ICONIX Design Document

Student Number: 483201
Student Name: Gary Bezuidenhout
School of Electrical & Information Engineering, University of the Witwatersrand

01 November 2017

Introduction

This document outlines the *ICONIX* process applied to the design of the Genesis Admixture Visualisation Tool. The *ICONIX* process is a design technique which is driven by the behavioural requirements of a proposed software application. The technique starts at the use case level and iterates through each step to arrive at a class diagram that can be used as a guideline for implementation of the design. This document will illustrate all design steps and iterations thereof undertaken for the design of this application.

1 Informal Problem Statement and Prototype GUI

The first step in the *ICONIX* process is the presentation of the informal problem statement and creation of the prototype GUI. The Informal Problem Statement presented below is an adaptation of the project description given for the Genome Admixture Visualisation Tool.

Problem Statement

With the use of *DNA* data, the exploration of genetic composition and admixture of individuals is possible. This capability allows individuals to better interpret data related to genetic diseases and provide answers to questions in relation to population history. Due to the positive impact provided by this data exploration, visualisation is a very critical component thereof. The Genesis tool was developed to provide individuals with capability to visualise *DNA* data. Despite the success of the tool, shortcomings in the application design have hindered the extension and maintenance of the tool. As a result, the tool is required to be re-engineered and redeveloped.

Prototype GUI

The images presented in Figures 1, 2 and 3, illustrate possible GUI layout for the application. The Welcome/Splash screen shown in Figure 1 welcomes the user to the application and provides the user with a choice to either load a previously saved project or start a new project. The Plot Type Selection screen in Figure 2 allows the user the select the plot type that would meet the required needs. Lastly the Plot Viewer/Editor presented in Figure 3 would be the view used most frequently. The view contains all plot related functions and would give the user the ability to both view and edit plots.

