SOFTWARE Project Management Plan (SPMP)



*Approvals:*

|  |  |  |  |
| --- | --- | --- | --- |
| Team Leader, Interface Specialist | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Alex Laird | Date: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Programming Specialist | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Ryan Morehart | Date: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Statistics and R Programmer | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Andrew Sterling | Date | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

*Document History:*

09-08-2010 – Initial Draft

10-04-2010 – Second Revision

Contents

[1 Introduction 4](#_Toc271894191)

[1.1 Project Overview 4](#_Toc271894192)

[1.2 Project Deliverables 4](#_Toc271894193)

[1.3 Reference Materials 5](#_Toc271894194)

[1.4 Definitions and Acronyms 5](#_Toc271894195)

[2 Management 6](#_Toc271894196)

[2.1 Managerial Objectives and Priorities 6](#_Toc271894197)

[2.2 Monitoring and Controlling Mechanisms 6](#_Toc271894198)

[2.4 Process Model 6](#_Toc271894199)

[2.5 Organizational Structure 7](#_Toc271894200)

[3 Risk 8](#_Toc271894201)

[3.1 Assumptions and Dependencies 8](#_Toc271894202)

[3.2 Risk management 8](#_Toc271894203)

[4 Quality Plan 9](#_Toc271894204)

[5 Configuration Management 9](#_Toc271894205)

[6 Plan/Schedule/Tasks 10](#_Toc271894206)

# 1 Introduction

## 1.1 Project Overview

* Context

Students entering into a first-semester probability and statistics course are often required to perform complex operations using the statistical command-line library of R. Students are also often required to format their homework neatly using the LaTeX framework. Both R and LaTeX are cumbersome hurdles that certain students have difficulty getting over to even be able to grasp the content of the course. This application is intended to eliminate the need for students to learn LaTeX and R as in depth, offering them a graphical interface to do the same sorts of things.

* Application

The application gives the user a simplistic graphical interface that allows them to perform complex statistical calculations. Problem sets can be created, solved, and exported to a neatly formatted Portable Document Format (PDF) file. Problem sets can be created easily using the simple graphical wizard for a new problem. Calculations can easily be made by dragging and dropping statistical commands onto the user’s data set, and the solutions to those calculations can be tied back to certain parts of the problem. To the average user, the complex calculations and exporting code will not be shown, but this code will be available at the request of a more advanced user.

* User(s)

The program is specifically targeted for students in their first-semester of a probability and statistics course. The scope could reach beyond students to anyone needing rudimentary functionality for simple probability and statistics equations, specifically if the user is seeking a simple graphical interface and not wanting to work with the command-line environment of R.

* Purpose

To eliminate the LaTeX and R learning curve from a first-semester probability and statistics course, and to provide an ease-of-use graphical interface for R that allows users to manipulate a drag-and-drop interface to perform complex statistics calculations in a simple manner.

## 1.2 Project Deliverables

* Documents
  + Help Content available to the user
* Application Components
  + A simple, clean Graphical User Interface (GUI)
  + A New Problem Wizard
  + Ability to Load/Save problem sets
  + Drag-and-Drop Ability for Calculations
  + View backend R commands and LaTeX code
  + Export formatted PDF for problem set

## 1.3 Reference Materials

* Educational References
  + Software Engineering:  A Practitioner's Approach, 7th Edition, Roger S. Pressman.  McGraw-Hill, 2010
  + Practices of an Agile Developer, Venkat Subramanaim and Andy Hunt.  Pragmatic Bookshelf, 2006
* Application References
  + R
  + LaTeX
  + texmaker
  + Foxit Reader
* Customer References
  + Professor Robert Schumacher
* Advisor References
  + Professor Robert Schumacher

## 1.4 Definitions and Acronyms

* API – Application Programming Interface
* GUI – Graphical User Interface
* PDF – Portable Document Format
* SPMP – Software Project Management Plan
* SRD – Software Requirements Document

# 2 Management

## 2.1 Managerial Objectives and Priorities

* Priorities:
  + Integration – Provides an application to students that is simple and powerful
  + Quality – Efficient, simple, and modular
  + Innovation – Does more than expected from the simple looking interface, and does it well

## 2.2 Monitoring and Controlling Mechanisms

* Team Meetings

Status checks are on Mondays, Wednesdays, and Fridays at 7:00pm. They last no more than fifteen minutes, and are for accountability and positive peer pressure as much as they are for information. Meetings with Professor Schumacher will only be scheduled when necessary.

