Software Requirements Specification

for

Voting System

Version 0.3 approved

Prepared by Yanjun Cui (cui00022), Jerry Nie (nie00008), Hanzhang Wu(wu000123), Yangjiawen Xu(xuxx1262)

University of Minnesota, Twin Cities

10/10/2019

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Voting System	30 September 2019	Create this document and finish the main part of this document.	Alpha0.1
Voting System	5 October 2019	Basically finish the document and leave specific questions for further development.	Alpha0.2

Voting System	10 October 2019	Finish the whole document.	Alpha0.3

1. Introduction

1.1 Purpose

Election, the formal process of selecting a person for public office or of accepting or rejecting a political proposition by voting. It is important to distinguish between the form and the substance of elections.
The purpose of this document is to present a detailed description of the voting system. 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 programmers, testers and election officials.

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

- Programmers work on the project by further developing it or fix existing bugs.
- Testers work on testing the unit parts of the software and the complete software.
- Election officials work on running the software and produce the results of the election result.

1.4 Product Scope

Voting System is a tool that displays the candidates who are elected through the open party list ballot or the closed party list ballot. The result about the winners and information about the election will be displayed to the screen. Also, the result will be shared through the media personnel. The users can read in the file, display the results to the screen, and share the results with media personnel.

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

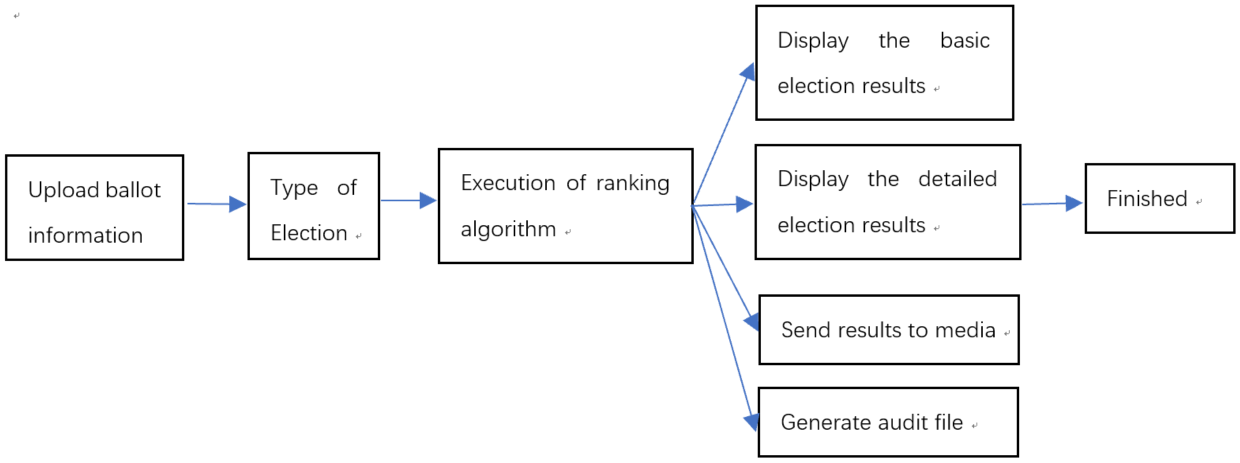
Party list voting systems are by far the most common form of proportional representation. Over 80% of the PR systems used worldwide are some form of party list voting. It remains the system used in most European democracies and in many newly democratized countries. Legislators are elected in large, multi-member districts. Each party puts up a list or slate of candidates equal to the number of seats in the district. Independent candidates may also run, and they are listed separately on the ballot as if they were their own party. In this system, according to the input ballot file, the system runs a sorting algorithm to determine the candidate's ranking. This system could also generate different type of results for different people to view.

2.2 Product Functions

File:
- Select: Selects a file
- Import Spreadsheet: Loads a CSV file

Execution:
- Run the ranking system

Results:
- Display the election results
- Generate audit file of election
- Send election results to media


Upload ballot information
Type of Election
Execution of ranking algorithm
Display the basic election results
Display the detailed election results
Send results to media
Generate audit file
Finished

2.3 User Classes and Characteristics

- Programmers work on the project by further developing it or fix existing bugs.
- Testers work on testing the unit parts of the software and the complete software.
- Election officials work on running the software and produce the results of the election result.

