Software Requirements Specification

for

Election Tabulation System

Version 1.0 approved

Prepared by: Zachary Larsen (lars7566), Amelia Lunning (lunni022), Baanee Singh (sing0931), Josh Subhan (subha013)

CSCI 5801 Group 10

February 10, 2025

Table of Contents

Table of Contents **ii**
Revision History **iii**
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 2
2.4 Operating Environment 2
2.5 Design and Implementation Constraints 2
2.6 User Documentation 2
2.7 Assumptions and Dependencies 3
3. External Interface Requirements **3**
3.1 User Interfaces 3-4
3.2 Hardware Interfaces 4
3.3 Software Interfaces 5
3.4 Communications Interfaces 5
4. System Features **5**
4.1 Filename Prompt 5-6
4.2 Input Parameters 6-7
4.3 Processing CSV File 7-8
4.4 Plurality Voting Calculation 8-9
4.5 STV Voting Calculation 9-10
4.6 Audit 11
4.7 Election Summary Display 11-12
4.8 System Startup 13
5. Other Nonfunctional Requirements **13**
5.1 Performance Requirements 13
5.2 Safety Requirements 13
5.3 Security Requirements 14
5.4 Software Quality Attributes 14
5.5 Business Rules 14
Appendix A: Glossary **14**

Revision History

Name	Date	Reason For Changes	Version
Election Tabulation System	Feb 10, 2025	Initial product development	1.0

1. Introduction

1.1 Purpose

The Election Tabulation System v1.0 is a new, standalone system that efficiently and reliably calculates election outcomes that use plurality or single transferable vote (STV) with the Droop quota allocation methods. The system ensures accurate vote collection, processing, and result calculation, reducing the risk of human errors in manual vote counting.

1.2 Document Conventions

This software requirement specification (SRS) document is based on the IEEE Template for System Requirement Specification Documents.

1.3 Intended Audience and Reading Suggestions

This SRS is intended for readers involved in the development, management, and use of the election tabulation system.

- **Developers** should focus on the functional and non-functional requirements as these will provide the technical specifications, constraints, and implementation guidelines for the voting algorithms (plurality vs. STV).
- **Project managers** should review the system objectives, scope, and high-level requirements to ensure project timelines and deliverables align with the specified goals.
- **Testers** should concentrate on the system's expected behaviors, use cases, and validation criteria to develop test cases.
- **Users**, such as election officials, administrators, and auditors, should focus on the sections with system operations, user interfaces, and workflow processes.
- **Documentation writers** should refer to the sections that detail the system functions, user interactions, and error handling to create clear user guides.

This document should be read from top to bottom. All readers should begin with sections one and two, which provide an overview of the election tabulation system. Section 3 provides the interface requirements. Section 4 highlights the description of each feature of the system. Section 5 provides a list of requirements for nonfunctional parts of the system.

1.4 Product Scope

This product is designed to tabulate the results of elections that use the plurality or STV allocation methods but does not support casting a vote. The system's ability to take in files with all ballot information speeds up the tabulation process while minimizing labor. System users can see election outcomes and statistics upon the completion of tabulation, which allows officials to release the information to the public and maintain historical significance.

1.5 References

- IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications*. IEEE Computer Society, 1998

2. Overall Description

2.1 Product Perspective

The Election Tabulation System is a newly developed, standalone system. The system originates from the need to tabulate the winner(s) of elections that use different vote allocation methods.

2.2 Product Functions

- Intake a CSV file with the votes of every cast ballot
- Choose between STV and plurality vote allocation types
- Input parameters for the election such as the number of seats
- Tabulate election results using the user-prescribed method
- Display election results and statistics
- Export an audit file outlining vote allocation

2.3 User Classes and Characteristics

- Election officials: Certified and trained individuals to use the system for election tabulation
- Testers: Certified and trained users who ensure the system is working properly before an election

2.4 Operating Environment

The software will operate on University of Minnesota College of Science and Engineering (CSE) lab machines that run the Linux-based Ubuntu operating system.

2.5 Design and Implementation Constraints

The Election Tabulation System is developed in Java and is only used on the CSE Lab machines at production time. Object-oriented programming principles and pillars are followed as closely as possible.

2.6 User Documentation

None at this time.

