A-level Coursework

# Analysis

## Background

### Introduction

In an attempt to find a problem I could apply my Computer Science Coursework to, I had a discussion with the head of music at our school to see if there were any problems facing the department that could be solved with the application of computer science.

In the discussion, it was revealed that student engagement was a problem within the music department, especially for the younger years. The teacher also explained that a lot of students built up the impression that music was not an amazingly diverse subject, which is contrary to reality. So, the basic requirement was to design a solution that would allow students to get engaged with the subject and demonstrate how diverse the subject could be.

During the discussion, it was decided that automating an instrument to play itself would match many of the criteria.

This would not only meet the criteria mentioned above, but also provide a tool that could be used to teach students how different instruments make different sounds.

### Current System

Student engagement is a problem that does not have a simple solution. My system would be one of many techniques that could be used to improve the overall standard of music lessons in the school.

### Clients and Users

The clients would be the music department at the school.

The users would be the music teachers but also the music students.

### Business case for change

### Overview of requirements

The system I design must be able to:

* Increase student attention in lesson
  + Must be interesting/unusual
  + Must be simple to explain how the music aspect of it works
* Be interactive
  + Have features that students can use to play with
  + Must be robust enough to let the students use and experiment with
* Be easy to set up
  + Students and teachers alike will be using this device and so it need to be simple to set up and use
    - Keep number of cables to plug in to a minimum
    - Keep number of first time set up steps to a minimum (4?)
    - Keep number of sequential set up steps to a minimum (2?)
* Be versatile
  + Allow students to play with as well as providing demonstration for the class
  + Allow for multiple different ways to interact with it
    - Downloading midi files from the internet
    - Plugging into existing midi keyboards and systems

### Technical Requirements

* Easily import MIDI files that have been downloaded or made by the music students
* Easily select what track should be played by the robot
  + GUI with all tacks displayed and a representation of what notes will be played
    - Either sheet music or a graphical score
* Be easy to set up
  + Single cable from computer to device (Power??)
  + No drivers required.
  + No looking for the right port to use
  + Simple, well documented user interface
* Robot needs to be able to play the song that it receives in real time.
* Robot should be compatible with other, existing, MIDI devices.
  + Follows MIDI protocol
  + Be able to be used with at least the current set of keyboards found in the music classrooms
* Robot needs to be able to play notes quickly to allow it to play as many different songs as possible.

## Analysis of the Problem

### Contstraints and limitations

The limitations I face with this project are that the Arduino only has a relatively small memory available to the program (32KB for the script and 2KB of RAM). This means the program I write for the Arduino will need to be very light weight so it can be run on the Arduino.

The Arduino is also slow. This means it may lag behind the MIDI inputs and drop bits of data and commands. This means I need to develop a program that can keep up with the torrent of data that is supplied by the serial input.

The Arduino serial input buffer is also quite small. This means that I would need to keep the data processing quick, so the buffer does not fill up and then miss bits of data and commands.

### Scope of the problem

### Model the System

#### Decomposition

(Structure diagram)

#### System Flow-Charts

#### Data flow diagrams

#### data dictionaries

#### Entity Relationship Diagram

#### Class diagrams

## Objectives

### SMART objectives & evaluation criteria

## Appendix

### Notes from interviews

# Documented Design

# Technical Solution

# Testing

# Evaluation