Scheduling System

Design Specification

CS 499

Developed by:

Andrew Beussink
Adam Estel
Ohtsuki Motomu
Bert Sanders
Abigail Young

Table of Contents

Feature List

Committed
The Scheduling System will allow a user to upload csv files containing class rosters which will then be processed into the database. The scheduling segment will then allow the user to create a new "Macro" Meeting, selecting which classes should be included in the meeting. The system will then generate a base schedule, reporting any necessary conflicts if applicable. The schedule can then be exported as a csv file.

Updates, time permitting
Additional features will be added to the scheduling system, time permitting:

- Allowing the user to manually adjust the schedule before exporting
- Setting restrictions on when certain students and teachers can meet

Architecture

The Architecture of the Scheduling System is broken up into three distinct categories: Input/Output, Algorithm Implementation, and Database

- State Diagram
  - Will be added after the algorithm has been designed more completely
- Class Diagram
  - Will be added after the algorithm has been designed more completely

Input

The Input module will be implemented in PHP. HTML will be used for the graphical user interface.

The module will present the user with a normal HTML file upload form (a text box for the address, a "browse" button and an "ok" button). Once the address has been received, it will check that the file exists and copy it onto the server. It will then open the file in read-only mode, create an associative array, parse the file into the array and close the file. It will then store the array into the database and call the scheduling module.

The input module will use the following functions:

- getCSV: gets file address from user, checks to make sure it exists and copies file to server
- csvToArray: another function receives the address, opens the file (read-only), creates the array parses the file into the array and closes the file
- storeArray: another function receives the array and stores it in the database

Output

The Output module will be implemented in PHP and will use HTML through a Smarty template for display.

The module will create a new .csv file and open it in write mode. It will then read the schedule from the database and store it in the file and close the file. It will then reopen the file in read-only mode and print the schedule to the screen.

The output module will use the following functions:

- createCSV: one function creates a .csv file, opens it, reads the schedule from the database , stores it in the file and closes the file.
- printCSV: Another function receives the file address, opens it (read-only), prints it to screen and closes the file.

Algorithm Design

The exact scheduling algorithm that we will use for our Scheduling System has not yet been determined. The goals that our algorithm needs to achieve can be broken down into two categories.

First, the requirement that our system must achieve in order to be a success is having each student meet with all of their teachers. This requires taking into account teachers schedules and student/parent availability.

Secondly, we would like to achieve other non-essential goals such as minimizing the gaps between meetings for teachers and students.

Two possible designs that are being considered generate the schedule in an iterative process by generating a schedule for one student or teacher at a time. These designs could be combined in a hybrid approach to take into consideration certain teachers with restricted schedules. In this approach we would generate schedules for those teachers first and then complete the schedule by iterating through the list of students. We will use the evolutionary model to first achieve our primary goal, and then in later versions refine the initial schedule to better meet secondary goals.

Database Structure

The database will be MySQL based. The structure of the database is being designed to allow for scalability in the application while remaining efficient. The Scheduling System will have a single database as a back end, it's structure will be as follows:

Tables

- students – holds all student information indexed by student ID's
- teachers – holds all teacher information indexed by teacher ID's
- classes – holds all classes that necessitate a conference indexed by class ID
- student_classes – holds the Student/Class Relationships, ie. It holds a single entry for every student that is in every class, indexed by student ID and Class ID
- teacher_classes – holds the Teacher/Class Relationships, ie. It holds a single entry for every teacher that is in every class, indexed by teacher ID and Class ID
- uploaded_files – this will keep track of files that have been uploaded, allowing access to them for processing by Input
- download_files – this will keep track of files that have been processed and are ready to be downloaded
- meetings – holds a list of all information for each large-scale meeting that is set up. Individual conferences will be determined based on the meetings listed here
- meetings_indiv – holds a list of all individual meetings, indexed by student, teacher, and overall meeting (see above)
- help_files – holds a list of help files that will be available to aide the user during

This list will be updated as needed as the Algorithm is more precisely defined.

Commenting Conventions

All php files will contain comments at the top of the file indicating the following information:

- Original author
- Original date of creation
- Description of what the file does or what functions it contains
- Modified Authors
- Modified Dates
- What changes have been made

External Documentation

External documentation will consist of a file/paper based document that will guide a non-technical user through using the Scheduling System. The System itself will also contain help documentation on every page describing the options available and the recommended process.