2.7 Assumptions and Dependencies

It is assumed that the latest version of Java is installed on the CSE lab machines. It is assumed that the first line of the file provides the correct names of the candidates, subsequent lines are each voter's ballot, the file contains a number of ballots that are at least equal to the number of seats, and for STV allocation each ballot has at least half of the candidates ranked.

3. External Interface Requirements

This document outlines the external interface requirements for the voting system, addressing user, hardware, software, and communication interfaces.

3.1 User Interfaces

The Election Tabulation System provides a text-based user interface for collecting the input file, election parameters, displaying the election summary, and providing a name for the audit file. Prompts are given to the user when they are required to enter information into the system, The system will inform the user of the actions it is currently taking. Input validation is performed where necessary, and the user is notified of invalid inputs. An election audit file is generated in text form where it is saved in the current working directory.

User Interface Example:

```
Welcome to the Election Tabulation System!
Please enter the name of your file (must be in .csv):
> voting.txt
File type is not .csv. Please try again.
Please enter the name of your file (must be in .csv):
> voting.csv (Case: not in the same directory)
File not found. Please try again.
Please enter the name of your file (must be in .csv):
> voting.csv
Processing the election file...
File has been processed.
Please enter the election type (STV or Plurality):
> Plurality
Please enter the number of seats to fill:
> [Number of Seats]
Calculating Plurality Votes...
Votes tabulated successfully. Winners and losers have been determined.
Election Summary:
Election Type: [Plurality/STV]
Number of Ballots: [X]
Seats to be Filled: [Y]
Number of Candidates: [Z]
Winners: [List of Winners and their Vote Percentage/Order]
Losers: [List of Losers and their Vote Percentage/Order]
Election summary has been displayed.
Please enter the name of audit file:
> [filename]
Audit has been run...
Results have been saved to the audit file.
Thank you for using the voting algorithm! Goodbye!
```

3.2 Hardware Interfaces

The software is run on the CSE Lab Machines. The user interacts with the system via a keyboard and mouse, and the system is displayed on a connected monitor. A device with I/O capabilities and basic display capabilities is required. The software exclusively handles comma-separated-value files as input and text files (.txt) as output for the audit file. This means that non-volatile storage is required to use this software.

3.3 Software Interfaces

This system is compiled and run on the CSE Lab machines running the Linux-based Ubuntu operating system. Support for comma-separated-value and .txt files is required to use the system. The program will utilize standard Java libraries for feature implementation such as *java.io.*, java.util.**, and *java.lang.math*. These packages are for implementing file I/O support, random number generation, and Droop quota calculations. The system interacts with the operating system's display mechanics, file I/O capabilities, and standard I/O.

3.4 Communication Interfaces

None at this time.

4. System Features

This section demonstrates the Voting Tabulation System's features and capabilities through use cases.

4.1 Filename Prompt

Name	Filename Prompt
ID	UC_01
Description	The system will prompt the user for the election filename.
Actors	The system and the user.
Organizational Benefits	Makes it much easier for the user to give the system the information, reducing errors that can happen with manual input and ensuring the process is quicker for future operations.
Frequency of Use	Once for every time the system is run from the beginning.
Triggers	The user starts the system.
Preconditions	The system has successfully started.
Postconditions	The system has found the file, created a file handler, and opened the file for reading.
Main Course	1. The user is prompted for the filename 2. The user enters the file name (AC1).

	3. System verifies that the filename ends in .csv (EX1). 4. System verifies that the file is found (EX2). 5. System opens file for reading (EX3).
Alt Courses	AC1 User does not enter a filename 1. System tells the user that a valid filename is required 2. Return to MC 1
Exceptions	EX1 Filename does not end in .csv 1. System tells the user that file type isn't supported and they must enter a .csv file 2. Return to MC1 EX2 File not found 1. System displays a file not found error message 2. Return to MC1 EX3 Invalid permissions 1. System tells the user that it does not have the appropriate permissions to read file 2. Program terminates

