Scheduling for Trinity School at Meadow View

Proposal

October 10, 2016

Prepared By: Mary Barthelson
Marissa Brienza
Zachary Nuzzo

Sponsor: Tim Maloney
Chair of Math and Science
Trinity School at Meadow View

Approval from Client

The Client has reviewed and accepted the Proposal for the Scheduling for Trinity School at Meadow View project. The Client agrees to the concept and content of the document. The Client further understands that information contained within this document will be refined as the project progresses.

This proposal describes the problem, scope, requirements, project plan, and proposed solution approaches for the Scheduling for Trinity School at Meadow View project.

Approved by: Timothy M. Malone 10/5/16

Client Signature Date

Table of Contents

1. Introduction

Don't rock the boat is the usual mantra of educational institutes when trying to schedule next semesters classes (Glassey). Typically, these institutes begin with the schedule of the previous semester and make the minimum necessary modifications to accommodate faculty requests. Trinity School at Meadow View is one of these institutes, and uses these practices when scheduling faculty to classrooms and time periods. Trinity School at Meadow View (hereafter referred to as the Client) is a small, private school that provides education to approximately 150 students in grades 7-12. Scheduling is a time intensive process that can take the Client days or even weeks to complete.

1.1 Background

Scheduling of student and teachers to classrooms and periods is a widely studied topic at the university level. Many universities with computer science, engineering, or mathematics departments have researched and developed algorithms that can optimally or feasibly solve their scheduling issues. Scheduling problems, sometimes called the timetabling problem, have been solved using a variety of strategies. Some techniques include: Integer Programming, Heuristic Algorithms, and Constraint Satisfiability Algorithms. These solution strategies and their advantages and drawbacks will be discussed in Section 2.2.

1.2 Problem Statement

The Client does not have access to the powerful computers or industrial/research grade linear program solvers, nor does it have the students and faculty with the requisite skill sets to develop a scheduling tool in house as a university or college does. The Client is seeking a tool that, given input parameters such as section sizes, teacher availability, and classroom constraints, can find a feasible schedule for future school semesters. The Client would like a tool that is easy to use and produces a set of feasible schedules that have teacher, section, and classroom assignments.

1.3 Scope

The scope of this project will be to create a capability for the Client to generate section/teacher/classroom/period schedules. The Client has stated that there are no metrics that they are seeking to optimize through the development of this scheduling tool. Due to this fact, the project team's approach will not require an optimal solution. The project team may choose to develop some metric for measuring fitness such as distance between two assigned classrooms for a specific section. The lack of the requirement of an optimal solution opens the door to a variety of solution approaches that will all be investigated and researched at length. The project team will choose a solution method, submit it for Client concurrence, and work to implement the solution by 14 October 2016.

The tool will not be a web-enabled application, nor will it interact with a separate backend (separate installations of the tool will not have access to the same information, unless the tool is a copy of a version with the data in it). The tool will be delivered to the Client via a CD-ROM or

DVD. The Client will be provided with a User Manual for the tool. The User Manual will include a high-level overview of the algorithms used, a guide on how to use the tool, and a troubleshooting section.

1.4 Document Motivation

This document serves as an overview of the Trinity School Scheduler (TSS) that the project team is currently developing to automate and streamline the scheduling process for Trinity School at Meadow View. This document will describe the requirements and objectives of the TSS, the technical approach for TSS, and will provide an overview of the management of the project.

2. Requirements and Objectives

The project team has examined the Client's stated requirements, needs, and desired outcome, and has developed the requirements below:

2.1 Nonfunctional Requirements

The nonfunctional solution requirements include:

2.1.1 The tool shall be supported by Windows OS.
2.1.2 The tool shall be available via a CD/DVD.
2.1.3 The tool shall be saved on a local Trinity School at Meadow View computer.
2.1.4 The tool shall run on software currently available to the client.
2.1.5 The tool shall have a User Interface that allows the user to add/edit all input parameters, but will not allow interaction with the scheduling algorithm.
2.1.6 The tool shall provide a User Manual outlining how to use the tool and troubleshooting best practices.

