# Software Design Document

for

Archaeology Robot

Group 13

| | |
|---|---|
| Yufeng Bai 1600095 | |
| Dawei Geng 1219181 | |
| Jun Chen 1206265 | |
| Quang Khoi Nguyen 1187070 | |
| Shikai Li 1214223 | |
| Yunyao Yao 1203525 | |
| Yatong Zhou 1204471 | |

Version 1.31

October 17, 2012

# Contents

## Revision History

| Name | Date | Reason For Changes | Version |
|---|---|---|---|
| Dawei Geng | 03 Sep 2012 | Template | 0.1 |
| Yufeng Bai | 06 Sep 2012 | Chapter 7 | 0.2 |
| Yufeng Bai | 06 Sep 2012 | Chapter 8 | 0.3 |
| Yufeng Bai | 06 Sep 2012 | Chapter 9 | 0.4 |
| Yufeng Bai | 07 Sep 2012 | Fix the layout | 0.5 |
| Jun Chen | 08 Sep 2012 | Chapter 1 | 0.6 |
| Jun Chen | 08 Sep 2012 | Chapter 2 | 0.7 |
| Yufeng Bai | 09 Sep 2012 | Fix the error | 0.8 |
| Khoi Nguyen | 10 Sep 2012 | Chapter 4,5 | 0.9 |
| Jun Chen | 11 Sep 2012 | Chapter 3.1,3.2 | 1.0 |
| Yufeng Bai | 12 Sep 2012 | Fix grammar and spelling errors | 1.1 |
| Yatong Zhou | 12 Sep 2012 | Chapter 6 and error fixing | 1.2 |
| Khoi Nguyen | 14 Sep 2012 | Chapter 4 review | 1.31 |
| Yufeng Bai | 10 October 2012 | fixing sdd document | fixing |
| Yufeng Bai | 10 October 2012 | fixing the pictures of add | fixing |
| Yufeng Bai | 16 Oct 2012 | fixing the SDD | fixing2 |

# Chapter 1

# Introduction

## 1.1 Purpose and Scope

### 1.1.1 Purpose

The purpose of making this Software Design Document (SDD) is to give the details of the design of the archaeology robot and its system, which is designed by our group (group 13). In this document, we will give an overall description of architectural design, system organisation and critical modules of the system. This document describes some general ideas of how do we design the system and how do we implement requirements into the system. This document will be given to the team members so that they can construct the system based on the requirements. Also this document will be used as a reference for further developments.

### 1.1.2 Scope

The document describes the host side, client side and some relevant details about the GUI. It is intended to give the developers and clients all specific details about how the system is constructed. This Software Design Document (SDD) is also used to give details of overall design of archaeology robot. It will contain nine parts : Introduction which will give an overall description of the design, System Overview which will show the design of the system, System Architecture and Components Design, Architectural Description which also includes alternatives and rationale, Data Design, Human Interface Design, Resource Estimates and Definitions, Acronyms and Abbreviations.

## 1.2 References

This Software Design Document (SDD) references these files below:

1. Software Requirements Document (SRS) : this document demonstrate all the requirements for this project.
2. Software Project Management Plan (SPMP) : this document shows the details of how do we manage our project.
3. UML files : the class diagram show how do we design and implement our system.

## 1.3 Overview

This Software Design Document will be organised in a way that describe System, Architectural Design, GUI Design and Resource Estimates specifically and logically.

In the beginning of the document, it will give an introduction of this document (Purpose and Scope) and System Overview. After that, the document will give some high level architectural designs of the whole system, which include System Architecture and Components Design, Architectural Description. In these parts, a explanation about the whole architecture and major component of the software and how they interact will be indicated. Then the document will move to the Data Design part, which demonstrates the Design Details of the system. After the Data Design, it will show the design of Human Interface and Resource Estimates. In Human Interface part, the major components of GUI and their functions will be demonstrated. In Resources Estimates, all resources which are used in the whole system will be provided. Lastly, there are some Definitions, Acronyms and Abbreviations at the end of the document.

This document will follow the developing stages and the template of the SDD is generated according to client's requirement.

## 1.4 Constraints

The follow restrictions, limitations, and constraints will affect the design and implementation of the system:

- The robot should be set up correctly before testing.
- The robot is forbidden to be damaged.
- The robot is not allowed to go out of the map.
- The robot is not allowed to push the wall and colliding is not permitted.
- The map should be set up correctly before testing
- The map is in fixed size (less than A1).
- Pixels will be the smallest component of a map, at 25mm by 25mm.
- There are some no-go zones on the map.
- The system must has save and load function.
- An acceptable latency is 300ms.
- There are some hidden walls on the map.
- The robot is only allowed to search the map by pixels on auto mode.
- 4 Pixels make up 1 zone on the map.
- The maximum speed of robot is 10 inches per second.
- The size of the wall is more than 3 pixels.

