***SOFTWARE PROJECT MANAGEMENT PLAN (SPMP)***

1. **Introduction**

This project is creating a database center to connect all components of the organization together. This will increase productivity of the current state of the company. Some components of the organization might need to be removed or altered to work for added functionality. We expect to add two new functionalities to the project which may result in deliverables being altered or removed completely. The first milestone will be to have the project approved. The second will be getting the two added functionalities approved. The third would be finalizing all documentation with the two functionalities to be added. The fourth will be completing objective 1, 2, 3, and 4. The fifth will be to complete database center. The sixth will be to test the datacenter with all components of the organization. The final milestone will be to launch the project.

* 1. **Problem statement**

The primary goal of this project is to make the chocAn company more efficient in their day to day processes. First off the providers currently handle all billing and paperwork manually, our aim is to delegate all “busy” work to a datacenter in order to make the providers job easier. In the current state of the company the provider takes to long with patients verifying membership as well as logging all necessary information manually which in turn reduces the likelihood that a patient will reschedule an appointment, by making it easier for the provider we hope to increase patient adherence in the long term.

**1.2 Project scope**

**1.2.1 Inclusions**

* Input: Patient Information sent from terminal to data center for verification
* Output: Verified patient information sent back to terminal or Non active member status can be sent back aswell
* Input: Patient membership card number is transmitted to the data center for billing of service & a description of services rendered is sent from terminal to data center for storage into a file.
* Input: Provider directory is stored(as a file) and updated at the datacenter.
* Output: email a provider the provider directory at anytime
* Output: every Friday the member receives a summary of the services rendered that week (stored on disk in a file).
* Output: every Friday the provider receives an email that matches their hand-held records of services rendered for verification (stored on disk in a file).
* Input: electronic funds transfer data is then written to disk (in a file) at the data center for provider compensation
* Output: EFT data is sent to banking computers
* Output: EFT data is transmitted to the manager
* Input: New member addition to the data center system
* Input: new provider addition to the data center system
* Input: update member record
* Input: delete the member record of members that have resigned
* Input: update provider record
* Input: removed provider record
* Output: relevant information for billing members for their monthly fee is sent to acme to bill the members
* Input: Acme updates our records according if the member has paid their monthly fee (this occurs every night at 9pm).
* Process: generate provider reports from various inputs/files

**1.2.2 Exclusions**

* Prover/manager terminal creation
* Banking functions such as depositing money into provider accounts (3rd party will handle that from our data)
* Billing monthly fees and keeping track of accounts that paid and that didn’t pay their monthly fee.
* Website development
* Mobile application development
* Creating Membership cards
* Emailing reports as files

**1.3 Major software functions**

* + - Data Center
      * Automatically emails reports to members, providers, and managers(simulation)
      * Contains information from
        + Provider directory

Holds service codes in a file

* + - * + Acme Financials

Update member status based on monthly fee billing(simulated by keyboard input)

* + - * + EFT

Gets information on who to pay and how much from our program (simulated by file input)

* + - Terminal(all simulated by keyboard input)
      * Connects to data center program to verify member information
      * Sends patient visit information and provider information
      * Checks provider directory for service codes from the datacenter program
      * Manager terminal
        + Sends request to data center to resend weekly reports

**1.4 Performance/Behavior constraints**

* All providers must have functional terminal
* Provider terminal must have card scanner
* All manager terminals must have access to data center
* All members must have card with required information
* All users must have an ID
* Data Storage must be able to contain at least 100 items
* All devices installed must have a device ID
* All services provided must have a service ID, and name
* Terminals must have a fast internet connection
* Providers must be able to contact managers to solve any issues

**1.5 Management and technical constraints**

**1.5.1 Management constraints**

* The project is due before the end of the semester
* Short 2 members for our group due to insufficient class size
* Must complete project with the members assigned from scratch (no outside help).

