**Software Project Management Plan**

**Project Management Program**

10/9/14

**Team Members**

Hayden McParlane

Matthew Parra

Anthony Laughlin

Jordan Larson

John Michaels

Document Control

**Change History**

Available on GitHub

**Document Storage**

This document is stored in the project’s SVN repository at: http://company.com/svn/project-name/docs/spmp.doc.

**Document Owner**

Larry Smith is responsible for developing and maintaining this document.**Table of Contents**

[1 Overview 4](#_Toc199052960)

[1.1 Purpose and Scope 4](#_Toc199052961)

[1.2 Goals and Objectives 4](#_Toc199052962)

[1.3 Project Deliverables 5](#_Toc199052963)

[1.4 Assumptions and Constraints 5](#_Toc199052964)

[1.5 Schedule and Budget Summary 6](#_Toc199052965)

[1.6 Success Criteria 6](#_Toc199052966)

[1.7 Definitions 6](#_Toc199052967)

[1.8 Evolution of the Project Plan 6](#_Toc199052968)

[2 Startup Plan 6](#_Toc199052969)

[2.1 Team Organization 6](#_Toc199052970)

[2.2 Project Communications 7](#_Toc199052971)

[2.3 Technical Process 7](#_Toc199052972)

[2.4 Tools 7](#_Toc199052973)

[3 Work Plan 7](#_Toc199052974)

[3.1 Activities and Tasks 7](#_Toc199052975)

[3.2 Release Plan 8](#_Toc199052976)

[3.3 Iteration Plans 8](#_Toc199052977)

[3.4 Budget 8](#_Toc199052978)

[4 Control Plan 8](#_Toc199052979)

[4.1 Monitoring and Control 8](#_Toc199052980)

[4.2 Metrics Collection 8](#_Toc199052981)

[5 Supporting Process Plans 8](#_Toc199052982)

[5.1 Risk Management Plan 8](#_Toc199052983)

[5.2 Configuration Management Plan 8](#_Toc199052984)

[5.3 Verification and Validation Plan 9](#_Toc199052985)

# Overview

## Purpose and Scope

This section provides an executive overview of the project. It explains why the project is being initiated and what can and cannot be expected from project. It may also include any background or contextual information necessary for understanding the project.

The purpose for the project explains the problem or opportunity the project will address. The statement of purpose isn’t a statement of what you are doing (“we plan to automate billing”), but rather why you are doing it (“The purpose of this project is to streamline billing in order to save time, money and resources.”).

Project scope defines the boundaries of the project—what will and won’t be included in the project. Defining project scope helps set expectations regarding what can be expected from the project. The scope definition may also play a role in evaluating requests for changes or new features. Project plans and estimates are based on the scope definition. A request for a change that is outside the current scope of the project can’t be accepted without a change in project scope.

*Partial Example*

A user’s guide and a system installation guide will be provided. One-on-one personalized training isn’t one of the project deliverables.

## Goals and Objectives

Goals and objectives define expected project outcomes. Goals are broad and inspirational. Objectives are narrow and measurable.

Project goals generally related project outcomes to business objectives (reduced cost, increased revenue, improved quality, etc).

A well-worded objective is SMART: Specific, Measurable, Attainable/Achievable, Realistic and Time-bound.

*Partial Example*

Project goals:

1. Establish a mobile porthole for real estate data.
2. Create a mobile porthole that appeals to homebuyers as well as home sellers.

Project objectives:

1. Create a database on homes offered for sale in a local area.
2. Create client software that allows access to the database from mobile devices.
3. Create an interface that home sellers and their agents can use to enter and update home data.
4. Extend mobile access to real estate data in the multiple listing service (MLS).

## Project Deliverables

This section lists the outputs of the project that are delivered to the customer.

*Partial Example*

The following items will be delivered to the customer on or before 1/1/2008:

1. Source code for both the client and server portions of the system.
2. User’s Guide
3. System Administrators Manual
4. Test Plan
5. System test Cases
6. Suite of regression tests
7. Data conversion program for migrating existing data to new database format.

