BlueRoll Design Specification

by Spencer Kulwicki, Grady Salzman, Maanas Varma Datla

1.0 Introduction

1.1 Purpose

The purpose of this document is to give a detailed description of the overall design of BlueRoll including the architecture of the system and class diagrams. This document is also intended to describe how the architecture of the system and class diagrams accommodate requirements specified in the BlueRoll System Requirements Specifications document.

1.2 Definitions

.csv file: a simple comma separated value file.

Bluetooth Address: a unique 48-bit address usually presented in the form of a 12-digit hexadecimal value identifying a Bluetooth device.

Class: a .csv file containing bluetooth addresses of registered students.

Instructor: main user of BlueRoll with the power to run functions.

Roll call: The function that matches all visible devices' Bluetooth addresses at that time and match them against addresses in the Class file compiling a Summary Report.

Semester Report: a compilation of all summary reports for a class.

Student: a Bluetooth address of a device registered in the class.

Summary Report: a compilation of attendance data for a class.

2.0 Architectural Strategies

BlueRoll implements a Centralized Control style of architectural structure. It was chosen because this program's features agreed most with a Centralised Control Architecture which allows for employing a Data-flow style architecture without compromising on the back-propagation features between data and Edit/Create classes. For the BlueRoll system, Data needs to transfer linearly from the load class to the Summary report generator and then to Semester report and so on, but we also need a way to alter the data itself when we the professor to deal with students dropping classes. In this case, the student's field needs to be deleted from the original data. To achieve this, the Centralized Control Model was adopted. Figure 1 shows the overall control flow of the program. Primary control is retained in the centralized controller while the Graphical User Interface is laid over the controller, allowing the user to make use of the functions in it.