2.4 Operating Environment

Initially we only support Linux(Ubuntu 16.04).

2.5 Design and Implementation Constraints

Voting System is developed in Java and built on Eclipse. The computer executing the program is recommended to have at least Intel 5-gen dual-core i3 CPU at 1.60GHz or AMD CPU that has similar performance, and with at least 1GB RAM. The recommended graphic requirement is at least DirectX 10 compatible GPU with at least 512mb RAM. We divided features into separate modules that depend on each other through well-written APIs available to make development easy. The program should be developed in English.

2.6 User Documentation

After the software has been developed, a video will be recorded for users to demonstrate how they implement the software. This video includes the introduction of the software, step-by-step instructions and subtitles.

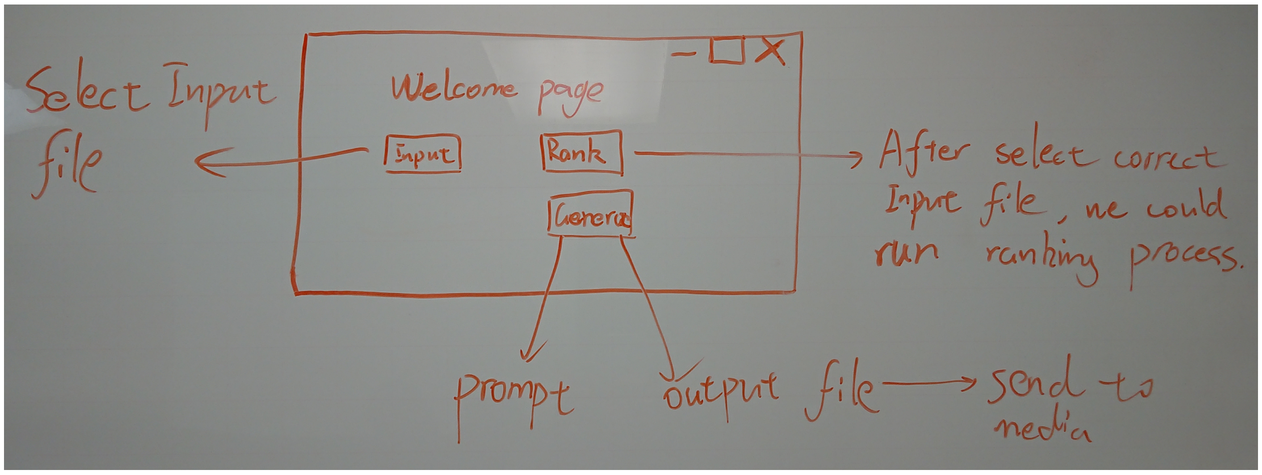
2.7 Assumptions and Dependencies

Voting System is developed in Java and therefore requires Java to be installed on the user's system. The latest stable version of Voting System requires Java version 8 or higher. This applies to Linux users.

3. External Interface Requirements

3.1 User Interfaces

The main page of Voting system has three buttons, "input", "rank" and "generate". Button "Input" is used to upload election file, button "rank" is used to begin ranking, while "generate" is used to generate results, which users can see the winners and generate the audit file. After generating results, if needed, the system can produce an overview and send it to the media.



3.2 Hardware Interfaces

This program is designed to run on a CSE lab machine. The minimum hardware requirements of Voting System are a 1.6GHz CPU and 1 Gb of RAM.

3.3 Software Interfaces

This program can be linked with .csv files. It can read the information from .csv file and write generated data into a new .csv file. Plus, it requires Java to be installed on the system, more specifically Java version 8 or higher. Additional information can be found on section 2.7 of this document.

3.4 Communications Interfaces

Voting System requires an internet connection to install new plugins, update already installed ones and update some of its components (APIs, modules etc.).

