Software Engineering and Project

Software Project Management Plan

UG-14

a1647816 Anthony Scaffidi
a1693014 Jingbin Zhang
a1710974 Jinshan Wang
a1727491 Lincoln Phillips
a1687940 Mitchell Phillips
a1696673 Nanye Zou
a1669907 Shaun Kraemer

1.2

21/10/2018

Table of Contents

Version History

Date	Version	Changer	Description
10/8/2018	0.1	Shaun Kraemer	Completed the first draft of the document filling in all sections of the provided template.
14/8/2018	0.2	Anthony Scaffidi	Added additional sections to the ‘Support Plan’ regarding documentation and project management.
22/8/2018	0.3	Shaun Kraemer	Added the ‘Testing Procedure’ section as well as a few additions to the ‘Scope’ and ‘Process Model’ sections
30/8/2018	0.4	Shaun Kraemer	Added ‘Milestone 1’ specification. Extracted from *Client Meeting Agenda & Minutes: Semester 1, week 6*, Written by Anthony Scaffidi.
12/10/2018	0.5	Jingbin Zhang	Additions/Changes made to the ‘Introduction’ and ‘Process Model’ sections.
12/10/2018	0.6	Jinshan Wang	Added the ‘Project Deliverables’ section.
12/10/2018	0.7	Nanye Zou	Additions/Changes made to the ‘Documentation plan’ as well as adding the following section: ‘Risk Management’, ‘Configuration Plan’, ‘Quality Assurance’. The ‘Gantt Chart’ has also been added to the Appendix A.
20/10/2018	0.8	Shaun Kraemer	Additions/Changes made to the ‘Process Model’ and ‘Overview’ sections as well as the addition of the ‘Milestone 2’ and ‘Code Review’ sections.
21/10/2018	0.9	Anthony Scaffidi	Additions/Changes made to the ‘Documentation Plan’.
21/10/2018	1.0	Anthony Scaffidi	Additions/Changes made to ‘Supporting Plan’ Added the ‘Updating Plan’. (has since been removed)
21/10/2018	1.1	Jinshan Wang Jingbing Zhang	Added the ‘Testing and Verification’ section.
21/10/2018	1.2	Nanye Zou Anthony Scaffidi	Additions/Changes made to the ‘Risk Management’ section.
21/10/2018	1.2	Lincoln Phillips	Proofreading and spell checking.
21/10/2018	1.2	Jingbin Zhang	Additions/Changes made to the documents layout.

1.Introduction

1.1 Scope

The purpose of the project is to produce code for a prototype unmanned rescue bot using the Lego Mindstorm EV3 kit. The prototype will be equipped with onboard sensors allowing for effective navigation around obstacles and finding survivors in an efficient manner. Once found, a survivor's coordinates will be returned, and the prototype will continue canvassing the area.

The prototype is required to demonstrate its searching capabilities on an A0 paper size map with several obstacles to overcome and survivors to find. The proprietary GPS painting technology and live stream video camera that will be used in the final product are not within the scope of this initial prototype.

1.2 Intended audience

This document is intended for viewing by the following:

- Members of the 2018 Software Engineering & Project UG-14 group
- The University Natural Disaster Search and Rescue Service (UNDSRS)

1.3 Project Deliverables

- Software Project Management Plan (SPMP) (Draft) due 14th Aug 2018 for client review on process model chosen and overall planning of project.

- Systems Requirements Specification (SRS) (Draft) due 21st Aug 2018 for client review and feedback of extracted requirements.

- Software Design Document (SDD) (Draft) due 28th Aug 2018 for client review and feedback on overall design criteria and technicalities.

- Milestone 1 demonstration due 13th Sep 2018 demonstrating selected set of functionalities presented to client.

- Milestone 2 demonstration due 4th Oct 2018 demonstrating selected set of functionalities presented to client.

- Final versions of SPMP , SRS and SDD due 21st Oct 2018 for a completed set of documents for the project.

- Final presentation due 25th Oct 2018 for client meeting.

1.4 References

Lego *Mindstorms EV3*, viewed August 2018:
https://www.lego.com/en-us/mindstorms/about-ev3

EV3 *Connecting to the Internet via* USB tutorial:
https://www.ev3dev.org/docs/tutorials/connecting-to-the-internet-via-usb/

Client Meeting Agenda & Minutes: Semester 1, week 6:
https://docs.google.com/document/d/1UXjqjjUnusfi-0fYrfaB56pozS090MX3jZO0oBTOfgY/edit

Include any references (as well as internal documentation) mentioned in this document

1.5 Glossary

Commonly used terms within the following document include:

UNDSRS – The University Natural Disaster Search and Rescue Service
SEP – Software Engineering Project
UG-14 – Undergraduate 14
SPMP – Software Project Management Plan
CMP – Configuration Management Plan
RMP – Risk Management Plan
SRS – Software Requirements Specification
SDD – Software Design Document
SMP – Test Management Plan
CRMP – Code Review Management Plan
GUI – Graphical Users Interface
EV3 – Lego Mindstorms Robot Generation 3
IDE – Integrated Development Environment
JDK – Java Development Kit

2.Process

2.1 Process Model

The project uses an Agile methodology with a Scrum framework based around a two-week sprint period, the first of which commenced on August 6, 2018. At the beginning of each sprint team members will determine what needs to be done and what can be done before the end of the current sprint. The team is then left to complete what was set out and the cycle is repeated with each sprint until the project is complete.

