Legendary Wasp

Software Development Plan for: *Statistical Analyzer*

Version: *1.2*

Approval date: *October 25th, 2022*

File Name:	Team 10 SDP	
File Location:	GoogleDrive/CS 499 - Group10/	
Version Number:	1.2	
Created By:	Justin Bushue	8/29/2022
Reviewed By:	Brandon Perry	9/06/2022
	Byron Thompson	9/06/2022
	Elizabeth Dooley	9/06/2022
	Justin Bushue	9/06/2022
Modified By:	Brandon Perry	9/06/2022
	Byron Thompson	9/05/2022
	Elizabeth Dooley	9/04/2022
	Justin Bushue	9/06/2022
	Brandon Perry	9/22/2022
	Brandon Perry	10/25/2022
Approved By:	Brandon Perry	10/25/2022

Table of Contents

List of Tables

List of Figures

List of Acronyms and Abbreviations

SDP	Software Development Plan
LW	Legendary Wasp
SA	Statistical Analysis
GUI	Graphical User Interface
QA	Quality Assurance
CSV	Comma Separated Value
TSV	Tab Separated Value
TXT	Text
UAH	University of Alabama in Huntsville
CM	Configuration Management
QA	Quality Assurance

1 Overview

The Software Development Plan (SDP) establishes the software development approach, methodologies, tools, and procedures to be used during the analysis, design, development, testing, integration, deployment, and maintenance of the software for Legendary Wasp (LW) Statistical Analysis (SA) project. This SDP is a dynamic document and shall be updated on a periodic basis to reflect organizational changes, lessons learned, new tools, and advances in methodologies. The SDP is a requirement for the project developers responsible for developing and submitting the SDP document for a software development effort.

1.1 Scope

The SDP provides the means to coordinate schedules, control resources, initiate actions, and monitor progress of the development effort. The purpose of the document is to provide a detailed plan for the use of resources, methodologies, and techniques that provide for the development of all software that comprise the product line. Due to the nature of the project (assignment during a semester), the scope may be limited and revised in the future.

1.2 Identification

The project SDP is managed and controlled in accordance with LWs configuration management (CM) practices.

This SDP applies to the Statistical Analyzer program assigned in CS 499-01 at the University of Alabama in Huntsville. Any future documentation, such as user manuals, will also follow the guidelines and processes within this document.

1.3 System Overview

The statistical analyzer program will allow a user to perform statistical measures on a wide range of data interfacing with a Graphical User Interface (GUI). Whether the data is input manually into a table or uploaded as a file, the user will be able to specify which measures they would like to perform. The user will also be able to select all or certain parts of the data inputted for analysis.

The program will provide a text file, graphical representations where applicable in JPEG format, CSV or TSV files usable by Excel, and text output of the results.

1.3.1 Operational Concept

The following methods of data entry will be implemented:

- Delimited file import
    - CSV
    - TSV
    - Other (delimiter can be specified)
- Excel file import
- Manual Table Entry

The following statistical measures will be implemented:

- Mean
- Median
- Mode
- Standard Deviation
- Variance
- Coefficient of Variance
- Percentiles
- Probability Distribution
- Binomial Distribution
- Least Square Line
- Chi Square
- Correlation Coefficient
- Sign Test
- Rank Sum Test
- Spearman Rank Correlation Coefficient

The following outputs (not mutually exclusive) will be implemented:

- Graphical
    - Vertical Bar Graph
    - Horizontal Bar Graph
    - Pie Chart
    - Normal Distribution Curve
    - X-Y Graph
- Text
    - Tabular Formatted
- File
    - CSV
    - TSV
    - Excel
    - JPEG
    - TXT

The following GUI elements will be implemented:

- Frames
    - The window which will contain all other GUI elements
    - A window to view the imported data from the chart, i.e. a subset of data
    - A window to view the chosen statistical measures to run
    - A pop up window to show which statistical measures can be added
- Textboxes
    - Editable by user to allow for manual data entry
    - Editable by user to allow for a file name
- Buttons
    - A button which prompts user to enter file name
    - A button for user to press to run calculations
    - A button for user to press to download created files
    - A button for user to press to download created images
    - A button for user to press to edit the table
    - A button for user to press to select graphs
    - A button for user to press to see settings
    - A button for user to press to view help
    - A button for user to press to add statistical measures
    - A button for user to press to remove statistical measures
    - A button for user to press to import data from chart for a subset
- Output
    - Area to view text output
    - Area to view graphs created
    - Area to see created files

