Testing Report

Un-Gerrymandered Software

Customer: Shawn Squire

DIAMOND DISTRICTING

Members: Corey Atkins, Matthew Hancher, Nahum Meherete, Joey Napolitano, Nirav Shah, Eric Yoo

Date: December 5, 2017

Un-Gerrymandered Software

Testing Report

Table of Contents

Section 1: Introduction

Section 1.1: Purpose of this Document

The purpose of this document is to describe the testing process and testing results for our recently-developed software product, *Un-Gerrymandered Software*. *Un-Gerrymandered Software* performs efficient, unbiased districting for three different states: Maryland, North Carolina, and Wyoming. Testing information in this document is provided for each of the product's four use cases. The intended readership of this document includes members of the state legislature, political scientists, independent organizations responsible for drawing district lines, and most importantly our primary client, Prof. Shawn Squire.

Section 1.2: References

Diamond Districting. (2017, October 24). *Un-Gerrymandered Software System Design Document.*

Diamond Districting. (2017, October 19). *Un-Gerrymandered Software System Requirements Specification.*

Diamond Districting. (2017, November 28). *Un-Gerrymandered Software Code Inspection Report.*

Ingraham, Christopher. (2015, March 1). *This is the best explanation of gerrymandering you will ever see*. Retrieved from https://www.washingtonpost.com/news/wonk/wp/2015/03/01/this-is-the-best-explanation-of-gerrymandering-you-will-ever-see/?utm_term=.8c8e3fe95ad1

(2017, August 12). *Redistricting*. Retrieved from https://en.wikipedia.org/wiki/Redistricting#Gerrymandering

UMBC CSEE. (2015, Spring Semester). *C++ Coding Standards*. https://www.csee.umbc.edu/courses/undergraduate/202/spring15/projects/coding-standards.shtml.

Kevin Baas. (2012, November 12) *Auto-Redistrict*. http://autoredistrict.org/

The Qt Company. (2017) *Qt*. 5.6.x Offline Installers. https://www.qt.io/

Section 2: Testing Process

Section 2.1: Description

We had a fairly standard testing process. Since we followed a waterfall/spiral pattern of software development, we were able to test each section of the waterfall or spiral when completed. The first thing we worked on was the User Interface (UI), we began to work on the district selection and state selection. Although we did not have a working algorithm at the time, we repeatedly tested that the right state and number of districts would work for any given scenario. As we approached our final product, we went back and tested each use case that should exist up until that point.

After finishing up our algorithm we were then able to test the algorithm. We had three states to test; Maryland, North Carolina, and Wyoming. Since our algorithm took a very long time to compile and produce a final product, we decided to test the even, odd, and regular numbered districts for Maryland, North Carolina and Wyoming respectively. After testing each state we realized that the algorithm did not display the same final product if we try the same number of districts again. Once we got our algorithm to work properly, we were finally able to connect the algorithm and UI by making screenshots of the output from the algorithm, then placed it in the UI. We tested the connection by using the UI to produce the right state and number of districts from the pre-selected output. After testing each state and a plethora of districts, we concluded that the UI and algorithm were fully integrated as one.

Our testing process involved a lot of black box and white box testing to check if we get the desired output and if the inner mechanisms of the algorithm ran to the most optimal speed that we and the client wanted.

Section 2.2: Testing Sessions

Table 1 provides a list of five completed testing sessions for *Un-Gerrymandered Software*. For each session, the date, location, time started, time ended, the performer of the test, and the covered use case.

Table 1. Testing Sessions

Test Session Number	Date	Location	Time Started	Time Ended	Performer of the Test	Use Case Covered
1	11/18/17	UMBC	2:00PM	3:30PM	Corey & Eric	District Selection, State Selection, & Image Production of UI
2	11/30/17	UMBC	6:00PM	7:00PM	Matt & Nahum	Algorithm Accuracy/Willingness of Algorithm to work
3	12/02/17	Remote	5:30PM	6:00PM	Matt	Algorithm Accuracy/Willingness of Algorithm to work
4	12/03/17	Remote	2:00PM	3:00PM	Nirav & Joey	Connection of Algorithm with UI
5	12/04/17	Remote	3:00PM	4:00PM	Diamond Districting	Overall Product Testing

Section 2.3: Impressions of the Process

Overall the testing process we used was effective since we implemented a incremental testing process. Rather than us finishing the *Un-Gerrymandered Software* and then testing for flaws, we finished a small step of the product and then tested accordingly. The only downside was we as a group were not able to meet up in person as often as we should. Outside of that issue, our process was optimal in testing many different use cases and making the debugging effort less strenuous after the product was completely finished. Prior to testing, the program ran how we expected it to, with major issues at each stage of development. After testing at each stage we were able to overcome the major obstacles that were associated with that stage of the product. This process decreased our code flaws after completing the product. Our process minimized the repairs that were needed at the end and we were able to meet all the requirements.