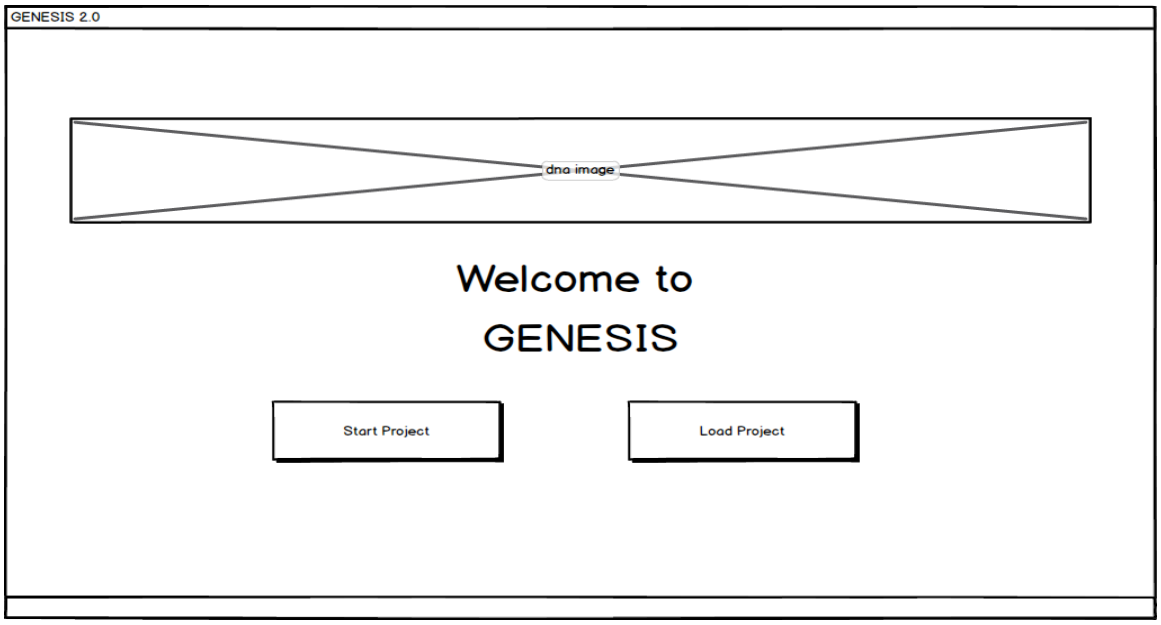

Figure 1: Genesis Splash Screen Mockup

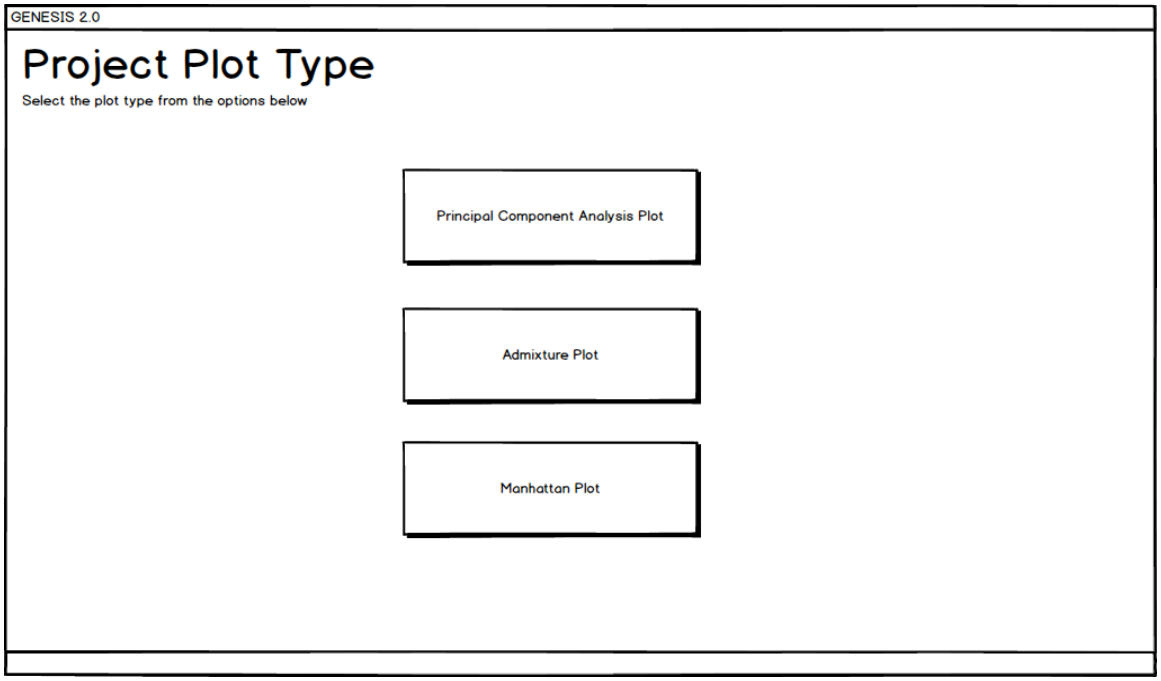

Figure 2: Genesis Plot Type Selection Screen Mockup

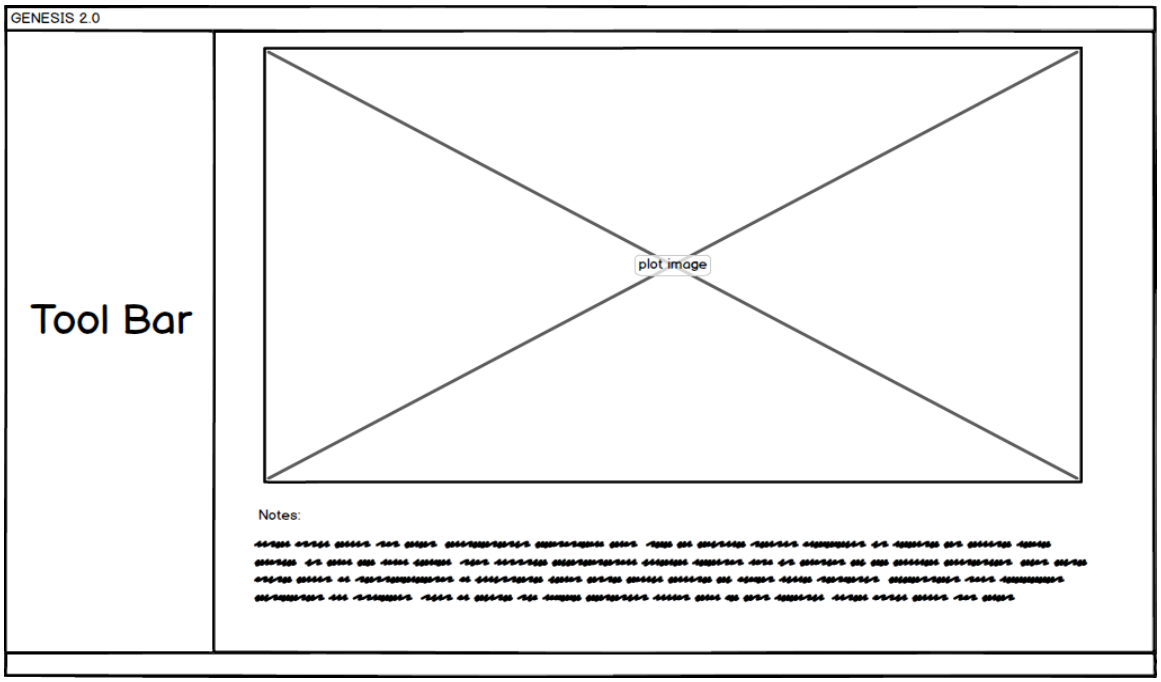

Figure 3: Genesis Plot Viewer/Editor Screen Mockup

2 Domain Modelling

The next step in the *ICONIX* process is creating a domain model. The aim of this task is to identify abstractions in the real world. These are the conceptual objects present in problem scope, that have a high possibility of participating in the system. This model is considered the foundation of the static branch of the UML model. Lastly, the only relationships present in the domain model are aggregation and generalisation. The domain model created for the Genesis tool is presented in Figure 4.

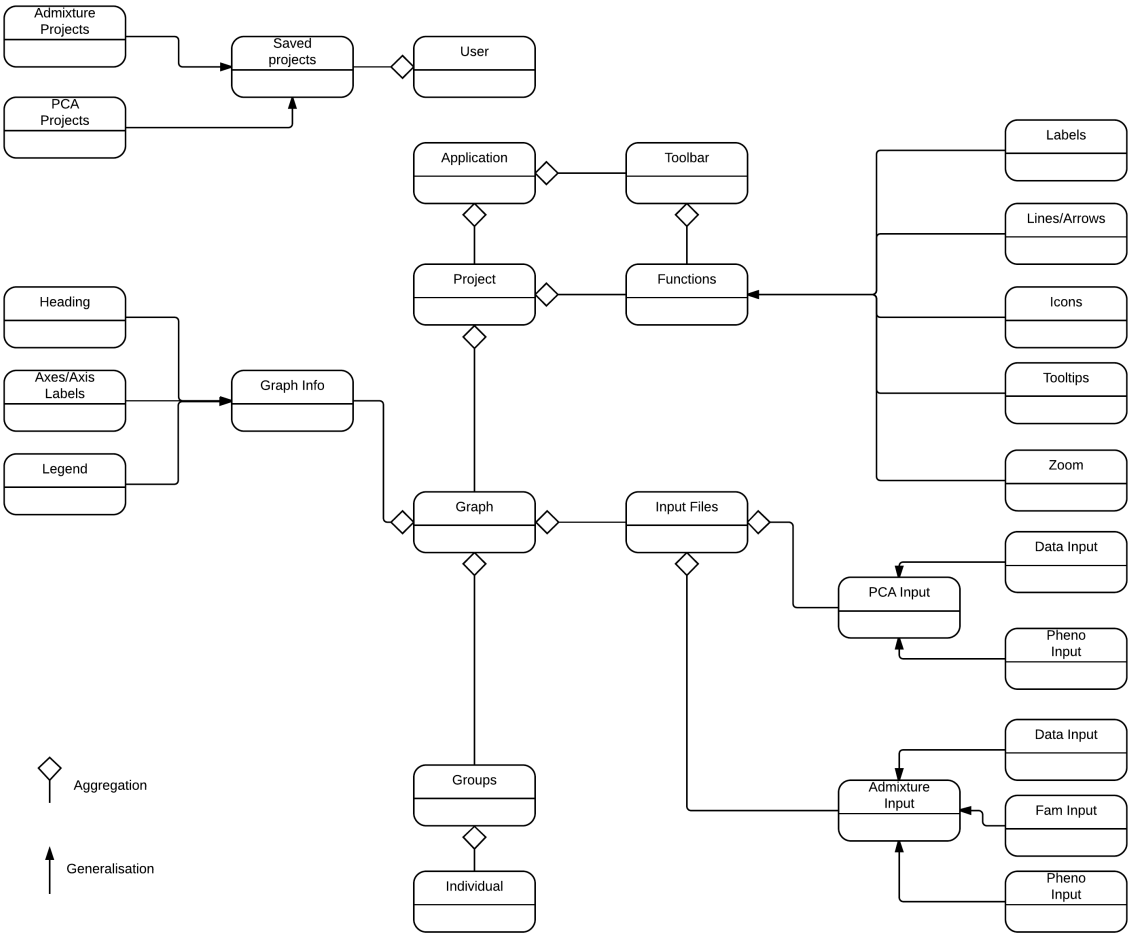

Figure 4: Genesis Domain Model

3 Use Case Modelling

The use case analysis step of the *ICONIX* process is the main driver of the design technique. This is the beginning of the creation of the dynamic branch of the UML model. The use cases created for the Genesis application are highlighted below.

