Design Document I

1.1 Context viewpoint

The Data Visualization Project aims to take several data spreadsheets to be used as input and discover patterns and trends that are found in the variables within the sets automatically without requiring the user to have to provide anything but the spreadsheets themselves, an important step towards solving the big data problem. At the current state of the design, the expectation is that the datasets follow a specific layout: the first row containing the column names that will represent the categorical variables and the rows that follow being the measured values of the said variable. The users and clients of this product are expected to follow this format when inputting the data into the program. This step marks the only real direct interaction that the consumers will have with the software, as the process is largely automated. The only other step in which the software will prompt the user to respond is the methods of analysis they wish to be conducted. The user has the opportunity to select certain forms of statistical analysis that they want to see from the variables in their spreadsheet, and the program returns only those analytics. The interaction between the client and the server can be seen in figure 1 below.

1.1.1 Design concerns

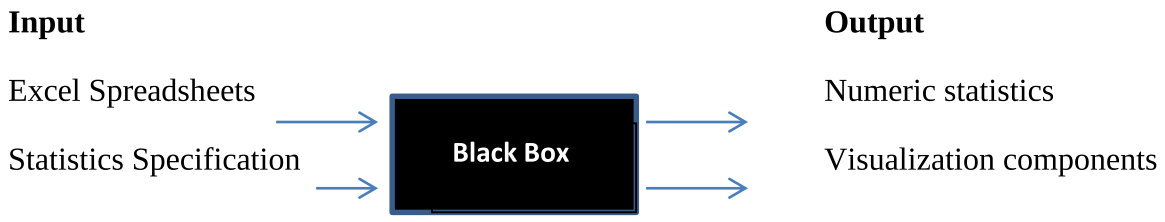
Potential design concerns that may arise when the user interprets the results from their spreadsheets include a potential system overflow from the size of the file the client may attach as input. Although the goal of the project is to be able to look at as many datasets at one time, size limitations will continue to be a constraint moving forward. Currently, a SQL database is being used to store each datasheet in its own table, both to avoid inner joins within files that would lead to a single massive file, and for organizational purposes. Another concern is by asking the user what methods of analysis they wish to see, they may not have an answer to the question. This can be due to a number of reasons, but most likely that the person may not have a strong enough statistical background to understand what each tool does in its calculation, and therefore not being able to comprehend the nature/distribution of the variables at hand to make such a decision (in the case of choosing correlation among variables in a dataset that aren’t related to each other). Finally, the project also returns a corresponding visual graphic that represents what is seen in the numeric values. Although a weightage system will be used to allocate points towards choosing the appropriate graph to return to the user in the results section, there is a chance the incorrect graph could be chosen by the computer, especially in instances of distributions that have not been encountered before. The hope is that through machine learning, the decisions made will become increasingly accurate as the familiarity with spreads of variables by the computer improves.