The tasks for each sprint will be broken down into smaller tasks and divided amongst the team. This allows for better time management as if a task it is incomplete it is easier judge how long it will take to complete based upon how many smaller tasks within that task remain. Thus, any incomplete tasks can be added to the next sprint and the time required will have a better estimate. These smaller tasks allow for more quantifiable progress and provide simple progress updates for the client. Each sprint will have a checklist for tasks provided on Trello to determine which tasks are complete and which are not. As the project progresses task completion percentage should improve since the team will gather a better understand of what they are capable of.

2.2 Determine Objectives

Define and determine all system requirements in detail, including functional requirements, user requirements, performance, non-functional requirements, hardware/software interfaces, milestones.

The project commenced on the 5th of August 2018 and will be concluded on the 24th of October. The project will include two Milestones, the first was negotiated on August 29, 2018 and delivered on September 12, 2018; the second having been negotiated on September 12, 2018 and delivered on September 19, 2018.

2.3 Milestone 1

The negationations took place for the first milestone at 4:10pm Wednesday the 29th of August 2018 and drew the following requirements for the deliverable.

Primary functional requirements:

I. Ability to locate the boundary of the map and navigate within it.
II. Detect objects and avoid/reroute around them.

III. UI developed in java (layout produced but buttons, etc. events/links not implemented at this stage in development).

General capabilities:

I. Robot will be capable of mapping the map boundary via line following in its first pass.
II. Search grid resolution will be a 20x20cm grid - increased precision to follow later in later milestones.
III. Robot will search the interior of the map following the grid system row by row. Searched grids will be denoted as such and the robot will continue to a new grid location.
IV. If an object is encountered in a specific grid reference, the robot will turn and reroute to another undiscovered grid location; backtracking if no new grid location is unobstructed.
V. Once all grid locations are denoted as discovered, the search is complete.
VI. All objects will be detected as obstacles at this stage.

2.4 Milestone 2

The negationations took place for the first milestone at 4:10pm Wednesday the 12th of September 2018 and drew the following requirements for the deliverable.

Primary functional requirements:

I. UI integration will be complete and a dynamic map will be displayed as the robot surveys the map.
II. Robot's design will be changed to minimize it's size thus allowing for an increase in grid spaces.

General capabilities:

I. Updated map featuring 16x16cm grids allowing for a greater number of discoverable map locations and potential obstacle/survivor locations.
II. Dimensional reduction of robot to facilitate the smaller grids and avoid more densely packed obstacles.
III. Map displayed on GUI will resize dynamically to

2.5 Evaluate Alternatives

After determining the initial plan, it is necessary to assess the risks that may occur, try to propose all the prototypes that may be used for implementation, and make plans to start implementation, try to predict the risks that may be encountered in the project realisation process and propose corresponding solutions.

2.6 Overview

The team aims to have two group conference calls per sprint. The first taking place at the beginning of the sprint and will be used to finalize completed tasks from the previous sprint as well as to determine what will be completed in the current sprint. The second conference call scheduled mid-way through each sprint will be used as a status meeting whereby each member can update the team on what they have completed and what still needs to be done before the end of the current sprint. Alongside this Facebook Messenger will be used for team members to keep each other up-to-date between conference calls and a weekly meeting with the client will be conducted. Trello will be used to keep the team organized, providing each member with a clear view of what tasks have been completed and what still needs to be completed.

Face-to-face meetings and workshops with the prototype will be conducted on a weekly basis beginning August 21, 2018. Eclipse will be used as the development platform with GitHub being used for version control.