Use Case 1: Create a simple PCA plot

Basic Flow:

- User selects the new project button on the start screen
- User is directed to the graph selection screen
- User selects graph type
- User directed to data selection screen
- User clicks add data file button
- Popup window appears allowing user to select PCA data file
- User selects the required data file
- Required PCAs are selected from dropdown list
- User clicks done
- Basic PCA with default axis labels and legend is plotted

Alternative Flow:

- The incorrect file type is selected for the PCA data file
- The file selected does not exist when selected by the user.

Use Case 2: Create a PCA plot with phenotype data included

Basic Flow:

- User selects the new project button on the start screen
- User is directed to the graph selection screen
- User selects graph type
- User directed to data selection screen
- User clicks add data file button
- Popup window appears allowing user to select PCA data file
- User selects the required data file
- User selects required phenotype file
- Required PCAs are selected from dropdown list
- Phenotype file column is selected from dropdown list
- User clicks done
- Basic PCA with default axis labels and legend is plotted

Alternative Flow:

- The incorrect file type is selected for the PCA data file
- The incorrect file type is selected for the PCA pheno file.
- The file selected does not exist when selected by the user.

Use Case 3: Create a PCA plot with phenotype data and user defined plot labels

Basic Flow:

- User selects the new project button on the start screen
- User is directed to the graph selection screen
- User selects graph type
- User directed to data selection screen
- User clicks add data file button
- Popup window appears allowing user to select PCA data file
- User selects the required data file
- User selects required phenotype file
- Required PCAs are selected from dropdown list
- Phenotype file column is selected from dropdown list
- User clicks Next
- User is directed to graph axis and heading customization screen
- User inputs required heading and axis labels
- User clicks done
- Basic PCA with custom axis and heading labels and legend is plotted

Alternative Flow:

- The incorrect file type is selected for the PCA data file
- The incorrect file type is selected for the PCA pheno file.
- The file selected does not exist when selected by the user.
- Invalid character is used for axis and heading label input.

Use Case 4: EditModify a selected icon present on the PCA plot

Basic Flow:

- User clicks on icon required to change
- Data associated with point is displayed.
- User clicks edit Icon button
- Edit Icon popup appears
- User can edit icon colour, icon shape and icon size
- Once user has set the required quantities/data, they click the done button.
- The PCA plot is updated with the new Icon with the parameters set by the user.

Alternative Flow:

- A colour that does not exist is set for the icon colour.
- A shape that does not exist is set for the icon shape.

Use Case 5: Add a line to the PCA plot

Basic Flow:

- User clicks on add line button.
- User clicks on two points to set beginning and end points of the line.
- Line is drawn.

Alternative Flow:

- User sets a start or end point for the line out of the scatter plot range.

Use Case 6: Add an arrow to the PCA plot

Basic Flow:

- User clicks on add arrow button.
- User clicks on two points to set beginning and end points of the arrow.
- Arrow is drawn.

Alternative Flow:

- User sets a start or end point for the arrow out of the scatter plot range.

Use Case 7: Add a label to the PCA plot

Basic Flow:

- User clicks on add label button.
- User clicks on position to set label on the scatter plot.
- Popup appears to input text for the new label.
- User inputs text and clicks done
- Label is created and placed in selected position on the scatter chart.

Alternative Flow:

- User selects a point for the new label not in range of the scatter plot.
- User inputs invalid characters for label text.

Use Case 8: Save a user created PCA plot to file

Basic Flow:

- With a user created plot present, the user clicks the save button.
- A popup window appears to allow user to select directory and name to save the graph file.
- User sets the directory.
- User inputs the name to save the file.
- User clicks the save button on the popup and the file is saved in the chosen directory with the chosen file name.

Alternative Flow:

- User selects an invalid directory to save the file.
- User inputs a file name that already exists in the chosen directory.
- User input and invalid character to save file.

Use Case 9: Load a PCA plot save file

Basic Flow:

- Either from the start screen or the plot edit screen, the user clicks the load file button.
- A popup window appears to allow user to select file required to be loaded to the application.
- User selects the required file.
- User clicks the load button.
- The chosen file is loaded to the application and the user can update/edit the plot.

Alternative Flow:

- User selects an invalid/restricted directory to load the file.
- User selects a file of the incorrect file type to load to the system.

Use Case 10: Export PCA plot project to PDF file

Basic Flow:

- With a user created plot present, the user selects the export to PDF button.
- A popup window appears to allow user to select the directory and name to create the PDF file.
- User set the directory.
- User inputs the PDF file name.
- The user clicks done and the system generates a PDF with the given location and name

Alternative Flow:

- User selects an invalid directory to save the file.
- User inputs a file name that already exists in the chosen directory.
- User input and invalid character to save file.

4 Requirements & Review

At this stage of the design, the use cases and domain model are evaluated to ensure they collectively capture the requirements of the software to be developed. Upon evaluation at this stage it was noted that the domain model required some alteration, however the use case were deemed acceptable. As a result the domain model was revised, and Figure 5 illustrates the changes made.