1.3.2 Computer Software Configuration Items

Configuration of the computer on which this program will be installed requires:

- Windows
    - 11 (64 bit only)
    - 10
    - 8
    - 7
- System Minimum Requirements
    - RAM - 4 GB
    - Disk - **TBD** MB
    - Processor - Pentium 2 266 MHz
- macOS
    - 12
    - 11
    - 10.9
    - 10.8.3
- Java 8 or newer

1.4 **Relationship to Other Plans**

Due to being a self contained assignment to be completed within a semester, there are no relationships to other plans. The only other plan is the Test Plan found in the CS499 Group 10 Google Drive Folder

2 Reference Documents

Table 1 - Reference Documents

Document Title	Revision	Date	Notes
Requirements	1.0	8/24/2022	Provided by Instructor
Team Weekly Meeting	1.2	8/31/2022	Revision is <Sprint Number>.<Week Number>
Java TableSaw	0.43.1	4/3/2022	TableSaw JavaDoc URL
Test Plan	1.0	10/25/2022	Plan for testing the program
Testing Results	1.0	10/20/2022	Function Test Output
UML Diagram	1.0	9/26/2022	Provides the Software Architecture
Statistical Measure Instruction Manual	1.0	10/29/2022	A User Manual to Guide the End User on the programs functionality

3 Overview of Software Development Planning

This section establishes the context for the planning described in later sections.

Two team meetings will be held each week after class to discuss the progress made thus far, future efforts, problems that have arisen, concerns, etc. More meetings will be added on an as needed basis. A requirements document was provided for the Statistical Analyzer assignment. The weekly team reports also contain a detailed list of system and software requirements.

For documentation, weekly team and individual reports will be submitted at the end of each week. A presentation slide deck will be created at the end of each sprint as well. A user guide will be created at the end of the development process. This software development plan will be created and updated throughout the development process.

Each sprint will be two weeks long with a presentation week in between. The sprints follow the general order of: SDP, User Stories/Backlog, Architecture, GUI, Final Deliverable.

3.1 Requirements and Development

System and software requirements will be documented in the requirements section of the Weekly Team Reports. Software requirements will be derived from the system requirements and also put in the requirements section of the Weekly Team Reports.

The Software shall support the use of CSV files to load data to be analyzed
The Software shall provide the user the ability to manually input data into an on-screen table
The Software shall be able to calculate the Mean of an input data set
The Software shall be able to calculate the Median of an input data set
The Software shall be able to calculate the Mode of an input data set
The Software shall be able to calculate the Standard Deviation of an input data set
The Software shall be able to calculate the Variance of an input data set
The Software shall be able to calculate the Coefficient of Variance of an input data set
The Software shall be able to calculate the Percentiles of an input data set
The Software shall be able to calculate the Probability Distribution of an input data set
The Software shall be able to calculate the Binomial Distribution of an input data set
The Software shall be able to calculate the Least Squre Line of an input data set
The Software shall be able to perform a Chi Square test on an input data set
The Software shall be able to calculate the Correlation Coefficient of an input data set
The Software shall be able to perform a Sign Test on an input data set
The Software shall be able to perform a Rank Sum Test on an input data set
The Software shall be able to calculate the Spearman Rank Correlation Coefficient of a data set
The Software shall create consistent, meaningful data without variation
In the event that a Statistical Measure is not appropriate for a set of data, the Software shall reject the data

When appropriate, the Software shall allow the user to display a Horizontal Bar Chart to display the output of a statistical measure
When appropriate, the Software shall allow the user to display a Vertical Bar Chart to display the output of a statistical measure
When appropriate, the Software shall allow the user to display a Pie Chart to display the output of a statistical measure
When appropriate, the Software shall allow the user to display a Normal Distribution Curve to display the output of a statistical measure
When appropriate, the Software shall allow the user to display an X-Y Graph to display the output of a statistical measure
The Software shall support the addition of new statistical measures for a user to select from whithout the need for significant changes to existing code
The Software shall use a Graphical User Interface for user interaction
The Software shall allow a user to select multiple Statistical Measures to use on a set of data
The Software shall allow the user to select a subset of their data, by use of the Graphical User Interface, to perform selected measures on
Where appropriate, The Software shall be capable of exporting the results of any statistical measure to a CSV file
Where appropriate, The Software shall be capable of exporting the results of any statistical measure to a TSV (Tab Separated Values) file
When tabular output is not appropriate, the Software shall be capable of exporting the results of a statistical measure to a txt file
The Software shall be capable of producing JPEG images of any graph it displays
The Software Shall be capable of producing text files summarizing the results of any and all statistical analyses it performed on a data set
The Software shall handle any unexpected manual input gracefully and allow the user to reenter with correct input.
The Software shall handle incorrect CSV input files gracefully and inform the user.

