Software Requirements Specification

for

V-Pal

Version 1.0 approved

Prepared by: Joel Alfveby, alfve012, Team 17

Bat-Ider Ganbold, ganbo011, Team 17

Kanoog Moua, mouax677, Team 17

Ramanish Singh, singh891, Team 17

Dept. of Computer Science & Engineering, University of Minnesota

CSci 5801: Software Engineering

February 19, 2021

Table of Contents

All authors had equal contribution to this document.

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of V-Pal and its implications. 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 software will be able to process two types of ballots based on a voting system which are *IRV (Instant Runoff Voting)* and *OPLV (Open Party List Voting)*, respectively. When given an input of formatted ballots, this program will process the ballots and output a winner for the election as well as an audit file and media summary file with the election information at the time. This document is intended for potential developers, testers, and election officials.

1.2 Document Conventions

This document follows the IEEE standard Software Requirements Specification documentation guidelines. The conventions stated in this section applies to every requirement statement outlined in this document. Statements or words of high and low level importance in this document are both bold and italicized: ***important***. Any terms that are of significant note to understanding the document is *italicized.*

1.3 Intended Audience and Reading Suggestions

- Election officials who will be using the software for counting ballots

  - As election officials who would be familiar with the voting systems they should directly go to the 2.1 for program functions and familiarize themselves with section 4 as it will allow deeper understanding of the software.

- Testers, such as (in this case) professors or TA, who would want to test the program to see if it is in working condition.
  - Testers who are experimenting with this software should read the section 1.1,1.2 as a start. Then they should move on to 2.1 to familiarize themselves with the voting systems. Section 2.2 should prove helpful in figuring out the functional aspect of the software. Then they should finish the document by reading Section 4.
- Programmers who are interested in working on the project by further developing it or fix existing bugs
  - Programmers who are interested in working on the project should first familiarize themselves with section 1.1 and 1.2. They should also read 2.1 alongside 2.4 and 2.5. Also they must fully read and understand sections 3 as well as 4.
- Typical users, such as students, who want to use this program for analyzing or projecting a typical election.
  - Typical users who analyze and tinker with this software should be aware of its hardware and software requirements before doing so. First they should accustom themselves with section 1 as it will explain in detail what the software is about and how it functions. They should then move on to section 2 and 4 for more detail.

1.4 Product Scope

V-Pal is a software system that processes the counting of ballot votes and to determine a winner for the election seasons. This system is designed to be compatible with *Instant Runoff Voting ballots* and *Open Party Listing* Voting systems, which are outlined in Section 2.1. This software synthesizes ballot counts in accordance with the selected voting system to determine the election outcome.

V-Pal is intuitive and easy to navigate as there is a provided step-by-step guide that walks the user through on how to use it. It auto-generates a separate audit file and media summary file after each run of the software system. The simple application and usability of V-Pal maximizes the election official's work efficiency and decreases the amount of time it takes to process ballot counting for the elections.

The mission of V-Pal is to efficiently carry out IRV and OPLV (refer to Section 2.1) ballot counting for the elections. It seeks to reduce the amount of time it takes to process ballot counts for these these voting systems, thus allowing for the media to promptly receive a summary of the election results.

1.5 References

FairVoteMinnesota. (n.d.). *How RCV Works | FairVote MN*. Retrieved February 6, 2021, from https://www.fairvotemn.org/rcv/

Wiegers, K. E. (n.d.). *IEEE Template for System Requirement Specification Documents*. IEEE Template for System Requirement Specification Documents. Retrieved February 6, 2021, from https://goo.gl/nsUFwy

Watters, S. (n.d.). *Software Requirements Specification (SRS) Document for Voting System*. (SRS) Document for Voting System. Retrieved February 6, 2021, from https://canvas.umn.edu/courses/217913/files/18790599?wrap=1

Alfveby, J., Ganbold, B.-I., Moua, K., & Singh, R. (2021, February 19). *Use Cases of Team 17*. Use Cases of Voting Systems (Team 17).

Varvoutas, K. (2017, February). *Software Requirements Specification for Gephi*. Software Requirements Specification for Gephi. https://gephi.org/users/gephi_srs_document.pdf

Teamleader, J., Adams, P., Baker, B., & Charlie, C. (2004, April 15). *Software Requirements Specification of Web Publishing System*. Software Requirements Specification of Web Publishing System <>. https://canvas.umn.edu/courses/217913/files/18708947?wrap=1

2. Overall Description

2.1 Product Perspective

<Software name> was developed for the purpose of processing election ballots for IRV and OPL voting systems.