1.1.2 Design elements

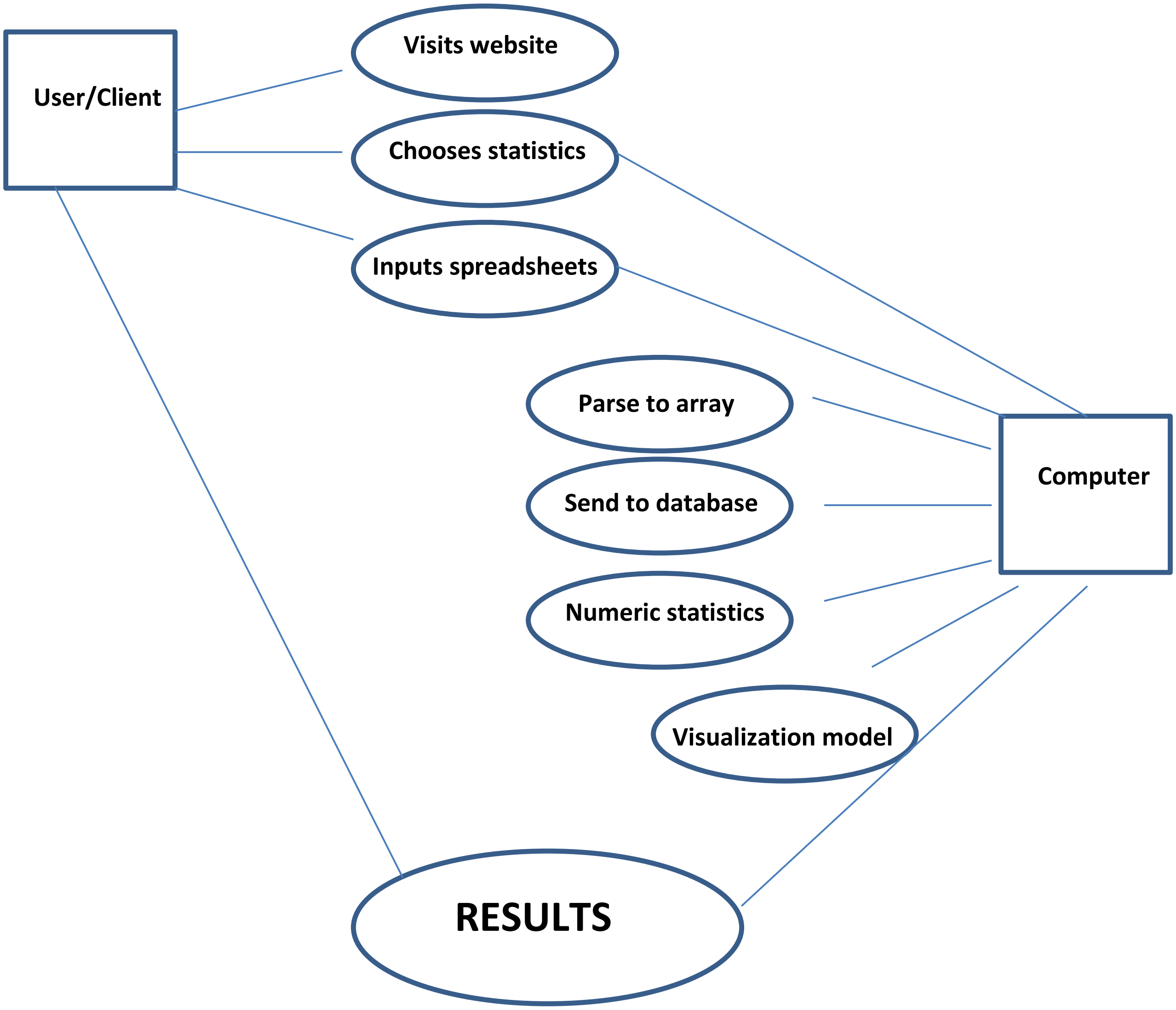
The chronological order in which the software operates is as follows: the user accesses the software by visiting the website domain for which it resides, and enters the appropriate information that they are requested, including the summary statistics that they wish to see. The spreadsheets that the user sends, assuming they follow the correct format, are downloaded and enter the program. The program launches, and begins by parsing each column into its own individual array that will be combined into a single large matrix, with the first row being the names of the columns from the spreadsheet, and the rest of the cells representing the corresponding rows of data values. The two-dimensional array is then sent to the SQL database to be stored. Not only does the SQL database enable more datasets to be stored at a single point, but it makes the retrieval of values easier. The analysis that the consumer wished to see in the final results is then calculated, which may include the measures of central tendency (mean, median, and mode), range, interquartile range, correlation, and outlier values. With the numeric values being calculated, the visualization component is run. If the user chose correlation as one of the statistics to be shown, for example, a scatterplot featuring two variables that were found to be highly related will be shown with annotations reflecting specific observations that are seen (for example, extreme data points or unusual summary statistic findings). If the dataset contains nominal variables with percentages that add up to a whole, a pie chart will be shown if they are all part of a common group. Once both the numeric and visual components have been obtained, the results will be printed on a separate page of the website that the user can then look at and copy for reference. The whole sequence can be seen in figure 2 (shown below).