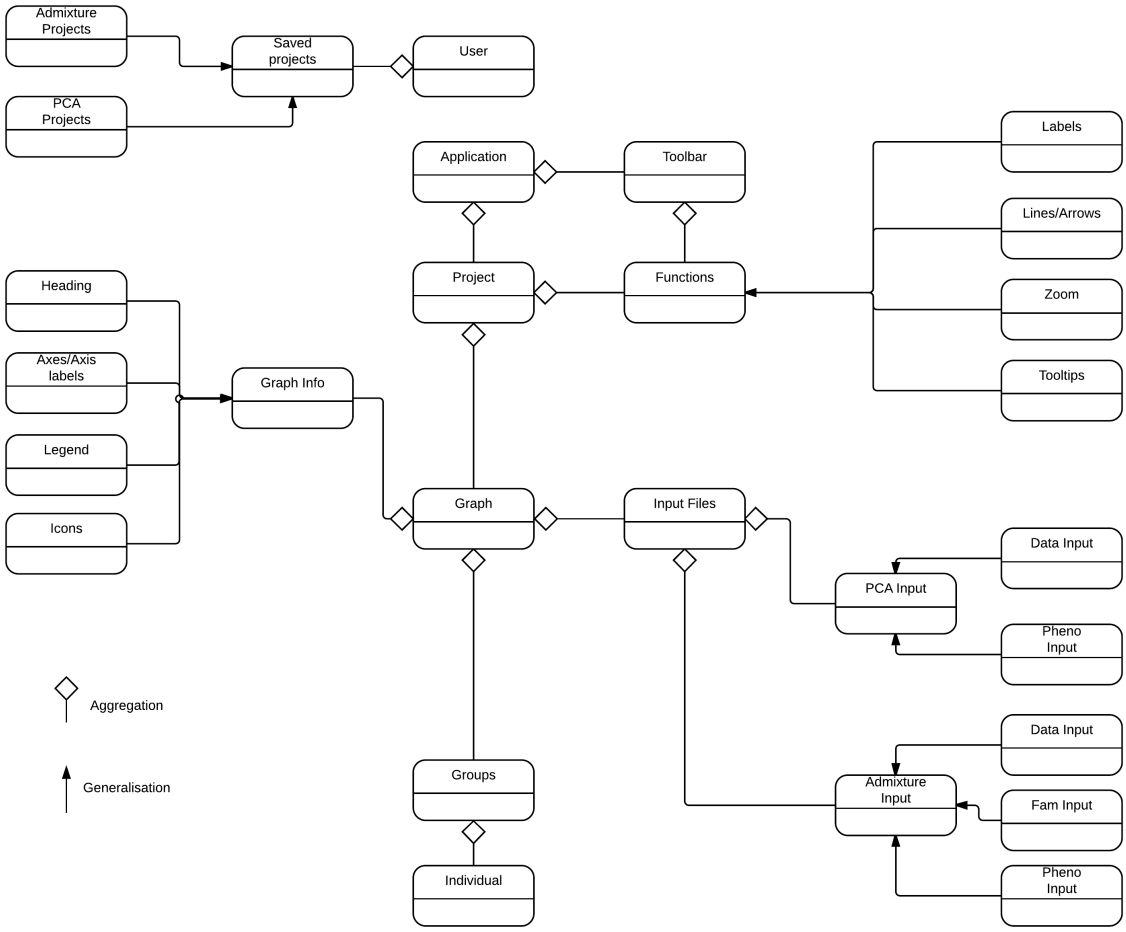

Figure 5: Genesis Domain Model Revised

5 Robustness Analysis

The collaboration of objects based on functionality required for each use case is decided at this stage of the design. The robustness diagrams illustrate the interactions between objects based on the specified scenario. The robustness diagrams developed for all presented use cases are shown below.

