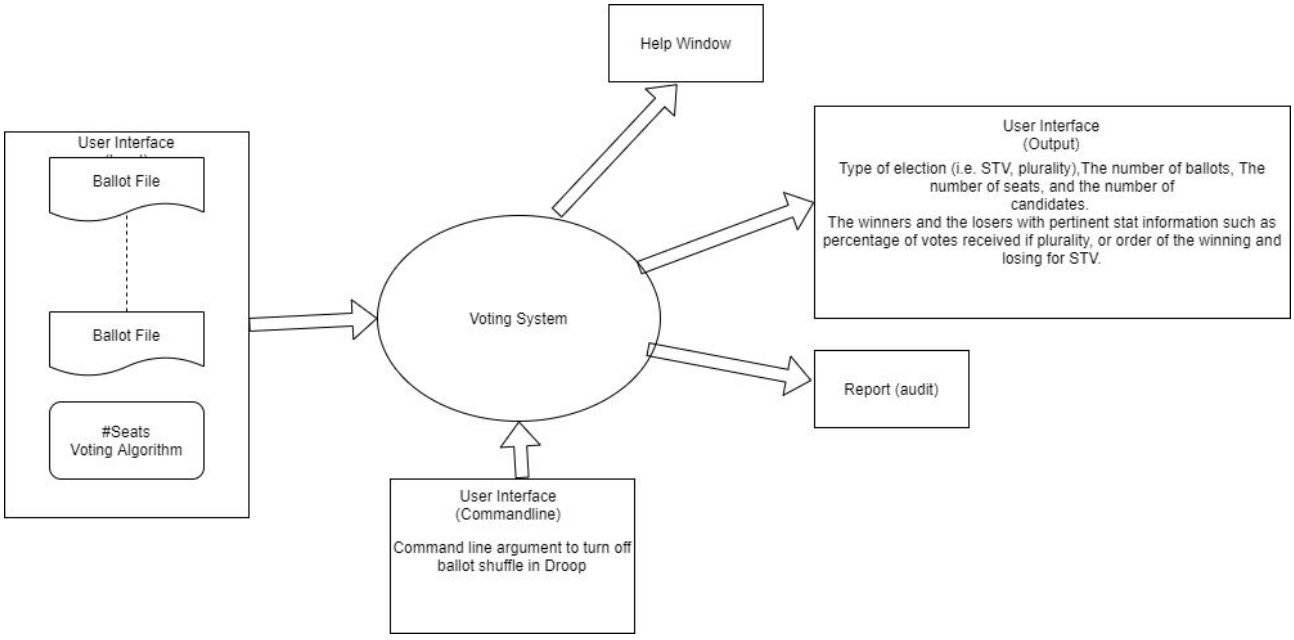


Fig. 1 Major Components of the Voting System

2.2 Product Functions

The system shall be able to:

- Accept user inputs including:

    - The number of election seats to fill
    - The type of election to run
    - The file location for the election ballots
- Import election ballots
    - Provide the ability to shuffle the election ballots
- Run an election
    - Run a plurality election type
    - Run an STV election type
    - Provide the ability to break ties in an election
- Provide election results
    - Provide elected and non-elected candidate lists
    - Calculate election statistics
    - Provide election audit file
- Provide the ability to calibrate or test the system under repeatable circumstances
    - Provide the ability to turn off the election ballot shuffle
- Provide a help window for the user

2.3 User Classes and Characteristics

There are two primary classes of users for the voting system:

Election Official

- **Definition**: The election official is in charge of setting up and running the election.
- **Functionality Required**: The election official needs to be able to set up the election by specifying the election type (STV/plurality), the number of seats, and inputting the ballot files. After the election is run, the election official will need to be able to see the results of the election.
- **Security Required**: None

Developer/Tester

- **Definition**: A tester is a user who checks the software to make sure that it is properly calibrated and is outputting the correct results. A tester should also have the ability to use command line arguments to test the program.
- **Functionality Required:** A tester needs to be able to set the software into calibration mode by disabling the ballot shuffling functionality. Once they have entered the software into calibration mode they need to be able to run an election. A tester will also need to be able to understand how to use the command line interface for the voting system.
- **Security Required:** None

2.4 Operating Environment

The voting system application will be capable of running on the Linux operating systems as found on CSE lab machines at the University of Minnesota - Twin Cities. (for hardware requirements, see section 3.2)

2.5 Design and Implementation Constraints

The voting system shall be written in C++. It shall run on a CSE labs computer. Testers shall be able to build the program with a single makefile. Running of a plurality or STV election with 100,000 ballots shall be completed in under 5 minutes.

