Software Requirements Specification

for

Voting System

Version 1.0 approved

Prepared by Philip Siedlecki (siedl009),
Anthony Ross-Sapienza (rosss001),
Jack Levine (levin520),
Xiuyu Yan (yanxx401)

Team 24

October 10th, 2019

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Voting System	10/10/19	Initial document creation	1.0

1. Introduction

1.1 Purpose

This document will explain the Voting system program. The following sections will define and describe the system in detail. The features, limitations, interfaces, constraints, and expected behavior will be laid out. This document will be of use to both programmers and users as a guide for use and modification of the Voting system software.

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

- Initial design and maintenance software engineers, who will use this document to guide the construction and maintenance of the Voting system software
- Testers of the software, who will use this guide to know and understand expected behavior to aide in finding bugs in the system
- Election officials, who will use this document to ensure the system meets all requirements and to identify the total scope of the system

1.4 Product Scope

This Voting system is a comprehensive program that will provide analysis and results in Open Party List (OPL) and Closed Party List (CPL) elections. When provided with a .csv file containing election information and ballots cast, the system will be able to read in the data and produce useful information. The system will declare the winner(s) of the election provided, and break ties fairly and randomly when necessary. After the data is tabulated and analyzed, the system will also provide an audit for election officials to verify results along with an easily readable file to be given to media outlets to report on the election.

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

Voting system is a new, self-contained vote processing system designed to accept a formatted .csv file and to output detailed information regarding the election. The system was developed to help improve the modern digital voting scene, offering a simple and effective means to tally voting data and produce an informative breakdown of an election. The system is designed to handle elections of either open or closed party list voting, awarding greater flexibility to potential consumers.

2.2 Product Functions

I/O:

- open_file: open a parameter file based on relative or absolute location
- process_file: validate the file is of either OPL or CPL and fill data structure with lines of input file.
- create_audit_file: based on tabulated data, create and fill a new file consisting of raw election data.
- create_media_file: based on tabulated data, create and fill a new file describing election outcome in digestible format.
- display_winner: based on output of tabulation, print winner(s) to terminal window.

Help:

- help: print use cases and formatting constraints to terminal window.

Party Listing:

- parse_input_line: parse input line based on line number and whether open or closed party listing, and assign values to appropriate member variable.
- tabulate_votes: from class members, tabulate votes for appropriate party or candidate.
- determine_winner: based on output of tabulate_votes, determine winning party or candidates.
- determine_tie: if there is a tie, randomly select a winner.

2.3 User Classes and Characteristics

- Election officials utilizing the system to parse and interpret the outcome of an election.
- System testers implementing bug fixes and analyzing the system for correct functionality.
- Programmers modifying or adapting the system functionality to suit changing needs.

2.4 Operating Environment

Voting system was designed to run on Linux Ubuntu 18.04 release version.

2.5 Design and Implementation Constraints

Voting system is developed in C++11 and designed to run through the Linux command terminal. The system must meet a benchmark performance of five minutes execution time for input of 100,000 ballots.

2.6 User Documentation

A quick start guide can be found at:
https://github.umn.edu/umn-csci-5801-f19/repo-Team24/vote_system_qstart.txt/

User Manual:
https://github.umn.edu/umn-csci-5801-f19/repo-Team24/vote_system_user_manual.txt/

2.7 Assumptions and Dependencies

Voting system is developed in C++11 and therefore requires C++11, or higher, to be installed on the user's system. Program compilation requires gcc 7 or higher to be installed on the user's system in order to run the Makefile.

3. External Interface Requirements

3.1 User Interfaces

Using the Makefile interface:
After downloading the Voting system, it needs to be installed.

1. This can be done by placing the downloaded files into the desired installation location.
2. Opening the downloaded folder.
3. Running the “make” command in the terminal.

This will compile the necessary executable file “Voting_System”.

Running the program:
The input file containing the raw vote data must be in the same directory as the executable.
The voting system can be run using the input file in two ways:

1. **Running directly through the command line:**
   The program can be run without user prompts by calling the executable with the file name.
   ./Voting_System vote_data.csv

2. **Running with user input:**
   When run using “./Voting_System” without additional arguments, the program will prompt the user for a file name.

Viewing the results:
After running the Voting System, the results will be stored in a media friendly file, along with an audit file. These files will not be overwritten by subsequent runs of the program. The output files will follow the following naming conventions:

1. **Media file:**
   District_(Date : mm-dd-yyyy)_Media.txt

2. **Audit file:**
   District_(Date : mm-dd-yyyy)_Audit.txt

Once the program is run the above files will be saved to the folder where the Voting System has been run. These files are text files and can be viewed using your favorite text editor / viewer.

File format Error:
The Voting System will check that the file exists within the folder:
“Error Invalid file name example.csv” will be displayed when the user has input an invalid file name.

The Voting System will also check that the header information of the file is correct.
The file must start with either “OPL” or “CPL” along with the expected file header information (found in Section 6).

3.2 Hardware Interfaces

The voting system is designed to work best on CSE lab machines. The recommended system specifications are 3.60GHz CPU and 32 gigabytes RAM.

3.3 Software Interfaces

The voting system is designed to work on Ubuntu 18.04. The voting system also requires that C++11, gcc 7.4.0, and Makefile are installed on the operating system.

3.4 Communications Interfaces

The Voting System does not require an internet connection, so long as the vote data exists on the executing machine.