# Chapter 2

# System Overview

The main goal of the system is to let the robot is able to be used for survey an archaeological site containing the remnants of an ancient city under both automated and manual control. The robot is not allowed to be damaged during the whole working process. The hardware system is the robot who execute all detection, the software system is used to control the robot and implement information transmission between the Client and Robot. Generally, there are three mainly part for this system: the Client, the Robot and the communicator.

## 2.1 Robot

The robot is used to execute the detection of the archeology in one sized map. The map contains the objects like hidden walls and obstacles, the robot is required to detect all these objects when it searches the map. The robot has the light and ultrasonic sensor which are used to implement the detection operation.

## 2.2 Client

The client is installed on the PC and the function of the Client includes:

1. Using GUI to demonstrate the map, include all objects on the map and the robot states.
2. Using Navigation to implement auto and manual control, which is combined with GUI.
3. Parse the XML file to the data structure which contains the map information.

## 2.3 Communicator

The communicator is used to connect the Client with the Robot. It allows the Client to send the command to the robot and robot to send the searching information to the Client. In the whole system, the communicator is implemented by the Bluetooth.

## 2.4 Conclusion

In conclusion, the whole system of this project contains three parts:

- Robot is the main body to implement all detection operations. It is also required to return the feedback about map and immediate robot status
- Client is used to implement the controlling of the robot and demonstrate the robot status. The Client also takes responsibility to store(load) the map to(from) XmL.

- Communicator is used to connect the Client with Robot. All transmission of the information has to use the communication part.

With these three parts, the system will work properly and do the task based on the client's requirements and Project Description.

Chapter 3

# System Architecture and Components Design

## 3.1 Architectural Description

The archaeology robot system we make will have three major parts that work together to achieve the goal. These three parts are Client System, Robot System, blue-tooth Connection. The Robot System is mainly used to execute the detection, the Client System is the controlling centre which is used to control the robot and record the information from remote interface. The role of the blue-tooth Connection is to connect the Client System with Robot System. The System connection diagram display below:

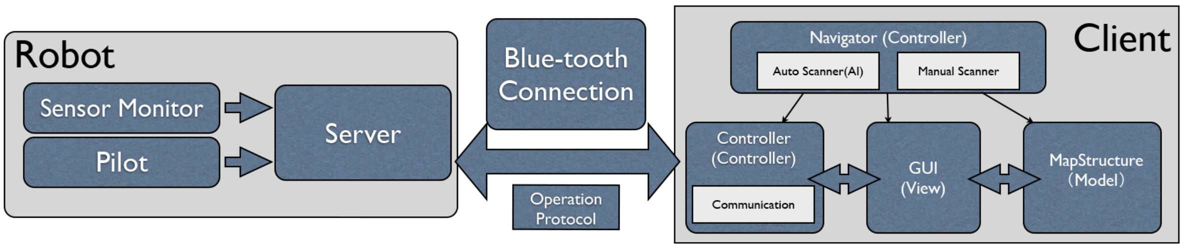

Figure 3.1: The system connection diagram

## 3.2 Component Decomposition Description

### 3.2.1 Robot System

The Robot System is installed in the archaeology robot. Basically the Robot System will be implemented all basic functionalities of the robot. Also it is required to respond to commands which are sent from the Client System. Moreover the Robot System should also include functionalities that can make the robot move according to the received command. There are three components in the Robot system:

**Server** is used to accept the commands from Client System and send the messages to the Client System. All the messages which will send to Client System need to packet here. All commands sending from Client System also need to unpacked in this server.

**Sensor Monitor** include light sensor and ultrasonic sensor. The light sensor is used to detect the hidden walls which will use different colour to mark on the map. The ultrasonic sensor is used to detect the obstacles on the map. As the safety priority principle, the robot is forbidden to collide the obstacle, the ultrasonic is the tool to protect the robot. In our design, when ultrasonic detects the obstacles, it will notify the robot, then robot will stop in a few distances front of the obstacle.

**Pilot** is used to control the movement of the robot and the detection of the sensors. It is also used to control the rotation of the robot by setting the speed of robot's wheels on both sides.

