Software Design Document (SDD) Template

Software design is a process by which the software requirements are translated into a representation of software components, interfaces, and data necessary for the implementation phase. The SDD shows how the software system will be structured to satisfy the requirements. It is the primary reference for code development and, therefore, it must contain all the information required by a programmer to write code. The SDD is performed in two stages. The first is a preliminary design in which the overall system architecture and data architecture is defined. In the second stage, i.e. the detailed design stage, more detailed data structures are defined and algorithms are developed for the defined architecture.

This template is an annotated outline for a software design document adapted from the IEEE Recommended Practice for Software Design Descriptions. The IEEE Recommended Practice for Software Design Descriptions have been reduced in order to simplify this assignment while still retaining the main components and providing a general idea of a project definition report. For your own information, please refer to [IEEE Std 1016­-1998](http://www.cs.concordia.ca/%7Eormandj/comp354/2003/Project/ieee-SDD.pdf)[[1]](#footnote-0) for the full IEEE Recommended Practice for Software Design Descriptions.

Team 3

**Voting System**

Software Design Document

Names:

Hailin Archer - deak0007

Bryan Baker - bake1358

Colin Kluegel - klue0037

Josh Spitzer-Resnick - spitz123

Class: CSCI 5801

Semester: Spring 2020

Date: (03/18/2020)

**TABLE OF CONTENTS**

[**1. INTRODUCTION**](#_heading=h.i8xywdq1z2j5) **2**

[1.1 Purpose](#_heading=h.mr4x8iq2v1zs) 2

[1.2 Scope](#_heading=h.wvhcc89do0tg) 2

[1.3 Overview](#_heading=h.razo2ge83a2s) 2

[1.4 Reference Material](#_heading=h.63c3ili9n86f) 2

[1.5 Definitions and Acronyms](#_heading=h.vy936vu13y5i) 2

[**2. SYSTEM OVERVIEW**](#_heading=h.houcqza1d72w) **3**

[**3. SYSTEM ARCHITECTURE**](#_heading=h.lbxc14vgrmo8) **4**

[3.1 Architectural Design](#_heading=h.b8obowzh707l) 4

[3.2 Decomposition Description](#_heading=h.8vbrf1z9kzsw) 5

[3.3 Design Rationale](#_heading=h.mehutlwou0o2) 5

[**4. DATA DESIGN**](#_heading=h.40jl9u8y5wjw) **6**

[4.1 Data Description](#_heading=h.vn8f2bu6kdoh) 6

[4.2 Data Dictionary](#_heading=h.wd4wm36waxlj) 6

[**5. COMPONENT DESIGN**](#_heading=h.21ixsy4vlzpx) **6**

[**6. HUMAN INTERFACE DESIGN**](#_heading=h.urhjepoexc8) **7**

[6.1 Overview of User Interface](#_heading=h.107c37n6zbbo) 7

[6.2 Screen Images](#_heading=h.rmunt9q6076u) 8

[6.3 Screen Objects and Actions](#_heading=h.w57zqsacz8cg) 10

[**7. REQUIREMENTS MATRIX**](#_heading=h.wlj6xyxxhi8k) **11**

[**8. APPENDICES**](#_heading=h.ykxehd2awlp0) **11**

## 1. INTRODUCTION

### 1.1 Purpose

*Identify the purpose of this SDD and its intended audience. (e.g. “This software design document describes the architecture and system design of XX. ….”).*

This Software Design Document provides the design details of the Voting system.

The expected audience are our class instructor, TA and our team members.

### 1.2 Scope

*Provide a description and scope of the software and explain the goals, objectives and benefits of your project. This will provide the basis for the brief description of your product.*

This document contains a complete description of the design of the Voting System.

The goal of the voting system is to provide users with election results of STV (Single Transferable Vote) algorithm and plurality algorithm. Users will need to provide the system with the number of seats, select election algorithm and csv files containing ballot information.

The basic architecture is a stand alone program consisting of following major objects: UserInterface, Election, ElectionRecord, ResultDisplay, Candidate, Ballot, Logger.

### 1.3 Overview

*Provide an overview of this document and its organization.*

The remaining chapters and their contents are listed below.

Section 2 is system overview.

