**Software Requirements Specification:**

Apply Yourself to the Field Ministry – Assignment Planner (AYFM-AP)

1-INTRODUCTION

This document describes the background, user goals, and requirements of the software solution.

1.1-*Purpose*

This document is intended for the software developer. This is a hobby project where there is expected to be only one developer. The goal is to practice with the Software Engineering life cycle activities and various tools (Java, Python, MySQL, Git). The need for a guided design has been observed by the single developer that has already started working on the software.

1.2-*Scope*

The software product is intended to automate the processing and assignment of tasks for the section of a Jehovah’s Witness’ meeting named *Apply Yourself to the Field Ministry* (AYFM). The software product will be named “AYFM *Assignment Planner*.” The most important feature of the product is the automatic generation of *suggestions*. Therefore, the task assignments must be reviewed by the user for approval and finalization.

1.3-*Definitions, Acronyms, Abbreviations*

User

Individual responsible for organizing *Apply Yourself to the Field Ministry* section (an obsolete but useful term for lack of a better one is *School Superintendent*)

Student

An enrolled person, who satisfies the requirements to participate

Assignee

Student assigned to participate in an assignment

Publisher

Student assigned responsibility for an assignment

Householder

Student assigned to role play as house holder

AYFM

Apply Yourself to the Field Ministry

AYFM-AP (AP)

Apply Yourself to the Field Ministry Assignment Planner

SRS

Software Requirement Specification

assgn.

Assignment

hhold.

Householder

1.4-*References*

The documents currently used by the School Superintendent will be considered as input and output formats. These are included as references in the ‘user document templates’ folder.

Three important documents that give specific instructions on AYFM are:

* Benefit from Theocratic Ministry School Education (‘Benefit’)
* Life and Ministry Meeting Workbook
* S-38-S

Benefit describes the goals of AYFM (formerly Theocratic Ministry School) and the lessons with exercises that students will be assigned.

The Workbook is a monthly publication available at jw.org. This document describes all sections of the meeting including AYFM. This is where details are given about 3 (Initial Call, Return Visit, Bible Study) of the 4 assignments scheduled weekly. The Bible Reading assignment is under the section “Treasures from God’s Word” but responsibility for assigning it is given to the School Superintendent.

The last document is more difficult to access. S-38-S is in Spanish as the last letter in the name suggests. An English version exists but it has not been provided. The document details the guidelines for all sections of the Life and Ministry Meeting and clarifies what is expected and how to handle weeks with special activities (Overseer visit, Convention, or cancelation for example). S-38-S has been made available as pictures of the original document in folder S-38-S.

There are many existing tools that facilitate the scheduling of assignments. TheocBase is one of these. There is interest in exploring this application for understanding and identifying use cases. It can be found at theocbase.net along with its supporting documentation. Note that TheocBase manages all sections of midweek meeting and weekend meeting as well.