4. System Features

This section demonstrates Voting system's most prominent features and explains how they can be used and the results they will back to the user.

4.1 Data structure feature

4.1.1 Description and Priority

In order to meet the execution speed requirements. We use a dictionary (hash map) to store ballots information of every election: use candidates' names as keys and store the number of ballots as values. When reading ballots csv file, each time a new candidate is read in, a K-V item is created for him or her. Every time an existing candidate is read in, add one to her/his vote.

4.1.2 Stimulus/Response Sequences

We first need to read the content of the files. The content read is stored in HashMaps by the system. We then need to use quicksort to sort HashMaps, generating the sorted result.

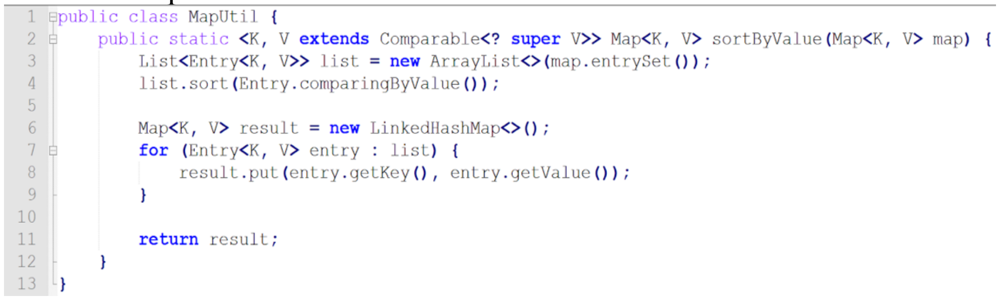
4.1.3 Functional Requirements

REQ-1: This data structure feature requires correct input data. We will set a process of checking input data. If the data is not correct.

4.2 Using quick sort as primary sorting algorithm

4.2.1 Description and Priority

For the result of election, we use java built-in quick sort to sort candidates by number of ballots. At first, we convert the election information dictionary to array, and call comparable interface to sort.



```java
public class MapUtil {
    public static <K, V extends Comparable<? super V>> Map<K, V> sortByValue(Map<K, V> map) {
        List<Entry<K, V>> list = new ArrayList<>(map.entrySet());
        list.sort(Entry.comparingByValue());

        Map<K, V> result = new LinkedHashMap<>();
        for (Entry<K, V> entry : list) {
            result.put(entry.getKey(), entry.getValue());
        }

        return result;
    }
}
```

4.2.2 Stimulus/Response Sequences

User executes the function and the system sorts the list to generate the sorted result.

4.2.3 Functional Requirements

REQ-1: Before sorting, we need to make sure dictionary has stored ballots information.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The computer executing the program is recommended to have at least Intel 5-gen dual-core i3 CPU at 1.60GHz or AMD CPU that has similar performance, and with at least 2GB RAM. The recommended graphic requirement is at least DirectX 10 compatible GPU with at least 512mb RAM.

5.2 Safety Requirements

Voting system only prompt the user for the filename. And all information that is needed to run the program should come through the file or you determine based on the file

5.3 Security Requirements

The file read in should have the fixed format will all required information.
Users need to log into the system before running it.

The file structure of voting system cannot be changed outside of the program since the election files will come in the predetermined format.

5.4 Software Quality Attributes

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 the user with basic features. Users can watch a video before using the software.

5.5 Business Rules

Voting System can be used for business purposes with authorization. If anyone uses this software for business purposes without consent, the perpetrator will be prosecuted to the fullest extent according to the law.