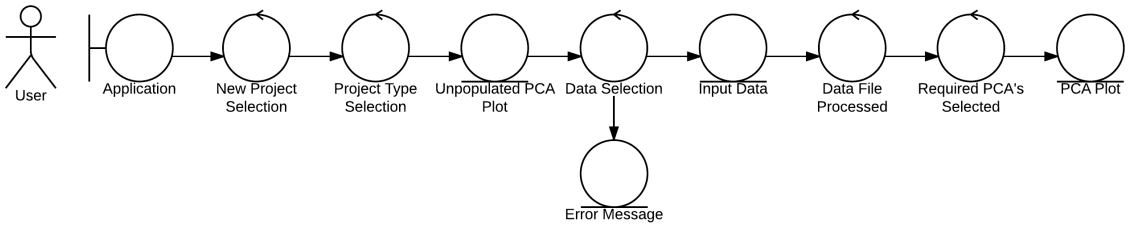

Figure 6: Robustness Diagram Use Case 1

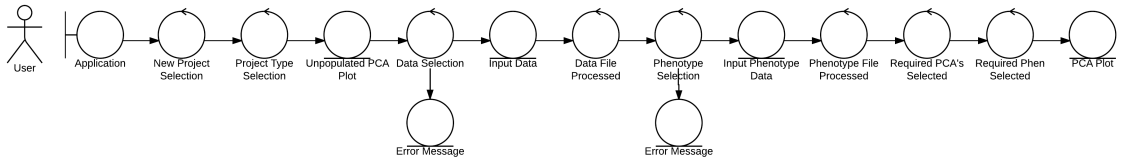

Figure 7: Robustness Diagram Use Case 2

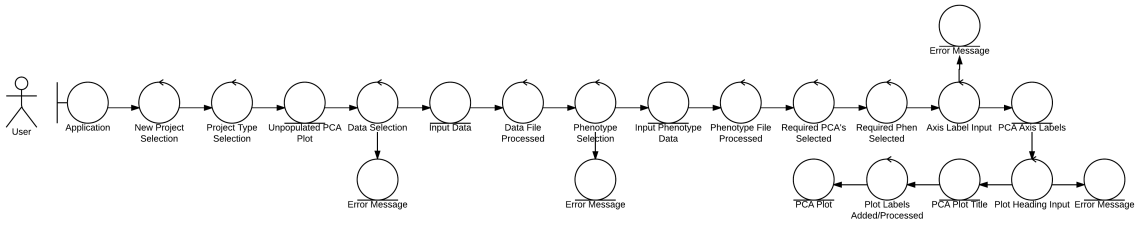

Figure 8: Robustness Diagram Use Case 3

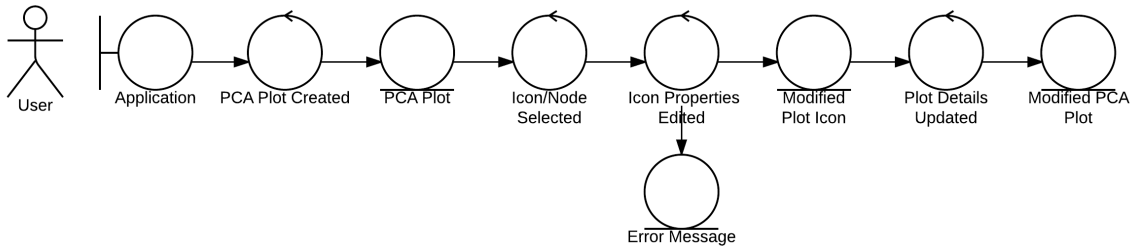

Figure 9: Robustness Diagram Use Case 4

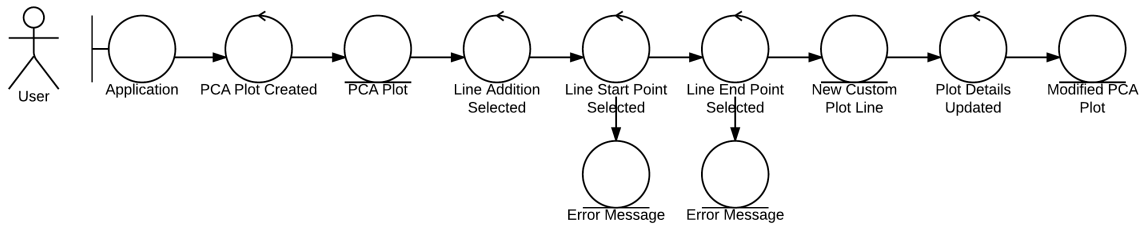

Figure 10: Robustness Diagram Use Case 5

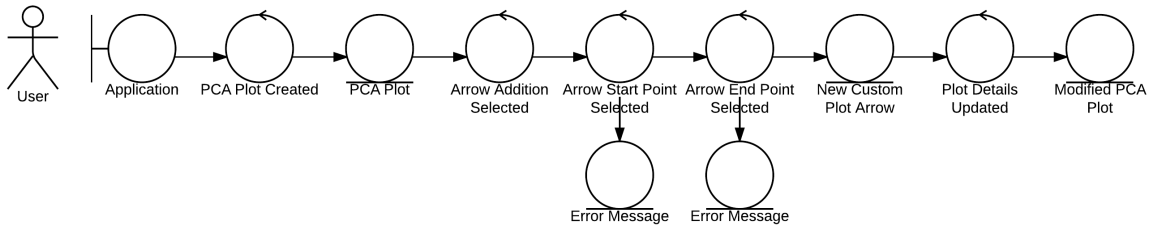

Figure 11: Robustness Diagram Use Case 6

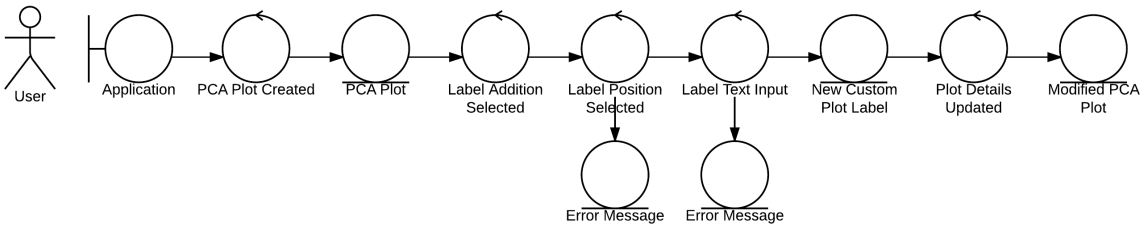

Figure 12: Robustness Diagram Use Case 7

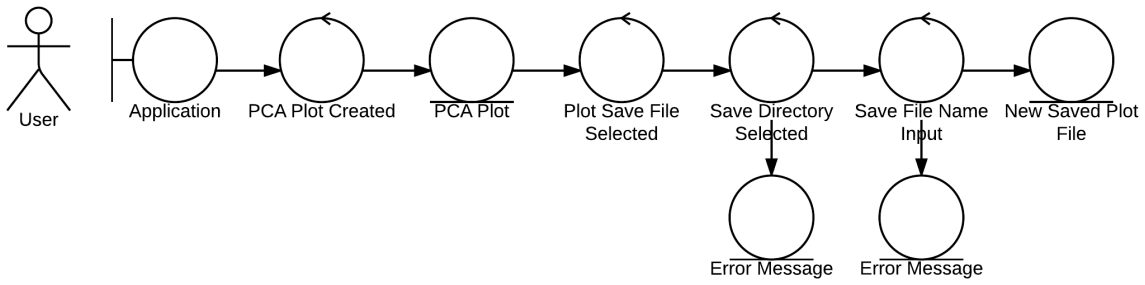

Figure 13: Robustness Diagram Use Case 8

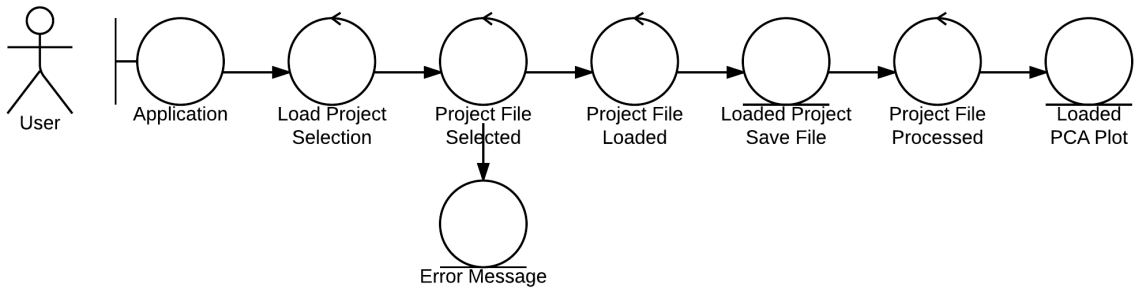

Figure 14: Robustness Diagram Use Case 9

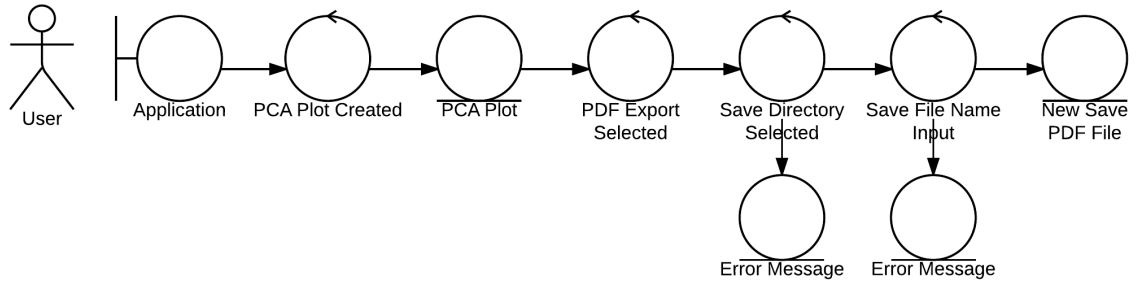

Figure 15: Robustness Diagram Use Case 10

6 Preliminary Design Review

The Preliminary Design Review (PDR) is a process by which robustness diagrams and use cases are compared to ensure functionality presented in the use cases is correctly presented by the interactions in the robustness diagrams. The entities in the robustness diagrams are also compared to those present in the domain model to ensure consistency is present between the two. The PDR was carried out for the Genesis software application, upon evaluation it was noted that no changes were required.