2.2 Functionality Requirements

The solutions provided to the Client will be developed with the following functional requirements in mind:

2.2.1 The tool shall output at least two feasible schedules.
2.2.2 The tool shall produce output in a spreadsheet/table format.
2.2.3 The tool shall allow the user to export and save feasible schedules.
2.2.4 The tool shall provide an option for the user to request alternate schedules.
2.2.5 The tool shall prompt the user for the following input parameters on first use and save the results: classrooms, corresponding room size for each classroom, course restrictions for each classroom, teacher names, hours of availability for each teacher, corresponding subjects taught by each teacher, and the size of student sections.
2.2.6 The tool shall allow the user to add/edit all input parameters directly in the tool through the use of tables.
2.2.7 The tool shall allow stored fields to be reset.
2.2.8 The tool shall allow the user the option to edit stored fields.
2.2.9 The tool shall notify the user if there is an error or additional information is required with the inputs.
2.2.10 The tool shall notify the user if the solution is infeasible and modification to inputs is needed.

2.3 Performance Requirements

The performance of the tool shall meet the following at a minimum:

2.3.1 The tool shall complete runs in 60 minutes.

2.4 Analysis of Alternative Approaches

Input from the Client is imperative when determining what solution approach will work best for a project. For this scheduling project, the Client specified the need for several, feasible schedules to be produced. The faculty will then examine these schedules, and one will be chosen that best fits the needs of Trinity School at Meadow View. This problem is a special case of timetabling problems, and has been studied by many researchers and academics due to its practicality. The class-scheduling problem has been approached with optimization and heuristic techniques. Section 3.2.1 will discuss some of these techniques and the associated benefits and drawbacks.

2.5 Success Criteria

The project team will meet or exceed all of the requirements listed in Section 2.1-2.3. If the project team feels that a requirement may not be met during the course of the project, it will be brought to the attention of the Client immediately and a resolution will be agreed upon.

3. Technical Approach

This TSS project will follow a Systems Engineering V-model approach. The left side of the V-model represents the problem definition and requirements construction phase, and the right side of the model represents the integration of the tool into the Client environment. It is important to notice that some of the phases occur simultaneously. The phases for this project are:

- Requirements Development
- Literature Review & Data Collection
- Tool & Solution Development
- Test & Evaluation
- Final Delivery

Figure 1 shows the customized V-Model for this project.