Section 3 is the Architectural Design that specifies the objects that collaborate to perform all the functions included in the system. Each of these objects has an Abstract Description concerning the services that it provides to the rest of the system.

Section 4 lists Data Structure Design.

Section 5 describes Components.

Section 6 discusses the User Interface Design.

Section 7 shows the relationship between the VS system’s components and SRS requirements.

### 1.4 Reference Material

*This section is optional.*

*List any documents, if any, which were used as sources of information for the test plan.*

https://www.slideshare.net/peny\_mg/sdd-software-des-sample

### 1.5 Definitions and Acronyms

*This section is optional.*

Provide definitions of all terms, acronyms, and abbreviations that might exist to properly interpret the SDD. These definitions should be items used in the SDD that are most likely not known to the audience.

|  |  |
| --- | --- |
| Term | Definition |
| VT | Voting System |
| STV | Single Transferable Vote |
| Plurality |  |
| SRS | System Requirements Specification |

## 2. SYSTEM OVERVIEW

Give a general description of the functionality, context and design of your project. Provide any background information if necessary.

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(s) 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.

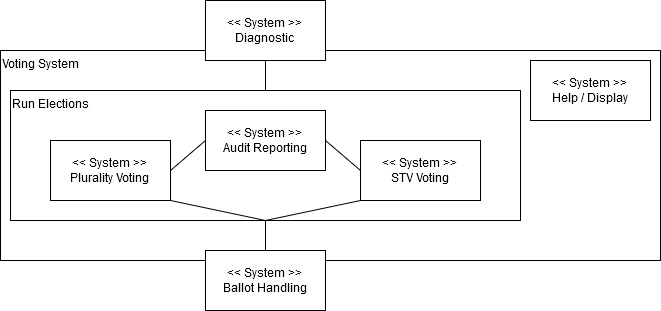
The software will also require a diagnostic mode. The diagnostic mode will be entered using a command line option. The diagnostic mode is required to support an option to disable ballot shuffling so the system can be calibrated. The diagnostic mode will also have options for developers to use to debug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

The above information leads to a list of the following sub-systems within the voting system:

* Ballot handling system
* Plurality voting system
* Single transferable voting (STV) system
* Audit reporting system
* Diagnostic system
* Help / display system

The interactions of these systems can be seen in the system diagram below.

