

**Computer Science and Engineering**

**DotaMatch**

**Project Management Plan**

**Version 1.0**

Document Number: SPMP-001

Project Team Number A11

Project Team Members:

Corey Chong cc3899

Albert Su as7353

Evans Yeung eby210

**REVIEW AND APPROVALS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Printed Name and Title** | **Function (Author, Reviewer, Approval)** | **Date** | **Signature** |
| Evans Yeung | Author | 4/6/2016 | Evans Yeung |
| Corey Chong | Author | 4/7/2016 | Corey Chong |
| Corey Chong | Reviewer | 4/7/2016 | Corey Chong |
| Albert Su | Reviewer | 4/8/2016 | Albert Su |
| Evans Yeung | Reviewer | 4/10/2016 | Evans Yeung |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**REVISION LEVEL**

|  |  |  |
| --- | --- | --- |
| **Date** | **Revision Number** | **Purpose** |
| April 6, 2016 | Version 1.0 | Initial Release |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Contents

[1. INTRODUCTION 5](#_Toc447668223)

[1.1 Project Summary 5](#_Toc447668224)

[1.2 Purpose, Scope, and Objectives 5](#_Toc447668225)

[1.3 Assumptions and Constraints 5](#_Toc447668226)

[1.4 Project Deliverables 6](#_Toc447668227)

[1.5 Schedule and Budget Summary 7](#_Toc447668228)

[1.6 Evolution of the Plan 7](#_Toc447668229)

[2. REFERENCES 8](#_Toc447668230)

[3. DEFINITIONS 8](#_Toc447668231)

[4. PROJECT ORGANIZATION 9](#_Toc447668232)

[4.1 External Interfaces 9](#_Toc447668233)

[4.2 Internal Structure 9](#_Toc447668234)

[4.3 Roles and Responsibilities 9](#_Toc447668235)

[*4.3.1 Roles* 9](#_Toc447668236)

[*4.3.2 Responsibilities* 10](#_Toc447668237)

[5 MANAGEMENT PROCESSES 10](#_Toc447668238)

[5.1 Start-up Plan 11](#_Toc447668239)

[5.1.1 Estimation Plan 11](#_Toc447668240)

[5.1.2 Staffing Plan 11](#_Toc447668241)

[5.1.3 Resources Acquisition Plan 11](#_Toc447668242)

[5.1.4 Training Plan 11](#_Toc447668243)

[5.2 Work Plan 11](#_Toc447668244)

[*5.2.1 Work Activities* 12](#_Toc447668245)

[*5.2.2 Schedule Allocation* 13](#_Toc447668246)

[*5.2.3 Resource Allocation* 13](#_Toc447668247)

[*5.2.4 Budget Allocation* 13](#_Toc447668248)

[5.3 Control Plan 13](#_Toc447668249)

[*5.3.1 Requirement Control and Traceability* 14](#_Toc447668250)

[5.3.2 Schedule Tracking and Adjustment 14](#_Toc447668251)

[*5.3.3 Budget Tracking and Adjustment* 14](#_Toc447668252)

[*5.3.4 Quality Control* 14](#_Toc447668253)

[*5.3.5 Reporting Mechanisms* 14](#_Toc447668254)

[*5.3.6 Metrics Collection Plan* 15](#_Toc447668255)

[5.4 Risk Management Plan 15](#_Toc447668256)

[5.5 Post Implementation Plan 15](#_Toc447668257)

[6. TECHNICAL PROCESSES 15](#_Toc447668258)

[6.1 Process Model 15](#_Toc447668259)

[6.2 Methods, Tools, and Techniques 16](#_Toc447668260)

[6.3 Infrastructure Plan 16](#_Toc447668261)

[6.4 Product Acceptance and Migration Plan 16](#_Toc447668262)

[7 SUPPORTING PROCESSES PLANS 16](#_Toc447668263)

[7.1 Configuration Management Plan 16](#_Toc447668264)

[7.2 Qualification (Verification and Validation) Plan 17](#_Toc447668265)

[7.3 Documentation (library) Plan 17](#_Toc447668266)

[7.4 Quality Assurance Plan 17](#_Toc447668267)

[7.5 Reviews and Audits 17](#_Toc447668268)

[7.6 Problem Resolution Plans 19](#_Toc447668269)

[7.7 Environment Management Plans 19](#_Toc447668270)