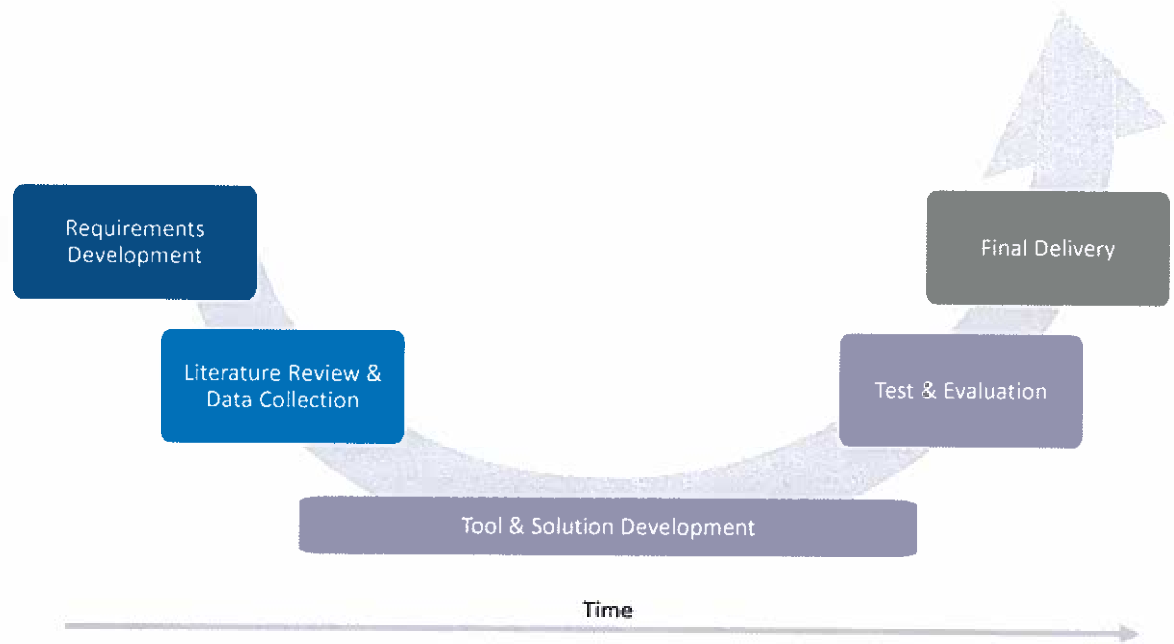


Figure 1: Scheduling for Trinity School at Meadow View V-Model

3.1 Requirements Development

The requirements developed for the TSS project were discussed is Section 2 of this Proposal.

3.2 Literature Review and Data Collection

It was imperative for the project team to conduct a thorough literature review before developing the tool and solution for the TSS project. Exploring all relative literature allowed the project team to develop an educated and knowledgeable approach to solve the current problem. Data was collected prior to the development of the tool to assist in constructing requirements and creating a strategy for the TSS project.

3.2.1 Literature Analysis

Integer Programming

Integer Programming (IP) is a subset of Linear Programming problems that has the additional constraint that some or all the variables have to be integers. This approach is one of the most popular in solving the class-scheduling problem. IP will find the optimal schedule given an objective function, a set (or sets) of variables, and a variety of constraints. It is important to note that as the problem gets more complex it becomes difficult to solve for optimality, and the program will run for a nondeterministic polynomial amount of time. While IP is very precise, the complexity and size of the problem can make this strategy inefficient in finding a solution in a reasonable amount of time. This may not be an issue for the Client's specific problem. An IP requires a fitness function (objective function) and the Client has stated there is not metric to optimize; therefore, a metric will have to be developed to implement this solution.

Heuristics

Heuristic algorithms search the solution space directly and do not require a formal mathematical formulation. These algorithms incrementally alter the solution to move towards a more feasible solution that satisfies the constraints. Although a heuristic cannot guarantee optimality, these algorithms are flexible and can be adjusted to solve many problem types. Heuristic algorithms also allow the developer to choose stopping criteria meaning run time will not be an issue with this strategy. The challenge with using heuristics is selecting the best algorithm for the problem, and then implementing the algorithm to solve the specific problem. For the Client's problem, a metric will have to be developed to compare different solutions. However, if this method is used, importance will not placed on the objective function, and instead on meeting all constraints ad exploring feasible solution neighbors. Using a heuristic algorithm will allow the project team to provide the Client with a set of feasible schedules.

Constraint Satisfiability Algorithms

Constraint Satisfaction Algorithms (CSA) are a subset of ability algorithms that solve a problem given a set of variables, their possible values, and a set of constraints restricting the variable values. Many class scheduling problems have used this approach to find an initial feasible schedule and then use a heuristic algorithm (i.e. simulated annealing) to improve the quality of the schedule. CSAs cannot guarantee optimality but the sole objective of a CSA is too assign values to all the variables in such a way that all constraints are satisfied. Much like the heuristic strategy, CSAs will produce a set of feasible schedules. These algorithms are flexible and typically have reasonable run times, but implementation of a CSA is more involved that using an IP. Accuracy of this solution method cannot be measured like when using an IP due the approach not requiring a fitness function.

3.2.2 Data Sources

The Client has provided the Project team with the following data:

- Classroom capacity
- Classroom content

- Teacher status (Full time or part time)
- Teacher Availability
- Section size
- Curriculum requirements

Mr. Tim Maloney, the point of contact from Trinity School at Meadow View, will provide data required to build the tool. The Client has, at this point, provided initial values for parameters and constraints necessary to begin development of the tool. Future requests for data will be directed to Tim Maloney. If a gap in the data exists, the project team will select a course of action to mitigate that gap. The project team will also research a portion of the vast body of academic research and publications for this scheduling problem.

3.3 Tool and Solution Development

3.3.1 Tool Development

The TSS will be developed either in a Microsoft (MS) product or via an open source software such as Java. The preferred solution is using an MS Access database, with forms and functionality enabled via Visual Basic for Applications. Members of the project team have had extensive experience developing tools in MS Access with additional VBA functionalities.