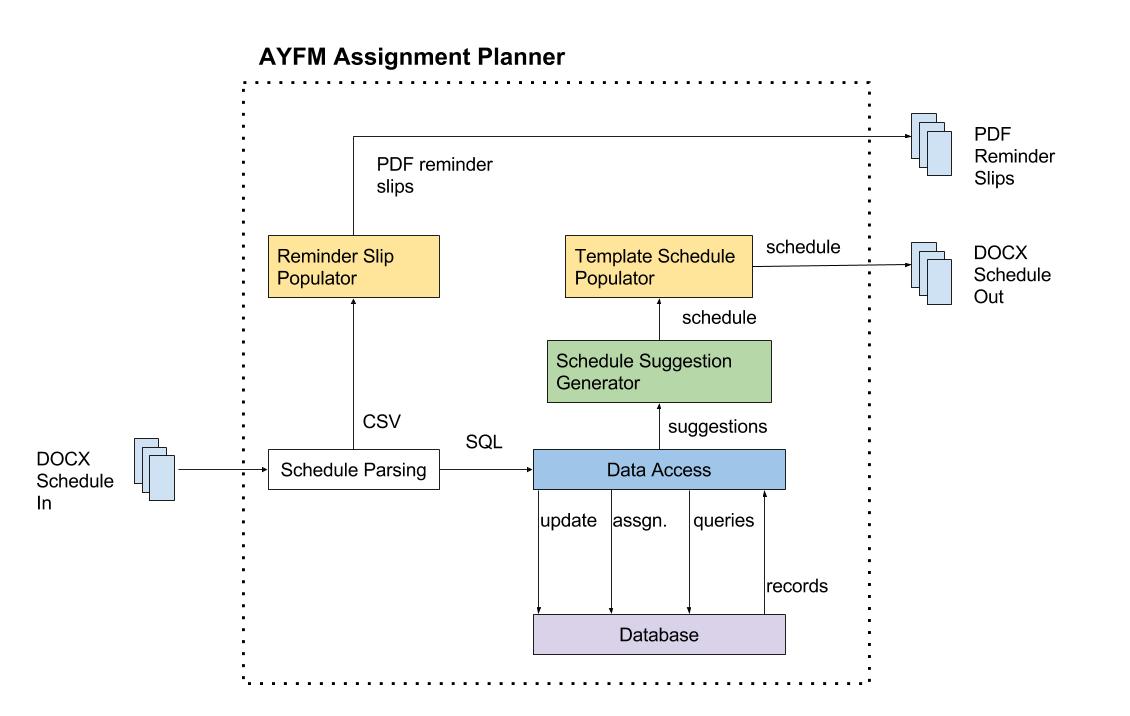
1.5-*Overview*

Section 2 is a description of the architecture and subsystems that make up AP. A block diagram outlines the subsystems. In that section, there is also a summary of the product, user background, and domain background. Section 3 gives a detailed description of the specific categories of software requirements.

2-OVERALL DESCRIPTION

AYFM-AP can be described as a feedback driven system. The schedules that AP produces can be used as input to AP. Some modification is expected to be made on the output schedules before being used as input.

2.1-*Product perspective*



Block Diagram

System Interfaces

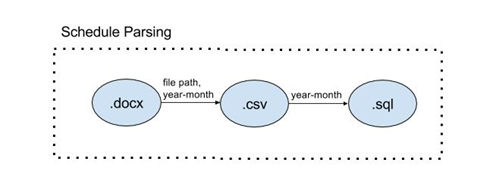
Above is a block diagram that outlines the subsystems. Note that the schedules that are output can be used as input. (If it seems like that diagram came out of nowhere, you’re right. It is premature to develop it here but that’s too bad.)

The ‘Reminder Slip Populator’ will be in C# to build some experience with that programing language, although writing it in Java makes the most sense. Presumably there is some tool available that will allow the C# program to be run from the overall application which will be in Java. Ideally the slips should be populated directly from data that has been committed to the database rather than from a csv file. It should be noted that this is generally undesirable because the information on the slips may not be inconsistent with the database. This might be an example of what are known as data silos.

The ‘Template Schedule Populator’ will likely be made in C# as well, although it is known how to easily accomplish the task with Python. The schedule passed from the ‘Suggestion Generator’ to the ‘Template Schedule Populator’ will be a temporary csv file to accommodate the communication between the two subsystems.

Schedule parsing will be done in python. Using Jython, the scripts will be runnable through the main Java application. This subsystem has the architectural design paradigm of a transformational system because the data is converted (“transformed”) several times into different formats. A new file is created each time the data is transformed and the original is left intact. Below is a Transformational Graph that depicts the data needed and its transformations. Each edge represents a process and the description on the edge represents parameters while the nodes represent the format state.

Transformational Graph



The ‘Schedule Suggestion Generator’ will use assignment suggestions generated by the database and will also request the last participation by a given student. This is written in Java. This subsystem belongs to the business layer.

‘Data Access’ is the data access layer component. This is written in Java. Here are the Data Access Objects and Services based on the requirements that will be allocated later. The ‘Data Access’ subsystem is set up to connect with a MySQL database specifically. A Driver Manager would allow a vendor neutral database access but this is not being used. The Data Access Object (DAO) design pattern describes a class whose responsibility is to encapsulate a class’s ORM (Object-Relational Mapping) code. There is a DAO defined for every persistent entity class. Data-Tier Services may not be needed. Since their primary purpose is to make transactions. The operation needed is simple so the service layer may not be necessary and direct access to the DAO is allowed.