### 3.2.2 Client System:

The main purpose of the Client System is to implement remote controlling of robot. Following the Requirement Description, the robot is required to be controlled by users. It is necessary to build a control panel on PC, then the user is able to control the robot and check the robot status from PC. In our design, the Client System will follow the Model-View-Controller pattern. There are three main parts representing the three components of this pattern: GUI(View), MapStructure(Model), Controller and Navigator(Controller).

#### MapStructure and XML Documents(Model)

1. MapStructure: The Map is composed by pixels. The size of each pixel is 25mmx25mm. The robot search the map using pixel as the unit. The map also need to include the information of all object like hidden walls and obstacles. In addition, some related robot status also is stored in the Map Structure(the current position and the start position of the robot).

2. XML Documents: The map is stored in the XML file. The map information is able to be checked by GUI, and the GUI also can store the map information in XML file.

**GUI(View)** The GUI will contain three components which are:

1. Main GUI: is the graphic interface for the host system which is used to show to the user. The main frame of it has many small components which is able to implement different functions. It includes buttons to control the robot(the movement, speed and mode change), robot status(battery level, current speed and current position), system status(the bluetooth connection), save, load and create map(the size and position of the map and no go zones, load(save) the map from(to) database) and menu(the group introduction).

2. Map Panel: is a component of the GUI that shows the surveying map. It is used to display the location of the robot and draw the real map into it, which means the map panel will display and draw everything the robot explored. After the survey, the map viewer will save the map into the database if it receives a 'save' command from the host. The map panel also contains the icon of the map, different colour represents different object on the map, for example, the red colour represents the no-go zones, the black colour represents unexplored area.

3. Information Window: All operation for the GUI will display on the Information window, also the Information window shows the brief introduction of every group member.

#### Controller and Navigator(Controller)

1. Controller: The controller mainly takes responsibility to connect with the robot, the function of controller is similar with the Server, its main function is packeting the commands which send from Client System to Robot System and unpacketing the messages which receive from the Robot System.

2. Navigator: The Navigator has two component, the Auto Scanner control and the Manual Scanner control. The main function of Navigator is to make the algorithm to control the robot scan the map on auto and manual modes. On Auto mode, the Navigator will make the algorithm to let the robot to find path automatically and send all objects of the map to the Client System. On Manual Mode, the Robot is controlled by the user. The user is able to send the command to control the movement of the robot. The safety is priority for both mode, the Navigator is forbidden the user to execute some unsafe operation. For example, if user wants the robot to collide the obstacles or the user wants the robot to access the no go zones, the robot will not move follow the commands. The Navigator is combined with the GUI.

### 3.2.3 Blue-tooth Connection

The bluetooth is used to connecting the Client System with the Robot System. All the information transmission need to go through the bluetooth, the bluetooth is wireless so that it will not limit the move range of the robot.

## 3.3 Detailed Components Design Description

### 3.3.1 Client

#### GUI

1. Component Identifier: C-0
2. Purpose: GUI is a interaction between the host and the robot. GUI is able to send the client's command to the robot and get the feedback from the robot.
3. Function: It provides the GUI for the client, which is convenient for client to control the robot. For example, the client can control the movement, speed, and the searching mode of the robot. The GUI is also able to display the feedback from the searching of the robot, for example, the GUI can display the hidden walls or obstacles of the map. Basically, GUI takes responsibility to translate the client's command to robot instruction and translate the robot feedback to client message, then send between the robot and host.
4. Subordinates: LEGOGUI, CommandTranslator, About, MainControlThread, Operation, TextAreaPrintStream.
5. Dependencies: MapStructure, Navigation
6. Interfaces: Client
7. Data: None

#### MapStructure

1. Component Identifier: C-1
2. Purpose: The map structure is used to build the map for the robot. The robot need to search in a sized map. The map needs to display on the map.
3. Function: The map is made by pixels, the robot is required to search on the map and map also contains some information of the robot(the current and start position of the robot), as a result, the robot status and some relevant operations of robot also are required to connect with map.
4. Subordinate: Map, Operation, Pixel, RobotStatus

5. Dependencies: XMLDocuments
6. Interface: Client
7. Data: Contain the current and start position of the robot.

**Navigation**

1. Component Identifier: C-2
2. Purpose: The navigation is used to control the robot. Robot is required to search and find all objects on the map in auto and manual mode, it is necessary to use navigation to control the searching of the robot. The navigation also contains the algorithm to define scan trace of the robot.
3. Function: The function of the navigation is to control the movement and scan of the robot. When the robot is in auto mode, the robot is able to scan the map and find all objects automatically. When the robot is in manual mode, the robot is required to be controlled by client, client is able to give the command to the robot, then robot executes the operation according to the instruction from host. The pathfinder is a method to define how the robot scan the map.
4. Subordinate: AutoNavigation, ManualNavigation, PathFinder, RobotAutoScanner, RobotManualScanner, RobotNavigator
5. Dependencies: MapStructure
6. Interface:Client
7. Data: None

