# Software Engineering and Project

## Software Project Management Plan

Team PG09

Congwei Bai
Anna Dai
Zhengjian Li
Li Qiu
Ruoxi Sun
Xinyi Xu
Zheng Xu

Version 1.0

14/08/2018

# Table of Contents

# Version History

| Date | Version | Changer | Description |
|---|---|---|---|
| 12/8/2018 | 0.1.1 | Xinyi Xu | Initial the Introduction section. |
| 12/8/2018 | 0.1.2 | Zheng Xu | Initial the section Process model and Overview. |
| 13/8/2018 | 0.1.3 | Ruoxi Sun | Added Agile model and project time schedule in Process Model section. Added detailed overview of events and activities during each phase of project in Overview section. |
| 13/8/2018 | 0.1.4 | Zheng Xu, Ruoxi Sun, Li Qiu, Anna Dai, Xinyi Xu, Zhengjian Li | Drafted activities in Supporting Plan:<br>• “Naming convention”, "Task estimation" by Zheng Xu;<br>• "Pair programming, Meetings" by Ruoxi Sun;<br>• "Versioning", "Task Tracking" by Li Qiu;<br>• "Documentation", "Git configuration", "Configuration management" by Anna Dai;<br>• "Risk management" by Anna Dai & Xinyi Xu;<br>• "Coding conventions", "Quality assurance" by Zhengjian Li |
| 13/8/2018 | 0.1.5 | Ruoxi Sun, Xinyi Xu | Modified Introduction section"<br>• “Intended audience”, “References”, and “Glossary” by Ruoxi Sun;<br>• "Scope" by Xinyi Xu |
| 14/8/2018 | 1.0 | Ruoxi Sun | Modify, update, and release version 1.0. |
| 14/9/2018 | 1.1 | Li Qiu | Update “milestone 1&2” to section 2.1 process model. Modify the scope in introduction. |
| 14/10/2018 | 1.2 | Li Qiu | Update “3.3 git flow control” and “3.7.2 Testing Methodology”. |
| 21/10/2018 | 2.0 | Li Qiu | Modify, update, and release version 2.0. |

# 1 Introduction

## 1.1 Scope

The main purpose and scope of this project are to develop a rescue robot equipped with Lego Mindstorm EV3 kit, with the functions of direction detecting and navigation, using color and distance sensors. The robot should be able to search a two-story building to locate all the survivors. More importantly, a real-time position should be presented clearly to display the location of all the identified objectives. As a result, the UNDSRS, the client of this project, can perform the search and rescue mission efficiently.

The software development team is primarily in charge of the code producing for the EV3 kit which assists the robot to recognize the direction and find the right path to the trapped survivors, with the support from on-broad sensors. Since the appearance design of this robot has already been stipulated, the code development has become the crucial task for this project.

## 1.2 Intended Audience

The intended audiences for this document are the project management team from University of Adelaide, and the client from the University Natural Disaster Search and Rescue Service (UNDSRS).

Firstly, the management team should be the main audience who should understand the processes in whole project, realize the crucial features of this project and organize the communication between different departments to perform the management smoothly.

Secondly, the client, UNDSRS, should also be involved in this management plan to supervise the project progress.

## 1.3 References

Client Specification, version 1.0

Group Project Specification, version 2.0

## 1.4 Glossary

| | |
|---|---|
| UNDSRS | University Natural Disaster Search and Rescue Service |
| SPMP | Software Project Management Plan |
| SRS | Software Requirement Specification |
| SDD | Software Design Document |

# 2 Process

## 2.1 Process Model

Considering the following reasons, an Agile process model that implements the incremental development methods will be applied in this project. With this model, the project team will make new versions released frequently with rapid reflections.

