Software Requirements Specification

for

Voting System CRSS

Version 1.0 approved

Prepared by Team 14

Cassandra Chanthamontry (chant077)

Jake Nippert (nippe014)

Meghann Silagan (silag001)

Christine Tsai (tsaix223)

University of Minnesota

CSCI 5801 Software Engineering I

October 8th, 2018

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Voting System CRSS	October. 8, 2018	Version 1.0, first document for system design.	1.0

1. Introduction

1.1 Purpose

This document outlines the software requirements of the Voting System which is capable of performing two types of voting structures: Instant Runoff Voting (plurality/majority type) and Party List Voting using Open Party List (proportional voting type). An overall description of the product will be provided along with descriptions of the external interface requirements, system features, and other nonfunctional requirements of the product. This SRS document covers the entire Voting System including the user interface, how users will interact with it, and the backend operations of the system.

1.2 Document Conventions

This Software Requirements Specification document was created using the IEEE Software Requirements Specification Template.

The convention for displaying questions is to italicize.

1.3 Intended Audience and Reading Suggestions

Election Officials, Testers, and Software Developers are the intended audience for this document. It is recommended that the document be followed sequentially for Software Developers and Election Officials in its entirety. Testers will likely benefit the most from focusing on sections 3 and 4 where external interface requirements and system features are covered.

1.4 Product Scope

The Voting System allows each User to produce election results for two main voting structures prominent across the globe. Testers, Software Developers, and Election Officials should be able to utilize the system to produce an output of election results for their respective needs. The objective of this software is to incorporate all informal user requests so that the system meets all the users' needs and functions without error. The software must be accurate, consistent, easy to use, and helpful for the users. The overarching goal of the system is to provide financial and time efficiencies and to extend the generality of the system to accommodate the processing of elections of the aforementioned types.

1.5 References

- Use cases will be included in a separate PDF document and turned in with the deliverance of the SRS document

2. Overall Description

2.1 Product Perspective

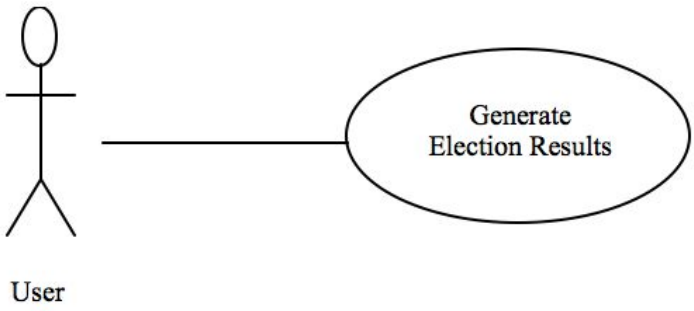
The product is designed for electoral officials to easily receive and analyze election results of two types of voting: Instant Runoff Voting and Party List Voting using Open Party list. This software is intended for use by electoral officials, testers, and software engineers. The voting system software will use a provided CSV file and implement the required voting system algorithm to output an audit file containing election information.

2.2 Product Functions

Actors/Actions:

1. User (Software Engineer/Tester/Electoral Official)
    a. Generate Election Results

2.2.1 User Use Cases



2.3 User Classes and Characteristics

- Software Engineers who will develop this program
- Testers who will test the program to ensure that the voting algorithm produces correct results for the audit file
- Electoral officials who will use this software to return correct election results and audit file information

2.4 Operating Environment

The Voting system is compatible with any operating system that contains a Java compiler (e.g. Windows 7, 8, 10, Mac OS X, and Linux).

2.5 Design and Implementation Constraints

- This system will be developed in Java 8 (The latest version of java).