**XMLDocuments**

1. Component Identifier: C-3
2. Purpose: According to the client requirement, the map and the relevant information need to store in the XML file.
3. Function: The function of the XMLDocuments is to create the xml file to store the map information and load the map information from the xml file.
4. Subordinate: XMLDocuments, XMLReaderWriter
5. Dependencies: MapStructure
6. Interface: Client
7. Data: Contain the map information.

### 3.3.2 Robot

1. Component Identifier: C-4
2. Purpose: The Robot is required to do the relevant operation on the actual map, the Robot is used to implement the movement, detection, robot state monitoring and sensor monitoring on Auto mode and Manual mode. The robot is also required to handle some safety situation include it has to stop automatically in some circumstance.

3. Function: The Robot contains the methods to implement the basic operation of the robot. In our design, we use different value to represent different operation, for example, 0x000 represents the forward pressed, which is more convenient for us to implement all function. Since the map is made by pixels, the robot will move use pixel as unit. The robot side also includes the function to monitor the sensor. The robot has light sensor and ultrasonic sensor, light sensor is used to detect the hidden wall and ultrasonic is used to detect the obstacles. The robot is required to use ultrasonic sensor to find the obstacle and avoid to collide the obstacles. In the sensor design part, it is necessary to stop the robot immediately in some danger situation no matter which mode the robot is.

4. Subordinate: BTRobotDriver, Operation, Robot, SensorMonitor.

5. Dependencies: None

6. Interface: Robot

7. Data: the different data represents different operation of the robot.

### 3.3.3 Class Files

#### GUI-LEGOGUI

1. Component Identifier: C-5

2. Purpose: LEGOGUI is used to generate the main GUI.

3. Function: The LEGOGUI is used to build the main GUI, include the size and the layout of the GUI. The LEGOGUI defines the different panel of the GUI, the position of these panels and the size of these panels. This class also build the connection between the GUI and database.

4. Dependencies: TextAreaPrintStream, About, CommandTranslator

5. File: GUI

6. Data: Node

#### GUI-TextAreaPrintStream

1. Component Identifier: C-6

2. Purpose: GUI need to receive the input message

3. Function: This class is used to get the message from the input, write these message to GUI and then print all these messages out, which is a part of the GUI

4. Dependencies: None

5. File: GUI

6. Data: Receive the data from the input and write them to GUI, then print them out.

### 3.3.4 GUI-About

1. Component Identifier: C-7
2. Purpose: GUI is required to contain some Group introduction.
3. Function: This class is used to build a window to display some group information, for example, the Group name and the team members' names. The About class also has the contact method.
4. Dependencies: None
5. File: GUI
6. Data: the introduction message.

### 3.3.5 GUI-CommandTranslator

1. Component Identifier: C-8
2. Purpose: GUI works as a controller which connect the host with robot. It is required to translate the client's input to the command which is able to be read by GUI.
3. Function: Basically, different number represents different input command, for example, the number 1 represents the input command " f ", which is able to be read by GUI. The client can input the command by GUI.
4. Dependencies: Node
5. File: GUI
6. Data: the numbers which represent different operation.

### 3.3.6 GUI-MainCntrolThread

1. Component Identifier: C-9
2. Purpose: The GUI need to connect with the robot and control it.
3. Function: In our design, it is required to use bluetooth to connect the robot, the connection function is from lejos API, then the GUI is able to use InputStream and OutStream function to input the command to the robot, different number represents different command. When connection is successful, when client input different string, the robot will read the relevant number, then do the corresponding operation.
4. Dependencies: None
5. File: GUI
6. Data: the string as command input by client and the numbers which represent differnt command.

### 3.3.7 GUI, MapStructure, Navigation, Robot-Operations

1. Component Identifier: C-10
2. Purpose: Collect different number to represent different command
3. Function: Display the relevant numbers which is used for the system
4. Dependencies: None
5. File: GUI, MapStructure, Navigation, Robot
6. Data: the relevant numbers which represent different operation

### 3.3.8 MapStructure-Map

1. Component Identifier: C-11
2. Purpose: The GUI need to contain a map to demonstrate the searching of the robot.
3. Function: This class generates the map by pixel, it also demonstrate the position of the robot, which includes the current position and start position. In addition, this class defines the layout of the no-go zones.
4. Dependencies: Pixel
5. File: MapStructure
6. Data: the input data which is used to define the size of the map, and the position of the robot.