- There is only a high-level system specification of this project, which is to develop a prototype of the unmanned vehicle to assist in search and rescue operations, using the Lego Mindstorm EV3 kit. According to the client requirements, there is no limits on the features to be developed and neither the working environment.
- The system will be developed in series of versions. The client will be closely involved throughout the development process and help project team to specify and evaluate incrementally released version. The incrementally deliverables will be created and made available to client every two weeks.
- The system requirements are expected to be changed during developing and the system design should be agile to these changes.
- The project team is a small co-located team with effective communication, both formally and informally.

In the perspectives of users/clients, they can evaluate the project in each increment, even in early versions, to verify if their requirements have been delivered, instead of reading documents. The next increment may or may not be changed according to the feedbacks from the clients. It is possible to add new requirements in later increments as well.

From the view of developers, the cost is relatively lower than other models to alter the software in the development with the feedbacks. It is feasible to break up the task into each small interleaved increment assigned to group members.

There are several sizes of time frames by using agile method. The intervals of releasing new version and seeking for verification of functionality from the clients are about two weeks while the software could be build a few times a day. The sprint cycle is corresponded to the releasing cycles which are roughly two weeks. ‘Stand-up’ meetings are held every day to share the progress, problems and plans.

The Agile model used in this project combines the Scrum approach, which focuses on iterative development managing, with the Extreme Programming Release Cycle (XP), which is a more technical agile approach.

As shown in Figure 1 below, there are three main phases in this project management framework. The first phase is the Planning Phase. In this phase, the project team will figure out the general objectives for the project and work on the software architecture. The following phase is the Development Phase, which is consists of a series of Scrum sprint cycles. In each sprint cycle, an increment of this project will be developed. At last, the project will be finally delivered in the Closing Phase after final evaluation and receiving by client. An overview of detailed activities in each phase is described in the next section.

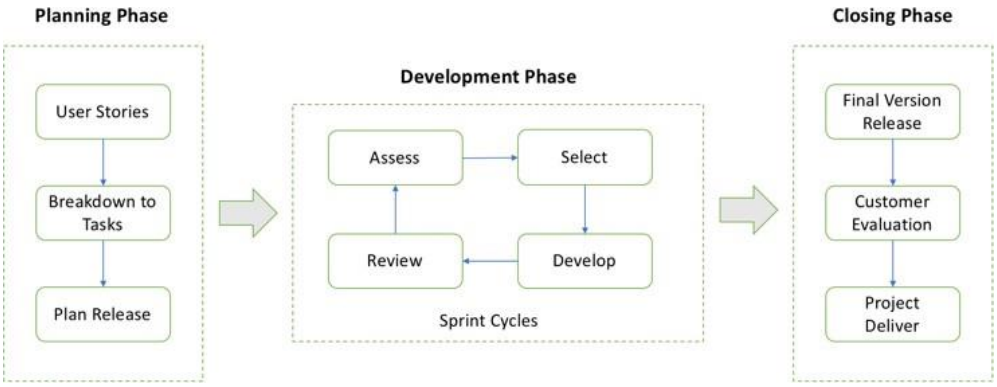


Figure 1 The Processing Model

According to the requirements in Group Project Specification, the project was kicked-off on 6th Aug 2018 and will be closed by 26th Oct 2018. Notice that there will be a two-week mid-break after week 8, which makes the total duration of this project to 10 weeks. For each two week, there will be a sprint cycle as a planning unit. The project team will deliver an increment development, which will then be reviewed and presented to the stakeholder, the client. The time frame, tasks and main milestones of this project is planned as following.