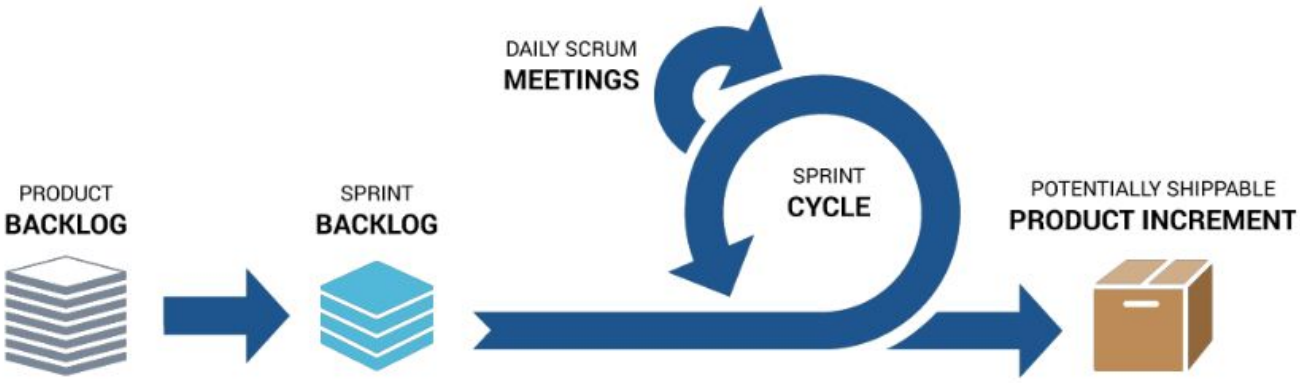


Figure 1: The SCRUM Process

3.Supporting Plan

3.1 Git configuration - structure & branching

The team identified the following project requirements: documentation, GUI design, and source code.The GUI was designed using java. The team provided the customer version 0.1 GUI during the customer meeting of milestone 1 and upgraded to version 0.2 according to the client feedback. During the meeting session of milestone 2, the team will submit the optimised version 0.2 to the customer.

The source code consists of two main parts: GUI implementation and robot motion control.
The finished version allows the robot to move forward, backwards, turn 90 degrees to the right/left and stop, detect boundaries and identify objects. The team will complete the source code for the internal tests before the final customer delivers. The final version will be called version 1.0 and will be delivered to the customer at the end of the project. For more information on configuration documents, see the Documentation Plan section.

Different functionalities of the system will be implemented using feature branches, developing branches should have descriptive names for the features, and they Should be merged into the master branch once finished implementing or featured demonstrations have concluded.
The GitHub repository structure used by the team, the hierarchy diagram is shown in the figure 2. The top-level directory is the source code of the document and contains four Level 2 directories: meeting, documentation, source code, use case.

The documentation includes three Level 2 directories: SRS (Software Requirements Specification), SPMP (Software Project Management Plan), SDD (Software Design Document)
The meeting consists of two levels of 3: the agenda and the minutes of the meeting.
The source code contains two level 3 directories: Src (source code).

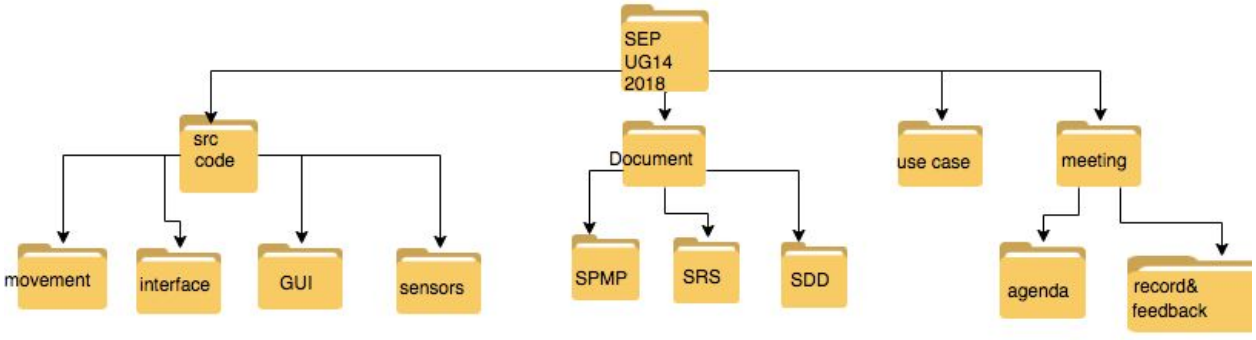


Figure 2 : Git Repository Structure

3.2 Documentation Plan

The deliverable documents in this project are as follows:

- Software Requirements Specification (SRS)
- Software Design Document (SDD)
- Software Project Management Plan (SPMP)

The SRS, SPMP and SDD documents are all deliverable for final submission and will therefore be required to be continuously updated. Templates for each of these have been provided via Google Documents and will be used as the basis for ongoing documentation development. These will facilitate improved collaboration and team contributions will greatly assist in the preparation of agendas for upcoming client and group meetings.

Deliverables will be discussed early in the project between during group meetings and a draft prepared for each document as required for the client review dates. Ongoing amendments will be made based on project progress as well as customer feedback during client meetings. Document changes are to be discussed and reviewed by the team before submission to the client to ensure document integrity. Any amendments will be updated within the document version history to ensure clear documentation of any changes. Each amendment will adhere to an incremental version history model with the initial document being entitled version 0.1 and progressing until the final release of the submission document entitled version 1.0.

3.3 Quality Assurance Plan

Development of the robot will be performed using an agile sprint method. Starting with the requirements analysis, the team creates user documents (SRS) that record each project requirement, physical part assembly requirements, and other non-functional requirements. For each demand analysis classification, each requirement has a corresponding implementation priority and easily identifiable number or task name. It is convenient for each team member to follow up on each process of the project and ensure that the priority portion with high priority is prioritized.