### 3.3.9 MapStructure-Pixel

1. Component Identifier: C-12
2. Purpose: Since the map is made by pixels, it is necessary to define the pixel.
3. Function: This class defined the specific parameters for each pixel, which contains the size of one pixel and the size of different object. For example, the smallest hidden wall's size is one pixels.
4. Dependencies: None
5. File: MapStructure
6. Data: the specific number represents the size of different object.

### 3.3.10 MapStructure-RobotStatus

1. Component Identifier: C-13
2. Purpose: Some robot information is required to store on the map.
3. Function: The class defines the specific location of the robot.
4. Dependencies: Pixel
5. File: MapStructure
6. Data: None

### 3.3.11 Navigation-AutoNavigation, RobotAutoScanner

1. Component Identifier: C-14
2. Purpose: Implement the auto mode of the robot, which is required robot to move and detect automatically on the premise of safety.
3. Function: Using PathFinder class to let the robot scan the whole map automatically, the robot need to record the current position of the robot and the path from start position to current position. The robot also is required to find the path from current position to nearest unexplored Pixel.
4. Dependencies: Map, PathFinder, Pixel, RobotNavigator
5. File: Navigaiton
6. Data: None

### 3.3.12 Navigation-ManualNavigation, RobotManualScanner

1. Component Identifier: C-15
2. Purpose: Implement the manual mode of the robot, which is required robot to move and detect by the controlling of the client via GUI on the premise of safety.
3. Function: The robot need to receive the commands from GUI. In our design, we will use one queue to receive the command from host machine, the scan method is also from PathFiner class. The different direction also use different value to represent.
4. Dependencies: Map, PathFinder, Pixel, RobotNavigator
5. File: Navigation
6. Data: None

### 3.3.13 Navigation-PathFinder

1. Component Identifier: C-16
2. Purpose: It is necessary to define one algorithm to let robot scan all pixels of the map.
3. Function: Firstly, Using Depth First Search to traverse all pixels in the map. The robot decides whether the robot finish the traverse or not according to the return value of the DFS. Second, Using PathFinder function to find the path from the current position to next position. Thirdly, Using findNearestUnexplorePixel function to find the nearest unexplored pixel when the 4 directions(east, west, north, south) of the current position have already explored. Finally, Using Robot Scanner class to decide when and how the robot moves.
4. Dependencies: Map, Pixel, RobotNavigator
5. File: Navigation
6. Data: None

### 3.3.14 Navigation-RobotNavigator

1. Component Identifier: C-17
2. Purpose: The robotNavigator is a class to implement robot and read all messages from map it is a unify class to let the robot detects following the design.
3. Function: The robotNavigator is able to control the movement of robot and detect all objects, and translate the GUI command to Robot instruction.
4. Dependencies: Map, Pixel, MainControlhread
5. File: Navigation
6. Data: the number which represents different command.

### 3.3.15 XMLDocuments-XMLDocument, XMLReaderWriter

1. Component Identifier: C-18
2. Purpose: According to the client requirement, the map and the relevant information need to store in the XML file.
3. Function: Define the method to create a xml file to save the new map, and another method to load the map from xml file.
4. Dependencies: Map
5. File: XMLDocuments
6. Data: None

### 3.3.16 Robot-BTRobotDriver

1. Component Identifier: C-19
2. Purpose: The unify class to implement all functions of the robot, include the bluetooth connection and the sensor monitoring.
3. Function: The function of this class includes connect the host machine with robot, and receive the command from the host machine. The robot is able to move following the command from host machine. All sensors' functions are also able to implement in this class.
4. Dependencies: BTConnection, Robot,
5. File: Robot
6. Data: the numbers which represent different command.

### 3.3.17 Robot-Robot

1. Component Identifier: C-20
2. Purpose: Collect all operation of the robot, include the movement and stop in manual and auto mode
3. Function: The Robot is required to control the robot move forward, backward, rotate and stop in auto and manual mode.

4. Dependencies: None
5. File: Robot
6. Data: None

### 3.3.18 Robot-sensorMonitor

1. Component Identifier: C-21
2. Purpose: It is necessary to build a Thread that monitor the ultrasonic sensor when moving to avoid near walls.
3. Function: This class is using sensor to detect all object from the map, this function contains when the robot is need to stop and the range of the monitoring.
4. Dependencies: Robot
5. File: Robot
6. Data: None

## 3.4 Architectural Alternatives

The other alternative design which is considered is letting the robot do the exploration without waiting command from the host. The idea is shelved because the robot's resource is not capable of performing the algorithm.