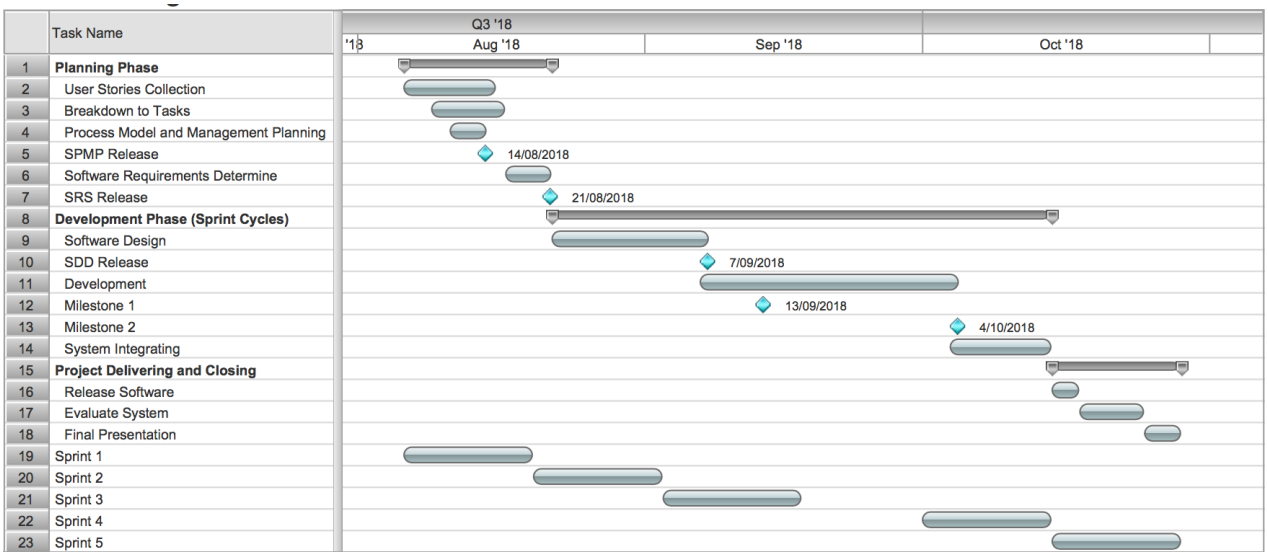


Figure 2 Project Schedule

## 2.2 Overview

As described in Figure 1 above, there are multiple events in each phase of project. Here is an overview of the activities that the project team aims to perform throughout the project lifecycle.

In Planning Phase:

- User stories collection

The user requirements will be described as user stories/scenarios, which will be prioritized by the client. The user requirements will be collected from the documents (Client Specification and Group Project Specification) provided by client and also from the client meeting in week 3-5.

- Breakdown user stories to tasks

After the user requirements are collected, the project team will assess the effort and resource of each scenario and breakdown the user stories to development tasks. If new requirements released by client during the development phase, the project team will repeat the breakdown activities and plan new incremental development tasks in next sprint cycle.

- Plan release

After the user stories breakdown, the project plan and the software requirement specification documents will be reviewed by the whole project team, including the client, and then released as a product backlog, which lists the work to be done on the project. The plan and requirement specification will be updated during the development phase according to new changes.

In development phase:

- Sprint cycle

The length of a sprint cycle in this project is set to 2 weeks. With each sprint cycle, the project team will develop an increment of the system. A cycle of sprint consists of four sub phases, the Assess, the Select, the Develop, and the Review.

- Assess

During assessment phase of sprint, the project plan and SRS document will be reviewed by the project team. The priorities and risks of tasks will be estimated in the weekly client meeting. The client will be closely involved in this assess process and new requirements and tasks could be introduced at the beginning of each sprint cycle.

- Select

After the assess phase, the project team will work together with the client to determine which feature or function will be developed in the sprint. The task selection process should consider the priorities and risks of each tasks, which are determined in the assess phase, previously.

- Develop

The develop phase of sprint consists of multiple activities, including not only code implementation, but also the testing and integrating activities.

All project team members should attend short daily "stand up" meetings to review the project status of each day. All team members should share information, present their progress, discuss about the issues and plan for the next day activities. The stand-up meeting is compulsory to each team member, however, if a team member cannot join the meeting, the project team could have an online video/audio meeting. One of the project team members will act as the Scrum master to arrange this daily meeting, track the project progress and schedule, record decisions, and communicate with external resources, such as the client.