[7.8 Process Improvement Plan 19](#_Toc447668271)

[8. ADDITIONAL PLANS 19](#_Toc447668272)

[9. INDEX 20](#_Toc447668273)

[10. RATIONALE 20](#_Toc447668274)

[11. NOTES 20](#_Toc447668275)

[12. APPENDICES 21](#_Toc447668276)

[12.1 Schedule Tracking 21](#_Toc447668277)

[12.2 Defect Tracking 22](#_Toc447668278)

[12.3 Gantt Chart 24](#_Toc447668279)

# 1. INTRODUCTION

## 1.1 Project Summary

The motivation for DotaMatch is to allow users to be matched with players of their choice rather than being randomly placed on a team by Dota’s in-game matchmaking system. As a result, player’s unequal skill, toxic, or negative players can be placed onto the same team. One solution to avoiding such players is to create a team by inviting one’s friends, however, there are players who cannot. DotaMatch is a matchmaking application in which players can find potential matches (players who fit their preferences) and start games with them.

The purpose of the Software Project Management Plan (SPMP) briefly describes the project, the client deliverables, the project milestones, and expected documented. The intended audience of this document includes the management and the developers.

## 1.2 Purpose, Scope, and Objectives

DotaMatch is a matchmaking application that will make the process of finding fellow players a more fun and enjoyable experience by introducing other factors besides skill into the equation. Users will be able to set their own preferences for the type of players they would like to be matched with. Our algorithm takes into account these preferences and other factors to decisively match players in the database. Once matchmaking is complete, the user will be presented with a matched player’s profile which includes: in-game statistics, preferred role, favorite characters, etc. A player rating system is implemented so that the user will be able to rate the players they have played with. Overall this application is designed with the user’s freedom of choice in mind in order to create a positive experience for all of our users.

## 1.3 Assumptions and Constraints

**Assumption**:

* *Security*

System will be secure. Users must provide login credentials.

* *Maintainability*

System will undergo postdelivery maintenance. Postdelivery maintenance is specified in Section 8.

**Constraints**:

* *Deliverable Deadlines*

All deliverables must be submitted electronically to NYU Classes by deliverable deadlines. All deliverables and delivery dates are listed in section 1.4.

* *Software*

Software constantly updates.

* *Staff/Skill*

Staff is limited to the three members in Team A11. Skill of each staff varies.

* *Reliability*

System must provide the right matches more than 99% of the time.

* *Availability*

System must be available to users more than 99% of the time.

## 1.4 Project Deliverables

|  |  |  |
| --- | --- | --- |
| Deliverables | Delivery Date | Format |
| Software Requirements (SRS) Version 1.0 | 3/7/2016 | Electronic form |
| Software Requirements (SRS) Version 2.0 | 3/21/2016 | Electronic form |
| Software Project Management Plan (SPMP) Version 1.0 | 4/6/2016 | Electronic form |
| Software Analysis Specification (SAS) | 4/11/2016 | Electronic form |
| Software Design Document (SDD) | 4/25/2016 | Electronic form |
| Project Completion and Finished Project | Fall 2016 | Electronic form |

## 1.5 Schedule and Budget Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Schedule Summary | | | |
| Task Name | Start Date | End Date | Duration (days) |
| Project Team Selection | 2/3/2016 | 2/17/2016 | 14 |
| Brainstorm | 2/7/2016 | 2/10/2016 | 3 |
| Outline Functions of Idea | 2/10/2016 | 2/12/16 | 2 |
| Project Proposal | 2/13/16 | 2/17/16 | 4 |
| Outline Requirements | 2/17/16 | 2/20/16 | 3 |
| Systems Requirements Specification (SRS) Version 1.0 | 2/20/16 | 3/7/16 | 17 |
| Review SRS Version 1.0 | 3/7/16 | 3/10/16 | 3 |
| Systems Requirements Specification (SRS) Version 2.0 | 3/10/16 | 3/23/16 | 13 |
| Software Project Management Plan (SPMP) Version 1.0 | 3/23/16 | 4/6/16 | 13 |
| Software Analysis Specification (SAS) Version 1.0 | 4/6/16 | 4/11/16 | 5 |
| Software Design Document (SDD) Version 1.0 | 4/11/16 | 4/25/16 | 14 |
| Oral Presentation | 5/2/16 | 5/9/16 | 7 |

Budget summary not required.