## Assumptions and Constraints

Assumptions are conditions, usually outside the control of the project team, that are taken for granted. Project plans (i.e. estimates) typically depend on certain assumptions being true. Assumptions that turn out to be false, may jeopardize project success. In order to reduce project risk, the project manager may elect to validate certain assumptions as part of the risk management process.

This is also a good place to document verbal promises or assurances given to you.

Constraints are limits or restrictions on freedom. Projects may have technical as well as non-technical constraints. Priorities for schedule and budget can impose non-technical constraints on a project. Restrictions on programming language or delivery platform are examples of technical constraints that limit design and implementation options.

*Partial Example*

Assumptions:

1. The location API works on the test hardware.
2. A senior architect will be assigned to the project during the first 4 weeks.
3. The Unix server and RAID controller can be purchased and delivered by 7/1/2008.
4. Facilities will provide private office space for 3 outside contractors for the duration of the project.

Constraints:

1. The software must run on a Windows Mobile 6 device.
2. The database must be open source.
3. The software must be ready by 1/1/2008.

Note, the following is not a reasonable assumption for inclusion in this section: “We assume that our group has the necessary skills and knowledge needed to complete the project.” This might be something you are taking for granted, but it is not something worth documenting in the project plan. The assumptions you want to list here are those that are outside your control. Once the development team is established, it is their responsibility to possess or develop the skills and knowledge needed to complete the project. If there is a concern that the existing team doesn’t have the skills and knowledge needed to complete the project successfully, add it as a risk and develop a plan for mitigating the risk.

## Schedule and Budget Summary

The schedule summary shows start and end dates for high-level activities ending in major milestones or deliverables. Milestones are major events in the project life cycle that are used to measure progress.

A Gantt chart is an excellent tool for visualizing the start and stop dates of major scheduled activities.

The budget summary shows total project cost, possibly broken down into separate categories for such things as salaries, equipment, travel, overhead, etc.

## Success Criteria

Success criteria spell out what has to happen before the project can be considered a success. Having explicit success criteria serve two purposes. First, during a project success criteria help to focus attention on what is important. Second, at the conclusion of a project (project closure) success criteria are used to assess whether or not the goals and objectives of the project have been achieved.

To be effective in both of these endeavors, success criteria must be defined in a way that is both quantifiable and verifiable.

For more advise on how to define the success criteria for a project, I recommend: *Success Criteria Breed Success*, by Karl Wiegers. It is available on the web.

*Partial Example*

* Total project cost does not exceed 20% of the post-requirements phase estimate.
* All high-priority use cases in the requirements specification are delivered before May 15.

## Definitions

This section should define potentially unfamiliar or ambiguous words, acronyms and abbreviations.

## Evolution of the Project Plan

This section describes plans for updating the project plan throughout the project.

*Partial Example*

Before the start of an iteration, the project plan will be updated to include a schedule of detailed tasks for the upcoming iteration. At the conclusion of an iteration, the project plan will be updated to include the actual effort for each completed task.

Risk mitigation efforts will be evaluated at the start of each iteration. Severe risks will be analyzed and added to the project plan as soon as they materialize.

# Startup Plan

## Team Organization

This section explains project roles and the authorities and responsibilities associated with these roles. Lines of communication, authority and reporting relationships are often shown with an org chart. If development team is known, actual names can be associated with roles.

*Partial Example*

Project Manager: The project manager is responsible for creating the project plan (with input from those doing the work), managing risks, running the weekly team meeting and providing monthly status reports to senior management.

Programmers (3): Programmers are primary responsible for coding and unit testing modules. They are also expected to take part in architecture planning and review meetings.

Build Coordinator: The build coordinator is responsible for setting up, running and distributing the results of the nightly build.

## Project Communications

This section contains the project communications plan. The communications plan describes how information is gathered and distributed.

## Technical Process

This section describes the software development methodology or conventions the team agrees to live by. When following an organization standard process, this section will refer to the standard process and state any deviations that are planned for this project. In the absence of an organization standard process, this section will define planned phases, entry and exit criteria for each phase, major milestones, workflows, and other aspects of the proposed development process.