The current application has a MySQL Database in use. The use of mysqlite is highly recommended because the requirements are for a single user and the database is known to be quite simple at this stage of development. (Views, triggers and other such objects are not currently implemented but may be in the future.)

Site adaption requirements

The goal would be to make this a web app where the only needed software on the client side should be a browser. Presently there are already four tools that need to be installed:

* Java 1.8
* python-docx (python module installation)
* unidecode (python module installation)
* MySQL

The C# processes might require additional libraries to be installed for operation to be integrated with the Java application.

2.2-*Product Functions*

2.3-*User characteristics*

2.4-*Constraints*

2.5-*Assumptions and Dependencies*

The factors that affect the requirements should go here but I don’t know what that means.

3-SPECIFIC REQUIREMENTS

Functional Requirements

Performance Requirements

4-APPENDIXES

4.1-*Overall Software Requirements Specification*

1. AYFM-AP must allow the user to input schedules that have been previously created.
   1. AYFM-AP will use a formatted table in a MS-DOCX (or RTF) document for the user to fill in manually.
   2. AYFM-AP will parse the document to extract assignment information and detect new students.
2. AYFM-AP must allow the user to make corrections to existing schedules.
   1. AYFM-AP will not write back to the MS-DOCX document any changes made in the database. (The user will need to do so manually to prevent lost information.)
3. AYFM-AP must allow the user to update completion information about assignments.
   1. AYFM-AP will store whether the assignment was completed.
   2. AYFM-AP will store whether the lesson was passed.
4. AYFM-AP must allow the user to create student profiles.
   1. AYFM-AP will suggest student profiles to be created when an inserted schedule includes a name not in the database. (see R1.2)
   2. AYFM-AP must allow the user to insert new profiles into the database.
5. AYFM-AP must allow the user to edit student profiles.
   1. AYFM-AP must allow the user to approve students for different assignment types.
   2. AYFM-AP must allow the user to approve students for different assignment roles.
   3. AYFM-AP must allow students to be deactivated so that they will not be recommended for assignments.
6. AYFM-AP must generate participation summaries for each profile.
   1. AYFM-AP will have *assignment-type participation summaries* which will include
      * Name
      * assignments type
      * number of assignments (completed?)
      * number of assignments in section A
      * number of assignments in section B
      * number of householder assignments
      * number of householder assignments in section A
      * number of householder assignments in section B.
   2. AYFM-AP *total participation summaries* will aggregate the “assignment-type participation summaries” and will include
      * Name
      * total assignments
      * total section A assignments
      * total section B assignments
      * total householder assignments
      * total section A householder assignments
      * total section B householder assignments
      * most recent assignment date
      * next lesson to be assigned (?).
7. AYFM-AP must generate suggestions for assignments based on history for fairness.
   1. AYFM-AP will suggest the student who has given the assignment (for a given type and role; possibly the section as well) the longest ago.
   2. AYFM-AP will also avoid suggesting the same student across multiple assignments for an upcoming schedule.
   3. AYFM-AP will suggest the same student across multiple assignments only if there is at least a minimum lapsed time between the assignments.
   4. AYFM-AP will allow the user to edit the suggested schedule.
8. AYFM-AP must suggest lessons to be assigned next.
   1. AYFM-AP will suggest the lesson that the user has entered to be assigned next.
   2. AYFM-AP will not suggest a lesson if the user has not provided the next lesson.
9. AYFM-AP must generate the S-89-S form for each assignment.
   1. AYFM-AP will use the schedule approved by the user to fill in the form.
   2. AYFM-AP will use the suggested lessons or the lessons assigned by the user if there are no suggestions.
10. AYFM-AP must handle unexpected input values by informing the user.
    1. AYFM-AP may ignore the values.
    2. AYFM-AP may correct the values without write back. (user is expected to make corrections in the input files)
    3. AYFM-AP may abort the operation.