Instant runoff voting is a system where voters rank the candidates based on preference. It is also referred to as alternative voting. This voting system is often adopted to ensure that the winning candidate has the support of the majority of voters in the district.

How it works: All candidates are listed on the ballot and instead of voters only choosing one candidate, they will rank them from 1 to *n*, in order of preference. Where a mark of “1” indicates that that is the most preferred candidate and a mark of “2” indicates that the candidate is the second preference, etc. This is done by filling in the corresponding column boxes next to the candidates name, where the boxes are number from 1 to *n*.

To determine the winner, the candidate with the highest amount of number one preferences is counted and if they receive more than 50% of the votes, then they are the elected winner. If none of the candidates on the ballot received more than 50% of the number one preference vote, then the candidate with the lowest amount of number one votes is eliminated. The ballots of voters that listed the eliminated candidate as their number one preference, is then redistributed to their second preferred candidate and the votes are recounted. This process of elimination continues until one candidate receives the majority of votes. If it comes down to a tie between candidates, then a fair process is followed to determine the winning candidate from there.

Example IRV:

	First Count	Second Count		Third Count	
Candidates & Parties	Original First Choice Votes	Transfer of Royce's Votes	New Totals	Transfer of Chou's Votes	New Totals
Susan Rosen (Dem.)	43,000	+ 0	43,000	+ 5,000	48,000
*Nina Kleinberg (Rep.)	42,000	+ 6,000	48,000	+ 4,000	52,000
Thomas Chou (Ind.)	8,000	+ 1,000	9,000	-------	-------
Edward Royce (Libert.)	7,000	-------	-------	-------	-------

*Winning candidate

Figure 1: Example of instant runoff voting scenario

An example IRV scenario is present in Figure 1. Total number of votes casted is 100,000. No candidate has received a majority. Therefore in the first stage the candidate who has the least number of votes is determined. In this case, it is Royce and he is eliminated. Then the counting progresses into stage two where the ballots casted for Edward are checked, and distributed to the second choice mentioned in those ballots. The votes are counted again and the candidate with the least votes is eliminated (Chou is eliminated in stage two). The 9000 ballots that Chou had in second stage will be redistributed to Susan and Nina depending on who was listed at a higher preference in the ballots that Chou had in stage 2. Since even after stage two, no candidate has a clear majority, the calculation process goes into the third stage. In the final stage, it is determined that Nina has the most votes so she is declared the winner.

Open party listing is a component of Party list voting systems. In this system, candidates are elected in large, multi-member districts. The ballot consists of each party and their list of candidates running in that party equal to the number of seats in the district. In an open party listing system, voters express their preference for certain candidates and not just the party itself. This system allows for voters to decide which candidates in the party they want to elect into office. Voters will then vote for the candidate that they prefer and by voting for the candidate, they indirectly vote for the party.

The most common way to calculate seats for the party, is the “largest remainder formula”. A quota is first calculated by dividing the number of valid votes by the number of seats available. Each vote for the party that reaches the quota or threshold, gains them another seat in the legislation, which will equal the number of candidates that wins a seat in that party.

Example OPLV:

Parties	Votes	First Allocation Of Seats	Remaining Votes	Second Allocation of Seats	Final Seat Total	% of Vote to % of Seats
Republican	38,000	3	**8,000**	1	4	38% / 40%
Democratic	23,000	2	3,000	0	2	23% / 20%
Reform	21,000	2	1,000	0	2	21% / 20%
Green	12,000	1	2,000	0	1	12% / 10%
Moll	6,000	0	**6,000**	1	1	6% / 10%

Figure 2: Example of open party list voting scenario

Figure 2 contains the details of votes obtained by each candidate. The total number of votes is 100,000 and the total number of seats is 10. So the quota comes out to be equal to 10000. In the first allocation of seats, Republicans win 3 seats as floor(38000/100000) is 3 and they have 8000 remaining votes. Similarly, Democrats,

Reform, Green, and Moll win 2, 2, 1, and 0 seats respectively. In the second allocation, we sort the parties depending on the remaining votes they have (higher to lower) and allocate the remaining seats between them. In the example, Republicans, Democrats, Reform, Green, and Moll will 4, 2, 2, 1, and 1 seats respectively.