## 1.6 Evolution of the Plan

Team A11 is solely responsible for managing and updating the Software Project Management Plan (SPMP). The Software Project Management Plan (SPMP) is subject to changes. Version numbers will be incremented by a whole number if major changes occur. While decimal increments will occur if minor changes occur. The assumptions, dependencies, constraints, deliverable dates and resourcing planning can change during the project. Scheduled updates will occur based on delivery date. Unscheduled updates will occur to fix faults or any changes in deliverable dates. Throughout the lifecycle, the Software Project Management Plan (SPMP) will undergo changes based upon faults, deficiencies, shortcoming, inaccuracies, changes in system environments, or changes to deliverable dates.

# 2. REFERENCES

All documents are electronically submitted to NYU Classes and can be referenced through NYU Classes.

* Project Proposal was electronically submitted to NYU Classes on February 17, 2016.
* Software Requirements Specifications (SRS) Version 1.0 was electronically submitted to NYU Classes on March 7, 2016.
* Software Requirements Specifications (SRS) Version 2.0 was electronically submitted to NYU Classes on March 21, 2016.
* Software Project Management Plan (SPMP) Version 1.0 will be submitted to NYU Classes on April 6, 2016. Once submitted it can be referenced through NYU Classes. Software Project Management Plan (SPMP) Version 1.0 can currently be referenced through Google Doc.

# 3. DEFINITIONS

No additional acronyms needed to be defined.

# 4. PROJECT ORGANIZATION

## 4.1 External Interfaces

The product will be able to parse dotabuff.com and yasp.co profile data and migrate them to their respective DotaMatch profiles. Linking accounts should require additional authentication through the Steam OpenID service.

## 4.2 Internal Structure

The internal structure of Team A11 is a democratic system. The democratic system promotes an ethos (group identity) instead of individual identities. All members have mutual respect for each other and are able share opinions and decisions without an authoritative figure. All ideas about the functionality, documentation, progress, etc. are discussed between all members. Coding will be implemented through egoless programming where no member is attached to their code. As a result, members are open to finding faults in their own code and pointing out faults in another member’s code.

## 4.3 Roles and Responsibilities

### *4.3.1 Roles*

|  |  |  |
| --- | --- | --- |
| Document | Name | Role(s) |
| Project Proposal | Corey Chong | Author and Reviewer |
| Project Proposal | Albert Su | Author and Reviewer |
| Project Proposal | Evans Yeung | Author and Reviewer |
| Software Requirements (SRS) Version 1.0 | Corey Chong | Reviewer |
| Software Requirements (SRS) Version 1.0 | Albert Su | Reviewer |
| Software Requirements (SRS) Version 1.0 | Evans Yeung | Author and Reviewer |
| Software Requirements (SRS) Version 2.0 | Corey Chong | Reviewer |
| Software Requirements (SRS) Version 2.0 | Albert Su | Reviewer |
| Software Requirements (SRS) Version 2.0 | Evans Yeung | Author and Reviewer |
| Software Project Management Plan (SPMP) Version 1.0 | Corey Chong | Author and Reviewer |
| Software Project Management Plan (SPMP) Version 1.0 | Albert Su | Reviewer |
| Software Project Management Plan (SPMP) Version 1.0 | Evans Yeung | Author and Reviewer |

### *4.3.2 Responsibilities*

**Author**

Responsible for writing the documents based upon the decisions discussed between group members. Author must self-check their work without being attached to it.

**Reviewer**

All members are required to review each document throughout each workflow. All documents must be reviewed for correctness, ambiguity, completeness, consistency, stability, verifiability, modifiability (malleable), and traceability.

# 5 MANAGEMENT PROCESSES

## 5.1 Start-up Plan

Staff members were chosen at the beginning of the semester.

## 5.1.1 Estimation Plan

Not required.

## 5.1.2 Staffing Plan

Staff consist of the three members from Team A11. Staff may leave the project at any time during the project. Additional staff may be assigned to the team upon approval from Professor Strauss. The skill level of all staff members are students of NYU Tandon School of Engineering.

## 5.1.3 Resources Acquisition Plan

NYU will provide the server and database. Each member will have their own development machine.

## 5.1.4 Training Plan

The staff will be trained in-house by other teammates, in addition to any additional training that may be provided online. Staff will also learn required skills individually during their own time.