For System Design Documentation (SDD), each project requirement is updated according to user requirements, including some diagrams to aid analysis. The code section will be improved based on each newly generated test. This allows the robot development solution to be changed at any time based on test results and more flexible. And developers can easily set up the way they develop their choices based on how each part of the development is associated with a functional requirement.

Verification process

1. Create a new class or update an existing one.

2. Create test cases based on new class requirements.
3. Write code that implements functionality.
4. Test and verify.
5. Run on the robot and record the results.
6. Code review - repeat steps 4 and 5 if necessary.
7. Submit on the master branch.

It is vital that the code produced for the project is of high quality to ensure the prototype meets all of the client's requirements. Easy to read, well structured code also allows for easier modifications and additions should the prototype not perform as expected or additional requirements arise. A Naming and whitespace convention will be used across all code produced.

3.4 Code Review

All team members will be notified upon a request to push a new feature to the master branch, allowing individuals to each first access the code. This will further prompt collaboration between team members and ensure everyone is familiar with new code and functionality being added to the system.

3.5 Test Procedure

As the project is limited to one Lego Mindstorm block, the multiple team members developing independent code for various features will unfortunately inhibit testing efficiency, thus virtual simulators will be used whenever possible as an initial means of testing before using the physical prototype. As all team members will be familiar with the prototype code and expected behavior most of the testing will be performed using whitebox testing methodology. Effort will be made to incorporate blackbox testing when possible for major additions and changes to the prototype.

Each new feature will be tested extensively individually in an isolated environment before being added to the functioning code where it will then undergo further testing.

3.6 Testing and verification

3.6.1 Black box testing

Black box testing, also known as functional testing, data driven testing, or testing that gives a specification of the requirements, is testing focused on the functional requirements of the software.
When Using this testing methodology, our team regards the test object as a black box meaning the logic structure and characteristics of the program are not considered. The team only needs

to check whether the function of the program conforms to its functional description according to the requirements specification. Black box testing can give a better and more realistically look at the implementation of the system's functional requirements under testing similar to an end-users perspective. It plays an important role in all stages of software testing, such as unit testing, integration testing, system testing, and validation testing, especially in system-testing and validation-testing, its role can not be replaced by other testing methods.

For the pathfinder, Integration testing was used, using dependency injection.

3.6.2 White box testing

White box testing is often referred to as structural testing, logic driven testing, or testing based upon the internal structure of the program code. The team is required to have a deep software development site, proficient in the corresponding development language.

3.6.3 Static testing

Static testing is a static process of finding defects without executing the code of the object in question. Static testing compares program code and documentation with the requirements listed in the software requirements specification(SRS) document and identifies any unreasonable design in the program code or errors in the documentation.

3.6.4 Dynamic testing

Dynamic testing involves executing the program code for a measured object, inputting the pre-designed test case, checking whether the result of running the program code is different from the expected result designed in the test case, and to then determine whether the received result is consistent with the expected result. Thus it is the process of verifying the correctness, reliability and effectiveness of the program, and analyze the system's operating efficiency and robustness.

The dynamic test consists of four parts: designing test cases, executing test cases, analyzing and comparing output results, and finally outputting test reports.

3.7 Risk Management

3.7.1 Convention

This section identifies the risk management of the project. Risk is assessed taking into consideration severity and likelihood and and a classification assigned based on these ratings from high to low.

The chart is as follows.

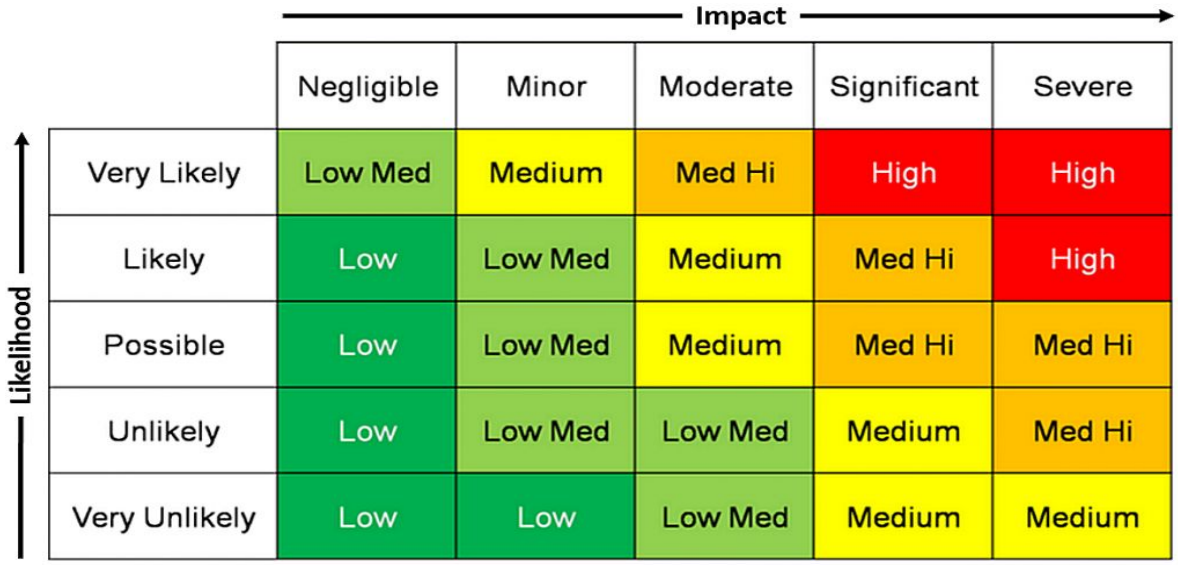
- **A low (green) risk** indicative of a negligible amount of risk, low impact on the overall project if risk is disregarded.

