The Computing Practical Project

School Timetabler

Analysis

Background to and Identification of Problem

My client for my timetable is Mrs Lansdown, Head of Sixth Form at Hitchin Boys’ School and constructs timetables for the whole school.. Hitchin Boys’ School is a secondary educational establishment with a sixth form which is part of the Hitchin Sixth Form consortium with Hitchin Girls’ School and The Priory School. The school provides a secondary level education with the aim to carry out GCSEs, followed by further education towards AS and A2 exams. This is given by well trained staff with the use of projectors, interactive whiteboards and textbooks. Currently, I am studying form A2 at Hitchin Boys’ School sixth form and Mrs Lansdown is the head of the sixth form.

Description of the Current System

Currently, Mrs Lansdown uses Nova T6 to construct timetables for the whole school and, in conjunction with the other heads of sixth in the consortium, the sixth form. Nova T6 works in conjunction with the school management system SIMS, which has an SQL server containing staff and student information which Nova T6 has access to. Sadly, Nova T6’s automatic timetable construction functionality causes staff and students to travel unnecessary distances between lessons, even though classrooms are available within a small area. Currently, the timetable is initially constructed by Nova T6, then Mrs Lansdown manually adjusts the timetable to minimise distance travelled and other problems with the automatic system. In order to improve the automatic timetabling construction functionality, the new system would have to take in to account the distance between classrooms to reduce the distance travelled by staff over a day.

Identification of the Prospective User

Mrs Lansdown will be the immediate user of the system, but the system will be designed to allow for fast tutoring in the usage of the system, to allow all following heads of sixth form to continue to use the system.

Identification of User Needs and Acceptable Limitations

After having an interview with the client, I have ascertained the client wants a system which generates a timetable using staff, student and subject data from the current database in the SIMS system. Unfortunately, the SIMS system and its database requires a licence to access the database and interface provides by SIMS, which requires a fee to be paid to Capita and a non-free training course in how to use the interface and database. The system must have simple and intuitive user interface which utilises blocks to represent the periods in a day and which lessons are running within the block. As each teacher must have at least a specific number of ‘contact’ hours with students, the system must be able to calculate and display the number of hours worked by each member of staff and specify what subjects the members of staff are qualified to teach. Since the new system will take into account the distances between the class rooms, each member of staff should have a preferred class room where the system will attempt to base as many lessons of their lessons as possible and classrooms should have a subject linked to them to avoid large distances between lessons for subject staff.

TODO: current docs

Data Sources and Destinations

The current system utilises a Microsoft SQL server to store all data for use within the system, which is updated using the interface provided by the system. Since the database requires a non-free licence to access, the source of the data for the new system will have to different to the current system. In order to make the data entry as easy as possible, the data about the staff, subjects, classes and map of the school will all be in the CSV format, each will be their own file. This allows the user to use an external program like Microsoft Office Excel to populate the files. A configuration file will be used to specify information about the data files, such as their locations and file types, which will be in the YAML format as it is easy to modify and process. The staff, subject and class data will be dynamic as it has to be updated at runtime and as the school changes through out the program's life cycle. The map will be treated as static data, because the layout of the school will vary rarely. Also, the configuration file will be treated as static data as it will only be modified when the data about the school has been heavily modified.

TODO: output data and possible CSV export from SIMS

Data Volumes

The volume of data to process will be large, as the school teaches many subjects, has a vast array of members of staff and a large number of individual classes, but this data will all be text data which is small and can be efficiently compressed. This data will only have to accessed and processed roughly once per academic year, as the school will only need to generate timetables at the beginning of an academic year, and will only be accessed by one member of staff.

Data Dictionary

| **Data** | **Data Type** | **Description** |
| --- | --- | --- |
| Staff Data | CSV Text File | A list of all the members of staff available for timetabling. Each entry will contain: staff ID, name, available subjects, preferred contact hours. |
| Subject Data | CSV Text File | A list of all subjects taught by the school. Each entry will contain: subject ID, subject name. |
| Class Data | CSV Text File | A list of all classes at the school. Each entry will contain: class ID, subject ID. |
| Configuration File | YAML Text File | A 4-space indented YAML file which contain information about the school data, such as file type and location. |

Data Flow Diagrams

Entity Relationship Diagrams (for databases)

Object Orientation planning (for Object Orientation)

Objectives

|  |  |
| --- | --- |
| **Objective Number** | **Details** |
| **1** | The system must be able to parse a map of the school and calculate the distances between every classroom to every other classroom. The map does not need to be kept in memory after the distances are calculated as the distances are the only information which is required from the map. |
| **2** | The system must be able to parse files containing information about the school, such as subject, staff and class data. This data should be stored in a database rather than in memory. |
| **3** | The system must allow the user to modify the school data at runtime to allow the user to perform validation on the data within the system before any processing is done to the data. |
| **4** | The system must allow the user to input a predetermined lesson timetable where the lessons are already timetabled. This must be done using a grid-based interface, allowing the user to specify colours for subjects within the grid. |
| **5** | The system must allow the user to specify the file type of the school data, the location of the map files, and the top map, using a configuration file. |

[https://www.mindmup.com/#m:a17ff69ab0ad720133e69d2110f3315c98](https://www.mindmup.com/" \l "m:a17ff69ab0ad720133e69d2110f3315c98)

Potential Solutions

Chosen Solution

Design

Overall System Design

Modular Design

Definition of data requirements (Design Data Dictionary)

Definition of record structure

Validation required

File organisation and processing (if appropriate)

Database design including normalised relations (if appropriate)

Sample of planned SQL queries (if appropriate)

Identification of storage media

Identification of suitable algorithms for data transformation, pseudocode of these algorithms

Class definitions(diagrams) and detail of object behaviours and methods (if appropriate)

User interface design (HCI) rationale

Description of measures planned for security and integrity of data

Overall test strategy

Implmentation

System Testing

Testing Plan

Figure Reference

System Maintainence

System Overview

Detailed Algorithm Design

Procedure and variable Lists

Annotated listings of program code / macro code and tailoring

Database tables in design view

User Manual

Title page

Contents page

A brief introduction and installation instructions

Detailed description of the use of the full system

Samples of actual screen displays

Samples of error messages and error recovery procedures