V-Pal is designed to be a self-contained system that interacts with csv files containing ballots from either the IRV or OPL voting system. It auto-generates a unique audit file detailing the process of counting the votes and the calculations of determining the winning party or person after each run. It will also auto-generate a summary file specifically for the media to report the results to the public.

Diagram of system components:

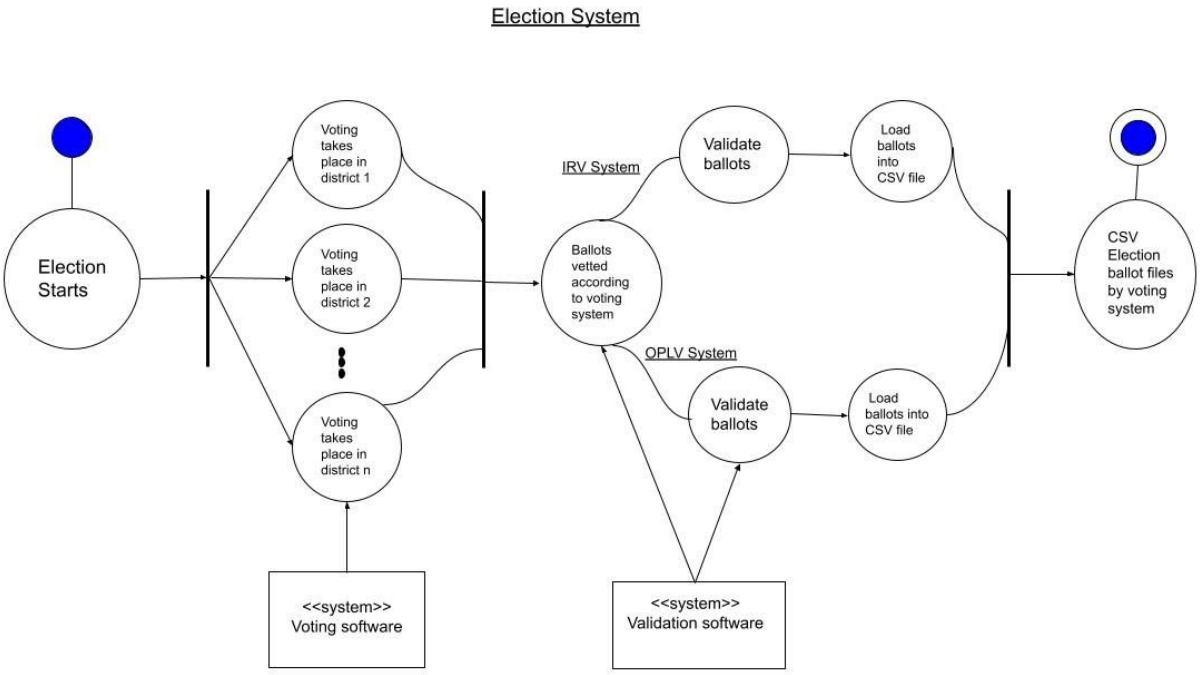

Figure 3. This diagram depicts the election process of voting and synthesizing the ballots to the respective csv file according to the two different voting systems.