## Tools

* Programming Language – ASP .Net, HTML
* Version Control – source code and written artifacts will be stored in a Github repository.
* Automated testing – Visual Studio’s Unity will be utilized for unit testing.

# Work Plan

## Activities and Tasks

Depending on the needs of the project, some or all of the following attributes will be recorded for each task in a separate Team4 Estimated Effort Document (TBD):

* Task name
* Task Description
* Owner
* Effort estimate
* Actual effort
* Planned start and stop dates
* Actual start and stop dates
* Dependencies with other tasks

## Release Plan

***3.2.1 Release Plan By Feature***

**Iteration #1 9/29 – 10/12**

**Summary:** Develop an initial architecture design and populate an early database with tables that will be used in future iterations.

|  |  |  |
| --- | --- | --- |
| Features / Deliverables | **Estimated Effort** | **Actual Effort** |
| Database data-bank established | 10 | 6 |
| Architecture design implemented | 40 | 55 |

**Iteration #2 10/13 – 10/26**

**Summary:** Include user authentication along with a user table to be added to the database. Also include the ability to propose a new project, edit an existing project, and pagination for existing views.

|  |  |  |
| --- | --- | --- |
| Features / Deliverables | **Estimated Effort** | **Actual Effort** |
| Authentication | 20 |  |
| Propose a new project | 50 |  |
| Edit a project | 30 |  |
| Pagination | 30 |  |

**Iteration #3 10/27 – 11/09**

**Summary:** Develop all status change types for projects. Write unit testing for all existing features up until iteration 3 and begin unit testing for iteration 3 items.

|  |  |  |
| --- | --- | --- |
| Features / Deliverables | **Estimated Effort** | **Actual Effort** |
| Change the status of a project | 50 |  |
| Begin unit testing | 50 |  |

**Iteration #4 11/10 – 11/30**

**Summary:** Implement admin ability to view reports

|  |  |  |
| --- | --- | --- |
| Features / Deliverables | **Estimated Effort** | **Actual Effort** |
| Admin – View reports | 50 |  |

**Iteration #5 12/01 – 12/08**

**Summary:** Provide finishing touches and implement any extra features as time permits.

|  |  |  |
| --- | --- | --- |
| Features / Deliverables | **Estimated Effort** | **Actual Effort** |
| Finishing touches | 30 |  |

**Features not scheduled, but under consideration**

|  |  |  |
| --- | --- | --- |
| Features | **Estimated Effort** | **Actual Effort** |
| Display recently viewed projects | 20 |  |
| Display # vouches / supporters | 20 |  |

***3.2.2 Release Plan Flow Chart***

## Iteration Plans

***3.3.1 First Iteration***

Develop an initial architecture design and populate an early database with tables that will be used in future iterations.

***3.3.2 Second Iteration***

Include user authentication along with a user table to be added to the database. Also include the ability to propose a new project, edit an existing project, and pagination for existing views.

***3.3.3 Third Iteration***

Develop all status change types for projects. Write unit testing for all existing features up until iteration 3 and begin unit testing for iteration 3 items.

***3.3.4 Fourth Iteration***

Implement admin ability to view reports. Admin should be able to view monthly or weekly reports ordered by date, submitter, ambassador, university, or mentor.

***3.3.5 Fifth Iteration***

Provide finishing touches and implement any extra features as time permits.

## Budget

At the end of iteration 1, the expended budget is 61 points. Projected budget for iterations 2 through 5 is 310-350 points, depending if optional features are added.

# Control Plan

## Monitoring and Control

|  |  |
| --- | --- |
| **Date** | **Review / Milestone** |
| Weekly | Group meetings with status updates |
| 09/03/2014 | Project initialized |
| 10/12/2014 | Milestone: Technical Prototype complete |
| 10/12/2014 | Milestone: Iteration #1 complete |
| 10/26/2014 | Milestone: Iteration #2 complete |
| 10/29/2014 | Managers’ Briefing |
| 11/09/2014 | Milestone: Iteration #3 complete |
| 11/30/2014 | Milestone: Iteration #4 complete |
| 12/08/2014 | Milestone: Iteration #5 complete |