## 5.2 Work Plan

Team members will meet and discuss progress and how to proceed onward with the project. The development team will work freely on the project artifacts and documents. Collaboration will be required on a particularly resource intensive components of the project. The work schedule will be allocated and adjusted according to the workload of each respective team member.

### *5.2.1 Work Activities*

|  |  |  |
| --- | --- | --- |
| **Workflow** | **Start Date** | **End Date** |
| **Requirements Workflow** | 2/3/16 | 3/23/16 |
| Risk Analysis (Inception Phase) | 2/7/16 | 2/20/16 |
| Execute (Elaboration and Construction Phase) | 2/20/16 | 3/21/16 |
| Review/Test (Transition Phase) | 3/21/16 | 3/22/16 |
| Post It  END OF MILESTONE 1 | 3/22/16 | 3/23/16 |
| **Analysis Workflow** | 3/23/16 | 4/25/16 |
| Risk Analysis (Inception Phase) | 3/23/16 | 3/26/16 |
| Execute (Elaboration and Construction Phase) | 3/26/16 | 4/4/16 |
| Review/Test (Transition Phase) | 4/4/16 | 4/5/16 |
| Post It  END OF MILESTONE 2 | 4/24/16 | 4/25/16 |
| **Design Workflow** (To be completed in CS-4523 Senior Project) | 9/6/16 | N/A |
| Risk Analysis (Inception Phase) | 9/6/16 | N/A |
| Execute (Elaboration and Construction Phase) | N/A | N/A |
| Review/Test (Transition Phase) | N/A | N/A |
| Post It  END OF MILESTONE 3 | N/A | N/A |
| **Implementation Workflow**(To be completed in CS-4523 Senior Project) | N/A | 12/16/16 |
| Risk Analysis (Inception Phase) | N/A | N/A |
| Execute (Elaboration and Construction Phase) | N/A | N/A |
| Review/Test (Transition Phase) | N/A | N/A |
| Post It  END OF MILESTONE 4 | N/A | 12/16/16 |

### *5.2.2 Schedule Allocation*

Profile Creation

Credential Management

Party Management

Administration Tools

UI/UX

Release

Sept 2016

Dec 2016

Nov 2016

### *5.2.3 Resource Allocation*

The staff and resources will be distributed equally for all work activities shown in the schedule allocation. Hardware and software resources will be provided by the development team.

### *5.2.4 Budget Allocation*

Not required.

## 5.3 Control Plan

The project work schedule will document the total amount of human hours each team member has put in and the cumulative number of hours for each document. The defect tracking schedule will document the estimated number of faults per team member, the actual number of faults, and the cumulative number of faults detected. The Gantt chart will indicate the life cycle of the project and the number of days allocated to each milestone and its subtasks.

### *5.3.1 Requirement Control and Traceability*

GitHub will be used as the version control software for the product. Each member of the development team will submit commits to the repository in order to maintain revision control on the product. The repository may be forked if drastic modifications are needed (for testing purposes.)

### 5.3.2 Schedule Tracking and Adjustment

The team members will self-check the progress and completeness of each component in addition to a brief walkthrough performed by Professor Strauss. Validity of the artifact will be assessed by comparing it to the client’s requirements.

### *5.3.3 Budget Tracking and Adjustment*

Not required.

### *5.3.4 Quality Control*

Components will be tested for quality through cursory inspection by the development team, with occasional review by quality assurance testers. They may be subject to additional review upon modification of any reciprocating components that may rely on it. Acceptance test will be conducted by users to test whether it meets requirements. Software Quality Assurance (SQA) group will also test the product against the requirements.

### *5.3.5 Reporting Mechanisms*

Not required.

### *5.3.6 Metrics Collection Plan*

Metrics to be collected will be:

* Lines of code
* Time spent on each component
* Number of faults detected

These metrics shall be obtained on a regular basis of three weeks, which will be committed to a document stored in the GitHub repository.

## 5.4 Risk Management Plan

The product will be subjected to frequent testing to reduce the risk of errors, especially during the implementation phase of the component. Each team member will be responsible for testing their code, in addition to testing another member’s code for errors and integration. Throughout this phase each component will be tested repeatedly as described above to prevent risks in the event of frequent modifications. Each artifact is constructed piece by piece (increments). Each increment goes through multiple versions (iterations). The project as a whole is divided into smaller mini projects. Every mini project extends the workflows. Test occurs at the end of each workflow. Planning and documentation activities are performed throughout the life cycle.