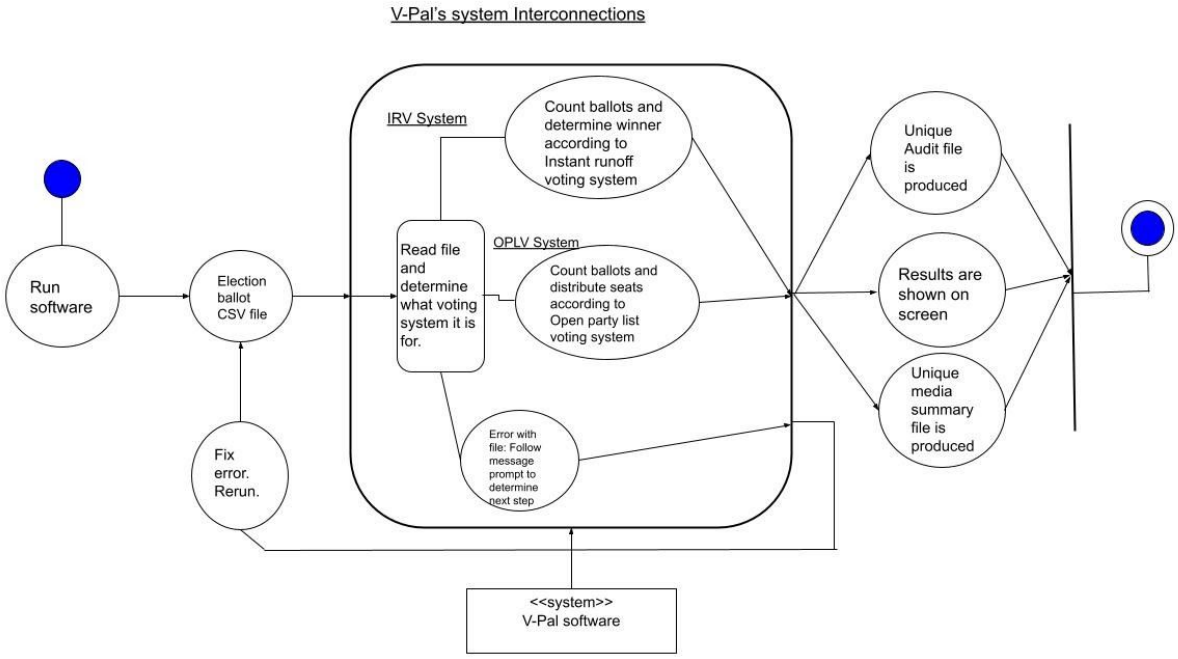

Figure 4. This diagram shows the connections between the election csv files in configuration with V-Pal's system and subsystem.

2.2 Product Functions

Before delving into this please refer to the Use Cases document (see section 1.5) for more in-depth information on each of the product functions.

- **Run**
  - To start processing the CSV file and produce output
- **New run**
  - A new run of the election with different CSV file
- **Rerun**
  - A rerun of the same election file to ensure correct calculation

- **Election method selection**
  - IRV or OPL : the voting method will be on the first line of CSV file
- **Benchmark**
  - A benchmark of the software as well as it runtime
- **Verification**
  - Recalculation of results that were calculated on a different system

- **Audit file creation**
  - Audit file containing information about the outcome as well as stats on the election are automatically created
- **Media file creation**
  - An automatic creation of election summary file for the media
- **Seat check**
  - A prompt with the information about how many seats were won by a candidate of choosing
- **Exit**
  - Program exits its current progress and terminates

2.3 User Classes and Characteristics

Election officials are expected to be adept at reading and comprehending prompts that will appear on the screen. They should possess the security clearance required to administer V-Pal.

Testers of V-Pal should have the knowledge required to read, run, and

analyze the software for testing purposes.

Future programmers of V-Pal are expected to have the knowledge required to read, run, analyze, and alter the software.

The users of V-Pal are expected to be adept at running it on their own operating system.

2.4 Operating Environment

- Windows 10

2.5 Design and Implementation Constraints

V-Pal was designed to be compatible with the outlined operating systems and hardware specifications outlined in Section 2.4. Furthermore, V-Pal was developed in C++ and necessitates a C++ compiler. It directly reads in csv files containing election ballots to be processed.

No security requirements were delineated.

Updates to V-Pal due to bugs or errors can be found on the following GitHub repository: https://github.umn.edu/umn-csci-5801-S21-002/repo-Team17

2.6 User Documentation

- Readme file will be provided with the software (in the GitHub repository)
- Doxygen to generate readable technical documentation
- UML diagrams
- Fairvote.org for more information on voting systems and which voting systems are

being used now

- User Manual provided after we complete the system
- Github repository for the source code

2.7 Assumptions and Dependencies

V-Pal was developed in C++, therefore the user(s) of this software should have the required software and hardware requirements as outlined in Section 2.4 installed beforehand.

No security requirements have been requested for this system, therefore it is assumed no unique security measures have been considered for V-Pal.

All csv files that are loaded into the software are assumed to be in the correct format (refer to Section 3.2 (i) and 3.2 (ii) for correct format) and should contain no errors. All ballots included in the csv file should be vetted beforehand so that it only contains valid ballots for the election and voting system.

V-Pal should run in the same location storage or file path as the csv files that are to be read into V-Pal's system.

3. External Interface Requirements

3.1 User Interfaces

V-pal works in the terminal. Once run, the program will prompt the user for the csv file name. When provided, the program will run, outputting results (election winner) to the display and storing the audit and media files in the current directory. Program screenshots will be provided after the implementation is complete.