During the develop phase, a test-driven approach will be applied. Test cases should be determined before code implementation for each task. There could be multiple testing

cases designed for one task. The client will play an important role in the testing process, which is to help develop acceptance criteria of the testing cases. All new developed codes should be validated to ensure the client requirements are met. The new codes should be integrated into the system only after passing all the tests.

- Review

At the end of each sprint cycle, the sprint deliverables will be reviewed and presented to the client in the weekly client meeting. The issues and new requirements will be discussed and updated in the beginning of next sprint cycle.

In closing phase:

- Final version software release

The final version software, together with the hardware, will be released through GitHub after two milestones achieved and a system level testing completed. The final version software should be easy-to-use by client, bug-free, and ensure that all client requirements are met.

- Customer evaluation

After receiving the final version software and hardware, the project team will cooperate with the client to validate and evaluate all the features and functionalities of the product. If there are new issues/bugs found or unmet requirements, the project team will assess the severity of the problem and the time/workload required for remediation. If the problem has a small impact on the project results and cannot be fixed in time, the project team will negotiate the rationale with client.

- Project deliver

As the final events defined in the project, the project deliver process will close the project and present the final product to the client, including the hardware, the software, the instruction documents, and also the project closing report.

At the beginning of the project, the project team will set up the git configuration for later project managing purpose. If there is any authorisation problem, the team will report it to the client. Any technical problems should be solved by the end of the first week.

The team will make a testing procedure for the testing use. The weekly quality assurance plan will be based on the procedure. If there are any testing objectives missing confirmed during the development, they will be amended to the procedure immediately.

In making naming conventions, the team will make a set of rules for naming variables, functions, and other elements in the code and documentations. Similarly, the coding conventions are for regulating the programming style. If there is no proper rule for donating some entities or styles, the team will hold a short informal meeting to make the complement and make it formal in the next convention.

On a daily basis, the team will perform task tracking, coding conventions, stand up meetings, and code review.

Each team member should record their works, problems and plans to be reported on the

stand-up meetings. The task tracking should be finished before the meeting for the summary.

In the stand-up meeting, each team member should report their task tracking and be reviewed by others. At the end of the meeting, a summary of team's work will be formed to be documented. The meeting will discuss how to eliminate the obstructions.

On a weekly basis, the team will do task estimation and assignment, client meetings and documentation.

The team will estimate the workload of the tasks of the following days and assign each part to group members evenly. There will be some margins as overestimation for the working hours in case some parts of the tasks are underestimated. At the end of the week, all team members should report the actual working hours compared with the estimation as a reference for the following weeks.

The client meetings will be held every Wednesday in order to review documents and report the progress of the project to the client. The feedback from the client will be added to the later increments. If there is any progress that the client thinks is behind the plan it will be prime task to be accomplished.

Documentations will be drafted and finalised on a weekly basis including group and individual reports. Every team member will have to finish a part of the documents and combine them in a final draft, two days before the due day. There will be an additional meeting if any documentation is not finished one day before the due day.

On a fortnight's basis, there will be versioning, and code review meetings.

Versioning corresponds to the release of the system. It depends on the degrees of the change of the software.

Code review meetings will be held before release to check if the code follows the conventions to ensure a high-quality code. If there are any defects in the code, they will be fixed in the meeting.

- Milestone 1: Basic feature implementation – navigation, positioning, survivor identifying and obstacle avoiding.
    - Due Date 13/9/2018
    - Description:
        - navigate automatically
        - avoid collision with obstacles and boundary
        - find and identify survivors
        - display the vehicle position and the survivor's position on LCD

- Milestone 2: improved feature implementation
    - Due Date 4/10/2018
    - Description:
        - map building
        - travel in grids
        - searching algorithm

- GUI
- threads

# 3 Supporting Plan

## 3.1 Configuration management

Configuration management is a process for ensuring consistency among physical and logical assets in an operational environment. For this project, the Git repository will be used as version control system to organize and control the development process.