2.6 User Documentation

- *Should our tutorials/training include information about using the system with accessibility accommodations incorporated?*
- *What types of tutorials will be required (video, text based, interactive, etc.)?*
- *Can we assume the user is well versed in using CSE lab machines?*

2.7 Assumptions and Dependencies

- The voting will be done outside of the voting system, and the results from the voting will be provided to us in a comma delimited text file.
- Only one CSV file will be passed into the system per election.
- The CSV file will have no numbering mistakes (i.e. voters will not make mistakes on the ballot), so each ballot will have at least 1 rank and no candidate will have more than 1 ranking.
- All preprocessing of the file will be done before it is passed into the Voting System.
- The first line of the CSV file will have the type of voting.
- Write-in candidates are not allowed.
- The Java program will be run either at the command prompt or through Eclipse on the most up-to-date CSE lab machine.
- The CSV file will be placed in the same directory as the program.
- The election results could be shared with media personnel.

3. External Interface Requirements

3.1 User Interfaces

The software will contain no UI at this moment. For now everything will be in a terminal. In future, UI may be implemented when prompting the user for the filename and other inputs.

3.2 Hardware Interfaces

The Voting system is compatible with any operating system that contains a Java compiler For example, Windows 7, 8, 10, Mac OS X, and Linux.

- *Will the system be interacting with external hardware such as flash drives or external hard drives or can it only load from a local directory?*

3.3 Software Interfaces

The program will be coded using Java 8 and will run and be tested on linux os using the CSE lab machines. The program can be run from Eclipse.

3.4 Communications Interfaces

- *Will the program require communication with the internet, networks, etc?*
- *What protocols will the system be using to communicate? Will it be using TCP/IP protocol/HTTP or something else?*

There are currently no constraints regarding the encryption or security of information as it pertains to communication. The program will not use any communication protocol. The onus is on the user to communicate with other parties.

4. System Features

4.1 Run an Election

4.1.1 Description and Priority

The user will execute the voting system on the provided csv ballot information file. The program will output election information on an audit file and display the immediate winner on the console. This feature is of the highest priority at 9 as this is the main functionality of the program. The benefits of automating the process of generating the election results saves user time and money. In addition to reducing human error.

4.1.2 Stimulus/Response Sequences

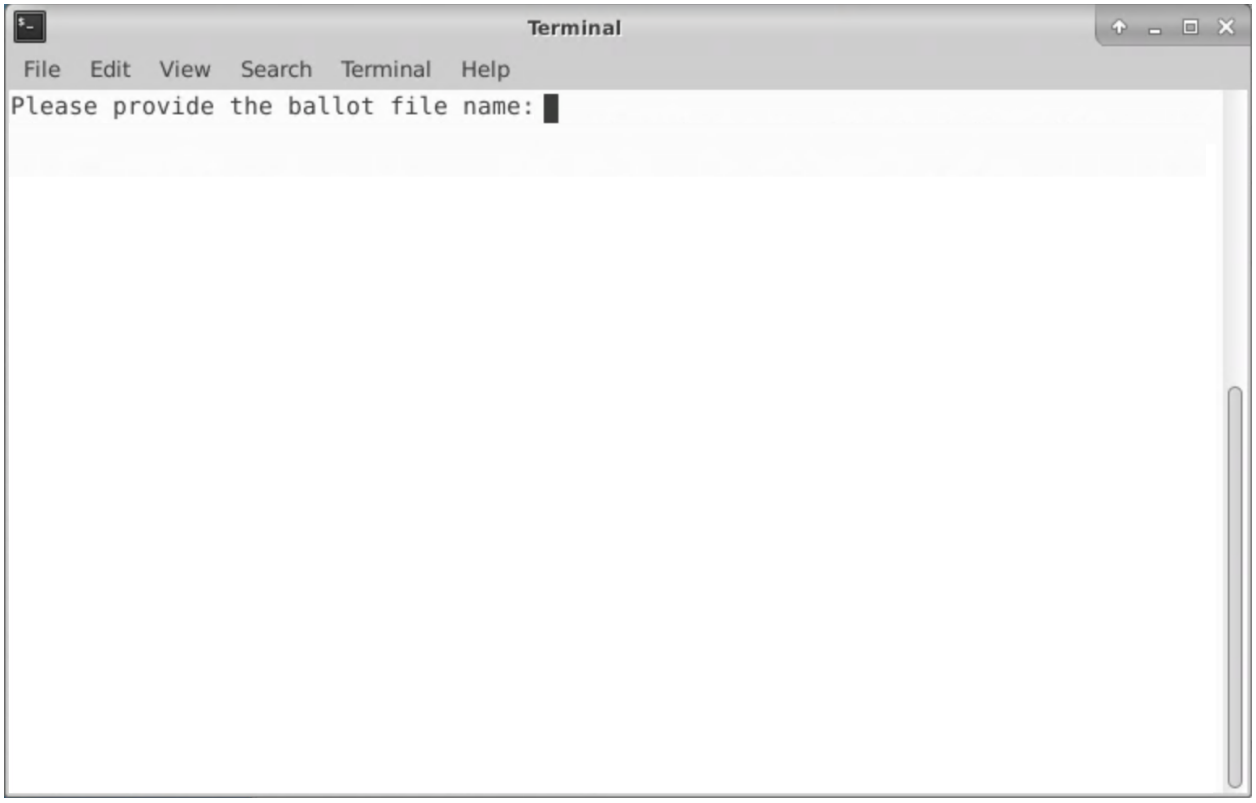
The user will be prompted to provide a file name if not passed in as a command line argument. The system will then parse the csv for ballot information and implement the proper algorithm. The system will then present the information as soon as available. Upon completion, the system will store all audit information in the audit file.