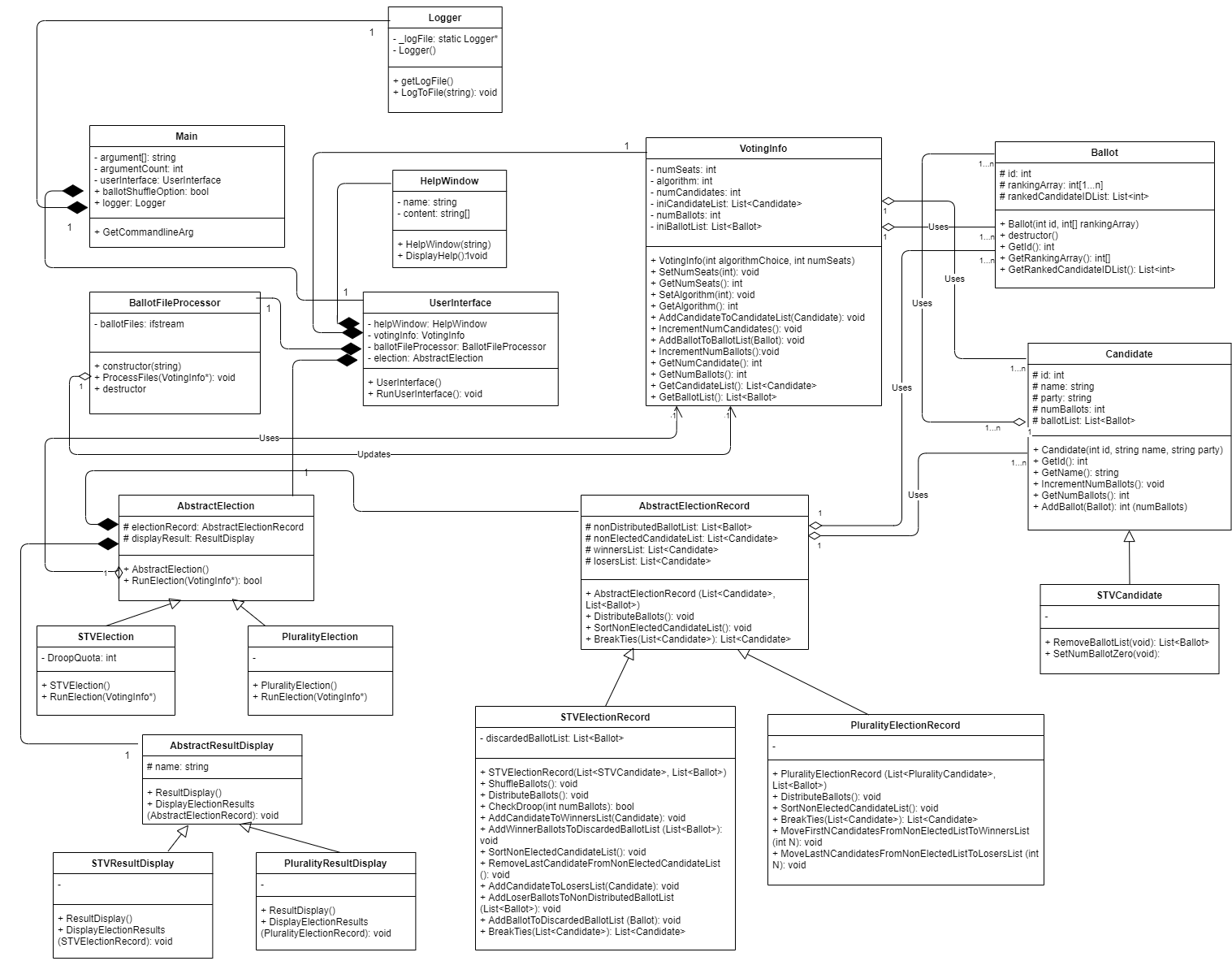


## 3. SYSTEM ARCHITECTURE

### 3.1 Architectural Design

*Develop a modular program structure and explain the relationships between the modules to achieve the complete functionality of the system. This is a high level overview of how responsibilities of the system were partitioned and then assigned to subsystems. Identify each high level subsystem and the roles or responsibilities assigned to it. Describe how these subsystems collaborate with each other in order to achieve the desired functionality. Don’t go into too much detail about the individual subsystems. The main purpose is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together. Provide a diagram showing the major subsystems and data repositories and their interconnections. Describe the diagram if required.*

Will insert the UML diagram here and make a detailed description of each class, like the SDD example.



Main class:

### 3.2 Decomposition Description

Provide a decomposition of the subsystems in the architectural design. Supplement with text as needed. You may choose to give a functional description or an object­ oriented description. For a functional description, put top ­level data flow diagram (DFD) and structural decomposition diagrams. For an OO description, put subsystem model, object diagrams, generalization hierarchy diagram(s) (if any), aggregation hierarchy diagram(s) (if any), interface specifications, and sequence diagrams here.

### 3.3 Design Rationale

Discuss the rationale for selecting the architecture described in 3.1 including critical issues and trade/offs that were considered. You may discuss other architectures that were considered, provided that you explain why you didn’t choose them.

The rationale is to have classes for all major system components.

## 4. DATA DESIGN

### 4.1 Data Description

*Explain how the information domain of your system is transformed into data structures. Describe how the major data or system entities are stored, processed and organized. List any databases or data storage items.*

The ballots are stored in Ballot objects, which contain lists of ranked Candidate ids and the current Candidate the Ballot is assigned to.

The candidates are stored in Candidate objects, which contain lists of Ballots assigned to that candidate.