**1.5.2 Technical constraints**

* Data must be in a data store that is maintainable long term
* Establish some sort of connection with 3rd party software to communicate with data store

**2.0 Project Estimates()**

This section provides cost, effort and time estimates for the projects

**2.1 Historical data used for estimates**

Based on the historical data of the project, it is estimated that the past time that it took to complete a project designing an internet user interface took about 5 hours. There was about an hour of planning, 2 hours of learning what to do, and another 2 hours to actually apply the work that had been learned. Prior to it all, when planning for a project, an estimated time was taken as a starter to every project. On a previous file system team members have created took about 3 to 6 hours and 300 lines of code. When designing a user interface for a tax calculator it took about 3 hours of work and 300 lines of code. The project itself took about 10 hours of work and 1000 lines of code. Based on all this historical data we believe each member can apply 20 lines of code a day with planning.

**2.2 Initial Estimate ()**

Based solely on 1.0 and 2.1 details provide.

**2.2.1 SPMP Completion Estimate**

The estimated effort to complete SPMP is 10 hours.

**2.2.2 Overall project estimate**

**2.2.2.1 Line-of Code Estimate**

Our initial guess would be around 1000 lines of code based on our past homework projects.

* + - 1. **Function Estimate**

Based on the functionality listed in 1.3 I think it should take around anywhere from 2000-3000 lines of code to complete this project, our estimate is based on past group projects we have had in class that also included a file system.

**2.2.2.3 Tasks Estimate**

* Create File system for input/output
* Create a repository to share major changes between members
* Store relevant information into relevant files
* Process file information in order to generate required reports
* Create the Classes to assign each member to work on

**2.2.2.4 Total overall project time estimate in hours of effort**

Total overall project time = 600 hours

**2.3 Estimation techniques applied and results(do this after all sections )**

**2.3.1 Estimation technique 1 – lines of code**

|  |  |
| --- | --- |
| **Functionality** | **Estimated LOC** |
| Data center automatically emails reports to members, providers, and managers(simulation) | 500 |
| Provider directory stores service code in data center files | 400 |
| Acme financials connects to data center to add, remove, or update memberships(simulation) | 300 |
| EFT connects to data center to view service records | 150 |
| Terminal sends member ID(keyboard input) and the data center program returns membership status(output onto screen) | 100 |
| Provider fills out Information about the visit(simulated by keyboard input) and logs service code to data center | 300 |
| Manager sends request to data center to resend weekly reports(simulated by keyboard input and output onto the screen) | 400 |
| Total lines of code | 2,150 |

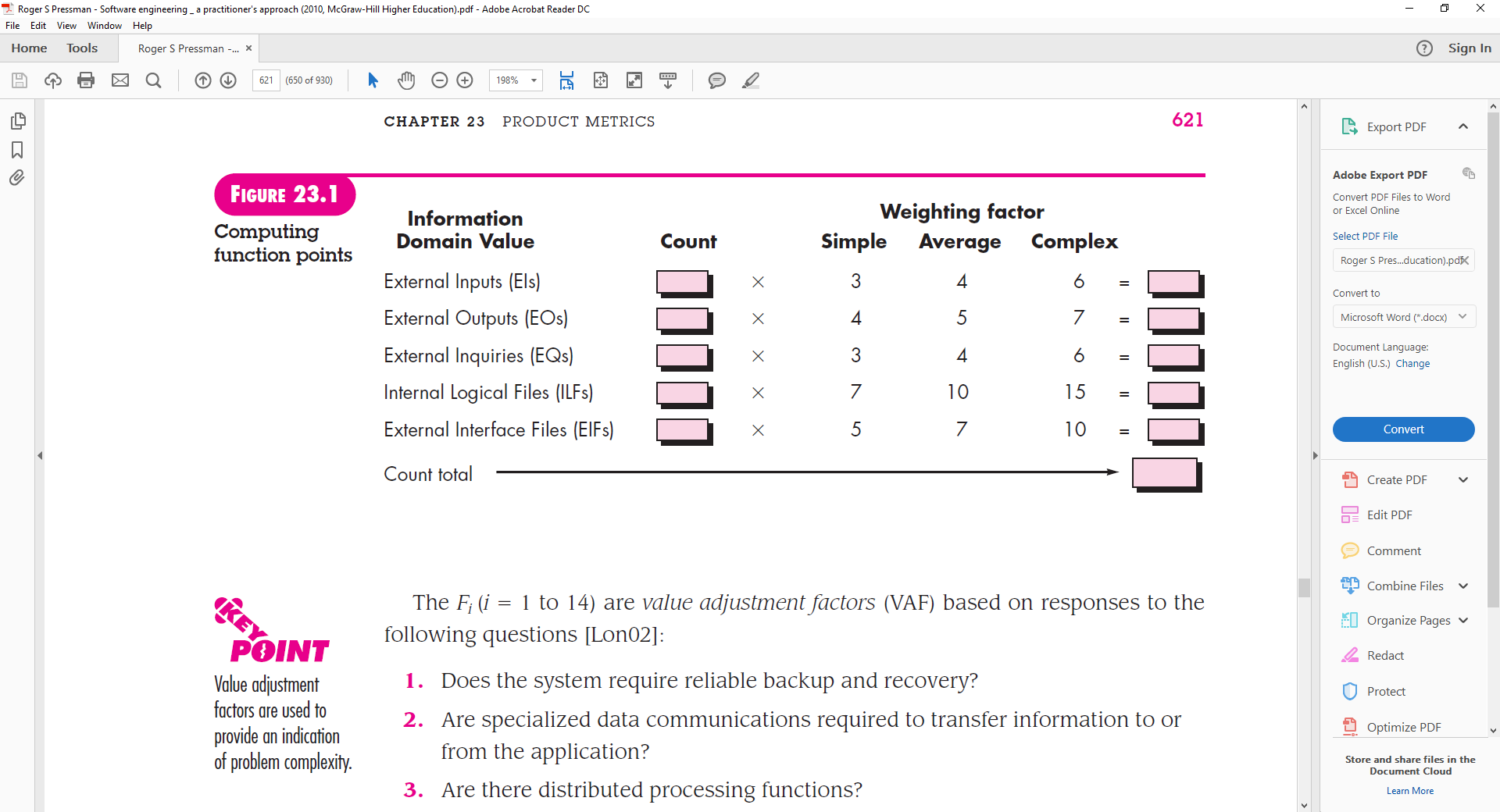
**2.3.2 Estimate for technique 1 – lines of code**