3.2 **Project Documentation**

Google Docs will be used to complete documentation. Google Drive will be used to store multiple revisions of documentation. This includes weekly team and individual reports, software development plan, UML diagrams, test plans, user manuals, etc.

3.3 **Schedules and Resources**

This section provides the detailed schedule for the project. The schedule depicts the assigned personnel and the critical path and dependencies. This schedule will be updated throughout the project once a better understanding of the tasks needed and time required is gained.

Table 2 - Detailed Scheduling

Date Due	Task	Dependencies	Assignee
Weekly	Individual Reports	N/A	All
Weekly	Team Reports	Team Input	Brandon Perry
9/07/2022	Software Development Plan	N/A	All
9/07/2022	Sprint #1 Presentation	SDP	All
9/26/2022	Epics, User Stories, Reqmts., Project Backlog	Basic understanding of the project broken down into parts	Brandon Perry / Byron Thompson
9/26/2022	Sprint #2 Presentation	U-S, Reqmnts., backlog, epics	Brandon Perry
9/26//2022	Backlog	Requirements	All
10/01/2022	Test Cases	Requirements	Brandon Perry
10/17/2022	Software Architecture	Requirements, User Stories	Justin Bushue
10/17/2022	UML Diagrams	Architecture, OOP pattern	Justin Bushue
10/17/2022	Sprint #3 Presentation	UML, OOP, Architecture	All
10/25/2022	Test Plan	Test Cases	Brandon Perry
11/02/2022	Backend Testing	Backend	All
11/02/2022	Final SDP	N/A	Brandon Perry
11/02/2022	Preliminary GUI	Requirements, User Stories, Potentially basic functions	Elizabeth Dooley
11/02/2022	Sprint #4 Presentation	GUI layout, Preliminary Code	All
11/05/2022	Statistical Functions	Foundational Code	Justin Bushue
11/10/2022	Graphing Functions	Backend, Statistical Functions	Byron Thompson
11/15/2022	Frontend to Backend	GUI and Backend complete	All
11/20/2022	GUI Testing	GUI Complete and Connected	All
11/20/2022	User Guide	GUI finished, Input/Output methods established	Brandon Perry
11/23/2022	Final Project Testing	Project Finished	All

11/25/2022	Final Deliverable	All code and deliverables	All
11/25/2022	Sprint #5 Presentation	Project Finished	All

3.4 **Training Requirements**

The project software engineers will need to review Java basics as well as any libraries used throughout the development cycle such as Tablesaw. Tutorials/reviews of software tools such as GitHub, Java IDE's, etc will need to take place on an as needed basis.

The end user will not require any training. Instructions will be provided within the application. A user manual will also be created to aid in any learning curve. This user manual will provide common scenarios that the user may encounter or actions they want to perform.

4 General Software Development Activities

4.1 Development Process

Our team will make use of the Agile Software Development practice, allowing us to more easily adapt to any issues or hurdles we may encounter.

Planned Builds:

- Architecture Build (10/17/2022)
  - Develop majority of software back-end, including file I/O, software data structure, and data processing.
- GUI Build (11/2/2022)
  - Develop preliminary GUI for user interaction with the program, including menus for importing files or entering data manually, selection of statistical measures to perform, and display of results including graphs for applicable measures.
- Final Build (11/28/2022)
  - Finalize design on GUI and software back-end. Resolve any remaining issues or bugs.

4.2 Development Methods

Object oriented programming will be followed. More details of the architecture can be found in the Sprint #3 Presentation found in the CS499 Group 10 Google Drive. The UML diagram can also be found in this folder as well.

4.3 Product Standards

General software engineering practices will be followed. We will also be adhering to the UAH Computer Science department policies at all times with regards to this project. These policies can be viewed at https://www.uah.edu/science/departments/computer-science/cs-policies.

4.4 Reusable Products

4.4.1 Incorporating Reusable Products

This section describes the approach to be followed for identifying, evaluating, and incorporating reusable software products, including the scope of the search for such products and the criteria to be used for their evaluation.

Focus will not be given to developing the software in a reusable manner. This is due to the limited scope of the development of this software. The graphical output functions may be reusable but only by chance. The libraries used in this software, on the other hand, have a high ability of reuse.