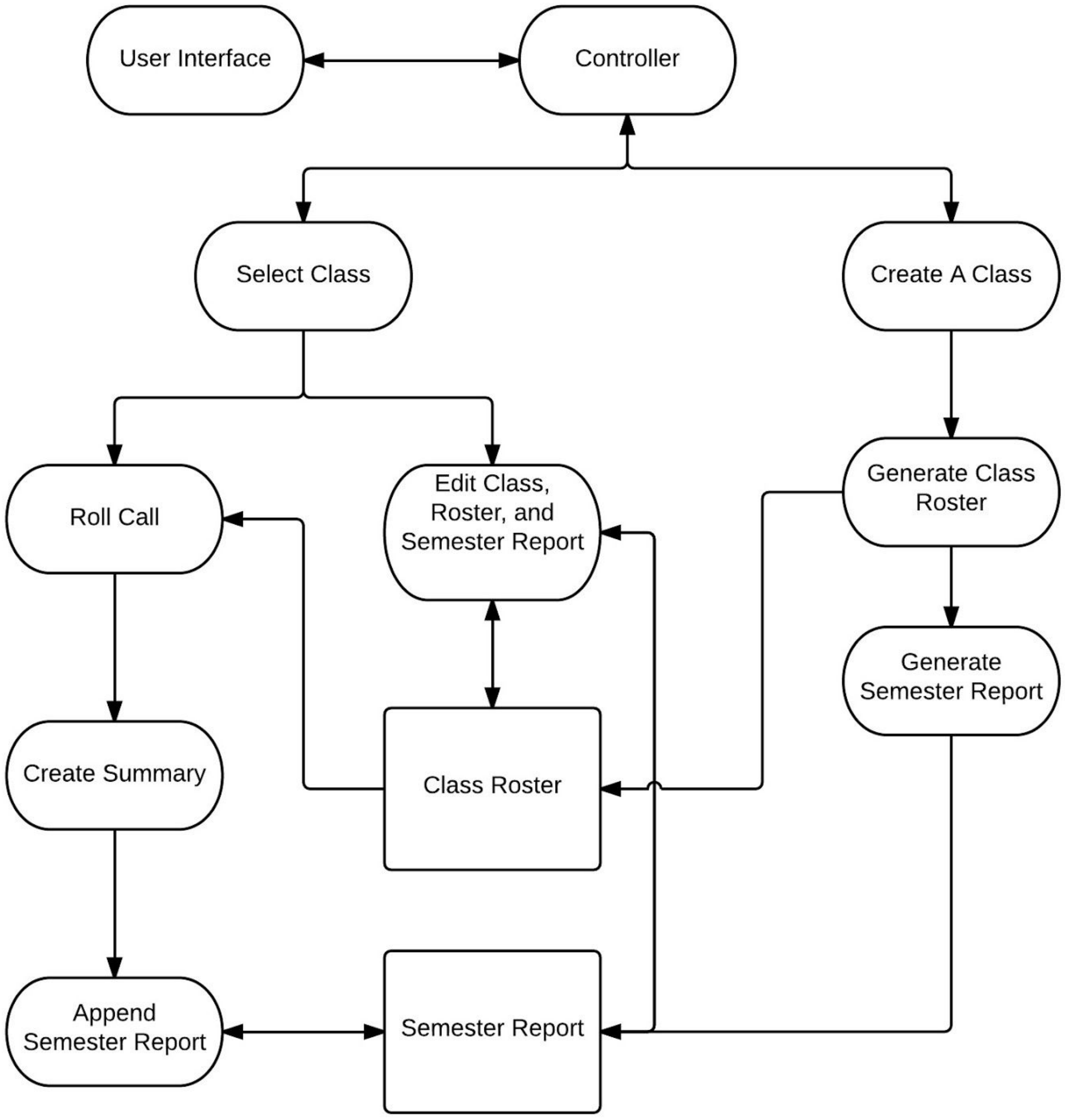

Figure 1 | Architecture Diagram

3.0 Class Diagrams

BlueRoll is decomposed into four different classes. The Controller class is used to manage the various functions and their interactions. The functions contained in the Controller class are detailed in Figure 3.1. The purpose of these functions are to mediate the interactions between the UserInterface, Class, and RollCall. Controller is related to Class via a one-to-many relationship, while the Controller - RollCall and Controller - UserInterface relationship is one to one. Class contains Semester_Report and Class_Roster as String Matrices and the Class name as a string. It implements the following funcitons:

editRoster() allowing to edit students in the class

create(Class_Name, Class_Roster) which creating a new class passing a class name

editSemesterReport() that allows a professor to manually change attendance of a student.

RollCall manages the primary part of the program which are searching for all visible devices, comparing them to a class roster, generating a Summary report and appending it to the Semester report. Rollcall and Class are related via a one to one relationship. It contains an instance of Class as well as the following fuctions:

findDevices() a function which uses Bluetooth to detect devices nearby

createSummaryReport() determines attendance of students append

SemesterReport() saves attendance to the semester report.

The final class is UserInterface which contains functions all relevant to displaying the information and options available to the user.

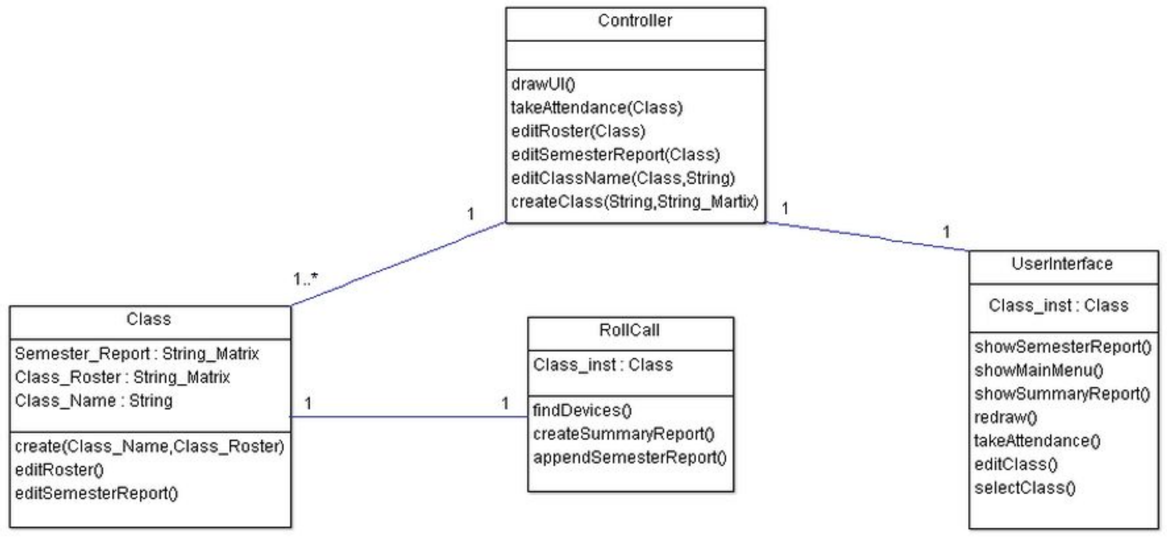

Figure 2 | UML Class Diagram

4.0 Traceability

All of BlueRoll's functions can be related back to a part of the BlueRoll System Requirements document, reference numbers are given.. From 3.2.1.2 Roll Call of the requirements, it is stated that the system will be able to locate and compile the bluetooth ids with in the vicinity of the professor. This is accomplished through the findDevices() function in the RollCall class. The findDevices() searches for all Bluetooth addresses within range accomplishing 3.2.1.2.2. These ids will then be matched against a main roster file and will mark students as present or absent if a the device matches satisfying 3.2.1.2.3 and 3.2.1.2.4. Then the results are used to create a summary report via the createSummaryReport function in the RollCall class. The semester report file will then be updated after each iteration of the program with the information from the summary report, which satisfies the requirement of a detailed semester report completing 3.2.1.3. From phase two (3.2.2), it is stated that the professor will be able to manually update the report in the event that a student does not have a bluetooth device on hand when the RollCall occurs (3.2.2.1). This requirement is satisfied via the editSemesterReport() function in the Class class. The instructor is allotted multiple classes, 3.2.2.2, which is satisfied by the one to many relationship between the controller and the Class class as well as the main menu's ability to display multiple classes. Finally all

UI requirements, 3.2.2.3, are implemented through the User Interface class. This class will satisfy this by implemening the showSemesterReport(), showSummaryReport(), showMainMenu(), and redraw() functions. Future Requirements are implementations of a mobile end for the professor (3.2.3.1) as well for the student (3.2.3.2). Our Architecture Structure and UML diagrams are made in a way that would be easily mimicked for a mobile platform using the same structure with a different user interface, therefore satisfying all future requirements.