- Configuration items
  - Design Documents
  - Software Requirement specification
  - Source code
  - Test cases
- Configuration roles
  - Configuration Manager
  - Configuration Item owner
  - Configuration Auditor
  - Support Manager
- Change control
  - Begin with a change request, group members decide to make or reject the request for change

## 3.2 Git Configuration

This project will use Git repository as a source code control system. There will be two kinds of branches: main branches and supporting branches.

The main repository holds master branch where the source code of HEAD always reflects a production-ready. There are two different types of supporting branches will be used in this project: feature branches and bug branches. These supporting branches are used to aid the parallel development between team members, ease tracking of features, and to assist in fixing problems. Once supporting branches are complete, at least two team members should review the code before code merging back to master branch.

## 3.3 Git flow control

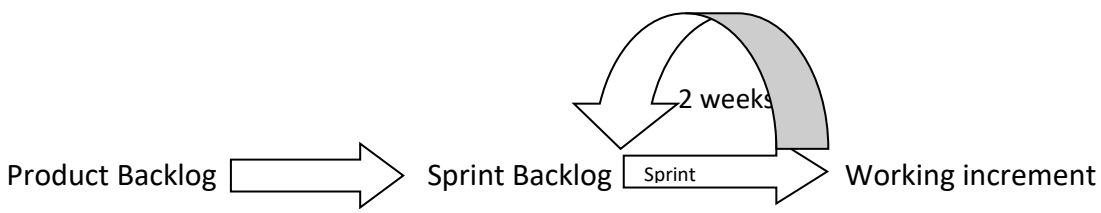


Figure 3 Scrum flow

The first process is product backlog which is the list of work items, that need our team to recognize the what is important to stakeholders. Trello is used for our team sprint. The group meeting is held every week, and the tasks are allocated in the meeting.
There are several branches in our team git. After each sprint, the branch shall be merged to master branch. When the main features are implemented, the project will be released with different version name such as V0.1. The final version shall be released to GitHub as version 1.0 after testing all features.

## 3.4 Documentation Management

Online collaborative tool Google Doc will be used for all group documentation. The deliverable document list is defined as below in Table 1.

Table 1 Documentation List

| Documentations | Releasing Date | Owner | Reviewer |
|---|---|---|---|
| Software Process and Management Plan (SPMP) | 14th Aug, 2018 | Project team | Project team |
| Software Requirement Specification (SRS) | 21st Aug, 2018 | Project team | Project team |
| Software Design Document (SDD) | 7th Sep, 2018 | Project team | Project team |
| Sprint report | At the end of each sprint cycle | Project team | Client |
| Meeting agenda & minutes | For each client meeting | Scrum master | Project team |

## 3.5 Versioning

The Versioning refers to the method of setting the unique version numbers or names. The sequence-base identifiers will be used in the project. Each updated software will be assigned a unique sequence number identifier.

The approach used in the project versioning is:

*major.minor(.build)*

The official version number is “1.0”. The version number less than “1.0” is a beta version, which means there are still some major errors, not launched officially.

When the new version is released, the major, minor or build version shall be updated. If there is a huge update, the major version number shall be increased. If there is an update, but it

does not update the major, the minor version number shall be increased. If the update changes a little, such as bug fixing, the build version number shall be updated. The following is an example:

0.1→0.1.1→0.1.2→0.2→0.2.1→0.3→1.0

In above example, the 0.1 to 0.1.1, 0.1.1 to 0.1.2 and 0.2 to 0.2.1 are build updates; the 0.1.2 to 0.2 and 0.2.1 to 0.3 are minor updates; the 0.3 to 1.0 is a major update.

## 3.6 Risk management

Table 2 below shows the risks which can be encountered during the project execution and provides required actions to mitigate those issue in best possible way.

The likelihood and consequence of risks are assessed (1-Low risk, 5-High risk). All the risks listed below are considered as major risks and a proper response or action must be planned to reduce project risk.