- **The low-medium (light green) risk** indicates a tolerable risk with no immediate effects but requires eventual resolution.

- **A medium (yellow) risk** indicates risk with potential negative effects that must be observed to avoid issues occuring. In the event of an incident, must be handled time-sensitively.

- **High Med (orange) risk** indicates a serious event that all team members must be notified about and a solution formulated to avoid long-term effects.

- **High (red) risk** indicates potentially catastrophic effects if risk is not eliminated prior to undertaking of project.



Likelihood	Negligible	Minor	Moderate	Significant	Severe
Very Likely	Low Med	Medium	Med Hi	High	High
Likely	Low	Low Med	Medium	Med Hi	High
Possible	Low	Low Med	Medium	Med Hi	Med Hi
Unlikely	Low	Low Med	Low Med	Medium	Med Hi
Very Unlikely	Low	Low	Low Med	Medium	Medium

Figure 3 : Risk analysis table

3.7.2 Risks

R01 Group member unavailable

Description Illness, travel or unforeseen circumstances may affect group availability.

Likelihood Possible

Severity Minor

Residual Risk Low Medium

Risk Indicator: Group members must be responsible for their own workload and either should plan in advance to redistribute work or actively try to balance workload and their commitments to limit issues.

Mitigation/control strategy Project work must be divided carefully and collaboration performed to assist one another to assure that deliverables are submitted on time or in advance to limit the potential for unforseen circumstances arising and adversely affecting time-sensitive content delivery.

R02 Specification change

Description Client or customer specifications may be subject to unforeseen change necessitating amendments to project requirements and thus development solutions.

Likelihood Likely

Severity Moderate

Residual Risk Medium

Risk indicator Communication difficulties between client and team during requirement elicitation or general correspondence.

Mitigation/control strategy All group members must be made aware of any changes at all times and alternative meeting time arranged with client to ensure that requirement change has been correctly understood.

R03 External delays

Description Any delay caused by factors outside the team such as change in deadline, client/team miscommunication, or unforseen hardware failure.

Likelihood Unlikely

Severity Significant

Residual Risk Medium

Risk indicator Miscommunication between the client and the project team resulting in unknown necessity to take action by the team.

Mitigation/control strategy Create Gantt charts to clearly show project time requirements, so that the likelihood of external factors suddenly causing team delays is reduced, and those responsible for risk should monitor any external factors that appear to be likely to cause delays.

R04 Time management

Description In completing the teamwork task, one or more members are behind or miss the deadline.

Likelihood Likely

Severity Significant

Residual Risk Medium High

Risk Indicator Lagging behind the original schedule on the Gantt chart.

Mitigation/Control Strategies Always pay attention to Gantt charts or trello, and continue to communicate with members and help each other where needed. Members should frequently check the status of the Gantt chart and the trello task and communicate it to the team members. If a team member may miss a deadline, they should actively seek additional resources for the task.

R05 Robot damage

Description Robots may experience irreparable physical damage due to lost parts, damaged collisions or electrical failure.

Likelihood Very likely

Severity Significant

Residual Risk High

Risk Indicator The functionality of the robot is compromised and results in a clear suffering in performance.

Mitigation/Control Strategies Ensure there are limiting factors in place during preliminary testing to ensure damage does not occur while reliability of core functionality cannot be assured. Such factors could include reduced motor speed, limiting angles, and reduced test run times.

R06 Data loss

Description Data loss (covering code, overwriting documents, corruption, etc.) may occur in all project-related documents and data, thereby resetting the project robot and redoing the lost work.

Likelihood Possible

Severity Severe

Residual Risk High

Risk Indicator Data and document corruption

Mitigation/Control Strategies Correct utilisation of repositories such as GitHub and google drive to maintain working copies of all produced documentation and development code.

3.7.3 Risk Monitoring and Review

All risks are monitored by all members. Weekly meetings should include risk results and all related updates. If any new risks or certain parameters are found to change, members should be notified and updated.

3.8 Meetings (internal and external)

3.8.1 Conference Calls (Start)

A conference call will be held on the first Tuesday of each sprint at 2pm consisting of all team member and will be no longer than 30 minutes. These calls aim to conclude the previous sprint and initialize the new one.