3.2 Hardware Interfaces

V-pal has a minimum requirement of:

- Intel Core i7 CPU (2.4 GHz or more)
- 16 GB RAM
- Windows 10 installation required
- C++ compiler installation required

V-pal supports Windows 10. C++ compiler should be pre-installed.

The input ballot data file should follow the following formats depending on the type of election (IRV and OPL):

i) Instant runoff election

IRV

4

Rosen (D), Kleinberg (R), Chou (I), Royce (L)

6

1,3,4,2

1,,2,

1,2,3,

3,2,1,4

,,1,2

,,,1

The first line contains the information about the type of voting. Second line specifies the number of candidates. Third line contains the candidate names and parties, separated by commas. The party name is put next to the candidate name in parentheses. Party names will always be a single word (can have multiple characters). Fourth line specifies the total number of votes. Remaining lines contain the information about each vote. Voters can specify a preference list for the candidates. The preferences are separated by a comma.

ii) Open party list election

OPL

6

[Pike,D], [Foster,D],[Deutsch,R], [Borg,R], [Jones,R],[Smith,I]

3

9

1,,,,,

1,,,,,

,1,,,,

,,,,1,

,,,,,1

,,,1,,

,,,1,,

1,,,,,

,1,,,,

First line contains the information about the election. Second line specifies the total number of candidates. Third line contains the names of the candidates and their parties separated by a comma. Candidate name and party name are put into square brackets together, and they are separated by a comma. The fourth line indicates the available seats. Fifth line indicates the total number of votes casted. Remaining lines contain the information about each vote. In each vote, the position of number “one” will indicate the preferred candidate of the voter.

3.3 Software Interfaces

V-Pal requires a C++ compiler. V-Pal requires the input data to be in the csv format. The input data csv file should be written according to this format provided in section 3.2. V-Pal analyzes the ballot file and produces the media summary file (.txt format), election audit file (.txt format), and announces the winner on the screen.

The data going into the system will only be the csv file, and its name that will be provided by the user. The data coming out will be the results on the display, the summary file, and the audit file.

Across software components, the system will share and compare the votes of each candidate, along with their name and party. This data will be used for calculations, along with printing to the screen and different files.

3.4 Communications Interfaces

V-Pal uses bash terminal as an interface to interact with the user.

V-Pal repository is uploaded on GitHub (https://github.umn.edu/umn-csci-5801-S21-002/repo-Team17) and the user can pull and install after every update. Software version number information can be found on the GitHub page and in the readme.md file in the repository.

4. System Features

The use cases for this software are stored in a separate file under the name use_cases_Team17.pdf . It is stored in https://github.umn.edu/umn-csci-5801-S21-002/repo-Team17/tree/master/SRS folder.

4.1 Vote/seat counting and determining the winner

4.1.1 Description

The primary purpose of V-Pal is to determine the winner of an election. The software takes the ballot data as a csv file and processes the information present in the file. The required structure of the csv file is presented in detail in section 3.2. Depending on the type of the election (mentioned in the ballot data file), V-pal calculates the seats won by each party (OPL)/candidate (IRV) and determines a winner. In case of a tie, the winner is randomly selected.

V-pal automates vote/seat counting and determines the winner. This reduces the risk occurrence of human errors.

4.1.2 Stimulus/Response sequence

User runs V-pal from the command line in the terminal. The software then asks for the name of the input file. Once the user inputs the file name and presses the Return key, V-pal begins to analyze the data. After the analysis, the result of the election (winner/seats) and election statistics are presented in the terminal.

4.1.3 Functional requirements

REQ-1: The input file must be provided in csv (comma separated values) format.

REQ-2: Input file should have the format as defined in section 3.2.

REQ-3: Input data must be stored on the system on which the software is used.

4.2 Creating an audit file for the election

4.2.1 Description

V-pal creates an audit file for the election based on the ballots casted.The audit file contains the following information:

i) Date and time when the calculation is being performed

ii) Type of voting

a) IRV:

i) Candidates and their party

ii) Original first choice votes

iii) If someone has >50% of the vote, they win. Else, transfer of last place's votes

iv) New total number of votes

v) If someone has >50% of the vote, they win. Else, jump to step iii

b) OPL:

i) Parties

ii) Total votes for each party

iii) Number of seats won in first allocation for each party

iv) Remaining votes for each party

v) Number of seats won in second allocation for each party

vi) Final seat total for each party

vii) Percentage of total votes to percentage of seats won

iii) Number of candidates

iv) Name of the candidates

v) Number of ballots

vi) Votes won by each candidate

vii) Seats won by each candidate

viii) Winner(s)

The audit file serves as a detailed description of the ballot counting and calculation process for the voting system that it is currently working on. It can be referred to in case of any disputes and confusions. The audit file should be copied in a safe place to minimize the chances of tampering. Each audit file contains a unique name.