Each person can do about 100 lines of code a day, programming for 5 hours a day.

We estimate 2150 lines of code

2150loc/100locperday= 21.5 days \*5 hours=107.5 hours to complete the project

**2.3.3 Estimation technique 2 – function points**



To compute function points (FP), the following relationship is used:

**FP= count total \* [0.65 +0.01 \* ∑ (*Fi*)]**

**The *Fi* (*i* = 1 to 14) are *value adjustment factors* (VAF) based on responses to the**

**following questions [Lon02]:**

**1.** Does the system require reliable backup and recovery?

1

**2.** Are specialized data communications required to transfer information to or

from the application?

7

**3.** Are there distributed processing functions?

12

**4.** Is performance critical?

4

**5.** Will the system run in an existing, heavily utilized operational environment?

**10**

**6.** Does the system require online data entry?

**9**

**7.** Does the online data entry require the input transaction to be built over multiple

screens or operations?

**4**

**8.** Are the ILFs updated online?

**1**

**9.** Are the inputs, outputs, files, or inquiries complex?

**11**

**10.** Is the internal processing complex?

**8**

**11.** Is the code designed to be reusable?

**14**

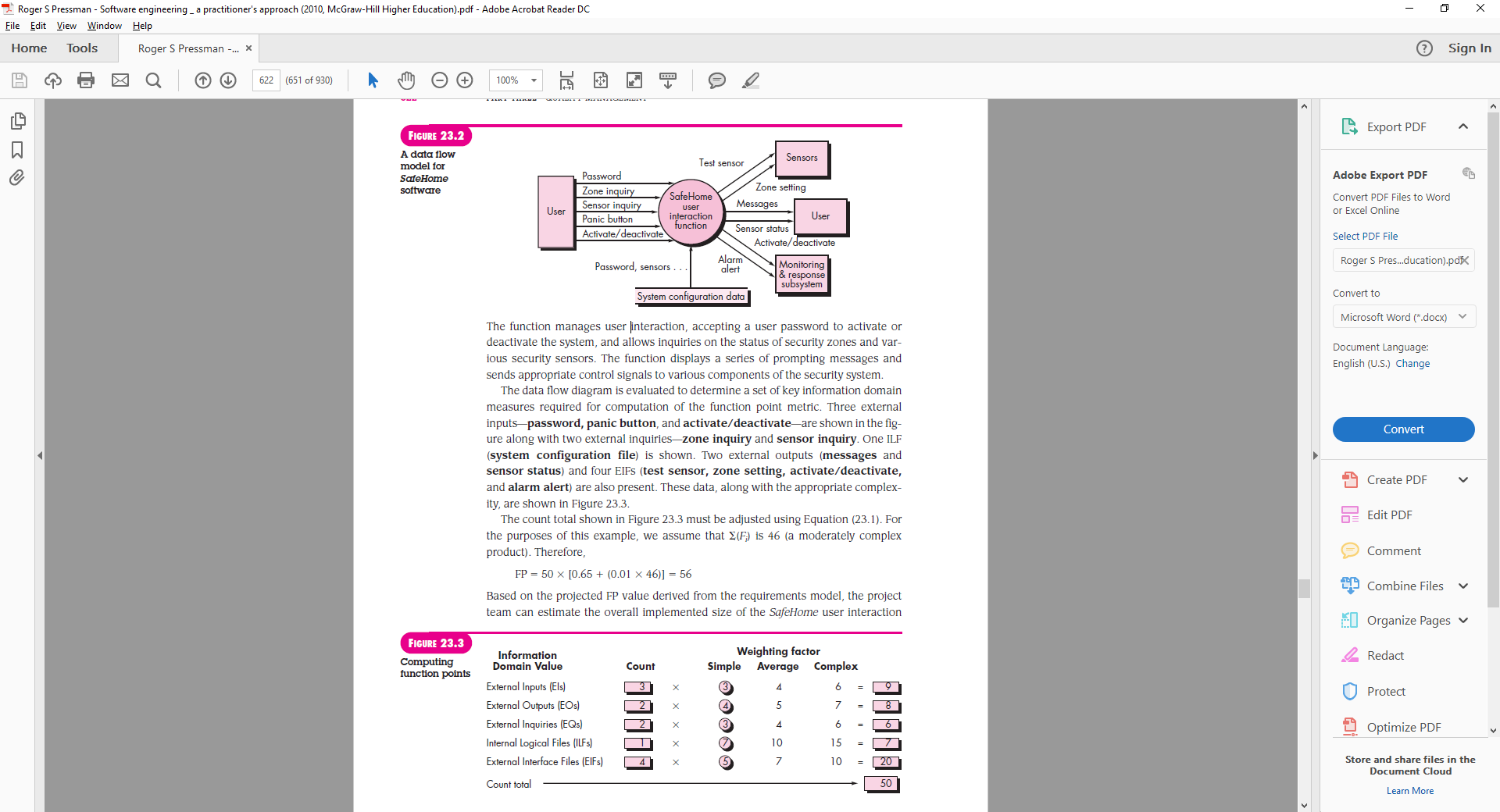
**12.** Are conversion and installation included in the design? 1