Each call will discuss the following:

- The outcome of each team members previous sprint task. Including any issues which may have been faced.
- Tasks which still need to be completed.
- Assigning of tasks to team members to be completed by the end of the new sprint.
- Any initial issues team members may expect to have.

Any other issues/updates will be addressed at the midway conference call or on Facebook Messenger.

3.8.2 Conference Calls (Midway)

Another conference call will be held on the second Tuesday of each sprint at 2pm consisting of all team members and will run for no longer than 15 minutes. These calls aim to act as a status update meeting.

Each call will discuss the following:

- What each team member has already completed.
- What each team member still needs to complete.
- If they are facing any issues and what can be done to assist them.

Any subsequent issues/updates team members may have will be addressed on Facebook Messenger.

3.8.3 Trello

Trello is the project management tool that will be used. It is a cloud-based software and thus can be accessed by all team members anywhere and allows for easy distribution and prioritization of tasks based upon importance and deadlines. The Trello board will be comprised of a list for each sprint containing cards outlining each task as well as a To Do list containing tasks yet to be complete for the current sprint. Each task card will be assigned one or more colours representing the type of task it is; these colours are defined in the Key list.

3.8.4 Client Meetings

Meetings will take place on a weekly basis every Wednesday at 4:10pm and will run for roughly 15 minutes. These meetings aim to deliver the client with meaningful information regarding the progress of their project as well as giving them the opportunity to provide useful feedback and inform the team of any changes in requirements.

Each client meeting will discuss the following:

- What tasks have been completed since the previous meeting.
- What tasks the team aims to complete prior to the next meeting.
- Any issues that have arisen with the current systems requirements.
- Address any issues or concerns the client may have.
- Any feedback the client wishes to give.

3.8.5 Face-to-Face Meetings

Face-to-face meetings will take place on a weekly basis prior to each workshop once coding and testing has begun. The day and time of these meetings have yet to be finalized. These meetings aim to initialize the workshop that follows.

Each face-to-face meeting will discuss the following:

- What tasks have been completed since the previous meeting.
- What tasks still need to be completed before the end of this sprint.
- What each team member wants to get out of the following workshop.

3.8.6 Workshops

The workshops give team members an opportunity to present code they have developed to the rest of the team and provides a way to visually update everyone on what has been done. Alongside this the time will also be spent working collaboratively to design code and resolve issues which may have arisen.

Each workshop aims to do the following:

- Update team members on the current code version.
- Collaboratively write code.
- Solve code related problems team members may be having.
- Test code on the prototype.

3.8.7 Facebook Messenger

Facebook messenger will be used as an informal means of communication between team members. The group chat will be available throughout the entire project and members can utilize it whenever they please.

Facebook messenger aims to provide the following:

- A means of informal communication between team members.
- A way for team members to receive feedback and help on their current task(s).
- A place to ask other team members questions regarding the project.
- A place to organize extra meetings and ensure everyone is aware of existing meeting times.

Appendix A

All	Task Name	Planned Start Date	Planned Finish Date	Planned Duration
1	**Group member allocation**	6/08/2018	7/08/2018	2 days
2	**Project initiation**	8/08/2018	26/10/2018	58 days
3	analyse project description	23/08/2018	23/08/2018	1 day
4	select process model	20/08/2018	23/08/2018	4 days
5	identify project environment	23/08/2018	30/08/2018	6 days
6	Client meeting week 3	8/08/2018	8/08/2018	1 day
7	**Review topics allocation**	8/08/2018	9/08/2018	2 days
8	**SPMP Draft**	10/08/2018	13/08/2018	2 days
9	overall description	10/08/2018	10/08/2018	1 day
10	**Group Sprint #1 (Week 3-4)**	15/08/2018	17/08/2018	3 days
11	Internal meeting week 4	14/08/2018	14/08/2018	1 day
12	Client meeting week 4	15/08/2018	15/08/2018	1 day
13	**SRS Draft**	17/08/2018	21/08/2018	3 days
14	non-functional requirements	17/08/2018	17/08/2018	1 day
15	external interface requiremer	17/08/2018	17/08/2018	1 day
16	Group meeting week 5	21/08/2018	21/08/2018	1 day
17	Client meeting week 5	22/08/2018	22/08/2018	1 day
18	**SDD Draft**	23/08/2018	28/08/2018	4 days
19	**Developing GUI**	21/08/2018	28/08/2018	6 days
20	analyse Java Gui developme	22/08/2018	22/08/2018	1 day
21	Group meeting week 6	28/08/2018	28/08/2018	1 day
22	Client meeting week 6	29/08/2018	29/08/2018	1 day
23	milestone 1 demo	30/08/2018	30/08/2018	1 day
24	**Group Sprint Report #2 (Wee**	30/08/2018	31/08/2018	2 days
25	Group meeting week 7	4/09/2018	4/09/2018	1 day
26	Client meeting week 7	5/09/2018	5/09/2018	1 day
27	Group meeting week 8	11/09/2018	11/09/2018	1 day
28	Client meeting week 8	12/09/2018	12/09/2018	1 day
29	**Milestone 1**	20/08/2018	13/09/2018	19 days
30	**Group Sprint Report #3 (Wee**	12/09/2018	14/09/2018	3 days
31	**Milestone 2**	13/09/2018	3/10/2018	15 days
32	Group meeting week 9	3/10/2018	3/10/2018	1 day
33	Client meeting week 9	3/10/2018	3/10/2018	1 day
34	**Group Sprint Report #4 (Wee**	10/10/2018	12/10/2018	3 days
35	Group meeting week 10	9/10/2018	9/10/2018	1 day
36	Client meeting week 10	10/10/2018	10/10/2018	1 day
37	**Mystery Presentation**	10/10/2018	10/10/2018	1 day
38	Group meeting week 11	16/10/2018	16/10/2018	1 day
39	Client meeting week 11	17/10/2018	17/10/2018	1 day
40	**SPMP Final Submission**	10/08/2018	19/10/2018	51 days
41	**SRS Final Submission**	23/08/2018	19/10/2018	42 days
42	**SDD Final Submission**	29/08/2018	19/10/2018	38 days
43	**Final Presentation + Demons**	19/10/2018	25/10/2018	5 days
44	Client meeting week 12	22/10/2018	22/10/2018	1 day
45	**Group Sprint Report #5 (Wee**	25/10/2018	26/10/2018	2 days
46				