Table 2 Risk Quantitation

| Risks | Description | Likelihood | Consequence | Action |
|---|---|---|---|---|
| Prototype rejected | The prototype may be rejected by the client. | 2 | 3 | Fully communicate with the client about product requirements. Report to client in case of problems found. |
| Member absent or quit | Team member may get sick, injured which will cause them take work day off. The high-intensity work may cause a member withdraws the course. | 1 | 4 | Pair programming. Development log. |
| Requirements changing | The client may continually changing or make late changing about project requirements and we cannot managing the change properly. | 2 | 3 | Communicate with client before each sprint cycle starts. Let client help to review the sprint plan and determine the priorities of tasks. |
| Misunderstanding/Unclear of project | Team misunderstanding or unclear the project scope, objectives and client requirements. | 1 | 2 | Team review on the key documents. Daily stand-up meeting. |
| Unrealistic time estimates | Failure to estimate the time consuming for coding and building the robot. | 2 | 4 | Team review on the sprint plan. Involve client at the beginning of each sprint cycle. |

| Risks | Description | Likelihood | Consequen | Action |
|---|---|---|---|---|
| Lack of knowledge | The project involves the use of new technology and require high level of technical complexity. | 4 | 1 | Plan learning time in workload. Pair programming. Search online for external reference resource. |
| Unstable performance and low accuracy of the sensors | The sensitivity of the sensors will greatly affect the performance of the robot, so the sensors must be well selected. | 1 | 3 | Communicate with client and looking for external help for sensor changing in case the sensor does not perform well. |
| The physical damage to the robot during the process | The hazardous environment that the robots have to work in may leads to some unexpected damages to the machine which may stop the robots from working. | 2 | 4 | Double check the working environment and control the hazardous before testing. Looking for external help for robot kit repairing and parts replacement. |

## 3.7 Quality Assurance

The quality assurance standard used in the project will follow a quality standard document. This document should be appropriately modified as required during project propulsion.

### 3.7.1 Code and Document standards

In order to ensure that the code and documentation meet the standards, a document of the quality standards needs to be drafted by one of the team members, and all members should refer to this document and to discuss if some details should be modified. Once the quality standards completed, all team members should strictly follow the above criteria to check the code and documentation they wrote, and all team members are responsible for alerting other team members when they find that the code and documents written by them do not meet the above standards.

### 3.7.2 Testing Methodology

The testing methodology shall be used in the project are black box, white box and unit test.

- Black Box Testing used in our project is from the users' aspects to test if each function is working properly.
  This testing methodology focuses on testing the output without considering the product internal struct. It attempts to detect the following issues:
  - Input and output errors
  - functions incorrect or missing
  - performance or requirements if meets the standards

- White Box Testing is used to test the internal structure or logic. When use this testing methodology, the tester shall check the product internal action whether it is working properly according to the requirements.
- Unit test used in the project is to test each class of JAVA file and each window or menu of GUI. Each test function shall be created for each JAVA function. In addition, the unit test shall be used in a very early stage of the software developing process. Each test function provides support the subsequent software developing.

## 3.8 Task Estimation

Task estimation shall be done at the beginning of every week in the stand-up meeting. The team will estimate the workload, work hours, and possible obstructions. Each member shall pass a motion to the final estimation. The team should break up the task according to the difficulties and urgency of the tasks, and abilities and schedules of team members. Roughly even tasks will be assigned to each member.

The task should be slightly overestimated rather than underestimated. There should be buffer zone to prevent impact from accidents. If any unexpected incident results in the delay of the tasks, the team should arrange an extraordinary meeting to re-estimate and re-assign the task.

Before the estimation, the team shall measure the accuracy of the last estimation by calculating the gaps between actual efforts and estimated efforts. The later estimation shall refer to previous results for a better outcome.

## 3.9 Task Tracking

Task tracking improves project management and team performance. The Trello which is a task management software will be used in the project. The project is divided into several tasks using Trello. Each team member responsible for some part of the task. It helps them to manage their time and improve their work more effective.