4.2 Input Parameters

Name	Input Parameters
ID	UC_02
Description	Prompt the user to enter the election type (STV or Plurality), the number of seats being elected, and whether to enable STV shuffle.
Actors	Election Officials / Testers
Organizational Benefits	The system needs to know the type of election and the number of seats in order to process the file, and count votes.
Frequency of Use	Every time the system is used
Triggers	The election file has been successfully opened for reading
Preconditions	The system has found the file, created a file

	handler, and opened the file for reading.
Postconditions	System completes file processing
Main Course	1. System prompts user for the ballot type 2. User inputs ballot type (EX1). 3. System prompts user for number of seats to fill 4. User inputs number of seats (EX2). 5. System prompts user for shuffle on/off 6. User inputs on/off (EX3).
Alt Courses	None
Exceptions	EX1: Invalid ballot type 1. System tells the user the ballot type entered is invalid 2. Return to MC1 EX2: Invalid number of seats 1. System tells the user the number of seats entered is invalid 2. Return to MC3 EX3: Invalid shuffle on/off input 1. System displays an error message that an invalid option was selected. 2. Return to MC5

4.3 Processing CSV File

Name	Processing CSV File
ID	UC_03
Description	The system reads in and processes the file that is opened for reading so the number of candidates and ballots can be collected.
Actors	The system.
Organizational Benefits	Automated ballot processing is much faster than doing it by hand, and can prevent counting errors that could alter the outcome of an election.
Frequency of Use	Once for every time the system run and a valid filename is entered

Triggers	The user has entered all valid parameters
Preconditions	There is an open file handler and all input parameters have been entered
Postconditions	The system has saved the number of candidates running and the number of ballots.
Main Course	1. The system tells the user that the file is being read. 2. The system extracts the required information and saves it. 3. The system tells the user that the file has been successfully read (EX1)
Alt Courses	None
Exceptions	EX1 System could not get the required information from the file 1. System tells the user that the file reading was unsuccessful, and checks the file for errors 2. Program terminates

4.4 Plurality Voting Calculation

Name	Plurality Voting Calculation
ID	UC_04
Description	The plurality voting system is used to tabulate votes and determine winners and losers of the election
Actors	System
Organizational Benefits	Decreases the time it takes to process an election using the Plurality voting system compared to doing it by hand.
Frequency of Use	Every time the user chooses Plurality as the method of choice
Triggers	File processing has successfully completed
Preconditions	The system has saved the number of candidates running and the number of ballots, and the plurality method has been chosen

Postconditions	Votes have been tabulated and winners and losers have been determined
Main Course	1. System reads through the ballots 2. System keeps track of the number of votes for each candidate 3. System assigns the correct number of winners based on the number of seats and amount of votes each candidate received (AC1). 4. All other candidates are determined to be losers
Alt Courses	AC1: A tie occurs between two or more candidates 1. Out of the tied candidates randomly choose someone to be the winner
Exceptions	None

4.5 STV Voting Calculation

Name	STV VotingCalculation
ID	UC_05
Description	This algorithm processes the ballots using the STV voting system and produces a list of elected and non-elected candidates
Actors	System
Organizational Benefits	Decreases the time it takes to process an election using the STV voting system compared to doing it by hand.
Frequency of Use	Every time the user chooses the STV election type
Triggers	File processing has successfully completed.
Preconditions	The system has saved the number of candidates running and the number of ballots, and the STV method has been chosen.
Postconditions	Votes have been tabulated and winners and losers have been determined
Main Course	1. System randomly shuffles ballots

	(AC1). 2. System calculates Droop quota 3. System reads through the ballot's 1st choice votes and distributes them into one-at-a-time piles for the candidates (AC2). 4. System adds any candidate that reaches the quota to a list of elected candidates until all seats are filled (AC3, AC4). 5. Steps 3 and 4 are repeated with the elected candidates' surplus votes, and they're distributed based on the ballot's next choice votes 6. If no candidate reaches the quota, then the candidate with the fewest votes is eliminated, and they're added to a list of non-elected candidates (AC3). 7. Steps 3 and 4 are repeated with the eliminated candidate's votes, and they're distributed based on the ballot's next choice votes 8. After all seats are filled, the system adds any remaining non-elected candidates to the list of non-elected candidates
Alt Courses	AC1: User selected shuffle off 1. Return to MC2 AC2: A ballot's choice candidate has been elected or eliminated 1. The next choice candidate that has not been elected or eliminated is counted, or the ballot is eliminated if there are no remaining ranked candidates on the ballot 2. Return to MC3 AC3: There is a tie 1. The candidate who was last to receive a vote is eliminated 2. Return to MC4 or MC6 AC4: More candidates reach quota than number of seats to fill 1. The candidates with the highest numbers of votes are elected until the seats are filled (AC3) 2. Return to MC5
Exceptions	None