2.6 User Documentation

An help screen will be provided for the user in the VS. This window will provide information on how to operate within the system. Comments will be included in the source code and a user manual will be offered.

2.7 Assumptions and Dependencies

The voting system assumes that the ballot files are correctly marked and in the correct format. The system also assumes that the election is set up correctly, when a plurality election is selected the ballots are plurality ballots and when a STV election is selected the ballots are STV ballots. When an election is set up the number of candidates should always be greater than the number of seats. There is no file naming convention for the ballot files. The system will always have enough space to generate an audit file.

3. External Interface Requirements

3.1 User Interfaces

The following interfaces shall be implemented in the voting system:

- User input screen
- Output screen
    - plurality election
    - STV election
- Output report (audit) file
    - The audit report is a text file in the same directory as the program executable and contains the ballots, the order in which they were shuffled (if STV), the order of candidates being assigned ballots, the order in which candidates are elected, results of any ties, the order of losing candidates
- Command line argument to turn off shuffling
- Help window

User input screen

```
/---------
|       Input number of seats in election: ___
|       Chose STV or plurality election: [dropdown]
|       Specify names of ballot files: ___
|                                       [Add another]
|       Click here to run election and display results
|       Click here to display help menu
\---------
```

Output screen - plurality election

```
/---------
|       Type of election:               Plurality
|       Number of ballots:              [#]
|       Number of seats:                [#]
|       Number of candidates:           [#]
|
|       Winning candidates, descending order:   A       B       C
|       Votes:                                  [#]     [#]     [#]
|       Percentage of total votes:              [#]%    [#]%    [#]%
|       Losing candidates, ascending order:     F       E       D
|       Votes:                                  [#]     [#]     [#]
|       Percentage of total votes:              [#]%    [#]%    [#]%
|
|       Location of audit report:       /path/audit.txt
\---------
```

Output screen - STV election

```
/---------
|       Type of election:               STV
|       Number of ballots:              [#]
|       Number of seats:                [#]
|       Number of candidates:           [#]
|       Droop Quota:                    [#]
|
|       Winning candidates, descending order:   A       B       C
|       Losing candidates, ascending order:     F       E       D
|       Number of invalidated ballots:          [#]
|
|       Location of audit report:       /path/audit.txt
\---------
```

Command line argument to turn off shuffling

```
$ > ./voting_system stv_test_file.txt -D SHUFFLE_OFF
```

Help window

```
/---------
|       Welcome to the Voting System 1.0 ~/~/Retro/~/~
|               (C) 2020 Archer, Baker, Kluegel, Spitzer-Resnick
|
|       How to get started:
|       1. Open your command line terminal on your Linux system
|       2. Type "./voting_system" and press enter
|
|       To calibrate the voting system:
|       1. Contact your IT department to enter the appropriate commands
|
|       How to run an election:
|       1. In the input screen:
|               a. enter the number of seats candidates may fill in this election
|               b. choose to run a plurality or STV election by choosing one from the
|                       dropdown menu
|               c. specify the name of your ballot files
|                       i. to add additional ballot files, click [Add another]
|               d. click "Click here to run election and display results"
|       2. Wait momentarily (up to 5 minutes) for all computations to have finished
|       3. When all computations have finished, a summary of the results with appropriate
|               statistics for the type of election you have chosen will appear on the screen,
|               as well as the location of the audit report file
|       4. Close the program
|       5. To run a new election, start the system again and repeat the above steps
\---------
```

3.2 Hardware Interfaces

The voting system is a stand alone program designed to run on CSE labs computers.

3.3 Software Interfaces

The following software shall be used:

- The VS shall be programmed in C/C++ version 7.4.0 as provided on CSE lab machines currently running Ubuntu 18.04.4 LTS and using the Linux 4.15.0-76-generic x86_64 kernel.
- User input shall be through a standard keyboard either on site or through a proxy (for example SSH).
- The VS shall read Windows standard comma separated (CSV) files.
- Audit reports shall be written to a standard text file.
- All user interfaces including input requests and help systems will be provided in a text based format.

3.4 Communications Interfaces

There are no communication interfaces required for the voting system.

4. System Features

System features are represented by use cases. Please refer to document UseCases_Team3.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system shall operate given the following function and performance constraints:

- Training for the user interface shall take less than 15 minutes.
- The election shall be processed in under 5 minutes.
- The VS shall run on a CSE lab machine from the command prompt.
- The VS shall be able to process a minimum of 100,000 ballots.