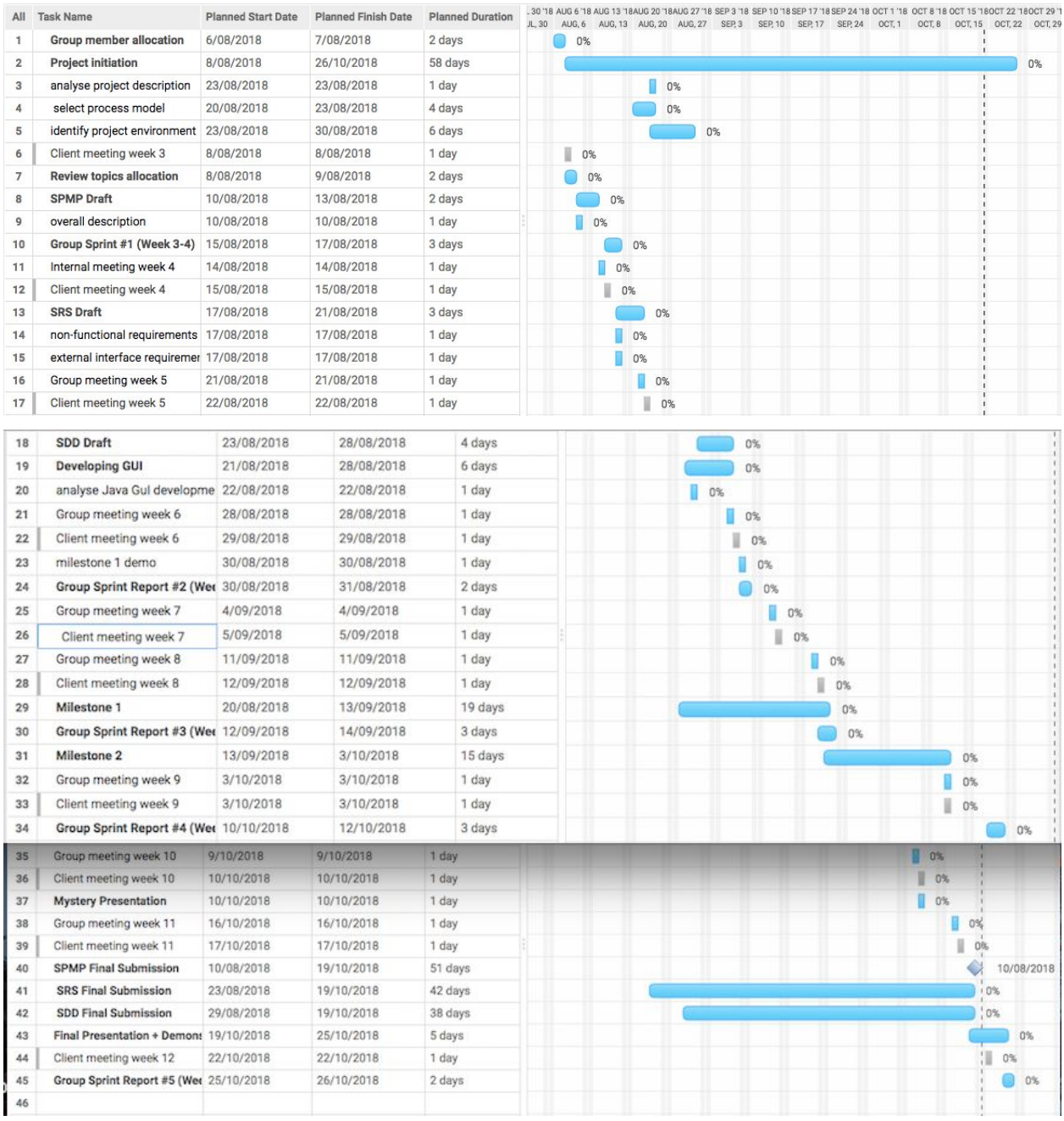
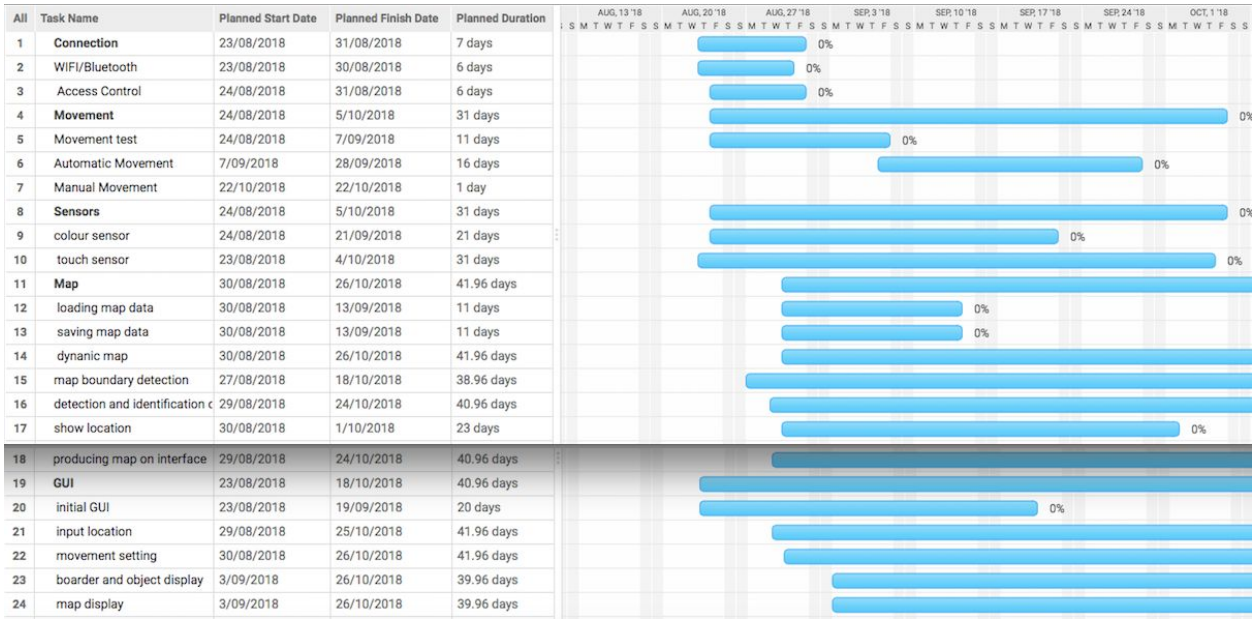