## Project Measurements

|  |  |  |
| --- | --- | --- |
| **Phase** | **Measurement** | **Source** |
| Release Planning | Record effort estimates for product features | Mgr |
| Iteration Planning | Record effort estimates for scheduled tasks  Update effort estimates for product features  Update estimated dates in release plan | Mgr |
| Iteration Closeout | Record actual effort for scheduled tasks  Record actual effort for product features  Record LOC count for modules written | Mgr/Pgr |
| System Test | Record the rate at which errors are found. | QA |
| Project Closeout | Archive project performance data in process database. (See process database definition for a list of measures to record.) | Mgr |
| Ongoing | Record defects found from integration testing through first year of release.  Assign each defect to one of the following categories: blocker, critical, major, minor or trivial. Keep track of the state of each defect: open, assigned, fixed, closed. | Mgr/Pgr/QA |

# Supporting Process Plans

## Risk Management Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Priority** | **Risk** | **Probability & Size of Loss** | **Response** |
| 1 | Lack of web dev. experience | Likely / Major | Mitigate – Invest extra time in research. Ask team members for help |
| 2 | Lack of database integration with MVC | Likely / Major | Mitigate – Invest extra time in research. Ask team members for help |
| 3 | Security | Unlikely / Major | Mitigate – Initialize several SQL injection preventative measures |

## Configuration Management Plan

Configuration management plans for this document and other baselined work products including review procedures and change management procedures.

1. All work will be stored and maintained on a GitHub repository located at <https://github.com/cs451/Fall2014Team4>
2. The naming convention for all documents on the repository will be Type\_Name\_Version.suffix:
   1. Type will represent any identifying word or phrase that categorizes the file. Types include Document, Baselined-Document, Dev for Development, and Prod for Production.
   2. Name will be unique for the file that is being maintained
   3. Version will be used so that older versions of documents are accessible
   4. An example: Document\_Project-Charter\_1.0.doc
   5. Any file(s) with more than 4 versions listed will have a subfolder created in order to reduce repository clutter
   6. As production continues beyond iteration 1, a separate “Prod” and “Dev” folder will be maintained as new features are implemented
      1. The Prod folder will be updated after each iteration OR if a large version change has taken place and the development team has decided to push their changes to Prod early.
   7. Versions for source code files will have a subversion implicit to their naming convention
      1. Version – Iteration#.Version#.Subversion#
      2. Example – version 1.1.12 will be the 1st iteration of the 2nd version of a file that has had 12 total minor alterations made to it. When this file is first moved to Prod, the version will be incremented to 1.2.0
      3. Version #’s will be maintained only by the development team.
      4. Subversions will be instantiated by any other contributor any time a source code item is modified.
3. Project items will all be stored on the GitHub repository but some will be under change control. These items will have a Type of Baselined-Document, and will include the following: System Requirements Summary (SRS), Architecture Document, and source code in Prod.
4. Before a baselined item can be considered complete, a group review must take place.
5. The change control procedure once a product is baselined is as follows:
   1. The entire group is notified when a member proposes a change to a baselined item.
      1. A description of the change
      2. The reason for the change
      3. The impact expected from the change
      4. A timeline for integrating the change
   2. If nobody has responded to the proposal within 24 hours, a reminder will be sent to the group. If another 24 hours passes, it will be assumed that the change is accepted.
   3. If a group member wants to alter the proposed change, a meeting will be scheduled for parties involved in order to make appropriate changes to the proposal or to reject the change altogether.
   4. After the proposed change is refined, a democratic vote will take place whether or not to finalize the change.
6. Each baselined item will contain a change history. The change history will be at the front of the work item and will include the following: (1) the name of the group member making the change, (2) a brief description of what has been changed, (3) reason(s) for the change, and (4) the date the change was integrated.

## Verification and Validation Plan

The verification and validation plan can be found in a separate document in GitHub at (TBD).

## Product Acceptance Plan

When each iteration is complete, the prototype will be tested to ensure each requirement is met. At least three days of testing will exist for each iteration. Product acceptance testing will ensure that the prototype handles all added features the way that is intended and, at the conclusion of the project, all necessary features will have been added and tested extensively.

If time permits, optional features will be added using prior guidelines for feature implementation.