4.2.2 Stimulus/Response sequence

User runs V-pal from the command line in the terminal. The software then asks for the name of the input file. Once the user inputs the file name and presses the Return key, V-pal begins to analyze the data. After the analysis, the audit file is saved in the same folder where the ballot data file was stored. It would be saved under the name audit_file_{election_type}_{date}_{time}.txt.

Audit file can be viewed through terminal using vim editor or using the following software:

Windows: Notepad

4.2.3 Functional requirements

REQ-1: The input file must be provided in csv (comma separated values) format.

REQ-2: Input file should have the format as defined in section 3.2

REQ-3: Input data must be stored on the system on which the software is used.

4.3 Create election summary file for media

4.3.1 Description

When the program decides the winner of the election, a text file will be created with the results and statistics. Unlike the audit file which contains the technical details of the calculations, this file will be more readable to a wider audience.

The file will include the

i) Type of election

ii) Number of ballots cast

iii) Winning candidate, their party, and number of votes

iv) Losing candidates, their parties, and number of votes

This will be enough information to understand the results of the election, but not too much that might confuse the reader. That much detail will be left to the audit file, described in section 4.2.

4.3.2 Stimulus/Response sequence

The program will generate the summary file automatically.

i. System finishes calculating the winner

ii. System generates the summary, outputting to the main folder

iii. User goes to that folder, reads summary

4.3.3 Functional Requirements

REQ-1: The program ran successfully and found a winner

REQ-2: The program displays the winner, party, and votes on screen

REQ-3: The program generates a summary file

5. Other Nonfunctional Requirements

5.1 Performance Requirements

V-Pal should process an election file (containing up to 100,000 ballots) in under 8 minutes.

5.2 Safety Requirements

1. Overwriting media summary file

   V-Pal creates a summary file of the election which can be sent to the media agencies for publication. V-Pal saves that file under the name "media_summary_{election_type}_{date}_{time}.txt" in the current folder. Any file already present in that folder with the name "media_summary_{election_type}_{date}_{time}.txt" will be overwritten. Please create a backup of those files in a safe location before running V-Pal.

2. Creating audit file

   The audit file is saved under the name audit_file_{election_type}_{date}_{time}.txt. If V-Pal is used for a batch of ballot data files, there can be cases where two audit files have the same name (if the execution time is very less). Therefore, please make a backup of audit files present in the folder before using V-Pal.

5.3 Security Requirements

There are no security requirements for this system.

5.4 Software Quality Attributes

1. Adaptability: The simple design of V-Pal allows for it to smoothly be altered to cover other types of voting such as closed party list voting.
2. Correctness: V-Pal was developed to auto-generate audit files after each run. This allows for all steps in the ballot counting process by V-Pal to be documented and reviewed if needed. Testing units were extensively used to test the accuracy of the algorithm.
3. Interoperability: V-Pal takes the ballot files in the comma separated values (csv) format. Files in other formats can be easily converted to this format. V-Pal creates the output files in the text format (.txt) that can be read easily in any editor.
4. Maintainability: V-Pal is easy to understand and is written to pass all unit tests.
5. Reliability: V-Pal is extensively tested during all iterations to ensure the program always runs correctly.
6. Reusability: V-Pal can be used on any election that uses IRV or OPLV.
7. Robustness: Plethora of unit tests confirm the robustness of each and every function.
8. Testability: Unit test will be provided in the github repository which makes testing easy.

9. Usability: The straightforward layout and use of V-Pal maximizes the user's (outlined in Section 2.3) productivity by decreasing the amount of time it takes to process ballot counting.

5.5 Business Rules

This is an open-source project, so anyone can use and modify this project. Sign-in is not required to use the program.

6. Other Requirements

Appendix A: Glossary

IRV: Instant runoff voting

OPLV: Open Party Listing voting

CSV: Comma Separated Values, a type of file that uses commas to separate data. Commonly used in Microsoft Excel..

Tester: Anybody wanting to test the integrity of the software

Election official: A person in charge of running the election, as well as counting the votes

User: Anybody using the software

Appendix B: Analysis Models

Appendix C: To Be Determined List