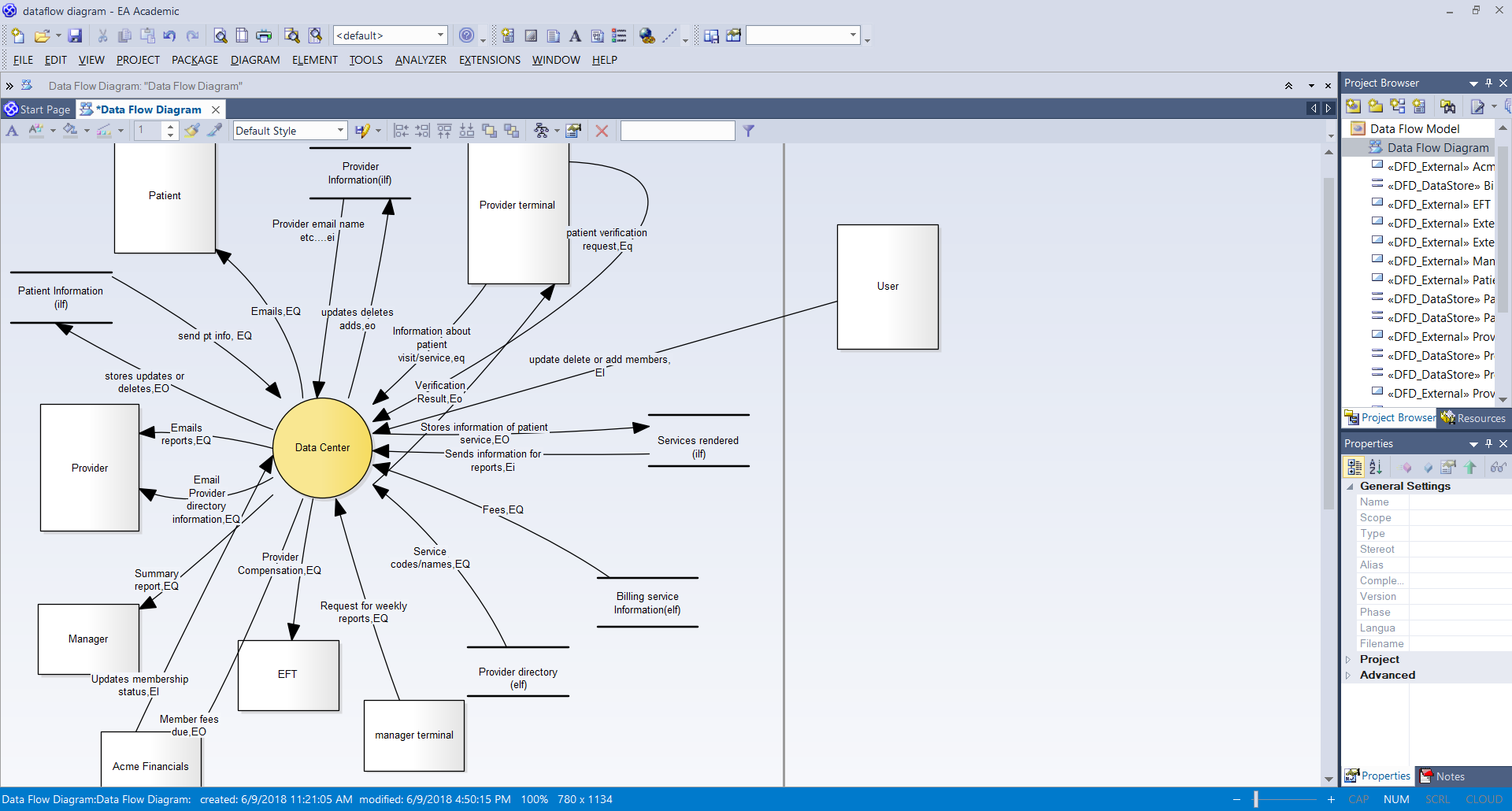
**13.** Is the system designed for multiple installations in different organizations?

1

**14.** Is the application designed to facilitate change and ease of use by the user?

1





Based on data flow model

Ilf= 2\*avg(10)=20

Eq=11\*avg(4)=44

Elf=2\*avg(7)=14

Ei= 4\*avg(4)=16

Eo=5\*average(5)=25

FP = 119 \*[.65+.01\*84]= 177

Loc=Function points\*53 lines of code per function point for c++.



[1] lines of code per function point estimation

<http://www.cs.bsu.edu/homepages/dmz/cs697/langtbl.htm>

Function point calculation equation and tables

Roger S. Pressman - Software Engineering\_ A Practitioner's Approach, 7th Edition

**2.3.4 Estimate for technique 2 – function points**

* + - Function Points= 177
    - Hours per function point for c++ code=8 hours
    - Hours spent=177\*8hours=1416 Hours

Now since the original function points are based on a lot of extra functionality that we won’t be using such as actually emailing the providers but simulating all input/output by files and keyboard input and a computer screen output we divided 1416 hours by a factor of 3 and got 472 hours of work needed

**2.3.5 Estimation technique 3 – process/task**

Tables or equations associated with lines of code estimation technique m are presented. Re-estimate effort based on actual tasks defined in completed SPMP.

|  |  |
| --- | --- |
| **Task** | **Process** |
| Problem statement | Reading through requirements and asking customer questions |
| Complete ERD | Break down project requirements and look at entities and relationships within the organization |
| Project Approval | Gather information for what is required to complete the project and have it approved |
| Software Project Management Plan | Gather information and break down all components of the project and constructing a detailed plan to construct the software and deliver on time and within budget |
| Business Case | A detailed view of all expenses |
| Software Models Requirements | Creating models to assist in visualizing the software |
| Initial Algorithm | Breaking down functionality into pseudo code |
| Test cases | Come up with test plans based off initial algorithm |
| Programing | Take initial algorithm and convert it into code |
| Test software | Run unit tests on software, black box testing, and white box testing |
| Launch software | Turn project in on time |

