SCORE MyCourses – A Course Scheduling System

Team members: Will Crawford, Sabba Petri, Erik Steggall, Justin Lazaro, Benjamin Ross

**Requirements**

**Document**

**Table of Contents**

**Change Record……………………………………………… 1**

**Overview…………………………………………………….. 1**

**Reference Documents………………………………………. 1**

**Definitions…………………………………………………... 2**

**Requirements……………………………………………….. 2**

* **[Functional: Algorithm, Database, User Interface, Backend]**

**Performance………………………………………………… 5**

* **[Usability, Constraints, Wish List, Coding Standards]**

**Change Record:**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Changes/Additions | Responsible Person |
|  |  |  |  |
|  |  |  |  |

**Overview:**

The MyCourses scheduling system will allow program managers, program administrators, lecturers, and students to create, assign, detail and sign up for courses. This will be an end-to-end system that incorporates an algorithm that will automatically schedule courses based on a list of constraints. The program administrator will designate and list out all the possible courses, times, and location. The program manager determines which courses and lecturers that will be offered in the particular quarter. The lecturer fills out his constraints and information about the course. Finally, the students signs up for the courses.

**Reference Documents:**

* **MyCourses Competition Description**

http://score-contest.org/2011/Projects.php#crnkovic

* **Python Language Reference**

http://docs.python.org/reference/

* **Django Framework Documentation**

http://docs.djangoproject.com/en/1.2/

* **Django Book, another Django reference**

http://www.djangobook.com/en/2.0/

* **MINION, a constraint solver**

http://minion.sourceforge.net/

**Definitions:**

1. **MyCourses** – The end-to-end course scheduling system
2. **PA** (Program Administrator) – Person who handles the entire MyCourses system
3. **PM** (Program Manager) – Person who handles individual department’s program and course offerings
4. **Quarter Course List** – This will be a list of courses to be offered in the particular quarter, every quarter will be unique and will be independent from the other quarters
5. **MyPortal** – This will be the main user interface (on the web browser) users will be interacting with
6. **Users** – Users can be considered any users of the MyCourses system

**Requirements:**

* **Functional –** 
  + ***Algorithm*** - The main algorithm that will be implemented into S.C.O.R.E. will be Minion, which is a solver for constraint satisfaction problems.
    1. Because we’ve identified our scheduling problem as a NP-Complete problem, whereby there is no known polynomial-time procedure to find a solution, we need to use an algorithm that can handle a dynamic amount of constraints with proper speed and scale. Especially given our project, we will have many constraints and variables that Minion will be tackling and optimizing.
    2. At a lower level, as with any constraint satisfaction problem, with every finite set of variables with each having its own finite domain, and every set of constraints over those variables, most constraint solvers set an assignment to each variable, such that all constraints are satisfied, meaning they are all true under the assignment.
    3. In other words, an assignment is set to each variable and domain, such that all constraints are true. After this is finished, Minion continues with its specific technique.
    4. Minion’s specific technique is by splitting and propagation.
    5. Splitting is the basic operation of search and propagation simplifies and improves the solution. The splitting search element typically uses depth-first chronological backtracking to select a variable and a value, and then the simplified component contains a queue of constraints, which need to be propagated.
    6. Once the constraint is propagated by Minion, it removes the values from the variable domains and the domain events cause more constraints to be added to the queue. This propagation is iterated until the queue is empty and complete.
    7. At a higher level and from a user’s point of view, the most important features are the set of constraints Minion can reason with, and the types of variables which are supported. It is at this level that we can simply pull together the constraints and variables, run the Minion algorithm, and receive the output.
  + ***Database*** – Our database system will be segregated into several tables and some will be owned by a specified user. The user will be responsible for populating and managing their database table. In the following list, is the title of the database table, the owner of the table, and also the various attributes
    1. **Courses** – This is owned by the Program Administrator. This will hold the table of all the courses offered at UCSC
       - Course Name
       - Course ID
       - Department
       - Description
       - Units
       - Prerequisites
    2. **Classrooms**
       - Building
       - Room Number
       - Class size
       - Preferred Department
    3. **Lecturers –** This is owned by the lecturer. This will be used to provide lecturer’s teaching information for the quarter
       - ID
       - Lecturer Name
       - Department
       - Preferred courses
       - Time Constraints
       - Day Constraints
    4. **Quarter Course List** – This is owned by the Program Manager. This will be used to provide which classes will be offered for the quarter. This will be segregated into the different
       - Course Name
       - Course ID
    5. **Student** – This will be owned by both the student and the Program Administrator who will handle each student’s account. Some attributes can be modified by the student
       - ID
       - Name
       - Major
       - Classes Enrolled
       - Course transcript
    6. **Program Administrators** – This will be owned by a head Program Administrator. This will be the accounts of all the administrators under the head program administrators where they have control over the entire MyCourses system
       - ID
       - Name
    7. **Program Managers** – This will be owned by the Program Administrator. These will be the list of program managers who manage the individual department’s course programs and offerings
       - ID
       - Name
       - Department
  + ***User Interface*** – Each user who logs into the MyCourses page will have a customized user interface
    1. Interface
       - The interface itself will be within the browser utilizing HTML and CSS
       - Should be compatible with all major browsers (Internet Explorer, Firefox, Safari, Chrome, Opera)
    2. Logging in
       - Using their school email (without the @ucsc.edu) and password (requires at least 6 characters and requires numbers) as their credentials
         * Username: [JohnDoe@ucsc.edu](mailto:JohnDoe@ucsc.edu)
         * Password: abcd1234
    3. Home Screen
       - When a user logs into the MyCourses system, the interface will be customized to their specific login credentials and personal information
         * i.e. Student UI will display different information from a PM UI
       - On the left side of the screen will be a series of vertical buttons that will direct users to different areas to the site
         * All users