7 Sequence Diagrams

The first stage in the detailed part of the design is the sequence diagrams, these diagrams allow for the allocation of behaviour. The sequence diagrams are responsible for three main tasks,

- The allocation of behaviour to boundary, entity and control objects.
- Highlight detailed action of objects associated with each use case.
- Provide a distribution for the operations among classes.

The sequence diagrams related to the Genesis application are presented below.

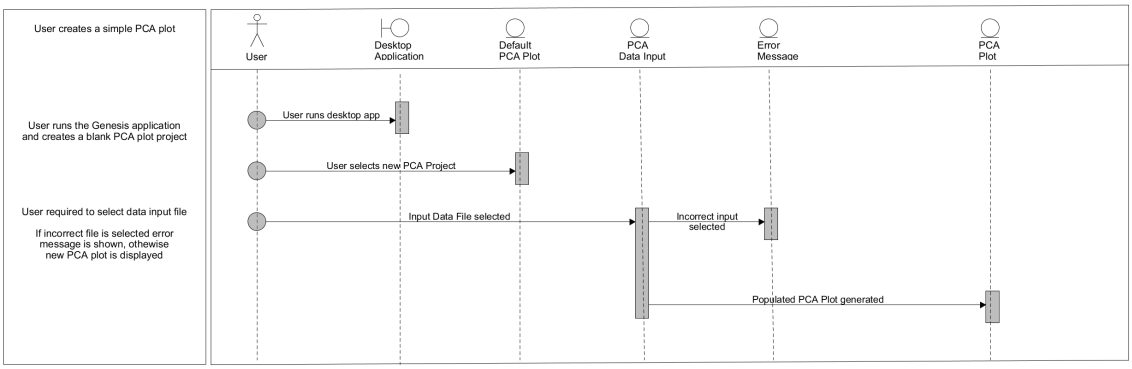

Figure 16: Sequence Diagram Use Case 1

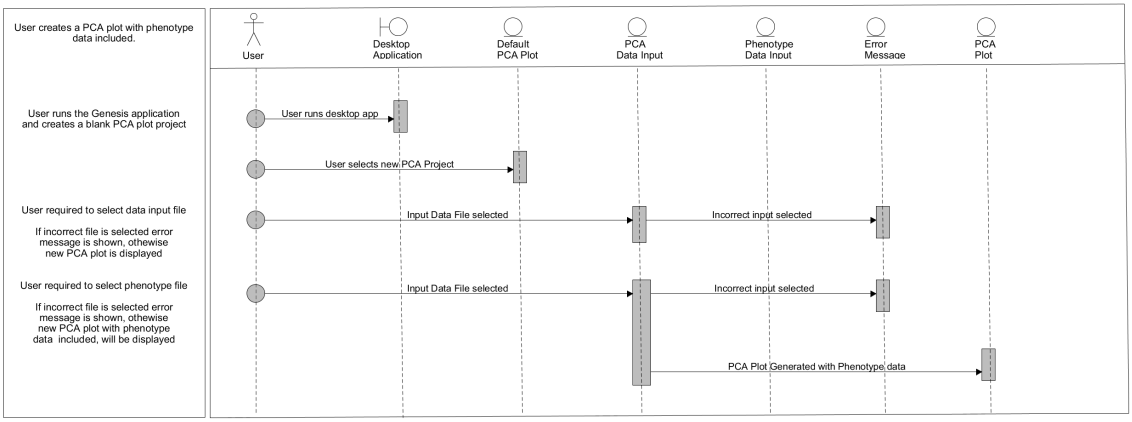

Figure 17: Sequence Diagram Use Case 2

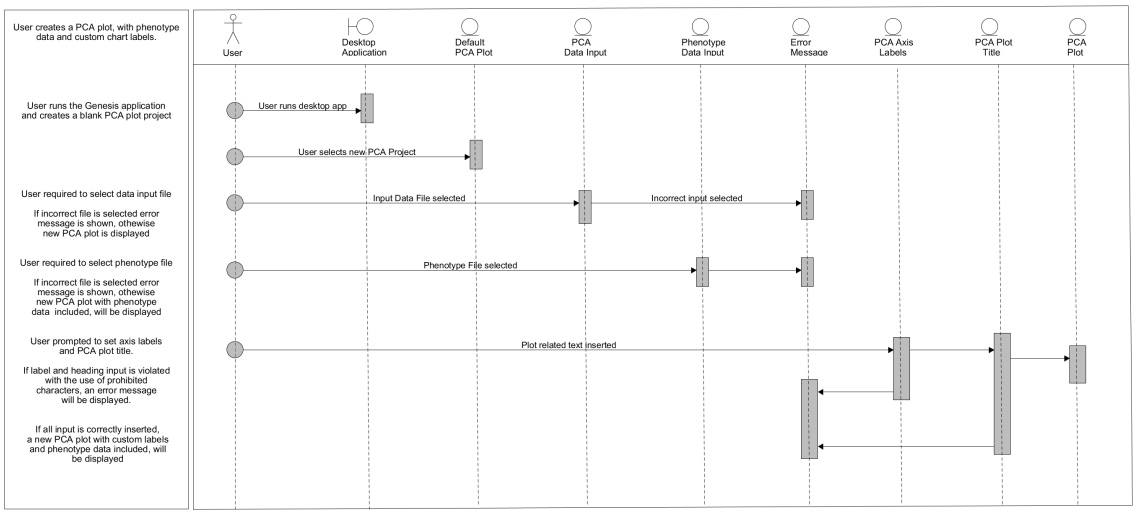

Figure 18: Sequence Diagram Use Case 3