**2.3.6 Estimate for technique 3 – process/task**

|  |  |
| --- | --- |
| **Task** | **Effort(hours)** |
| Problem statement | 5 |
| Complete ERD | 20 |
| Project Approval | 10 |
| Software Project Management Plan | 30 |
| Business Case | 10 |
| Software Models Requirements | 40 |
| Initial Algorithm | 40 |
| Test cases | 20 |
| Programing | 40 |
| Test software | 20 |
| Launch software | 1 |
| Total Effort | 236 hours |

* 1. **Reconciled Estimate**

We Think it will take about 150-200 hours to complete the project, initially we used a gut feel of what the code would take for each functionality then when we did a function point analysis of the project we realized it would be a little more difficult than we initially thought which was 108 hours so we agreed that anywhere from 150-200 hours would be sufficient.

“ According to one software development company, the average cost to build a function point at the beginning of the software development life cycle (SDLC) in the U.S. is $1,000”(<https://www.umsl.edu/~sauterv/analysis/function_point/UsingFPAS3.html>)

Since we divided by 3 for our function point analysis in the recent section we thought it should also apply to the function points aswell.

177 fp/3=59 function points

1000 dollars per function point =59\*1000=59000 dollars to produce this program.

1 work day = 8 hours of work since our estimate of hours to complete this project was 150-200 we decided to split the difference and use 175hours to estimate, so 175/8 =22 work days of effort.

**2.5 Project Resources**

**People:** Ali Tarraf, Hassan Harajly, Ali Al-mugoter, Mohamed Fawaz

**Harware:** Laptop running windows or mac os

**Software:** Visual Studio, Xcode, Enterprise Architect, Github, Word, Excel, and Power point.

**3.0 Risk Management**

This section discusses project risks and the approach to managing them.

**3.1 Project Risk Table**

Each project risk is described. Name of risk, probability, impact, mitigation and contingency plan

# Risk Analysis

| **Risk** | **Probability of Risk** | **Potential Impact of Risk** | **Risk Mitigation Strategy**  **and Contingent Action(s)** |
| --- | --- | --- | --- |
|  | H/M/L \* | C/H/M/L \* |  |
| Not enough work done due to inability to sync schedules | L | M | Mitigation: have Members agree to a day to work on project every week  Contingency: have more than one meeting time for members who can’t show up |
| Project performance not 100% due to members fasting | H | M | Mitigation: is to spend time together efficiently, by pre-planning what me be discussed and done.  Contingency: have members who can work at night work on the harder components of project while members with work, do the stuff that requires the least time |
| Project scheduling will become hard to deal with due all members having a class currently or starting soon | M | M | Mitigation: try to finish the hardest parts before the semester continues and classes increase in difficulty.  Contingency: keep track of members exams for other classes and see if it can be worked around.  Contingency: change schedule to do more work during down times of semester. |

**3.2 Overview of Risk Mitigation, Monitoring, Management**

We have created a mitigation for every risk that might occur. For each risk the team lead is communicating with the group members it Involves. For the first risk, which is the team lead going on vacation is to train someone to do his job, a team member has been selected and is helping the team lead with his work so he can do it when needed, also another person is being trained just incase anything else happens. Second risk mitigation is to use are time wisely and meeting during days where all members are able to work only on the project. This is communicated through our group chat, everyone says the day they are able to meet and then come in ready to work, if they are not able to work they are then accountable to finish the work at home the team lead picks the day we meet and manages if the there work is done. Third risk has the longest impact on the group & project some members are already taking other classes and or will have a class starting soon so members tell the team lead if they have any exams coming up and they are unable to complete a task. The team lead will then assign someone else to help and then monitor the task and make sure it finishes on time.

**4.0 Project Schedule**

This section presents an overview of project tasks and the output of a project scheduling tool.