1.1.3 Example languages (shown through a black box diagram and UML use case)

Black Box Overview (Figure 1)



Use Case Diagram (Figure 2)



1.2 Composition viewpoint

1.2.1 Design concerns

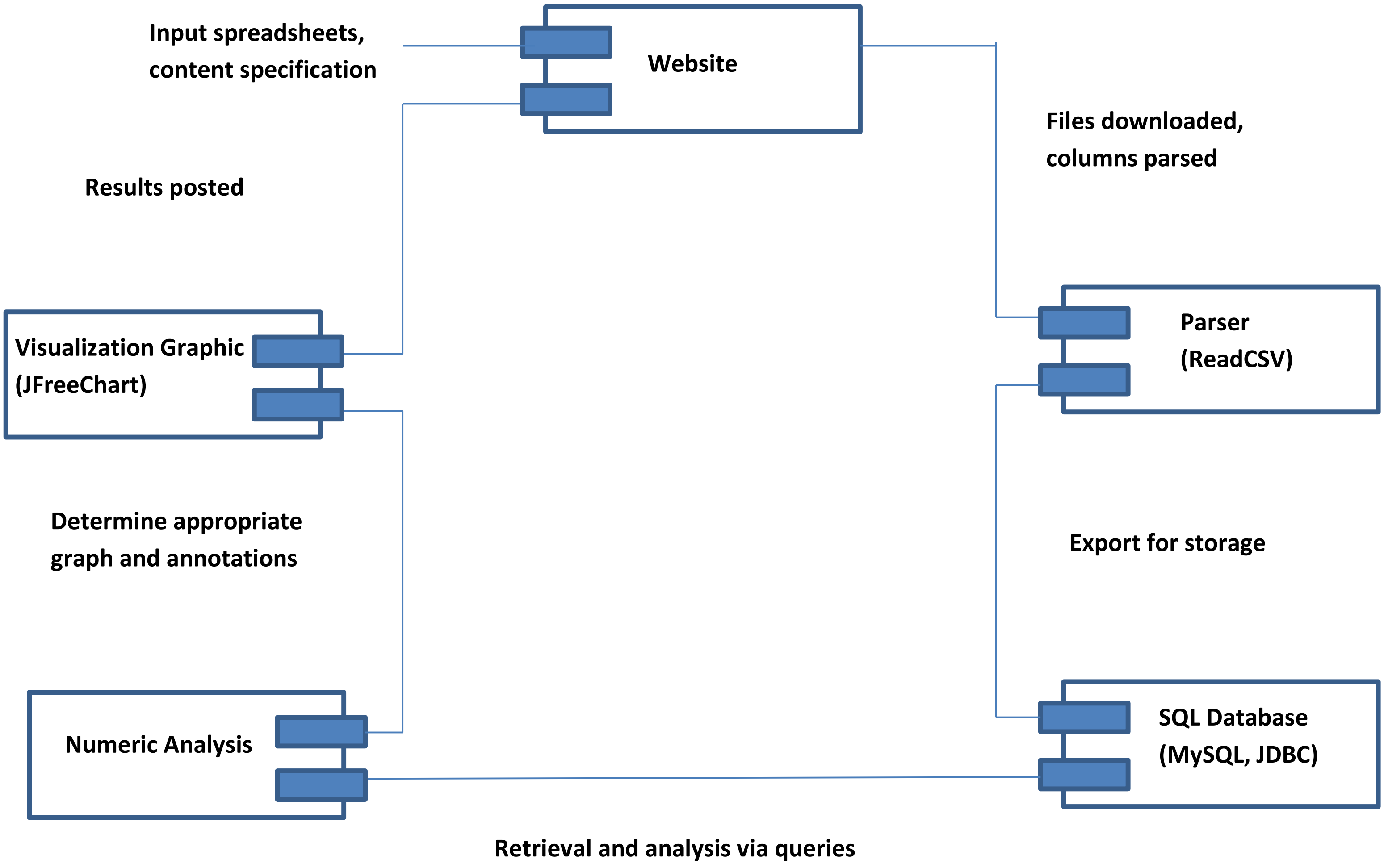
There are a few concerns involved in ensuring all of the submodules are functioning effectively. Since the website requires an online connection, there may be a need further down the road to make the software an application (with a small payment required to get the application version). This would enable users who frequently use the software to be able to use it whenever they need to without any problems if their internet were to go out. This workaround would also be a potential solution to high internet traffic on the domain. Another concern would be a spreadsheet being attached that does not fit the format guidelines. In such a scenario, the parser would not be able to do its job as intended and values would be processed in an incorrect way, which would also affect all other submodules that interact with it. The values retrieved from the database would be off and throw off the corresponding statistics estimates and final visualization model. A final concern would be the aforementioned exceeding of size constraints when the files are sent to the database. An overly large file or too many files being exported to the database could cause the server to crash or slow down the processing time it takes for the analyzing to get done.