Another architecture we considered was the Object Oriented model. The java is a object oriented model, meanwhile, the program of the system mainly use java language. OOM is a effective way to use. This model is helpful to let every developer understand each other's work content and process. However, the robot is required to terminate at any time, as the result, this model is not work well effectively. That is the reason why we abandon this model.

## 3.5 Design Rationale

The design of the system is layer architecture. The model is chosen for its support of separation and independence. The development is spread across teams with each team being responsible for a different part. The use of interface allows smooth integration without risk of incompatible method calls. If there is change in the interface, only the adjacent layer is affected so change is localised.

Chapter 4

# Data Design

## 4.1 Database Description

The map is represented in XML format. It is created when the robot starts to explore.The map updates every time when the robot explores an area, detects a wall or a border of the survey area. The information which is stored in the array is translated and saved into XML. The XML file can be loaded as well. The detailed functionalities are as follow:

**Creating a map** The map is initialised to be a size of a grid, which is two by two. The size of the map is stored as digital number and the coordinates of the border are stored in an array list.

**Recording the features of the area**

- Clear Area: containing no features and is considered to be safe. The status and coordinate is saved if the area is explored.
- Wall Area: containing walls.The status and coordinate is saved if the area is explored.
- No-go-zone area: the robot is not permitted to enter the area. The status and coordinate is entered by the operator.

**Recording the position of the robot** The coordinates of current position are stored in an array list and are flagged as visited.

**Saving** The map which is explored so far can be saved.

**Loading** The previous map can be loaded,

## 4.2 Data Structures

The underlying data structure of the map database is a two dimensional array. Each element is the zone (grid)'s information, which is indexed by its coordinatting in the map. Each zone(grid) consisted of the information of its 4 sub-grids, which is stored in a two-by-two array

# Chapter 5

# Design Details

## 5.1 Class Diagrams

SEP_13_Class_Diagram.png

Figure 5.1: Class Diagram

The diagram above shows the package and class structures of the entire project. Note that the database is included the view section. The class model follows the classic MVC design pattern. Model is the controller part of the project, which controls the robot. GUI is the view part, displays the map in the run-time. Model is the robot itself mechanism, which is able to be accepted the command from the controller.

## 5.2 State Diagrams

SEP_13_RobotStateDiagram.jpg

Figure 5.2: State Diagram

The state diagram above shows the all possible status of the robot. The robot started with initial state, after connecting, it jumps to “Idle” state, which means doing nothing but waiting for the commands from the host. After initial the map and establish connection. It wait operator to chose control mode. It transfers to either manual or auto control state to explore the unmapped areas. When transfer to auto mode(whatever in start or in the middle), robot will checks whether the map is fully explored. If yes, it reaches the end state. If not, the robot will do more exploration follow searching algorithms, if any wall is detected in the process, robot will record current location and start wall detection state, it will keep check the wall on left side until on more wall is found, and then check right side. After detected the wall, robot will come back to auto exploration mode. When transfer to Manual mode state, the robot can be control by operator, but for safety reason, if any wall is detected or go into no go zone in manual control, the robot will still pose its movement in until get next instruction that is safe to move. User can switch mode, and terminates exploring at any time.

## 5.3 Use Case Diagrams

SEP_13_UseCase.jpg

Figure 5.3: Use Case Diagram

The use case diagram above shows all the cases that user can operate on the graphic user interface. Start with initial map, this allow user to set initial value of map like map height, weight and start coordinate. Also a serial of instruction to to set up robot. That include establish robot connection, set operator mode (User/Developer), set control mode(Auto/Manual) and set robot movement speed. After robot set up, user can control robot through the control panel to explore the map. In addition user can save map or load map in the process. User also can chose disconnect with the robot or terminate this exploration at any time.

## 5.4 Interaction Diagrams

SEP_13_Sequence_Diagram.png

Figure 5.4: Interaction Diagram

The above diagram demonstrates how control is passed during the simple operation of the user manually moving the robot forward.
The GUI calls the hosts move function, which will do the required safeguard checks and then forward it onto the client, who will return the distance moved.
Afterwards the host will update the map data structure, so when the GUI updates the robot's new position and map display.