4. System Features

4.1 System Feature

4.1.1 Description and Priority

Feature	Description	Priority	Benefit	Penalty	Cost	Risk
CSV file loaded into program	A CSV file is loaded providing the ballot csat information	6	7	6	5	4
Tie breaking	A random generator would randomly select the winner	4	5	4	4	3
File validation	Check the file has the expected header information	4	4	3	2	3
Run ballot tabulation for OPL/CPL	The OPL/CPL file is read into the system and analyzed	7	7	8	6	6
Determine the winners for OPL/CPL	The voting result is analyzed to determine the winner	6	8	8	6	5
Produce the audit file	An audit file is produced that has in depth information about the election	6	8	8	6	6
Produce the media file	A media file is produced that contains the summary of the election	6	7	6	5	5
Show winner	A short summary of the election result is shown on the screen	6	8	8	3	4

4.1.2 Stimulus/Response Sequences

For the users of the program (programmers. testers, and election officials):
St-1: Run the program through command line with the filename of the read-in file provided

- The file can't be changed outside of the program.

- The read-in file is in the CSV format

St-2: The program runs to process the CPL or OPL file and determine the winner(s) of the election.

- The system will process the file based on the election type as indicated in the first line of the file.
- The election result is determined.

St-3: The election result is printed on the screen.

- The result contains the winner(s) and parties, type of the election, the number of ballots cast, and the number of votes received for all candidates and parties.

St-4: An audit file in TXT format is generated at the same directory of the program.

- The audit file includes the time, election type, number of open seats prior to the election, the number of ballots cast, the quota for winning a seat.

St-5: A media file in TXT format is generated at the same directory of the program

- The media file contains the time, election type, winner and a list containing percentages of the votes and the seat allocation.

4.1.3 Functional Requirements

REQ-1: The system should read in a file in CSV format that is at the same directory as the program, all information that is needed to run the program should come through this file.

- The system should only prompt the user for the file name.

REQ-2: The system needs to produce an audit file in TXT format with the election information at the time.

- The audit file contains the following information: time and day that the election results have been created, the election type, the number of open seats prior to the election, the number of ballots cast in the election, the parties participating in the election, the number of votes received, percentage of total votes and list of the candidates and quota for winning a seat.

REQ-3: The system needs to display to the screen the winner and information about the election.

- The following information should be displayed: number of ballots cast, the winners and the number of votes received for everyone or the party

REQ-4: If there is a tie of the result, the system needs a random generator to randomly select the winner in a fair coin toss.

REQ-5: The system needs to handle both types of elections: CPL (Closed Party List Voting) or OPL(Open Party List Voting).

- The election type is indicated on the first line of the read-in file.
- The system should be able to fully read and process CPL ballot and OPL ballot. For the CPL ballot, voters express a preference for a particular party. For the OPL ballot, voters express a preference for a particular candidate and party.

REQ-6: The system should be developed in either C++ or Java.

REQ-7: The system should be able to run 100,000 ballots in under 5 minutes.

REQ-8: The system needs to generate a media file in TXT format that contains the summary of the result.

- The media file contains the following information: The time and day that the election results have been created, the type of the election, the winner or the winning parties and a list containing percentages of the votes and the seat allocation.

REQ-9: The program needs to be viable for running multiple times during the year.

REQ-10: The file structure shouldn't be changed outside of the program.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Elections must be processed within a reasonable time frame. An election of 100,000 votes must be analyzed within 5 minutes. The system running it is recommended to have a 3.6 gigahertz CPU and 32 gigabytes of available RAM.

5.2 Safety Requirements

No safety requirements are currently in place for this system.

5.3 Security Requirements

No security requirements are currently in place for this system.

5.4 Software Quality Attributes

The Voting system will be easy to use and maintain. The audit provided will essentially show what the system has done, step-by-step, so as to easily test the system. It will also provide a clear method for election verification, so that if election officials are required to ensure the correctness of the results they can. The goal of the system is to be easily verifiable, so steps are taken to document what happens. The original file is also not edited in any way, to ensure that verification is possible in a different manner. The system will be able to handle any CPL or OPL election that is submitted in the proper format, and can be updated to handle other types of elections if necessary.

5.5 Business Rules

Anyone with access to it can run the system, but it will be maintained on a local machine that has limited access. The file loaded into the system must be in the same storage location as the system, and, similarly, the files produced by the system are saved in the same location. This limits the ability to tamper with the results of the elections.

6. Other Requirements

Input File Formatting:
The Input files must be in the following form:

1. **OPL (Open Party Listing):**
   1st Line: OPL for open party listing
   2nd Line: Number of Seats
   3rd Line: Number of Ballots
   4th Line: Number of Candidates
   Following the 4th line are the candidates (one per line) and in the format above. The candidates and their party are in [ ].
   Then the ballots are listed one per line and commas are used to separate the positions.

2. **CPL (Closed Party Listing):**
   1st Line: CPL for closed party listing
   2nd Line: Number of Parties
   3rd Line: Parties in Order of Ballot Ordering
   4th Line: Number of Seats
   5th Line: Number of Ballots
   6th Line: Number of Candidates
   Following line 6 are the candidates' information followed immediately by all of the ballots. The order of the party is indicates their position in the ballot.

Appendix A: Glossary

OPL: Open Party List voting system. A system in which available seats in an election are awarded to parties, but each member of the party is also voted for individually. The highest vote getters in a party are awarded seats that the party wins.

CPL: Closed Party List voting system. A system in which parties choose the rank of their candidates, and voters cast votes for the party as a whole. Seats that are awarded to parties are given to the candidates in the order the party ranked them.

.csv: Comma Separated Values file. A plain text file that contains a list of data, separated by commas. It is both human-readable and easily readable by the system.

Appendix B: Analysis Models

No applicable models.

Appendix C: References

No applicable references.