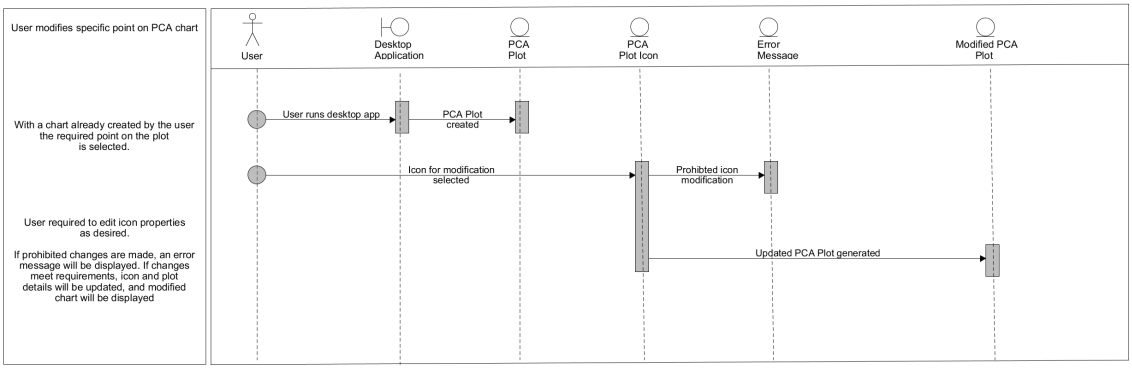

Figure 19: Sequence Diagram Use Case 4

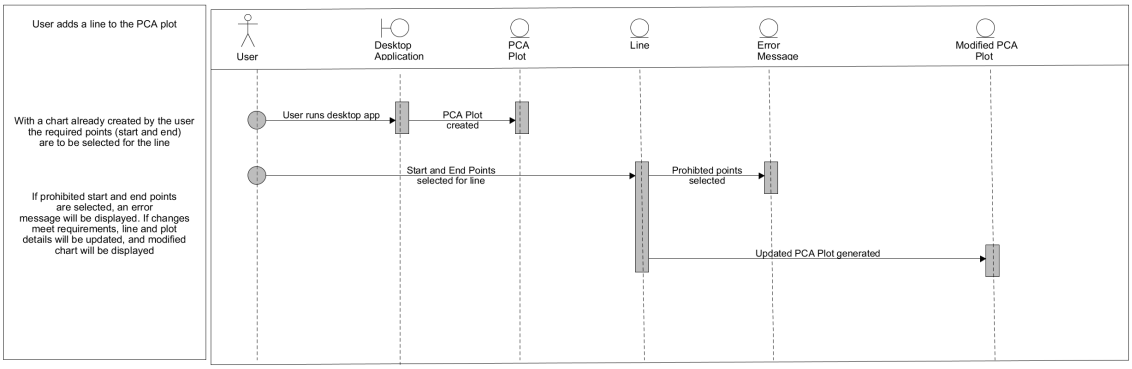

Figure 20: Sequence Diagram Use Case 5

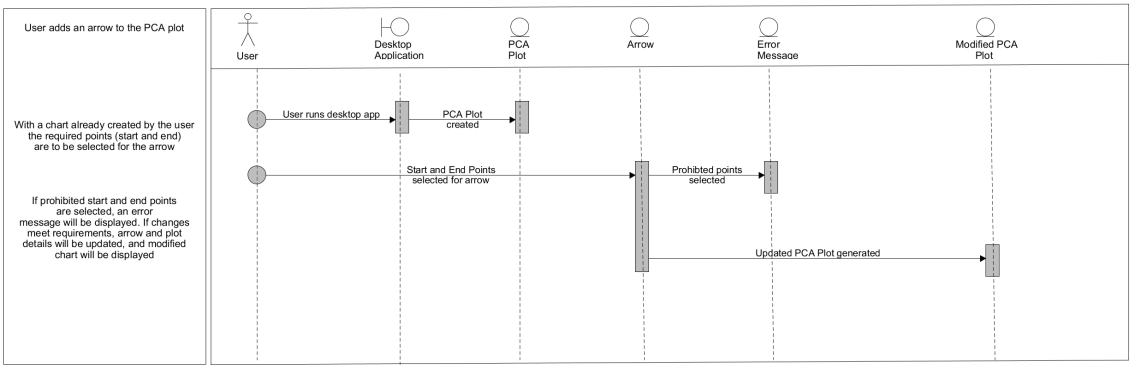

Figure 21: Sequence Diagram Use Case 6

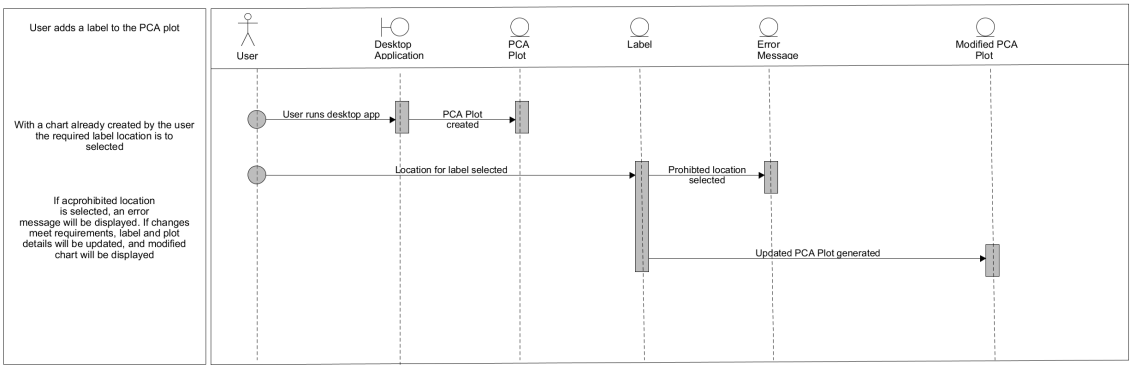

Figure 22: Sequence Diagram Use Case 7

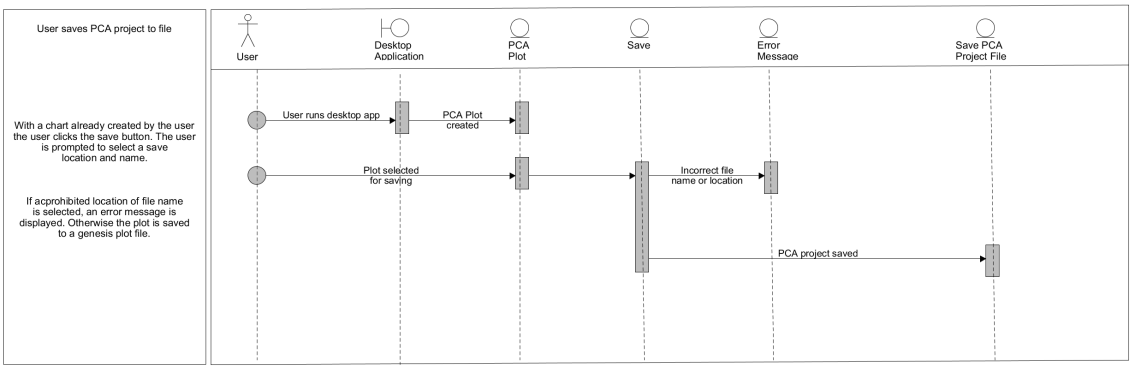

Figure 23: Sequence Diagram Use Case 8

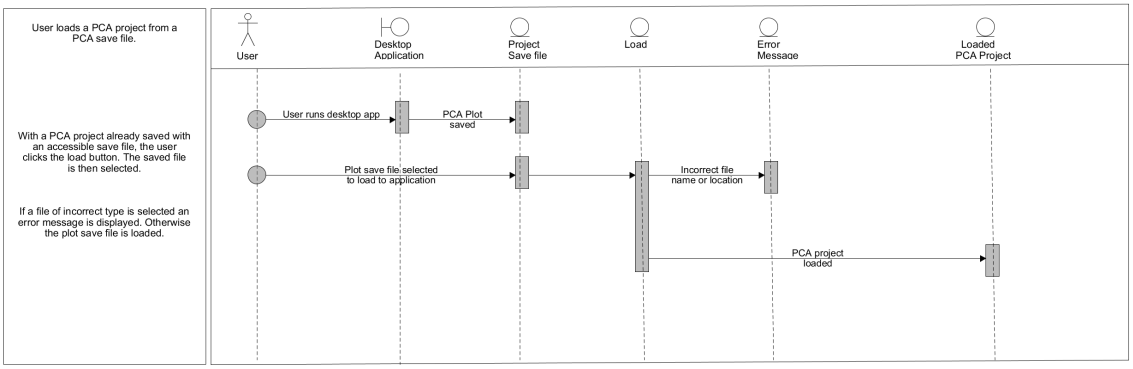

Figure 24: Sequence Diagram Use Case 9

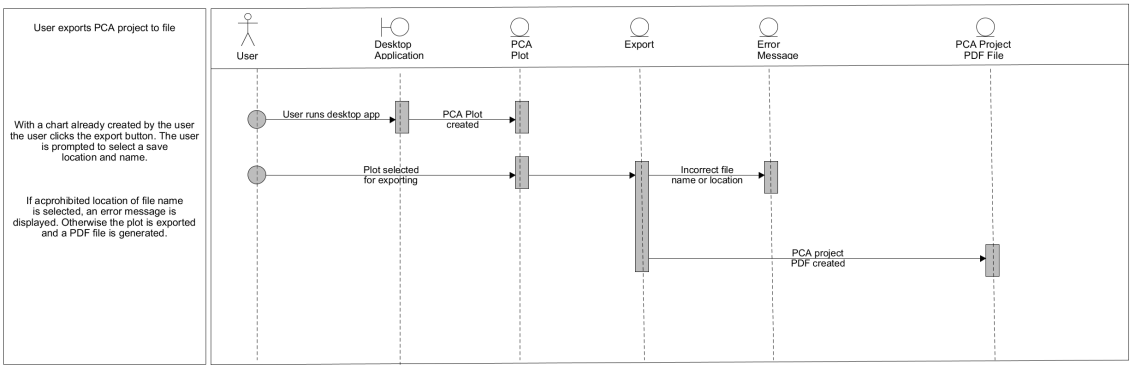

Figure 25: Sequence Diagram Use Case 10

8 Class Diagram

The penultimate stage of the design, this step is undertaken upon completion of all the sequence diagrams. At this stage all