## 5.5 Post Implementation Plan

Not required.

# 6. TECHNICAL PROCESSES

## 6.1 Process Model

The Iteration and Incrementation with Object-Oriented Paradigm will be used for this project. The workflows for this process include: Requirements workflow, Analysis workflow, Design workflow, Implementation workflow, and Test workflow. Each workflow will have phases: Inception phase, Elaboration phase, Construction phase, and Transition phase.

## 6.2 Methods, Tools, and Techniques

* Iterative and Incremental Life Cycle Model
* CASE Tools
* Use Case Model
* UML Diagram
* Microsoft Word
* Microsoft Excel
* Google Docs

## 6.3 Infrastructure Plan

Not required.

## 6.4 Product Acceptance and Migration Plan

The Software Quality Assurance (SQA) group will test the product against the requirements.

The users will implement a series of tests on the finished product. If product passes the tests and satisfies the specification then it is deemed satisfactory. Else, the product has not met the requirements of the client. Tests will be specified at a later date by client.

DotaMatch will be initially built on Android Lollipop Version 5.0 or higher. Migration to iOS will depend on multiple factors: popularity, number of user, compatibility, and available resources. If criteria are met, migration to iOS will be available in the future.

# 7 SUPPORTING PROCESSES PLANS

## 7.1 Configuration Management Plan

The product will be hosted on GitHub to provide a change control tool.

## 7.2 Qualification (Verification and Validation) Plan

All documents can be referenced through NYU Classes. All documents should have forward (to all artifacts spawned by this document) and backward (to previous stages of development) traceability. Forward traceability will allow tracing in a forward direction tracing from requirements to postdelivery workflows. While backward traceability allows tracing in the direction of postdelivery to requirements workflows.

Testing will occur at the end of each workflow. Product will also be tested with a small sample of users to determine the functions work and met requirements. If test with a small sample size is deemed to be successful, we will move onto the next testing phase. The next testing phase will test the product against a larger user base.

Each document will be reviewed through each of the review processes mentioned in Section 7.5

## 7.3 Documentation (library) Plan

Documentation will be generated, reviewed, and approved by Team A11. The initial baseline version will be delivered

## 7.4 Quality Assurance Plan

Each member of Team A11 is required to perform a self-check of their work. Testing must occur at the end of every workflow. Each document will undergo Tandon Management Review, self-check, walkthrough, and inspection.

## 7.5 Reviews and Audits

**Tandon University Management Review**  
Professor Strauss will perform review for each team’s documents during every deliverable deadline. Each document will review and a grade will be given to each team’s document. Comments indicating error will be provided on paper and electronically through NYU Classes.

**Self-Check**

Each member of the project is required to self-check the Software Requirements Specification (SRS). During the self-check, each reviewer will be thoroughly looking for correctness, ambiguity, completeness, consistency, stability, verifiability, modifiability (malleable) and traceability. Undiscovered faults can lead to future faults. After completing the self-check, each member must sign and date.

**Walkthrough**

A walkthrough team consist of four to six individuals. These members should be experienced senior technical staff members because they tend to find faults that would have a major negative impact on the project. There will be at least one representative for each workflow from the project group. There is also a client representative and a SQA representative. Material for the walkthrough needs to be distributed to the participants in advance. Each reviewer should study the material and develop two lists. One list should contain items the reviewer does not understand. The other list should contain items the reviewer believes to be incorrect. The person leading the walkthrough will be the SQA representative. There are two ways to conduct the walkthrough. The first is where the participant presents their list of unclear items and items they think are correct. The second is document driven. The person responsible for each document should walk the participant through the document. As a result the second type is interactive between the presenter and the participants. Faults found throughout the walkthrough are to be recorded and corrected later on.

**Inspection**

Inspections have five formal steps and should go beyond a walkthrough. The first step is an overview of the document to be inspected is provided to the participants. In the preparation step, the participants try to understand the document in detail. Next the participant walks through the document with the inspection team. Fault finding commences afterward. Within one day, the leader of the inspection team (the moderator) must produce a written report of the inspection. The next step is rework, where those responsible for the documents correct faults that were noted on the written report. The last step is follow-up, where the moderator must ensure that every issue raised has been resolved satisfactorily, by either fixing the document or clarifying items incorrectly flagged as faults. If more than 5 percent of the material inspected has been reworked, then the team must reconvene for a 100% re-inspection.