The TSS will be designed so that a user from a non-technical background can utilize the tool and generate schedules. In addition to a user friendly design, the Project team will also develop a User's Manual to instruct users how to use TSS.

3.3.2 Optimization/Algorithm Integration

The Optimization/Algorithm will be developed separately from the tool in either an MS product or using open source software. Based on final decisions on the method to solve the problem (either optimally or using a heuristic solution), some form of integration between the algorithm and TSS will be required.

3.4 Test & Evaluation

TSS will require extensive testing. The following validations will be needed:

- TSS generates a schedule that is at least feasible
- If TSS solves with a heuristic, a number of tests will be performed to quantify how far from optimal
- Testing will be arranged with the Client near delivery to collect any feedback on User Interface designs
- Testers who were not involved with the development of the tool and were merely given a set of inputs and the User's Manual will test the tool.

3.5 Final Delivery

Table 1 lists the deliverables that will be produced by the end of this project. The table identifies each deliverable and its delivery method.

Deliverable	Delivery Method
Website	Electronic
Final Paper	Electronic
Final Presentation	Electronic
Final Tool	CD
Tool User Manual	CD
Necessary Software	CD

Table 1: List of Deliverables

4. Project Management Approach

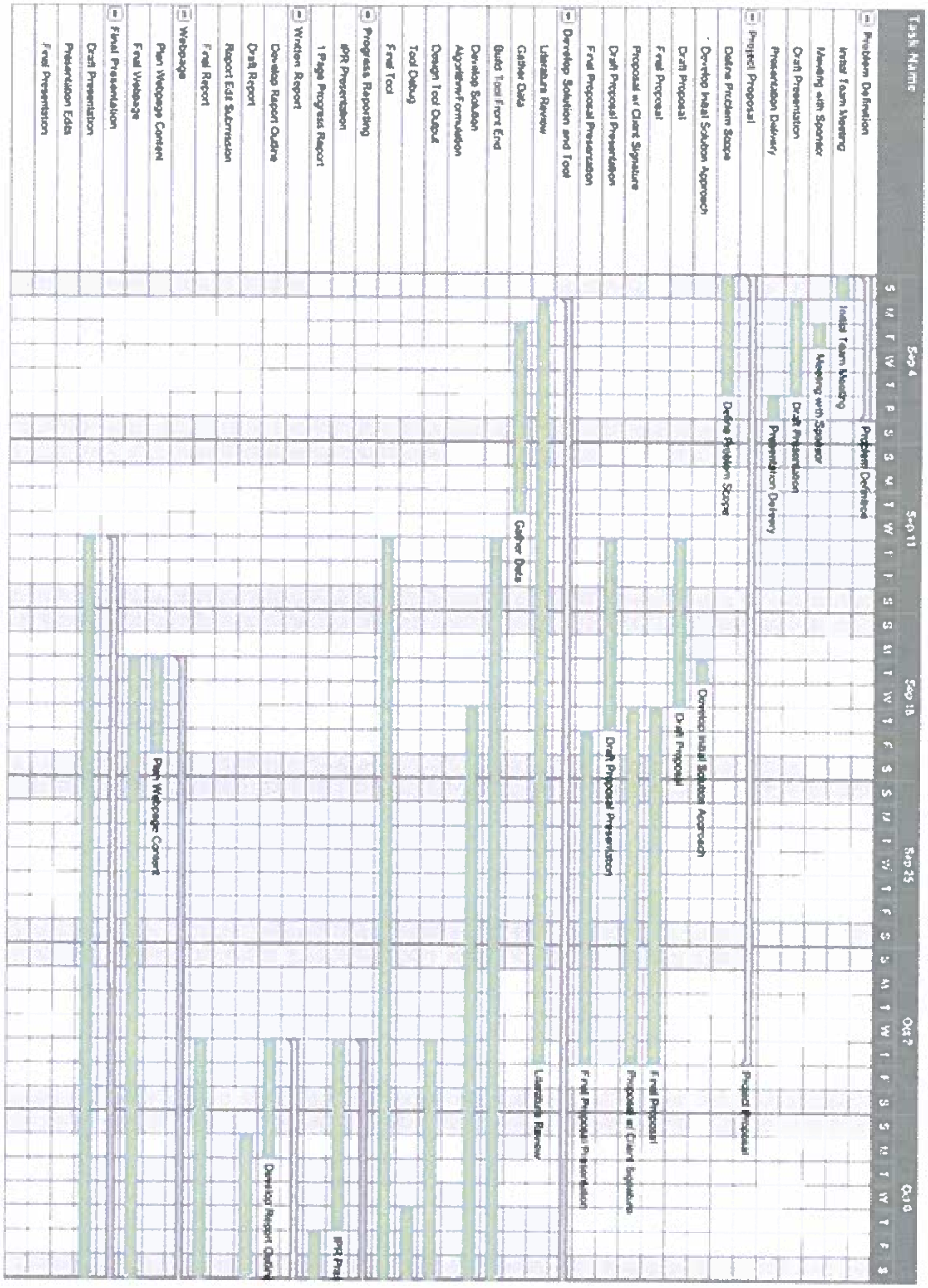
The project team will engage in project management best practices in order to ensure technical performance meets the requirements to build this tool successfully and in a timely manner.