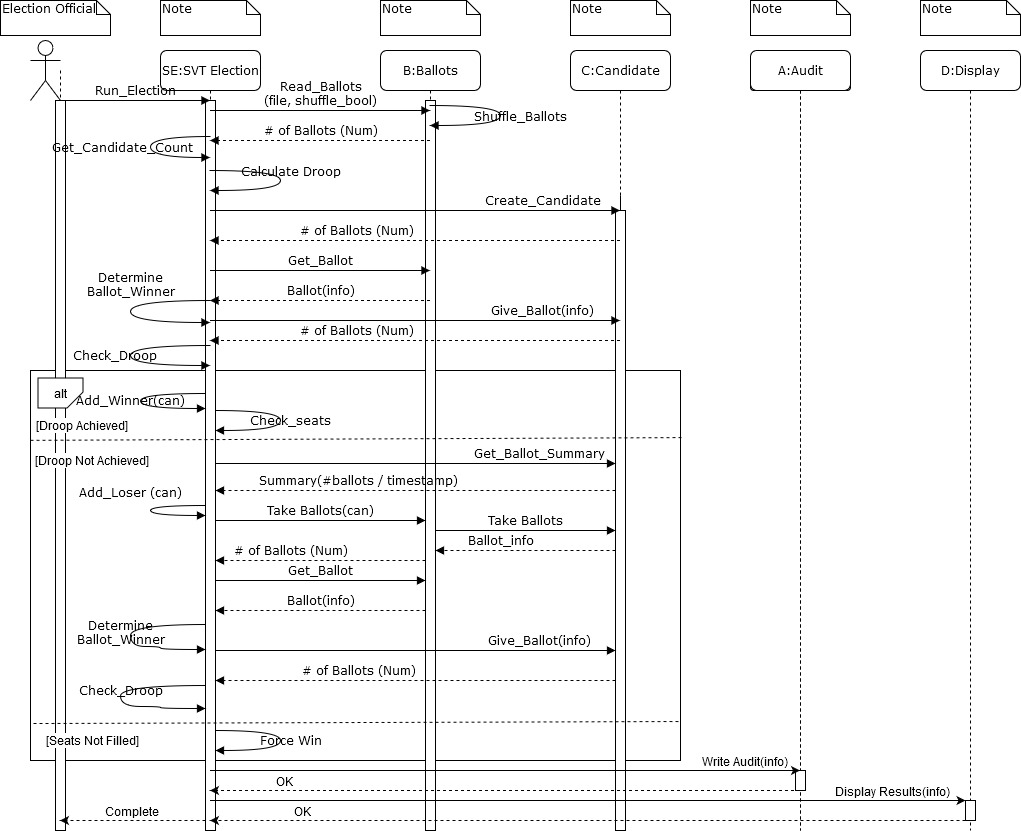
### 4.2 Data Dictionary

*Alphabetically list the system entities or major data along with their types and descriptions. If you provided a functional description in Section 3.2, list all the functions and function parameters. If you provided an OO description, list the objects and its attributes, methods and method parameters.*

## 5. COMPONENT DESIGN

In this section, we take a closer look at what each component does in a more systematic way. If you gave a functional description in section 3.2, provide a summary of your algorithm for each function listed in 3.2 in procedural description language (PDL) or pseudocode. If you gave an OO description, summarize each object member function for all the objects listed in 3.2 in PDL or pseudocode. Describe any local data when necessary.

The sequence diagram for the SVT system is given below.



## 6. HUMAN INTERFACE DESIGN

### 6.1 Overview of User Interface

Describe the functionality of the system from the user’s perspective. Explain how the user will be able to use your system to complete all the expected features and the feedback information that will be displayed for the user.

The user interface will consist of 3 separate windows, a startup/information gathering window, an election results window, and a help window.

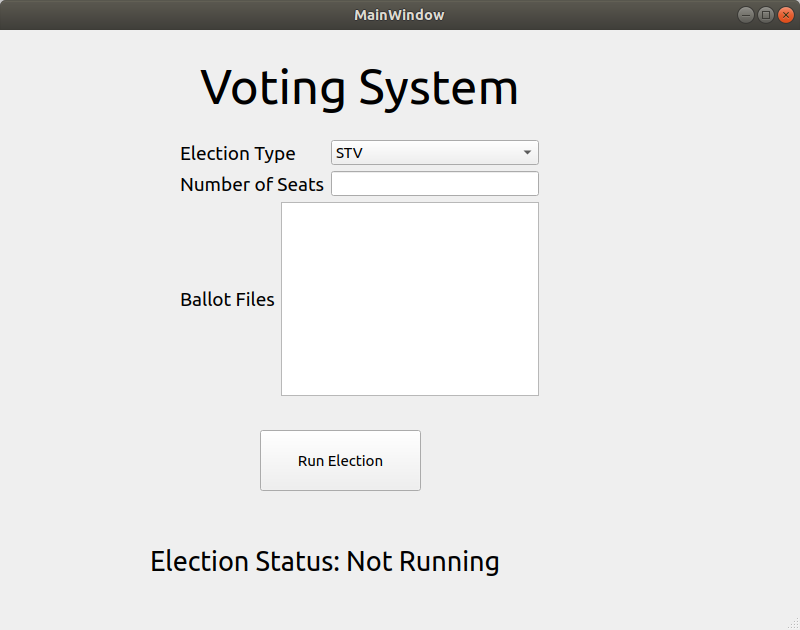
**Startup/Information gathering:** The Startup/information gathering window will be the first window that the user sees when starting the Voting System program. The window will have fields where the user can enter the necessary information to run the election, which includes the election type (STV/plurality), the number of seats, and a field to list the ballot files to be used. The window will also have a button to run election and will show the status of the election (not running/Running/Complete). The window will also have a menu bar where the user can access the help menu.