The best and worst modular units that were tested remain the same as the ones mentioned in the Code Inspection Report. The best module would be the drop down menu because not only we can make selections for the state, but also for the district in a all-in-one drop down menu. The worse module would the districts being hard-coded into the GUI, making the code less dynamic.

Section 3: Test Results

Use Case 1 - Drop down menu

The drop down menu consists of all three states that the software will redistrict, once the user selects a state, the product will then prompt the user to select the number of districts to be displayed. Since we have a drop down menu rather than inquiring for user input there are not any invalid input for this use case. Having a drop down menu reduces the case of user error. Our test cases involved creating a new state with various districts ranging from one to 16 districts. The reason for this was to see if our algorithm can handle many districts in states that are relatively too small to have many districts.

The testers for this use case were Corey and Eric, since they were responsible for the design and implementation of the GUI. At first they loaded some test images to check if displaying a state would actually work. Once completed, they attempted to use a spinbox for where the user could enter a desired number of districts. However they were not able to get the functionality of the spinbox to work. This resulted in setting a fixed number of districts for the user to select. After completing the algorithm we were able to load the final product of each state and a given number of districts. Each test case came out working properly and showed an accurate representation of the state if that state was to be redistricted with an equal population.

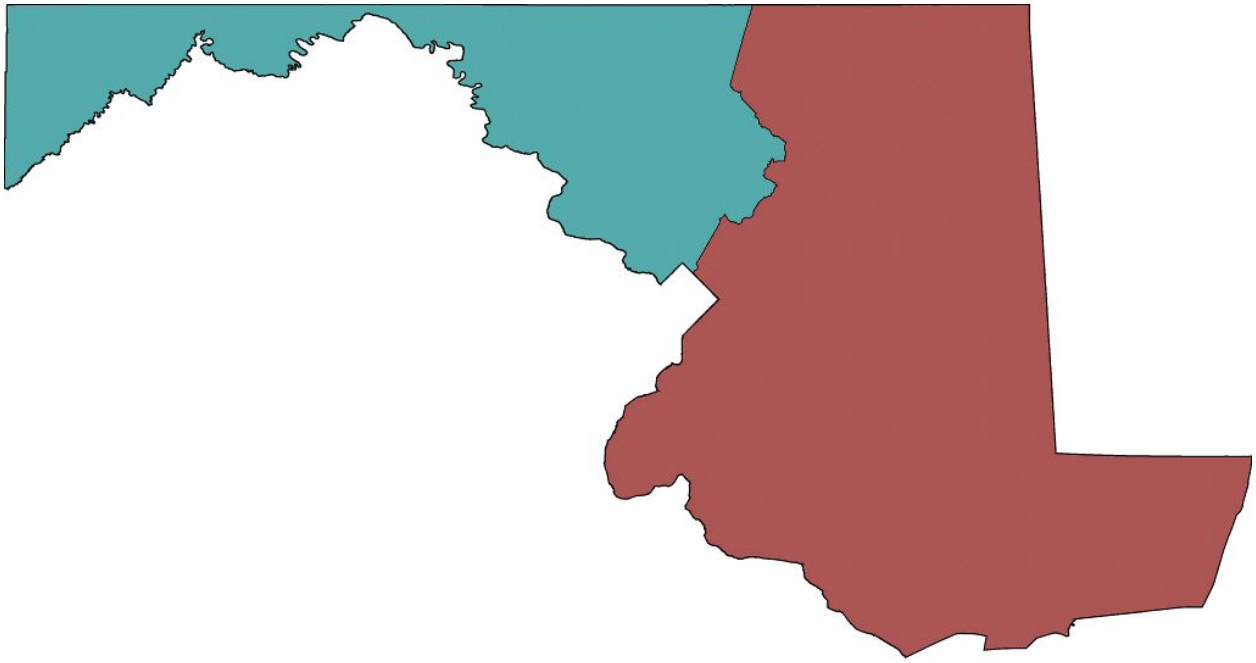
Use Case 2 - Algorithm

The hardest part of the product was trying to come up or find an algorithm that will work properly in a fashionable time. They tried many open source algorithms and at one point tried to create our own algorithm which ultimately failed. They ended up going with *Autoredistrict* as our final algorithm. The defects with *Autoredistrict* was the time to compile, the algorithm would take hours to run and produce a single shapefile of the new state. They could not really work around fixing this issue unfortunately, so to make up for the time we decided, with permission of the client, to run multiple instances of *Autoredistrict* and save the shapefiles and then load them to the GUI when the user prompts for it.

The testers for this case were Matt and Nahum. It took them a while to search, and code, the perfect algorithm. Once they discovered *Autoredistrict,* Matt and Nahum were faced with the issue of trying to fix the run time of the algorithm. Another issue that the faced was specifically the shapefile of Maryland *Autoredistrict* did not have the Chesapeake Bay included in the shapefile, it was seen as part of the land mass as seen in Figure 1. Unfortunately, they were not able to fix those two defect and had to work around them. Outside of those two defects, the