**4.1 Project task set**

* + - * + Documentation

Problem statement – (Completed) - A

Complete ERD – (Completed) - B

Project Approval Document – (Completed) - C

Software Project Management Plan - D

Business Case - E

Software Models Requirements - F

* + - * + Initial Algorithm - G
        + Create test cases - H
        + Programing - I

Design user interface

Code

Class for members

Class for providers

Provider directory

Access service class objects

Class for managers

Class for service

Class for Acme Financials

Class for EFT

Class for Data Center

Reports

Files system

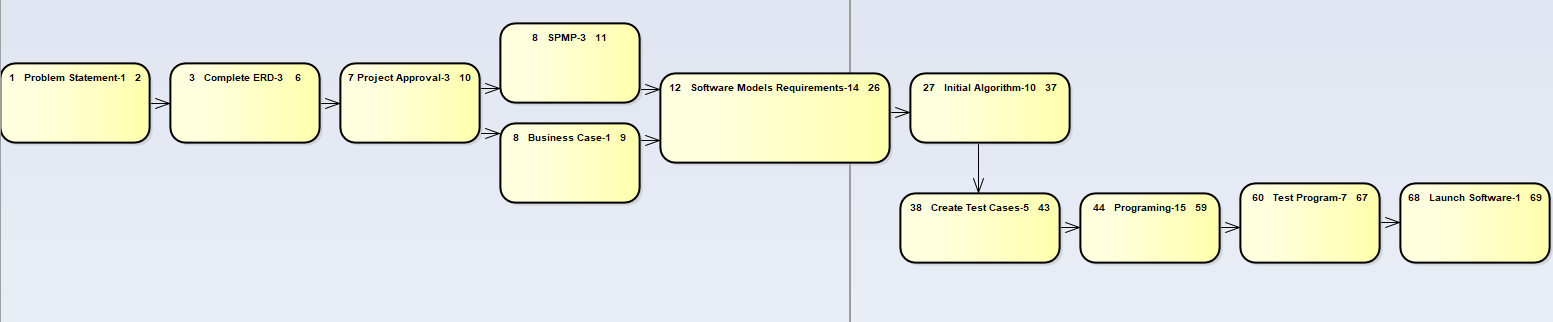
Unit tests

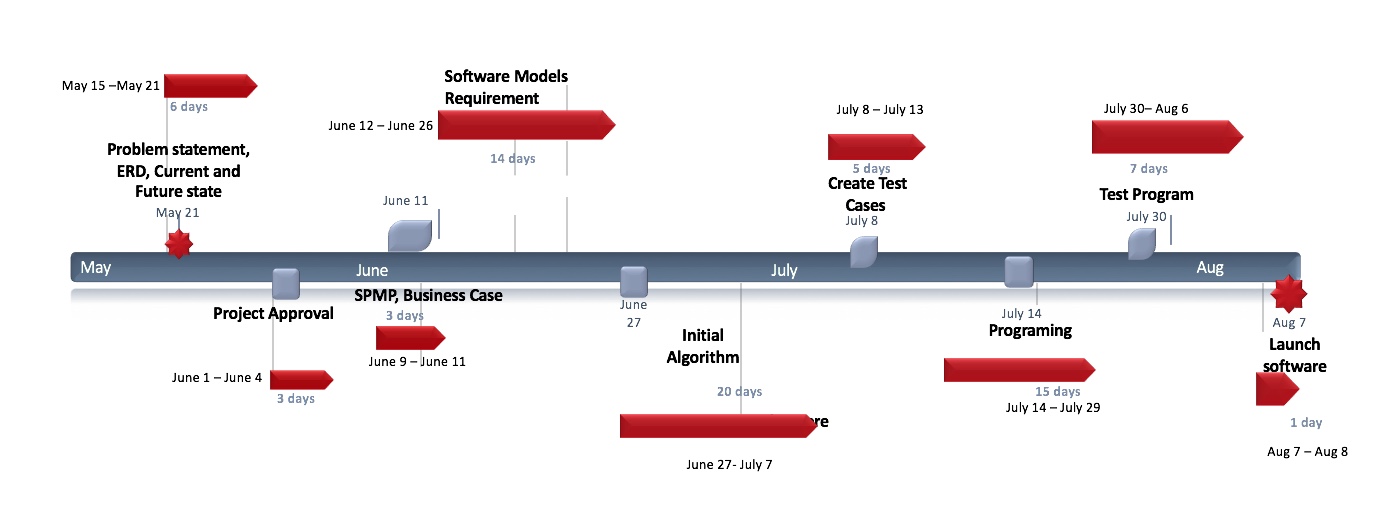
* + - * + Test Program - J

Run unit tests for all possible test cases

* + - * + Launch Software - K

The process model, framework activities and task set that have been selected for the project are presented in this section.