In the Trello boards, three main approaches are shown: “To Do”, “Doing” and “Done”. Current work to do to complete the task is allocated in “To Do”. The work being done is allocated in “Doing”. After team members completing their job, the task already done will be placed in “Done”. In addition, a list called “Backlog” is shown, that is a list to place the work may be done. After team discussion, some work may be allocated in “To Do” list.

## 3.10 Coding Conventions

The coding conventions are a set of guidelines for Java programming language using in Lego Mindstorm EV3 kit that recommends programming style and structure for each aspect of coding work.

The purpose of the coding conventions is to help improve the readability of the source code which may be from different team members and to make the software maintenance uncomplicated.

The coding conventions will be drafted by one of the team members before starting coding to the robot, and all members should refer to the draft version and put forward reasonable suggestions for modification after draft completion. In details, the coding conventions should include file organization, naming conventions, declarations, white space style, indentation style, interface specification, comments requirements, programming principles, etc. Once the coding conventions are completed, each member needs to copy it and to review it before programming, and members should make sure that the code is written in strict compliance with the specifications defined in the coding convention document.

This document should be appropriately modified as required during project propulsion.

## 3.11 Naming conventions

The team will make a set of rules for naming variables, functions, and other elements in the code and documentations. The naming conventions for the entities in the code and documents will be camelCase which is a name composed of words joined without separators with each word's initial letter in upper case.

Table 3 Naming Conventions

| Identifier type | Rules of naming |
|---|---|
| Class | The class names shall be the full names of the class with first letter of every letter capitalised. |
| Functions | The function names shall start with a verb with first letter in lower case. |
| Variables | Variables shall be written in “lowerCamelCase” with a short meaningful name. Temporary variables could be named with one letter. |
| constants | Constants shall be all in capitalised characters separated by underscores. |

## 3.12 Pair Programming

To increase the engagement of each team member and support the collective ownership and responsibility for the system, the project team members will work in pairs on one specific programming task during the software developing. The pairs should be created dynamically, which means that the same pair will not always program together. Actually, all team members will work with each other during the develop phase in sprint cycles.

The code implemented by one team member will be reviewed by another team member in the pair. Furthermore, as the test-driven approach is applied, pair members should design and review the test case for each other before the programming activity starts.

## 3.13 Meetings

- Weekly Stand-up Meeting

  The weekly stand-up meeting is a compulsory meeting for all the project team

members, which is scheduled at 1:30pm to 2:00pm every Wednesday in Hub Centre. The team members will present his/her newly progress, discuss about the problems, and make plan for the following week. If a team member cannot attend the meeting locally, he/she can share the information via email before the meeting starts or have an online video/audio meeting using communication tools, such as WeChat.

- Daily Progress Briefing

  In daily progress briefing, each team member should report his/her progress of the day before via online communicate tools, such as email or WeChat. Team members will discuss and make decision on the next day working schedule and task arrangement in the briefing. If a task is completed, the corresponded team member should update the task status in Trello. The Scrum master will update the project schedule according to the decision made by the whole team.

- Weekly Client Meeting

  The client meeting is scheduled on every Wednesday 2:10 pm to 2:35 pm, in 4.62 IW. The project team will discuss the project schedule, assess and select tasks for next sprint cycle, and review/present the deliverables with client. The client will help project team to determine task priorities, introduce changings and update the user requirements. A meeting agenda will be sent out to each team member by the Scrum master before client meeting, and a meeting minutes will be recorded each time.

- Biweekly Sprint Review Meeting

  The Sprint Review meeting will be held during the review phase of each sprint cycle. All team member should attend this meeting to review the project incremental deliverables, testing results, code implementation, and prepare the presentation to the client. If problems cannot be fixed before the sprint cycle ends, the project team should present the issues with the client in the next client meeting and discuss the solution or rationale for the issue.

  Furthermore, the plan for next sprint cycle should be discussed in this meeting. A task selection pool should be determined and will be present to the client in the next client meetings