Chapter 6

# Human Interface Design

## 6.1 Overview of the User Interface

The Graphical User Interface is used to communicate with the robot. The GUI is connected with robot, the client is able to control the robot by GUI. When the robot finishes its task, the client is allowed to switch off and disconnect with robot. In addition, the GUI demonstrates the status of the robot. The status includes the speed, the power and the location of the robot. The GUI also shows the obstacles, the "no-go" zone and hidden walls, which is able to be checked on the map window.

The GUI is able to demonstrate the follow functions for users:

- Save map to a XML file
- Load map from a XML file
- Change the control mode
- Demonstrate the Blue-tooth and Battery status
- Control the moving direction of the robot(i.e. forward, backward, rotating)
- Demonstrate the Icon of the map
- Demonstrate the Log Information
- Change the speed of the robot
- Demonstrate the coordinate of the robot

The GUI is displayed below:

GUI_Initial.jpg

**Figure 11: GUI**

Map.jpg

**Figure 12: GUI WITH MAP**

## 6.2 Detailed Design of the User Interface

### 6.2.1 Save map

The GUI includes a icon which is used to save the map from a XML file. When the client presses the File menu and choose the save option, there is a window jumping out, the client is able to choose the file where the client want to save.

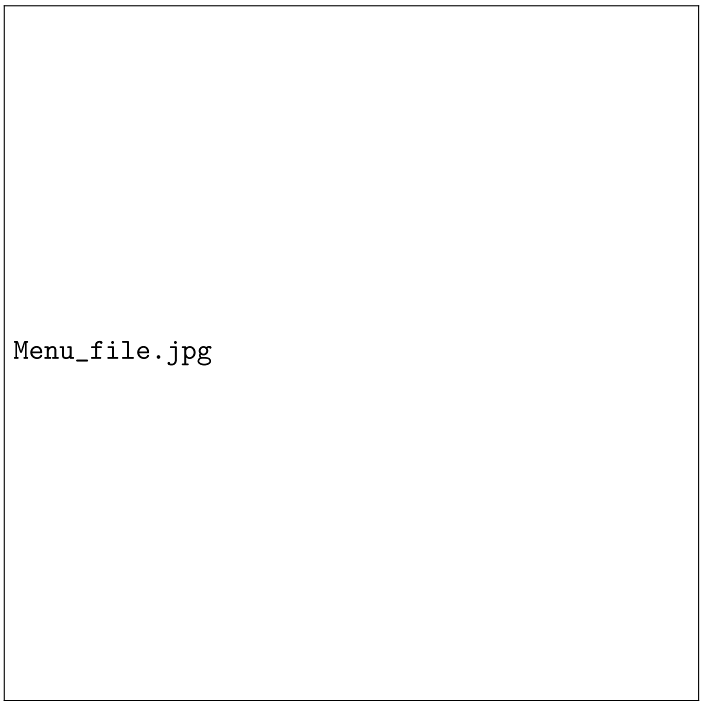

**Figure 13: SAVE MAP**

### 6.2.2 Load map

The GUI includes a icon which is used to load the map from a XML file. When the client presses that icon, there is a window jumping out, the client is able to choose the map from there.

**Figure 14: LOAD MAP Icon**

Menu_file.jpg

**Figure 15: Load MAP from menu bar**

LoadMap.jpg

**Figure 16: LOAD MAP**

### 6.2.3 Change the control mode

The client is allowed to change the control mode by pressing the mode button. When the client presses the Auto mode button, there is a widget jumping out. Then if the client presses “yes”, the robot changes to the Auto mode. The client is allowed to do the same operation to change to the Manual mode. The Log information display which mode it is currently.