## 7.6 Problem Resolution Plans

The developers will be responsible for arbitrating any problem that may arise during the product’s use. Each team member will resolve problems on a case-by-case basis relative to their respective fields of expertise. The product will implement a bug report interface to allow users to quickly contact the development team, or they may be contacted by email.

## 7.7 Environment Management Plans

Not required.

## 7.8 Process Improvement Plan

Will not be performed during CS-4513 Software Engineering. Process Improvement Plan will be completed in Fall 2016 CS-4523 Senior Project.

# 8. ADDITIONAL PLANS

**Migration**

Plans for migrating to iOS will depend on multiple factors:

* High popularity
* Large number of user
* Compatibility of systems and software
* Availability of resources

If criteria are met, plans for migration will be implemented.

**Postdelivery Maintenance**

Product will be maintained in three ways:

* Corrective Maintenance - removal of residual faults while leaving specifications unchanged.
* Perfective Maintenance - changes the client thinks will improve the effectiveness of the product.
* Adaptive maintenance - changes made in response to a change in environment in which the product operates.

# 9. INDEX

Additional Plans 19, 20

Constraints 6

Control Plan 13, 14

Documentation 20, 21, 22, 23, 24

Purpose 6

Project Deliverables 6, 7, 8

Project Tables 20, 21, 22, 23, 24

Methods, Tools, Techniques 16

Quality Assurance 14, 16, 17

References 8

Reviews 1, 17, 18

Risk Management 15

Roles and Responsivities 9, 10

Table of Contents 3, 4

Training Plan 11

Work Activities 12

Work Plan 11

# 10. RATIONALE

No additional rationales.

# 11. NOTES

No additional notes.

# 12. APPENDICES

## 12.1 Schedule Tracking

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| Software Requirements Specification (SRS) Version 1.0 | Corey Chong | 6 | 7 | 1 |
| Software Requirements Specification (SRS) Version 1.0 | Albert Su | 6 | 7 | 1 |
| Software Requirements Specification (SRS) Version 1.0 | Evans Yeung | 6 | 7 | 1 |
| Software Requirements Specification (SRS) Version 2.0 | Corey Chong | 5 | 4 | 1 |
| Software Requirements Specification (SRS) Version 2.0 | Albert Su | 5 | 5 | 0 |
| Software Requirements Specification (SRS) Version 2.0 | Evans Yeung | 6 | 8 | 2 |
| Software Project Management Plan (SPMP)  Version 1.0 | Corey Chong | 10 | 8 | 2 |
| Software Project Management Plan (SPMP)  Version 1.0 | Evans Yeung | 12 | 9 | 3 |
| Software Project Management Plan (SPMP)  Version 1.0 | Albert Su | 8 | 5 | 3 |

**Cumulative**

|  |  |  |  |
| --- | --- | --- | --- |
| Who (individual or Team) | Estimated | Actual | Difference |
| Corey Chong | 21 | 19 | 2 |
| Albert Su | 19 | 17 | 2 |
| Evans Yeung | 24 | 25 | 0 |

## 12.2 Defect Tracking

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| Software Requirements Specification (SRS) Version 1.0 | Corey Chong | 15 | 20 | 5 |
| Software Requirements Specification (SRS) Version 1.0 | Albert Su | 14 | 17 | 3 |
| Software Requirements Specification (SRS) Version 1.0 | Evans Yeung | 18 | 25 | 7 |
| Software Requirements Specification (SRS) Version 2.0 | Corey Chong | 15 | 22 | 7 |
| Software Requirements Specification (SRS) Version 2.0 | Albert Su | 20 | 24 | 4 |
| Software Requirements Specification (SRS) Version 2.0 | Evans Yeung | 14 | 15 | 1 |
| Software Project Management Plan (SPMP)  Version 1.0 | Corey Chong | 25 | 23 | 2 |
| Software Project Management Plan (SPMP)  Version 1.0 | Albert Su | 24 | 16 | 8 |
| Software Project Management Plan (SPMP)  Version 1.0 | Evans Yeung | 19 | 24 | 5 |

**Cumulative**

|  |  |  |  |
| --- | --- | --- | --- |
| Who (individual or Team) | Estimated | Actual | Difference |
| Corey Chong | 55 | 65 | 10 |
| Albert Su | 58 | 57 | 1 |
| Evans Yeung | 51 | 64 | 13 |

## 12.3 Gantt Chart