4.1.3 Functional Requirements

REQ-1: Must process election results from CSV file input and create audit file
REQ-2: Must display immediate winner to console

The program's main function is to allow the user to run an election process for one of two voting structures (IRV and PLV using open party lists).

Example of system prompting the user to provide a file name if one was not given as a command line argument:



This section provides detailed information on how the Voting System can be functionally utilized and the major services the system offers. Please refer to the attached pdf document for the use case.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The program will process up to 100,000 ballots taking fewer than eight (8) minutes.

5.2 Safety Requirements

There are no special safety requirements; this program will follow IEEE Code of Conduct guidelines. All information will be read in from the file once and then processed.

5.3 Security Requirements

This program will follow IEEE Code of Conduct guidelines. While the Voting System is running, the program should ensure that the calculations cannot be altered to output incorrect results. Once the election results are displayed and the audit file is produced, no one should be able to alter the results or the file. The user can only read the output and the audit file.

5.4 Software Quality Attributes

- The program should be general enough to work for all future elections that are of the specified structures with CSVs in the specified format.
- There is interoperability with operating system of CSE Labs Machine to write to files and pull information from ballot files.
- Testers, Software Developers and Election Officials will utilize the same software with the same results.
- The system should be able to run as many times as required up to the machine storage capacity of audit files.
- Immediate console results and the audit files should be displayed correctly, unambiguously, completely, and should be consistent across election runs (except for in the case of a randomly selected winner of a tie; in the case of a tie this should be explicitly stated).
- The audit file should be traceable and verifiable for a manual inspection of the results.

5.5 Business Rules

Voting systems operate under the understanding that any user of the program can utilize the output as they require for their respective needs and that no *specific* security, safety or other requirements necessitate the distinction of user identification for any processes.