Figure 4.1: Project Gantt Chart



All	Task Name	Planned Start Date	Planned Finish Date	Planned Duration
1	**Connection**	23/08/2018	31/08/2018	7 days
2	WIFI/Bluetooth	23/08/2018	30/08/2018	6 days
3	Access Control	24/08/2018	31/08/2018	6 days
4	**Movement**	24/08/2018	5/10/2018	31 days
5	Movement test	24/08/2018	7/09/2018	11 days
6	Automatic Movement	7/09/2018	28/09/2018	16 days
7	Manual Movement	22/10/2018	22/10/2018	1 day
8	**Sensors**	24/08/2018	5/10/2018	31 days
9	colour sensor	24/08/2018	21/09/2018	21 days
10	touch sensor	23/08/2018	4/10/2018	31 days
11	**Map**	30/08/2018	26/10/2018	41.96 days
12	loading map data	30/08/2018	13/09/2018	11 days
13	saving map data	30/08/2018	13/09/2018	11 days
14	dynanic map	30/08/2018	26/10/2018	41.96 days
15	map boundary detection	27/08/2018	18/10/2018	38.96 days
16	detection and identification c	29/08/2018	24/10/2018	40.96 days
17	show location	30/08/2018	1/10/2018	23 days
18	producing map on interface	29/08/2018	24/10/2018	40.96 days
19	**GUI**	23/08/2018	18/10/2018	40.96 days
20	initial GUI	23/08/2018	19/09/2018	20 days
21	input location	29/08/2018	25/10/2018	41.96 days
22	movement setting	30/08/2018	26/10/2018	41.96 days
23	boarder and object display	3/09/2018	26/10/2018	39.96 days
24	map display	3/09/2018	26/10/2018	39.96 days

Figure 4.2: Project Gantt Chart (continued)