4.6 Audit

Name	Audit
ID	UC_06
Description	Users can run an audit file with shuffle turned off to test the proper calibration of the system
Actors	Election official, system
Organizational Benefits	Makes sure the system works as intended and no bugs are present that would compromise the integrity of an election
Frequency of Use	Anytime a user wants to test the system
Triggers	The election summary has shown up on the screen.
Preconditions	Winners and losers of the election are determined.
Postconditions	A file with all audit information is saved to the user's machine
Main Course	1. Election tabulation is run and complete (UC-01-04 or 05) 2. Summary of results are displayed to the user (UC-07) 3. User presses any key to continue 4. The user is prompted to enter the name they wish to give to the audit file (EX1).
Alt Courses	None
Exceptions	EX1 User enters an invalid name for the audit file 1. Tell the user the allowed characters for a file 2. Return to MC4

4.7 Election Summary Display

Name	Election Summary Display
ID	UC_07

Description	The system displays key election details, including the type of election, the number of ballots, the number of seats, the number of candidates, the winner and losers with stat information (such as percentage of votes received if plurality, or order of winning/losing for STV).
Actors	System
Organizational Benefits	This allows the user to easily view the results and statistics of the election.
Frequency of Use	Once per system run
Triggers	Election tabulation has been completed
Preconditions	Votes have been tabulated and winners and losers have been determined
Postconditions	The system displays a summary of the election. The election results are available for review by the user.
Main Course	1. The system gathers all the necessary data for display. 2. The system displays: - Election Type (Plurality vs. STV) - Number of Ballots - Number of seats to be filled - Number of candidates - List of winners (such as percentage of votes received if plurality, or order of winning for STV). - List of losers (such as percentage of votes received if plurality, or order of losing for STV).
Alt Courses	None
Exceptions	None

4.8 System Startup

Name	System Startup

ID	UC_08
Description	The process in which the election tabulation system is initialized.
Actors	Election officials, testers
Organizational Benefits	Allows for the usage of the automated tabulation system
Frequency of Use	Each election or for testing
Triggers	The user starts the system
Preconditions	The appropriate software to run the system is installed on the user's machine
Postconditions	User is prompted for a filename (UC_01)
Main Course	1. User enters command line startup details and arguments (AC1)
Alt Courses	AC! User enters STV shuffle flag 1. STV shuffle is turned off
Exceptions	None

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The tabulation system must be able to process at most 100,000 ballots from a single CSV file. Tabulatim, displaying results, and producing the audit must not take more than 5 minutes. These requirements must be met to ensure that vote counting can happen quickly and seamlessly for an accurate election.

5.2 Safety Requirements

None at this time.

5.3 Security Requirements

Voting and election security are handled by the officials at city precincts. The Election Tabulation System does not need to accommodate security concerns at this time.

5.4 Software Quality Attributes

Development of the system should follow all object-oriented programming principles such as abstraction, encapsulation, inheritance, and polymorphism. Code redundancy should be minimized as much as possible. The system should be easily open to extension and closed to modification, such as adding additional voting methods.

5.5 Business Rules

The Election Tabulation System is only operated by trained election officials. Trained testing officials do all testing before an election.

Appendix A: Glossary

- STV: Single Transferable Vote: A voting method in which voters submit one ballot with candidates ranked by their preference of candidate. This method ensures that all winners have won by a majority vote.
- CSE: College of Science and Engineering: A college within the University of Minnesota that is home to all the science and engineering majors.
- CSV: Comma-separated value: A file type where columns of data are separated from one another with a comma.
- Plurality: The number of votes cast for a candidate who receives more than any other, but does not get an absolute majority
- Abstraction: The act of generalizing concrete details about the implementation and attributes of an entity
- Encapsulation: Ensures that the data and methods that act on it are bundled together, limiting access to the data by other actors
- Inheritance: Enables code reuse from other entities
- Polymorphism: The use of one symbol to represent multiple types
- Object-oriented programming: The idea that everything in a program is an object with attributes and methods