6. Other Requirements

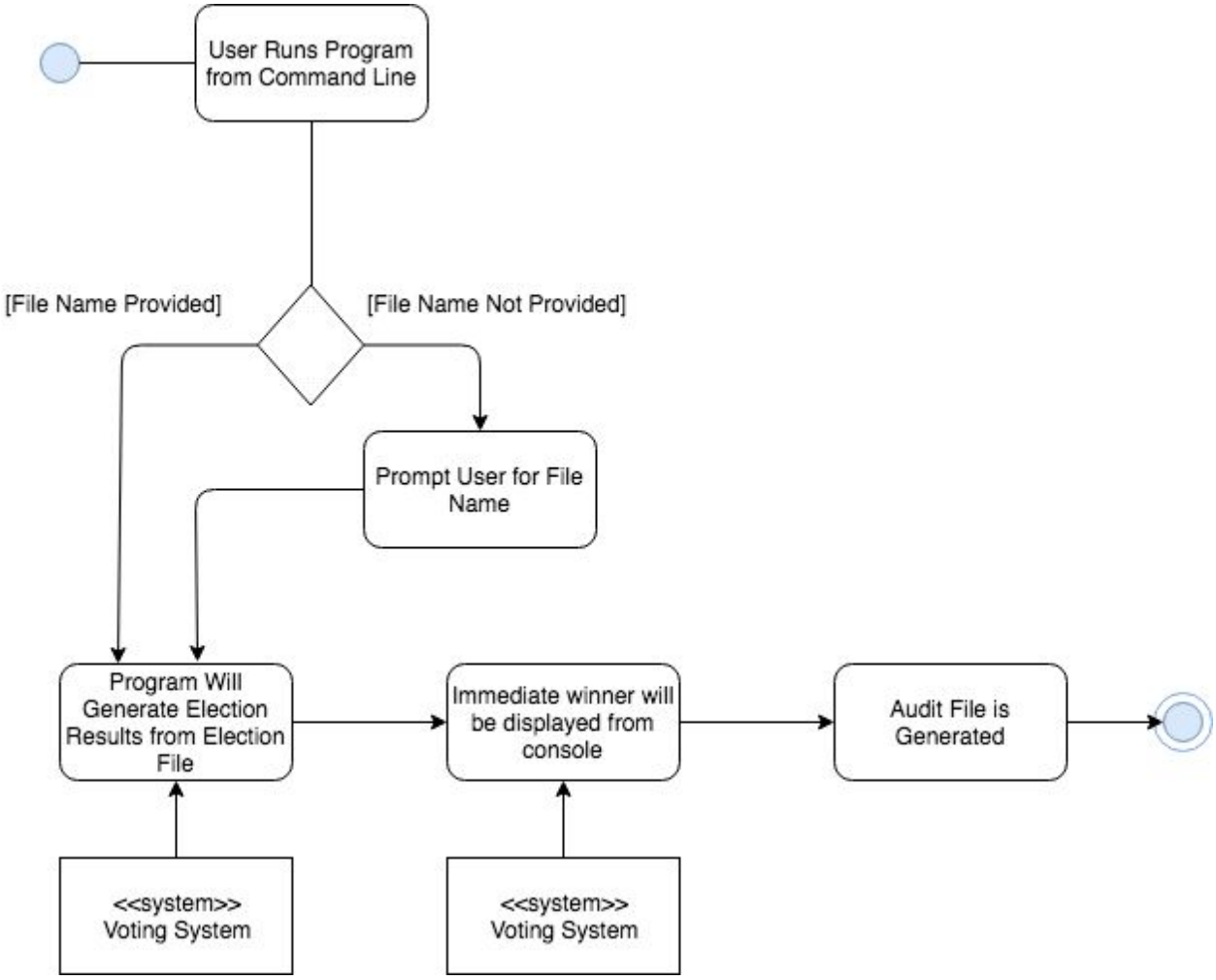
- *What legal requirements must our voting system abide to?*
- *What are other requirements not currently specified in the SRS?*

Appendix A: Glossary

1. Voting System - This is a system where users can provide a csv file with ballot information and choose which voting algorithm to implement to view results of the elections
2. SRS - abbreviation for software requirements specification. Usually implicative of the software requirements specification document
3. Instant runoff voting - also known as "IRV," and "majority preferential voting." Like two-round voting, this majority system a minor variation of single-member district plurality voting that was developed to ensure that the winning candidate enjoys the support of the majority of voters in the district. It was also thought to be an improvement over the two-round system because it does not require a separate election--it provides an "instant" runoff.
4. Party list voting - Legislators are elected in large, multi-member districts. Each party puts up a list or slate of candidates equal to the number of seats in the district.
5. Software - programs that determine operations of a system
6. Hardware - the physical components of a computer or other electronic system
7. OS - abbreviation for operating system

8. GUI - abbreviation for graphical user interface
9. UI - abbreviation for user interface
10. User - Tester, Software Developer or Election Official utilizing the Voting System program

Appendix B: Analysis Models



Appendix C: To Be Determined List

- *Can there be multiple elections that the program will process at a given time? (This would require the user to have the ability to select multiple files and for multiple audit files and outputs to the console to be produced.)*