  **4.2 Task network**

**4.3 Timeline chart**

**5.0 Staff Organization**

The manner in which staff are organized and the mechanisms for reporting are noted.

**5.1 Team structure**

The team structure for the project is identified. Roles are defined.

Team Leader: Ali Tarraf – Assign tasks, Documentation, Design user interface, test cases

Team members:

Ali Al-mugoter – Documentation, calculations, models, Files system

Hassan Harajly – Documentation, algorithms, testing, presenting

Mohamed Fawaz – Documentation, models, test cases, presenting

**5.2 Management reporting and communication**

* We will have weekly meet ups and time will be allocated for possible problems that any member may be experiencing
* Team lead will report weekly to project manager
* All members must agree at the end of the weekly meeting that project is going to be completed on time

**6.0 Tracking and Control Mechanisms**

Techniques to be used for project tracking and control are identified.

**6.1 Quality assurance and control**

An overview of SQA activities is provided. This is an outline at this point and will be used to create a detailed plan later in the project.

* Testing
  + Test every function for illegal and legal input by using keyboard/file input
  + Every member that tests their own function must test another partners function and vice versa
* Final review
  + When a function is completed and tested the team member is responsible for reporting completion of the project to the team lead
  + Team leader will determine if a function needs to be peer reviewed
* Risk analysis
  + Weekly discussion of project risk and status of the mitigation strategy
* Check ins
  + During weekly meetups will allocate time for possible problems that any member may be experiencing
  + Team lead will ask all members privately if they’re going to finish their program or need any help
  + All members must agree at the end of the weekly meeting that project is going to be completed on time

**6.2 Change management and control**

An overview of SCM activities is provided on how changes will be handled – communication of a change, how decision made of approval, defer or reject change request.

If a change is to be made the member must read over the appropriate documentation to insure they have a needed change. The team member will consult the team leader and if the team leader agrees will agree to bring it up at the next team meeting or over text(depending on urgency). After deliberation between members leads to an agreement then the change will be reflected in the appropriate documents and project. If a change doesn’t reach an agreement a vote will be held and in the case of a tie the team leader will break the tie.

**6.3 Tools**

What tools will be used to control access and versioning of artifacts.

* Tools
  + GitHub Repository
  + Enterprise architect will be used for all models

**7.0 Appendix**

Supplementary information is provided here.

|  |  |
| --- | --- |
|  | Software Project Management Plan    Appendix |
| Page 1  Page 2  Page 3 Page 4 Page 5  Page 6  Page 7  Page 8  Page 10  Page 14  Page 16  Page 17  Page 18  Page 19  Page 20  Page 21  Page 22  Page 23  Page 24  Page 25  Page 26 | 1. Introduction    1. Problem Statement    2. Project scope       1. Inclusions       2. Exclusions    3. Major software functions    4. Performance/Behavior constraints    5. Management and technical constraints       1. Management constraints       2. Technical constraints 2. Project Estimates    1. Historical data used for estimates    2. Initial Estimate       1. SPMP Completion Estimate       2. Overall project estimate          1. Line of Code Estimate          2. Function Estimate          3. Tasks Estimate          4. Total overall project time    3. Estimation techniques applied       1. Estimation technique 1-lines of code       2. Estimate for technique 1-lines of code       3. Estimation technique 2 – function point       4. Estimate for technique 2-function points       5. Estimation technique 3 – process/task       6. Estimation for technique 3-process/task    4. Reconciled Estimate    5. Project Resources 3. Risk Management    1. Project Risk Table    2. Overview of risk mitigation, monitoring, management 4. Project Schedule    1. Project task set    2. Task network    3. Timeline chart 5. Staff Organization    1. Team structure    2. Management reporting and communication 6. Tracking and Control Mechanisms    1. Quality assurance and control    2. Change management and control    3. Tools   7.0 Appendix |