Home – Home page to the site

Messages – Displays any messages that pertains to the user

* + - * + PA

Course System database – This will be a complete database separated into several tables

PA will be required to populate these tables

Manage Program Manager – PA’s can view the progress and status of the PM’s work

* + 1. Database (Program Manager and Program Administrator)
       - All the database tables (course listing, classrooms, Time slots, Day slots, etc) will be formatted in a excel table-style and each cell in the table will be used via pull down menu
         * The PA will be responsible for populating all the tables based on a previously set attributes in the database
         * The PM will be responsible for organizing and managing the populated tables

PMs can organize and manage the tables for their courses via pull-down menu

* + 1. Lecturer constraints (Lecturer)
       - This will be in a form-style interface where the lecturer will be able to select items
       - Items will be in the following format:
         * Preferred courses – singular column list of courses offered
         * Preferred days and times – Multi-selectable calendar of classes
    2. Student Portal
       - This will be the main area where students will be able to sign up for classes
       - The students will be able to see a course catalogue where it will display all the proper course listings
         * Students can select courses and add them to their shopping cart for later review and submission
       - There will also be a area where students can manage their courses in a table
       - Once the student chooses their courses, based on limitations on how many units they can sign up for, they can submit their courses and it will automatically sign them up
  + ***Backend (server)*** 
    - The default server for this application will run on Ubuntu 10.10; a minimum of 512mb RAM and 1.5GHz clockspeed CPU will be assumed.
* MySQL will be used as the database daemon.
* Lighttpd will be used as the server daemon.
* Lighttpd will use Python FastCGI to interface with the Django framework.

**Performance:**

The scheduling algorithm shall run in no worse than O(n^n) time. All other operations in the system, except starting the server, should take negligible amounts of time. Negligible here is defined as an absolute maximum of 1 second. Note that DNS resolution and HTTP transport are not within the scope of this requirements document; slow interaction with the interface over the internet may be a symptom of a poor connection.

* **Usability –** how the users must be able to interact with the MyCourses system
  + All users will interact with the system through a web browser
  + They will log in using their email credentials and passwords
  + Should be accessible within and outside of the system’s intranet
  + Buttons and links should be simple and easy to read and intuitive for the end user to understand the different aspects of the system
  + An online help page should be available
  + Database tables should be active and data should be actively pushing to the end user’s UI
  + Tables are populated only by the PA
  + Tables are managed by pull down menus
* **Constraints –** The Course Scheduling System will input the following constraints for   
  courses, classrooms and lecturers. It will then process all of the   
  constraints and find an optimal course schedule for a given school.
  1. **Courses** 
     + Number of Students
     + Prerequisites
  2. **Classrooms**
     + Size of classrooms
     + Preferred Department for the course
  3. **Lecturer**
     + Days available
     + Times available
     + Preferred Courses
* **Wish list** 
  + We would like to have an application for smartphones that will   
    give the students an interface where they can access the Course   
    Scheduling System. The items on our wish list will make our project more attractive to the vast majority of people using it; the students. Our hopes are to create an interface that will allow the student to use our software as a tool that they can use throughout their college experience.
    - Search for classes

Students will be able to search through the course schedule to browse potential courses.

* + - Add/Drop/Swap classes

Students will be able to modify their class schedule through their phone.

* + - Student Planner

Our student planner will give the students a clear picture of their history, and future at their University. It will give them information on courses that they need to take for their major, and courses that will fulfill GE’s.

* + - Connect to Google calendar

This option would allow the students to integrate their course schedule with Google’s calendar. The student will only have one calendar that they will need to manage, making it much easier to plan events without interfering with their classes.

* + - Connect via Google Maps

This feature will be especially useful for new students, it will pinpoint the student’s classroom within the campus and allow them to see exactly where their classes are on the map. When integrated with GPS it will give them the quickest route to their classroom.

* **Coding Standards –**
* For this project, we'll be borrowing a significant portion of Python's style guide (PEP 0008). PEP 0008 can be found at: http://www.python.org/dev/peps/pep-0008/
* Note that we will be ignoring the section on Version Bookkeeping. Any Python code should conform to this style guide as well as possible.
* We will also be using Django's design philosophy. Django's design philosophy can be found at: http://docs.djangoproject.com/en/dev/misc/design-philosophies/
* Most notable is the DRY principle - don't repeat yourself whenever possible. Additionally, spaghetti code should be avoided as much as possible.

**Preliminary User Interface:**