* Policy Enforcement

Positive reinforcement will be used when team members participate in the program and are present at team meetings. Verbal reprimanding will be used when team members show lack of progress or do not attend mandatory meetings. Further apathy and absence will result in a poor report to Dr. Shomper.

## 2.4 Process Model

* Guiding Model

Our formal model will be the Incremental Model.

* Rationale for Choosing

This model was chosen because, the way our application is set up, it will be very easy to ensure the core functionality works and to slowly add or modularize features from there. This allows us to place more emphasis on current features and, when complete, add more functionality. This should minimize the need for error correction near the end of production, since we will be error checking at the end of each increment.

* Adaptations

Initially, core functionality is what we are striving for. A simplistic interface and advanced functionality is not necessary to prove that the project’s ambitions are attainable. We will implement a rough interface to illustrate the interfacing of Java and R. Once we integrate this core functionality, other features can easily be wrapped around them, and the interface can be reconstructed to a simple, easy-to-use design.

## 2.5 Organizational Structure

* Team Structure and Responsibilities
  + Team Lead and Interface Specialist – Alex Laird
    - Organized
    - Good with communication to promote team unity and progress
    - Extensive Java GUI experience
    - Primarily responsible for the clean drag-and-drop GUI
  + Programming Specialist – Ryan Morehart
    - Excellent programmer
    - Primarily responsible for the backend of the application
    - Responsible for the Java side of the Java-to-R interface
  + Statistics and R Programmer – Andrew Sterling
    - Extensive knowledge of probability and statistics
    - Extensive R knowledge
    - Primarily responsible for the R side of the Java-to-R interface
* Peers, Advisors, Customers, and Information Sources

Advisors

1. Professor Robert Schumacher

Customer References

1. Professor Robert Schumacher

Information Sources

1. Professor Robert Schumacher
2. Java Application Programming Interface (API)
3. R API
4. LaTeX API

# 3 Risk

## 3.1 Assumptions and Dependencies

* Assumptions:
  + Available resources, skills, deliverable dates
  1. Simple Java-to-R interfacing is attainable and simple
  2. Understanding of probability and statistics
  3. Understanding what the average student will want and need
  4. Java GUIs are easily constructible
  + Customer assumptions
  1. The customer is communicative
  2. The customer does not change their mind
  3. The customer is available for meetings
* Dependences:
  + Customer availability
  + Team member availability

## 3.2 Risk management

* Identify Risks
  + We consider this project to be a low-risk project.
  + In the following table, 0 is unlikely and little impact, whereas 100 is imminent and has a grave impact on the project.

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Likelihood** | **Impact** | **Mitigation Plan** |
| R Interfacing | 10 | 100 | Write our own Java/R interface |
| LaTeX Interfacing | 5 | 20 | Eliminate export to PDF functionality |
| Data Manipulation (by User) | 10 | 20 | Require user to manipulate data before importing |
| Drag and Drop Difficulties | 15 | 33 | Popup windows and check boxes to input commands |
| Customer changes specification | 30 | 70 | We implement customer request |
| Cross-Platform Dependencies | 15 | 20 | We develop for Windows, possibly drop cross-platform support |

* Monitoring plan

Risks will be monitored weekly during team meetings. They will not be overreacted upon, but only countered if the risk is actually deemed imminent.

# 4 Quality Plan

* Quality Assurance Activities
  + Reviews will be held by the team for every document
  + Reviews will be held by the advisor for every milestone
  + Code will be thoroughly tested at the end of each increment of the build
* Quality Measurements
  + Customer satisfaction
  + Modularity
  + Simplicity and functionality

# 5 Configuration Management

* Tools
  + Mercurial will be used for version control
  + BitBucket.org will be the host of version control
  + BitBucket.org also provides effective issue tracking on project’s
* Process for Updating Production Objects

Mercurial allows for both local and repository commits. Local commits will be done when the team member wishes to secure their code before moving forward. Repository commits will be done *only* after a repository update has been done and files successfully merged. All files must compile and execute properly (no runtime exceptions) before a repository commit will be done. In general, repository commits should be done at the end of each significant functionality update.

* Testing

Extensive JUnit testing will be done at the end of each feature implementation. Specifically, testing by both the coding team member and other team members will be done at the end of each incremental build before moving to the next increment.

# 6 Plan/Schedule/Tasks

* Task List



* Team Member Assignment
  + Alex Laird – Emphasis on interface design and implementation
  + Ryan Morehart – Emphasis on backend programming, specifically Java-to-R integration
  + Andrew Sterling – Emphasis on statistics and R programming, specifically Java-to-R integration