**Election results window:** The election results window will display information of about the election after it is run. It will display the results of the election and addition information about the election. This additional information will include election type(STV/plurality), number of ballots, number of seats, number of candidates and the droop quota(STV election only). There will also be a menu bar where the user can access the help menu.

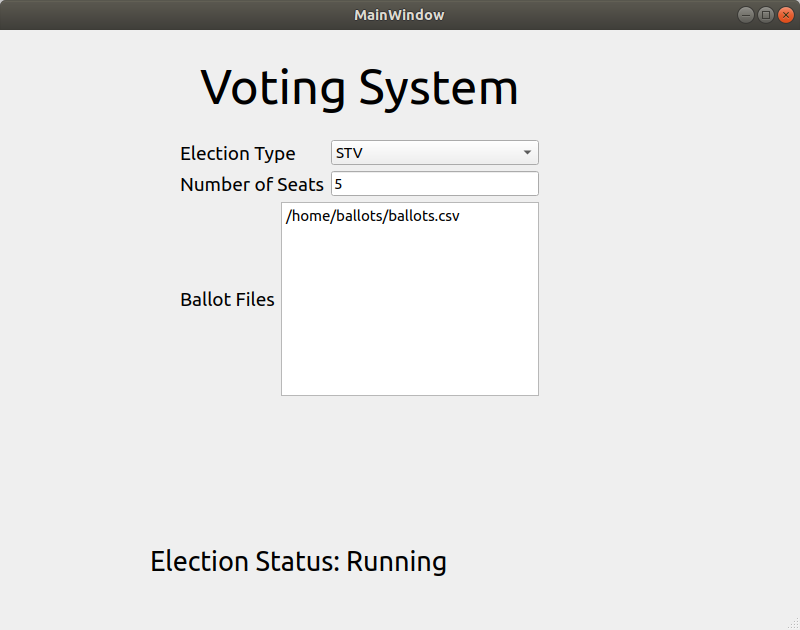
**Help Window:** The help window will display helpful information/ user guide to the user.

### 6.2 Screen Images

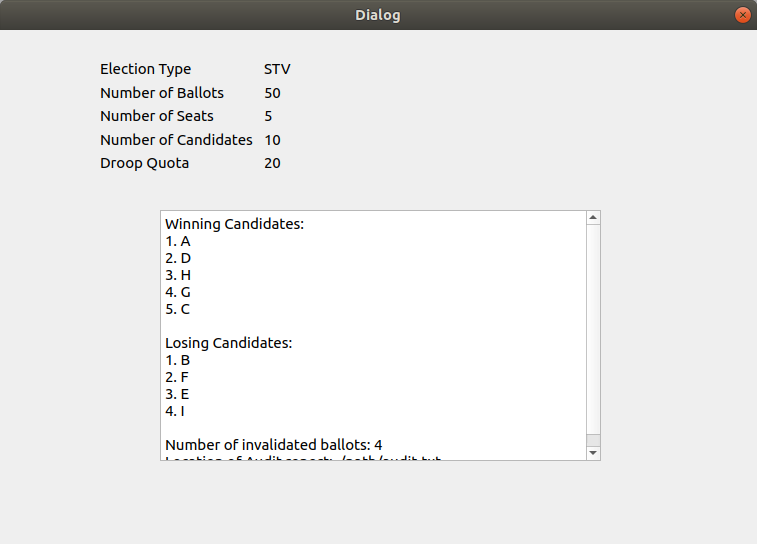
Startup Screen/Election Configuration Screen