4.4.2 Developing Reusable Products

Due to the limited scope of this project and assignment, focus will not be put on producing reusable software.

4.5 **Critical Requirements**

Due to the nature of Statistical Analysis, it is critical that any and all outputs from the software are repeatable, consistent, and accurate. To avoid issues with floating point mathematical errors, which could introduce inconsistency in data analysis, the program will make use of certain practices to avoid any issues that may arise from using floating point numbers, of which their exact implementation will be decided at a later date.

4.6 **Computer Hardware Resource Utilization**

Due to the self contained nature of this assignment, computer hardware resource utilization will not need to be monitored. Personal laptops will be used for the development and testing of the software project. Minimum hardware requirements will be provided in the future in section 1.3.2.

5 Detailed Software Development Activities

This section provides the plans for performing detailed software development activities, including the approach (i.e., methods, procedures, and tools).

5.1 Project Planning and Oversight

This section describes the approach to be followed for project planning and oversight.

At the beginning of each sprint, each member of the team will be assigned their tasks to complete. A test plan has been developed with test cases to unit test along the way, as well as to implement integration testing towards the end. Code reviews are held on an as needed basis to ensure good quality code.

A plan has been developed to have the final product finished one sprint early in order to allow time for error handling, installation/deployment issues, final merge conflicts, tidying up documentation, etc.

Due to the scope of the project, there is no software transition necessary. Any plans created, such as this SDP, will be updated at the end of each sprint. The team will then review the documents before final submission.

5.2 Establishing a Software Development Environment

This section describes the approach to be followed for establishing, controlling, and maintaining a software development environment.

Each team member will develop in the IDE of their choice. All source files are stored on GitHub in the project folder. Each member of the team has their own branch in order to make changes and updates to the code before merging with main.

5.3 System and Software Requirements Analysis

For the system and software requirements, each will be compared to its respected user story/stories. Each requirement will be traced from the beginning of the program, i.e. user input, all the way until the final output to see if the requirement has been met properly. Stress testing, boundary testing, edge case testing, and more will take place to analyze both that the requirement has been implemented and that the requirement is necessary and written well. If time allows, reviews held with the customer may take place as well.

5.4 **System and Software Design**

Due to the scope of the project, the system design is not too prevalent. Details can be found in the CS 499 Group 10 Google Drive folder as well as the UML diagram.

The software design process focuses on modularity and the easy flow of messages and data. Due to having data sets and subsets of data sets, focus has been placed on creating a consistent formatting of data to pass from the table to each statistical measure. Due to one of the requirements being modularity, each function is loosely coupled which allows for an ease of expansion. *UserSettings* allows for users to define their own functions that can be easily implemented. There is a main *Dataset* interface and *IMeasure* interface that interact with the data to allow the ease of data transmission. *UIServices* allows for easy communication between the front and back end.

5.5 **Software Implementation and Test/Code and Unit Test**

Test cases have been created in the Team Weekly Reports respective sheet. Unit testing will be completed as each function and action is completed. Any problems will be documented in the Team Weekly Reports document and resolved when necessary. It will be up to the team member to test correctly and ensure correct functionality.

Logging features have been implemented into the project in order to allow for easier troubleshooting, error chasing, and testing. At the end of each established build version, retesting will take place to ensure correct functionality.

A test plan has been developed with proper procedures and documentation. Test reports will also be generated and placed in the CS499 Group 10 Google Drive folder.

5.6 **Software and System Integration and Testing**

Due to the scope of the project, system integration is minimal. Software integration will take place in sprint four or sprint five to connect the front and backend together.

Testing processes and Integration testing is described in the Test Plan found in the CS499 Group 10 Google Drive Folder. The backend will be completed and tested before interacting with the front end. The GUI will eventually connect to the backend after testing the GUI layout. The interaction between the two will then be tested.

If errors are encountered, the tester will either fix the issue or defer it to another team member. After the fix, that functionality will then be retested and integrated into the main project. Test results will be documented in the Google Drive Folder.

This section describes the software and system integration activities that focus on integrating all the software components into a software element that will be further integrated with the other elements to form the systems.

5.7 **Preparing for Software Use**

A user manual will be created within the timeline of the fourth and fifth sprint. This document will provide instructions of how to use the software, ways to avoid potential errors, and provide visual depictions of the program in use.

The software will be distributed via source code and a packaged executable JAR file. The executable will be tested on multiple platforms.