1.2.2 Design elements (including 1.2.2.1 functional attributes and 1.2.2.2 subordinates attributes)

The Data Visualization Project can be divided into five subcomponents that contribute to the final functional model. The first part is the website, which is where the software itself resides. Although the website has not yet been constructed, the layout of the website is expected to include several components traditionally featured in graphical user interfaces. There will be JButtons to allow the user to choose the types of statistics that they want to see in the final results page. There will also likely be multiple text fields and corresponding buttons to allow the user to browse their computer for the desired .txt and .xls files they want to be analyzed. The second component of the project is the parser, which makes use of the ReadCSV library in Java to break down each column contained in the spreadsheet into its own unique array. For non-numeric variables, the string arguments are converted into doubles in order to do the analysis. These arrays are also used to determine the datatype of each variable, which allows the third component to work fully, the SQL database. The SQL database is the intermediary piece of the entire process. It is being used as a place to store the data in one compact place, but also easily retrieve only the necessary values to do the numeric analysis (the fourth component) using various SQL queries through the JDBC driver. The final component is the visualization tool, which is constructed with the help of the library JFreeChart. In making the decision as to which graph is the most optimal to choose to represent the distribution of a variable, the JFreeChart enables several different varieties of graphs to be constructed. The relationship between all modules can be seen in figure 3 below. Currently, the parser and SQL database have completed the development, testing, and integration stage. The numeric analysis has been tested and integrated but has not completed development. This is expected to be finished by late December. The development of the visualization graphics has recently commenced, and is expected to be completed with the decision-making algorithms in place by late February/Early March. The website will begin being worked on around January and continue until the end of February.

1.2.3 Example languages (shown through a UML component diagram)

UML Component Diagram (Figure 3)