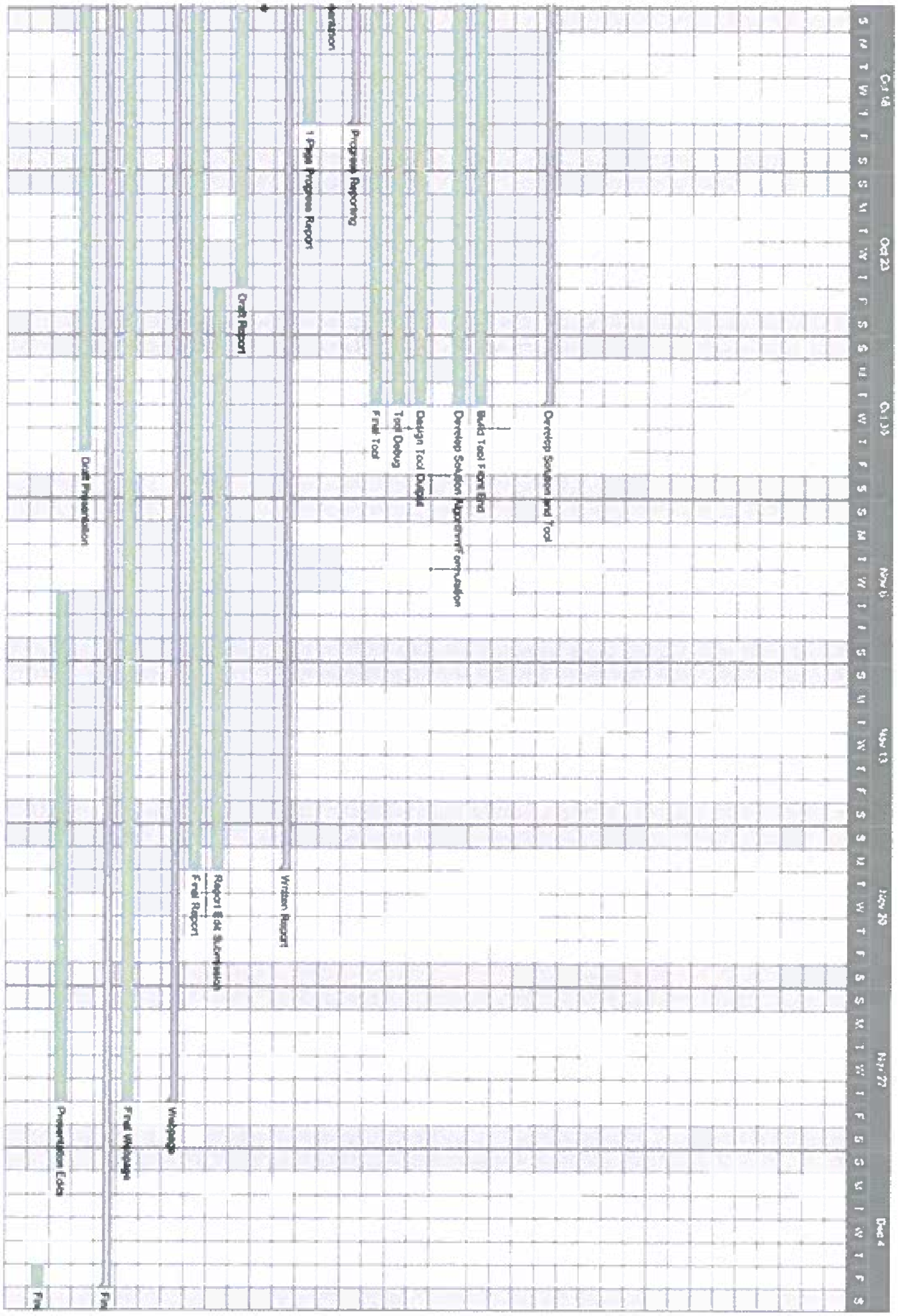
4.1 Resources

The team is made up of two Operations Research students and one Systems Engineering student. These students are full time employees and part time students. The team will use all software discussed in previous sections on their own personal computers.

4.2 Schedule

The schedule is broken down in to seven main tasks: Problem Definition, Project Proposal, Develop Solution and Tool, Progress Reporting, Written Report, Webpage, and Final Presentation. The schedule can be found on the next pages.


Task Name
Problem Definition
Initial Team Meeting
Meeting with Sponsor
Draft Presentation
Presentation Delivery
Project Proposal
Define Problem Scope
Develop Initial Solution Approach
Draft Proposal
Final Proposal
Proposal w/ Client Signature
Draft Proposal Presentation
Final Proposal Presentation
Develop Solution and Tool
Literature Review
Gather Data
Build Tool Front End
Develop Solution Algorithm Formulation
Design Tool Output
Tool Debug
Final Tool
Progress Reporting
IPR Presentation
1 Page Progress Report
Written Report
Develop Report Outline
Draft Report
Report Edit Submission
Final Report
Webpage
Plan Webpage Content
Final Webpage
Final Presentation
Draft Presentation
Presentation Edits
Final Presentation
Sep 4
Sep 11
Sep 18
Sep 25
Oct 2
Oct 9


Oct 16
Oct 23
Oct 30
Nov 6
Nov 13
Nov 20
Nov 27
Dec 4
Progress Reporting
1 Page Progress Report
Draft Report
Final Tool
Tool Debug
Build Tool Front End
Develop Solution and Tool
Draft Presentation
Written Report
Final Report
Webpage
Final Webpage

4.3 Key Milestones