work that has previously been done is captured to produce the application class diagram. This class diagram is ultimately used to implement the software that has been designed. The class diagram generated for the Genesis application is shown in Figure 26.

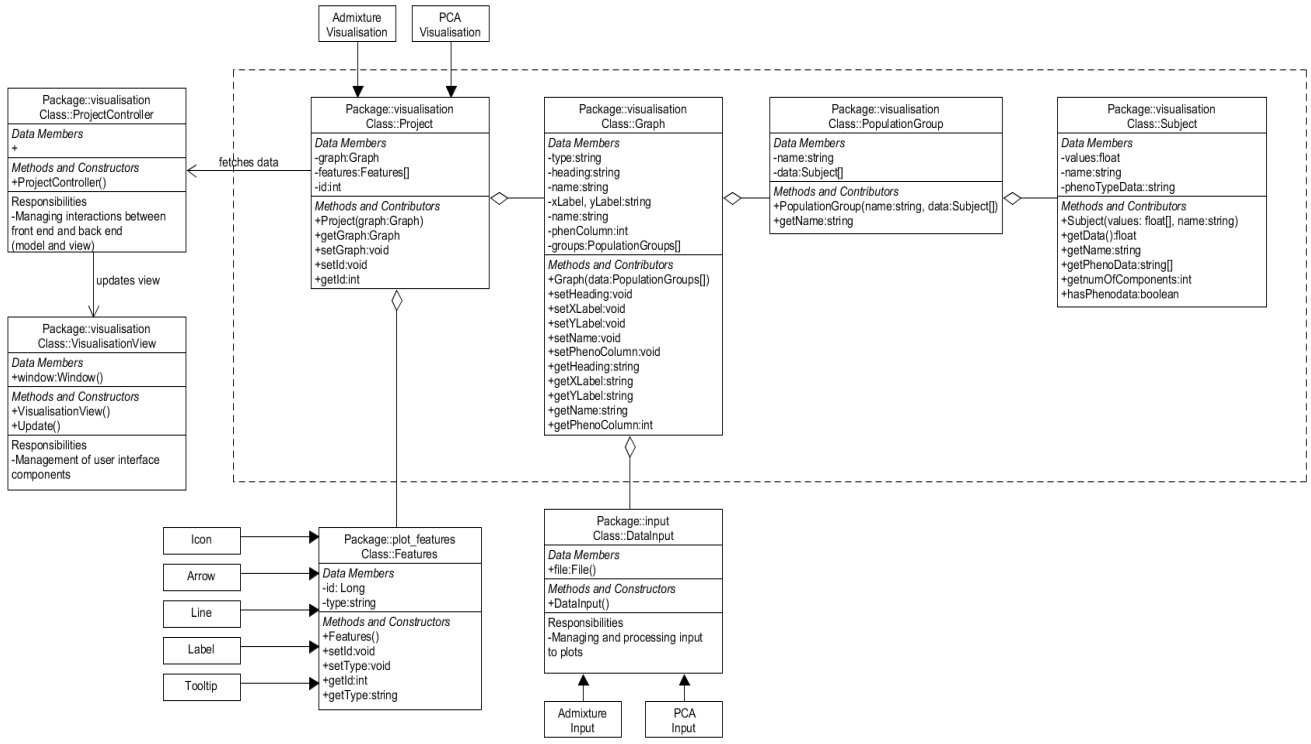

Figure 26: Genesis Class Diagram

9 Design Review

The main aim in the design of the Genesis application is to remedy the design flaws present in the current implementation of the system. By designing a system that promotes extendabiliy and modularity these design flaws are remedied. The design presented in this document achieves this goal while at the same time maintaining the functionality provided by the original application.