5.2 Safety Requirements

There are no safety requirements for the voting system.

5.3 Security Requirements

The voting system can only be used by election officials and testers. There are no user identity authentication, security, or privacy requirements.

5.4 Software Quality Attributes

The voting system provides the users with both simple and advanced features. Due to its well designed and easy to use interface it can be used by both experts and typical users. Voting system provides a help window where users can get instruction on how to use the system. However, users must already have a basic knowledge of Linux before using it.

5.5 Business Rules

Ballot rules:

- Plurality ballot rules
    - The first line of the file will provide the names of the candidates and each line after that will be a voter's ballot.
    - Each ballot will have only one candidate indicated as their choice. This will be indicated by the number "1".
    - An example of a ballot file provided for a plurality election is:
      A,B,C,D,E,F
      1,,,,,
      ,,1,,,

,,,,,1

- STV ballot rules
    - The first line of the file will provide the names of the candidates and each line after that will be a voter's ballot.
    - Each ballot must have at least half of the candidates ranked.
    - Each ballot will have numbers listed for each candidate in order of preference. A "1" is the first choice, a "2" is the second, etc. A ballot does not have to rank all candidates.
    - An example of a ballot file provided for an STV election is:
      A,B,C,D,E,F
      1,,2,,3,
      3,2,1,4,6,5
      1,2,,,3,4
      4,1,,2,,3

Plurality election rules:

- 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 are ties in an election the winner will be chosen from the pool of tied candidates.
- If there are seats to fill but the remaining candidates have no votes, winners will be chosen randomly.
- An election audit report shall be provided.

STV election rules:

- The STV election will be conducted using the Droop Quota algorithm as carried out using the following steps:
    a. Shuffle the ballots.
    b. Calculate the Droop Quota:

$$\left\lfloor \left(\frac{NumberOfBallots}{NumberOfSeats + 1} \right) \right\rfloor + 1$$

    c. Distribute all ballots
        - One-at-a-time to each candidate ordered by first vote received.
        - Any candidate who reaches the quota defined above is declared a winner and the ballots are removed from the election. The candidate is added to the winner list in the order elected.
        - Any ballot with a vote for a winning candidate will go toward the next candidate on that ballot's list.
    d. Redistribute ballots from losing candidates
        - Candidates with the fewest number of votes are removed from the voting pool and placed on the losing list.
        - In the case of a tie for the losing position, the candidate who was last to receive their first ballot is eliminated.
        - If, upon redistribution, a candidate receives the quota they will be added to the winning list.
    e. Continue until all ballots have been processed

- There will be two lists upon completion: a winning list and a losing list.
- These lists will be ordered based on when they were elected and when they were placed on the non-elected list.
- The candidates placed on the non-elected list last will be higher up on the election list itself.
- Both lists will be provided to the user.

f. Provide an election audit report
- Show the ballots that were assigned to a candidate as the election progressed.
- All election data must be included in the report at the top of the report. This includes: type of election, ballots shuffling order, number of seats, number of candidates, winners, losers, etc.
- This report should be output to a text file.

Election officials may:
- Input the number of seats to fill in an election
- Input the file location for election ballots
- Run a plurality type election
- Run an STV type election
- Review election results
- Perform election audits
- Ask for help through a help window
- Turn off the ballot shuffle option to test for system calibration

Developers / Testers may:
- Perform all activities described for election officials
- Use command line arguments to facilitate testing and development.

6. Other Requirements

Once the election starts, it must run to completion.

Appendix A: Glossary

ballot . . representation of voter preferences by a “1” denoting top choice for plurality and STV elections, extending sequentially to up to the number of candidates for a ranked choice election