5.7.1 Preparing for Software Transition

All software files will be gathered and merged within GitHub. This is also where the code reviews will be held. GitHub will also be responsible for holding the JAR executable file. Reviews will be held to make sure that each file has a proper header and description. The "Read Me" file in GitHub will also be created to properly document the code and project.No transition of support site will be necessary due to GitHub holding all of the code. Testing will take place to ensure the final product is ready for deployment.

All documentation will be reviewed including team reports, software development plan, the user guide, test documentation, and any supporting documentation that is necessary.

5.8 **Software Configuration Management**

The software will be distributed via source code and a packaged executable JAR file. There will be no settings to alter. GitHub will be used for managing all of the files, revision, branches, etc. Each team member will use the IDE of their choice.

5.9 **Software Product Evaluation**

Code reviews will be conducted at the end of each sprint when merging code to the final product. Documentation will follow a similar process by being peer reviewed before submitted.

The final software product will be evaluated by going through each user story and requirement to make sure that they have been met satisfactorily. Edge cases and further testing will also be conducted to ensure that a robust project has been made. All previous testing will be reviewed as well to ensure consistent testing quality. External users will be brought in to evaluate the look and feel of the product as well as critique any quality of life issues, bugs, etc.

5.10 **Software Quality Assurance**

Individuals will review their own code and make sure it is up to the standards set at the beginning of the project. Individuals will test their own code throughout the development cycle. Once ready to be merged, another member or the team as a whole will test the software code snippet or element. The Team Leader will make sure that all code has proper comments and documentation accompanying it. The team as a whole will test the final software product to make sure all requirements and user stories have been met for QA.

5.11 **Corrective Action**

When a problem is encountered, it will be reported in the Team Weekly Report spreadsheet in the "Problem Reports" sheet. A title, trace back to any applicable test cases, description, state, and open date will be provided. The problem will then be assigned to a team member, resolved, reviewed by another team member, and then the problem will be closed.

If a problem is outside the scope of a single team member, a group meeting will be held to discuss further steps to be taken.

5.12 **Technical and Management Reviews**

Once an individual is done with their elements of a sprint, they will reach out to a fellow team member to review their code, documentation, etc. A brief review that good coding practices have been followed, the code functions properly, and the user story has been met, the green light will be given to merge with the main project.

5.13 **Other Software Development Activities**

A list of known risks with mitigation steps is included in the Weekly Team Reports. No outside agencies, business units, or teams is expected due to the nature of the project. Improvements to the software development cycle will be made as seen fit if brought up at team meetings. Security and privacy, while needed to be kept in mind while coding, is not of high importance due to the nature of the project.

Software management indicators such as Scrum points and task descriptions are provided within the Scrumfast Kanban board. These are also provided within the Weekly Team Reports. Any suggestions for better metrics or more in depth processes / activities will be brought up during team meetings.

6 Schedules and Activity Network

Table 3 - Broad Scheduling

Date Due	**Description**
9/7/2022	Completion and submission of first revision of SDP
9/26/2022	Completion of Epics, User Stories, Requirements, and Project Backlog
10/17/2022	Completion of Software Architecture
11/2/2022	Completion of Preliminary GUI
11/28/2022	Completion of Final Deliverable

7 Program Organization and Resources

Due to the scope of the project, resources are not a concern. Each team member will provide their own technology and time to work on the project.

The project is organized in five sprints as described in previous sections. The Architecture is pioneered by Justin Bushue and Byron Thompson. Elizabeth Dooley is spearheading the GUI implementation. Brandon Perry is documenting the testing, SDP, reports, presentations, etc. Byron Thompson and Brandon Perry are jointly working on the graphical output functionality. Overall, each team member and their tasks are flexible and fluid.

The general flow of development is:

plan → architecture → backend → test → GUI → test → Integration → test → deliver

The project is broken into the following folders for ease of development:

- BackEndUtilities
- FrontEndUtilities
- GUI
- Interfaces
- Interop
- Measures
- Respository
- Settings
- TableUtilities
- Validators
- Main.java

Each build and its purpose is described in section 4.1.

DOCUMENT REVISION HISTORY			
Version Number	**Approved Date**	**Description of Change(s)**	**Created/ Modified By**
1.0	9/6/2022	Initial Commit	Justin Bushue
1.1	9/22/2022	Added information to fill in blanks in Section 4 and 5	Brandon Perry
1.2	10/25/2022	Filled in Incomplete sections in Section 3, 5, and 7. Updated schedule and other misc. information	Brandon Perry