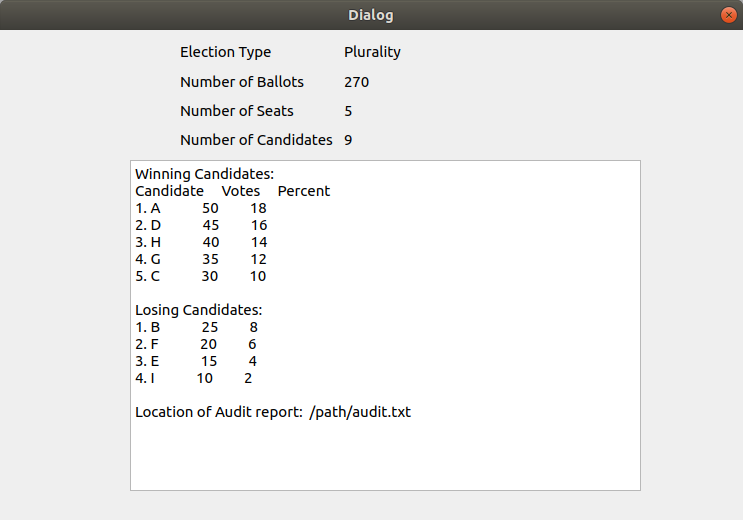
Screen while election is running



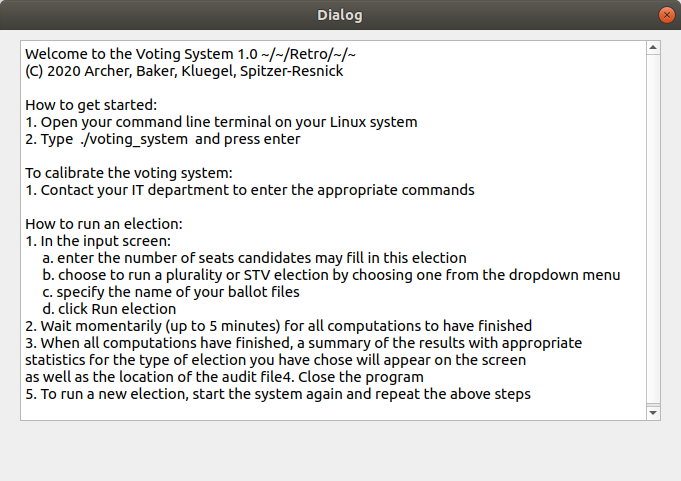
STV election results screen



Plurality election results screen



Help Screen



### 6.3 Screen Objects and Actions

A discussion of screen objects and actions associated with those objects.

**Startup/Information gathering window**: The Startup/information gathering window will have the following screen objects:

* Dropdown menu to allow the user to select the election type, STV or plurality.
* Free form text area where the user enters the number of seats in the election.
* Larger free form text area where the user can list the names of the ballot files to use.
* “Run Election” button that when clicked will start running an election from the user given inputs.
* Election status field that will show when an election is not running(has not been started), is running, or is complete.
* Menu bar that will allow the user to access the help menu.

**Election results window:**  The election election results window will have the following screen objects

* Fields listing information about the election such as Election type, number of ballots, number of seats, number of candidates, and droop quota.
* Non-editable text field that will display the results of the elections.
* Menu bar that will allow the user to access the help menu.

**Help menu:** The help menu will have the following screen objects

* Non-editable text field that will display the help text.
* Menu bar with an exit option to allow the user to exit the window and return to the main program.

## 7. REQUIREMENTS MATRIX

Provide a cross ­reference that traces components and data structures to the requirements in your SRS document.

Use a tabular format to show which system components satisfy each of the functional requirements from the SRS. Refer to the functional requirements by the numbers/codes that you gave them in the SRS.

## 8. APPENDICES

*This section is optional.*

Appendices may be included, either directly or by reference, to provide supporting details that could aid in the understanding of the Software Design Document.

1. [http://www.cs.concordia.ca/~ormandj/comp354/2003/Project/ieee](http://www.cs.concordia.ca/%7Eormandj/comp354/2003/Project/ieee)­SDD.pdf [↑](#footnote-ref-0)