candidate . person applying to fill a position by election

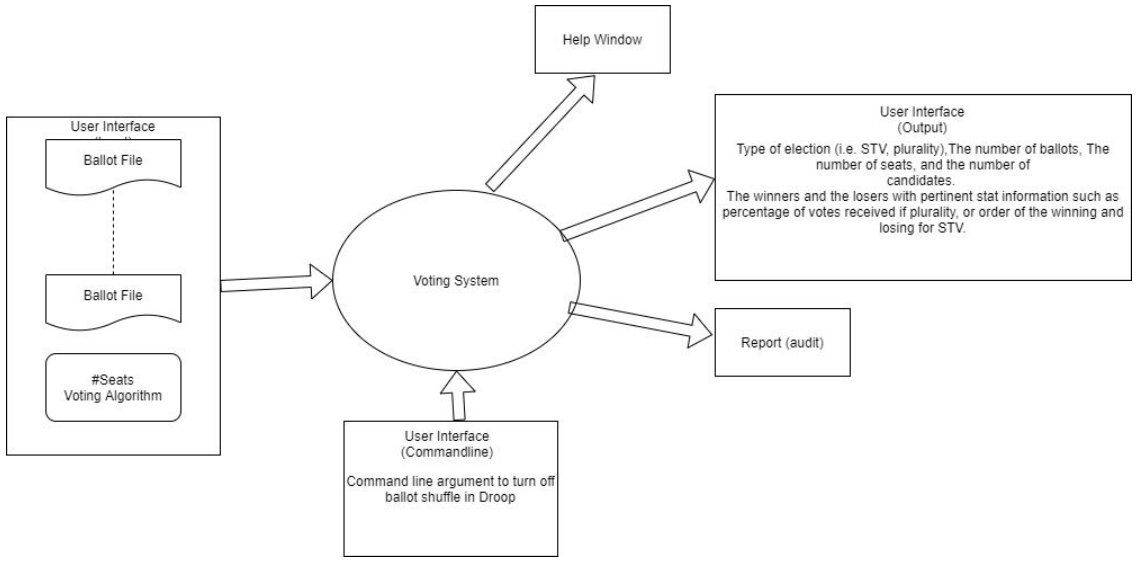
Droop Quota . (ranked choice voting) algorithm that allows voters to rank multiple candidates in order of preference, declares candidate(s) winner(s) once they have reached the quota of $floor\left(\frac{number\ of\ ballots}{number\ of\ seats + 1}\right) + 1$, after which candidate is removed from remaining ballots and next highest preference gains votes

election official inputting person tasked with responsibility of collecting, managing, and ballots into the Voting System to run elections and generate election audit reports

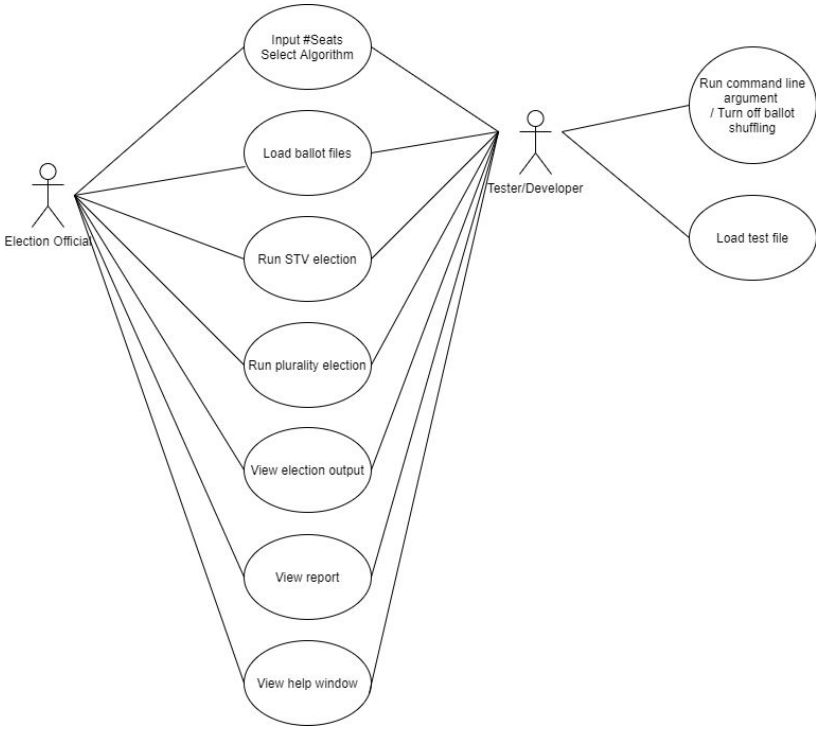
plurality	.	voting method in which each voter may only choose 1 candidate, and candidate(s) with greatest number of votes declared winner(s)
seat	. .	open position for which a candidate may be elected to fill
STV	. .	single transferable voting, family of algorithms including Droop Quota (ranked choice voting) in which voters may rank multiple candidates in order of preference
Voting System		name of this system used for running STV and plurality elections

Appendix B: Analysis Models

Voting System Major Components Diagram:



Use Case Diagram:



Appendix C: To Be Determined List

None