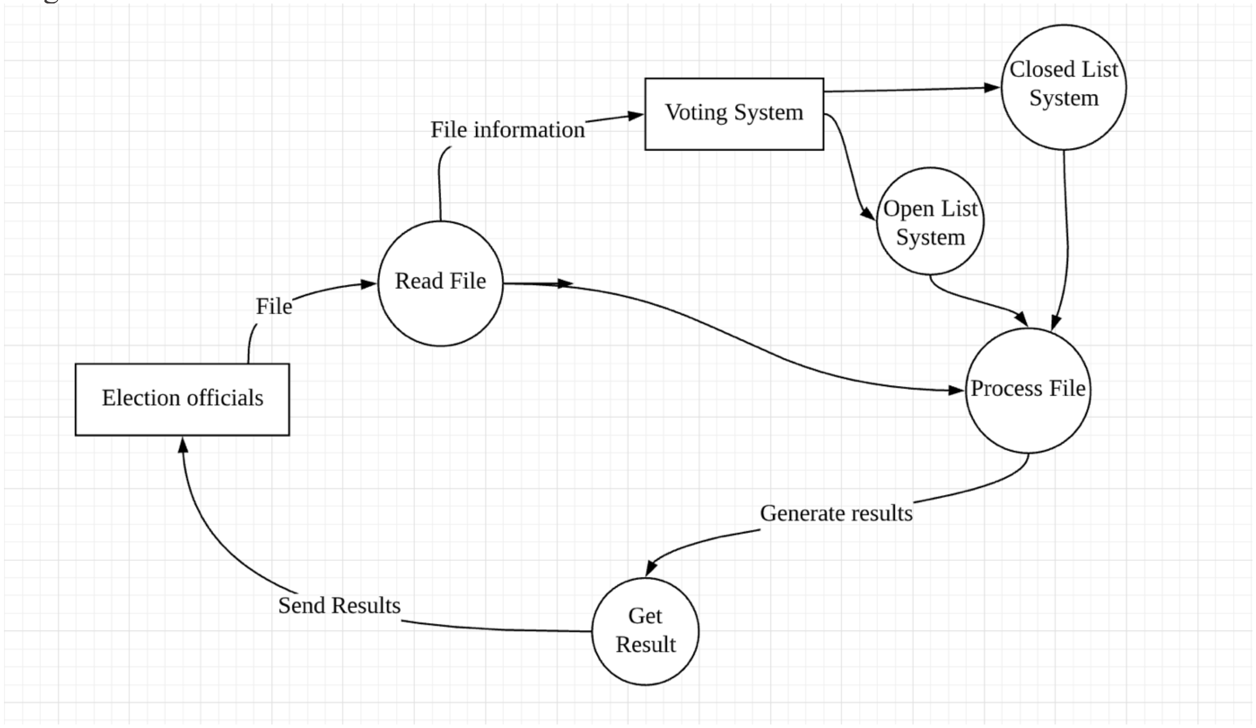
6. Other Requirements

Appendix A: Glossary

1. Package java.lang: Provide classes that are fundamental to the design of the Java programming language.
   https://docs.oracle.com/javase/7/docs/api/java/lang/package-summary.html#package_description
2. HashMap: It is a Map based collection class that is used for storing Key & value pairs, it is denoted as HashMap<Key, Value> or HashMap<K, V>. java.util.HashMap or its super class is imported to use the HashMap class and methods.
   https://beginnersbook.com/2013/12/hashmap-in-java-with-example/
3. Comparator: Comparator interface is used to order the objects of user-defined classes. A comparator object is capable of comparing two objects of two different classes. Following function compare obj1 with obj2. java.util.Comparator is imported.
   https://www.geeksforgeeks.org/comparator-interface-java/
4. Quicksort: QuickSort is a Divide and Conquer algorithm. It picks an element as pivot and partitions the given array around the picked pivot. Quicksort can operate in-place on an array, requiring small additional amounts of memory to perform the sorting.
   https://en.wikipedia.org/wiki/Quicksort
5. Package javax.mail.internet: It is imported to connect the system to email, then send the election results to media personnel.
   https://docs.oracle.com/javaee/6/api/javax/mail/internet/package-summary.html
6. Package javax.swing.JOptionPane: It is imported to pop up windows to hint errors and log in/out information.
   https://docs.oracle.com/javase/7/docs/api/javax/swing/JOptionPane.html

Appendix B: Analysis Models

Data flow diagram is used to show the way that data flows through a system. Below is the diagram.



Appendix C: To Be Determined List

To be determined.