Autoredistrict algorithm worked like a charm and creates shapefiles that accurately represents how the states should end up looking like.

Figure 1 *Autoredistrict* shapefile of Maryland with Chesapeake Bay not included.



Appendix A – Agreement Between Customer and Contractor

Client Agreement

Shawn Squire
1000 Hilltop Circle
Baltimore, MD 21250

The following represents an agreement between **Diamond Districting** (hereinafter referred to as “we”, “us”, or “Diamond Districting”) and **Shawn Squire** (hereinafter referred to as “you” or “Client”). The details of this agreement are as follows:

Professional Services. The Client hereby contracts with Diamond Districting to perform a visualization of non-gerrymandered states.

Description of Services. The following services will be provided:

· Files containing all source code to Un-Gerrymandered Software

Other Terms/Customer Comments

_____
_____
_____
_____
_____

Terms and Conditions

Limited Liability. We shall not be liable for any delay due to circumstances beyond our control to provide services, including acts of God, war, government regulations, disaster, or civil disorder.

Amendments. Any changes or modifications must be specifically placed in writing, attached, dated, signed, and approved by both parties.

Cancellation. Cancellation of services should be provided to **Diamond Districting** in writing to amend the current client agreement. In the event that the client cancels the contracted services outlined in this contract, the initial payment will be forfeited.

I have read and understand the terms of the entire agreement. I hereby agree to the terms of this agreement. We both agree to make the attached Terms and Conditions as part of this Agreement.

Client

Shawn Squire: ____Shawn Squire_____ **Date** __12/5/2017_____

Diamond Districting (Team)

Corey Atkins: _____Corey Atkins_____ **Date** _12/5/2017_

Matthew Hancher: _____Matthew Hancher_____ **Date** _12/5/2017__

Nahum Meherete: _____Nahum Meherete_____ **Date** __12/5/2017__

Joey Napolitano: _____Joey Napolitano_____ **Date** _12/5/2017__

Nirav Shah: _____Nirav Shah_____ **Date** _12/5/2017__

Eric Yoo: _____Eric Yoo_____ **Date** _12/5/2017____

Appendix B - Peer Review Sign-off

All members of Diamond Districting have reviewed the document and agree on its content and format.

Corey Atkins: _____Corey Atkins_____ **Date** _12/5/2017____

Matthew Hancher: ____Matthew Hancher_____ **Date** __12/5/2017____

Nahum Meherete: ___Nahum Meherete_____ **Date** _12/5/2017___

Joey Napolitano: ____Joey Napolitano_____ **Date** _12/5/2017_____

Nirav Shah: __Nirav Shah__ **Date** __12/5/2017__

Eric Yoo: _____Eric Yoo_____ **Date** _12/5/2017____

Comments:
__None_____

_____

_____

Appendix C – Document Contributions

Atkins, Corey - Description, References, Test Results (20%)
Hancher, Matthew - Test Results (5%)
Meherete, Nahum - Description, Impressions of the Process, Appendix A, B, & C (40%)
Napolitano, Joey - References (5%)
Shah, Nirav - Testing Sessions, References, Impressions of the Process (20%)
Yoo, Eric - Testing Sessions (10%)