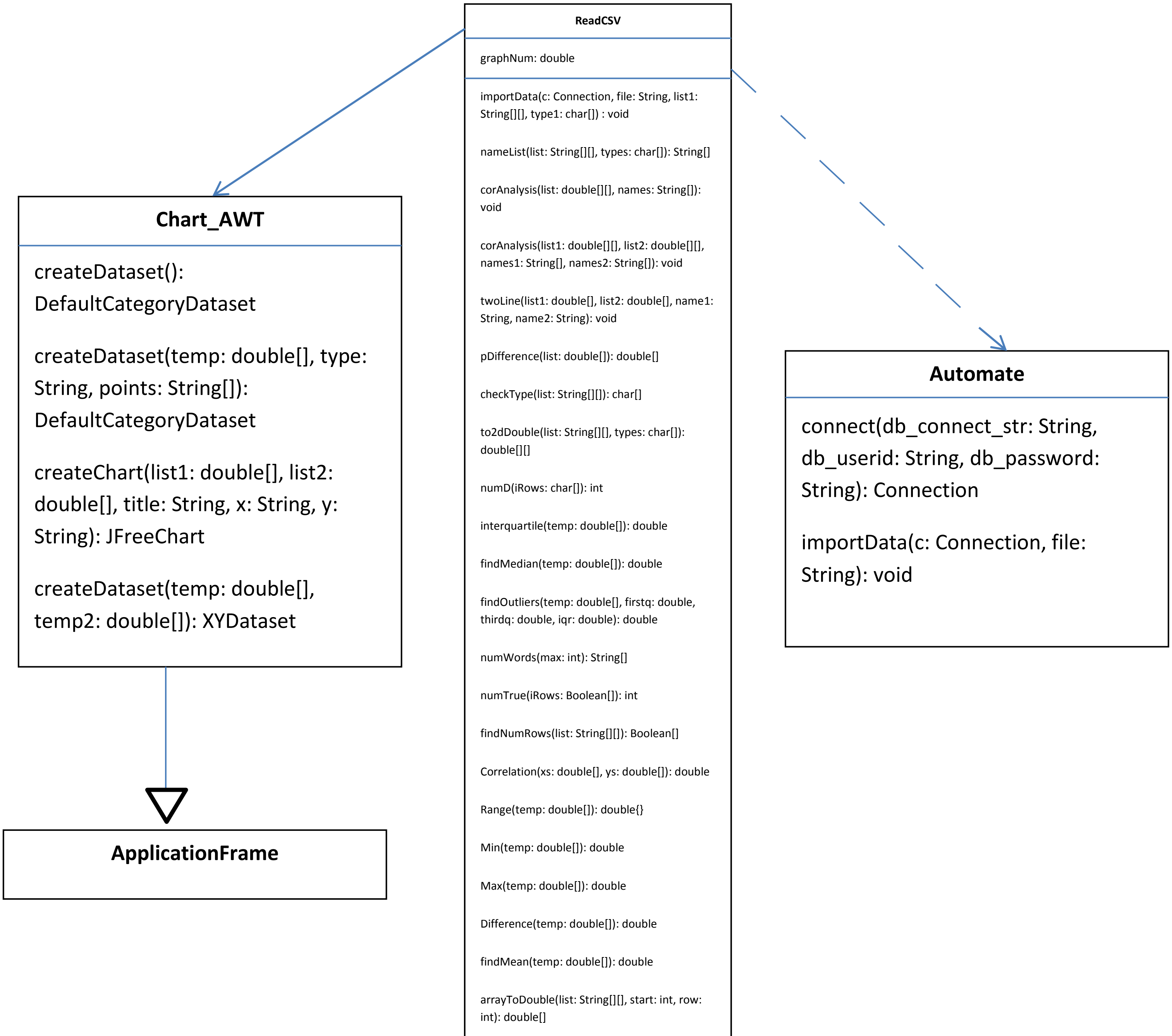


1.3 Logical viewpoint

1.3.1 Design concerns

In some of the predefined libraries being used for the project (such as ReadCSV, JDBC, and JFreeChart), abstraction has been used to implement certain functions whose method summary has already been created. For example, in the class that constructs graphs in JFreeChart, private methods are used to protect the information contained within them (such as creating the dataset content used to build the graph). However, in the other classes public methods have been used to call functions in other classes (i.e. the JDBC driver calling the class automate to get the server connection). In these methods, the code is much longer and there is a concern that as modifications are made to the code and more content is added, there is some repetition in how each row of data is being read. Therefore, it may be logical moving forward to break more of the code contained in the methods within ReadCSV into a greater number functions (perhaps marking some of them as private) to make the code more concise and compact.

1.3.2/1.3.3 Design elements and Example languages (using a UML class diagram)


ReadCSV
graphNum: double
importData(c: Connection, file: String, list1: String[][], type1: char[]) : void
nameList(list: String[][], types: char[]): String[]
corAnalysis(list: double[][], names: String[]): void
corAnalysis(list1: double[][], list2: double[][], names1: String[], names2: String[]): void
twoLine(list1: double[], list2: double[], name1: String, name2: String): void
pDifference(list: double[]): double[]
checkType(list: String[][]): char[]
to2dDouble(list: String[][], types: char[]): double[][]
numD(iRows: char[]): int
interquartile(temp: double[]): double
findMedian(temp: double[]): double
findOutliers(temp: double[], firstq: double, thirdq: double, iqr: double): double
numWords(max: int): String[]
numTrue(iRows: Boolean[]): int
findNumRows(list: String[][]): Boolean[]
Correlation(xs: double[], ys: double[]): double
Range(temp: double[]): double{}
Min(temp: double[]): double
Max(temp: double[]): double
Difference(temp: double[]): double
findMean(temp: double[]): double
arrayToDouble(list: String[][], start: int, row: int): double[]
Chart_AWT
createDataset(): DefaultCategoryDataset
createDataset(temp: double[], type: String, points: String[]): DefaultCategoryDataset
createChart(list1: double[], list2: double[], title: String, x: String, y: String): JFreeChart
createDataset(temp: double[], temp2: double[]): XYDataset
ApplicationFrame
Automate
connect(db_connect_str: String, db_userid: String, db_password: String): Connection
importData(c: Connection, file: String): void