Table 2 is a list of key milestones and the dates they are due.

Milestone	Due
Problem Definition Presentation	09/08/2016
Proposal w/ Client Signature	10/06/2016
In Progress Review Presentation	10/13/2016
Final Report	11/19/2016
Final Tool	11/19/2016
Final Webpage	12/1/2016
Final Presentation	12/9/2016

Table 2: Key Milestones

5. References

Aycan, Esra. (2008). Solving the Course Scheduling Problem by Constraint Programming and Simulated Annealing. Izmir Institutes of Technology.

Badri, Masood A. (1996). Two-Stage Multiobjective Scheduling Model for [Faculty- Course-Time] Assignments. European Journal of Operations Research.

Blanco, Juan Jose; Khatib, Lina. Course Scheduling as a Constraint Satisfaction Problem. Florida Institue of Technology.

Dinkel, John J.; Mote, John; Venkataramanan, M. A. (1989). An Efficient Decision Support System for Academic Course Scheduling. INFORMS

Glassey, C. Roger; Mizrach, Michael; Laetsch, W. M. (1986). A Decision Support System for Assigning Classes to Rooms. INFORMS.

Gunawan, Aldy; Ng, Kien Ming. (2011). Solving the Teacher Assignment Problem by Two Metaheuristics. International Journal of Information and Management Science.

Karmaker, D.; Islam, M. R.; Rahman, H.; Bhowmik, A.; Imteaj, M. N. A Heuristic Approach to Course Scheduling Problem. American International University- Bangladesh.

Martin, Clarence H. (2004). Ohio University's College of Business Uses Integer Programming to Schedule Classes. INFORMS.

Ramirez, Eugene Ruben. (2010). Using Genetic Algorithms to Solve High School Course Timetabling Problems. San Diego State University.