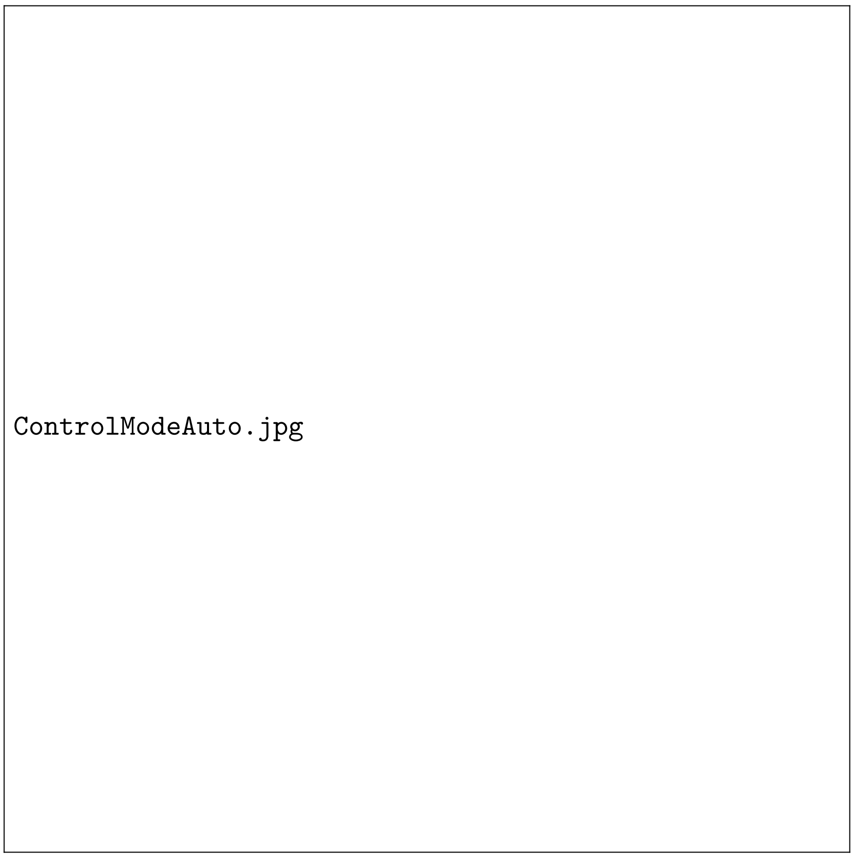

**Figure 17: Change Mode**

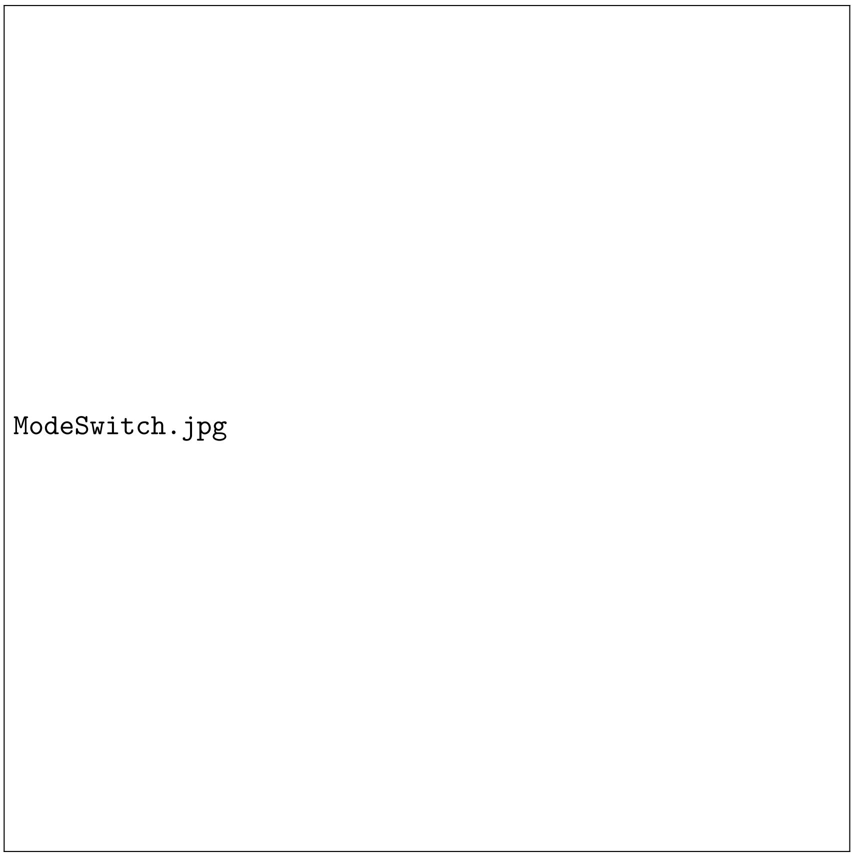

**Figure 18: Change Mode icon**

### 6.2.4 Demonstrate the Blue-tooth and battery status

The Blue-tooth and battery status are displayed on the GUI, if the Blue-tooth loses the connection, the log information board will demonstrate the lose information. The power of the battery is displayed by the percentage number and the colour of the battery bar will change along with the battery level.

**Figure 19: Blue-tooth and battery demonstration**

### 6.2.5 Control the moving direction of the robot

The robot is able to be controlled by GUI when the robot is changed to the Manual mode. The client is allowed to press the arrows to move the robot. The robot is able to move forward, backward and rotate(include rotate 90 angles and rotate 360 angles). The moving direction controlling only can be used in Manual mode.

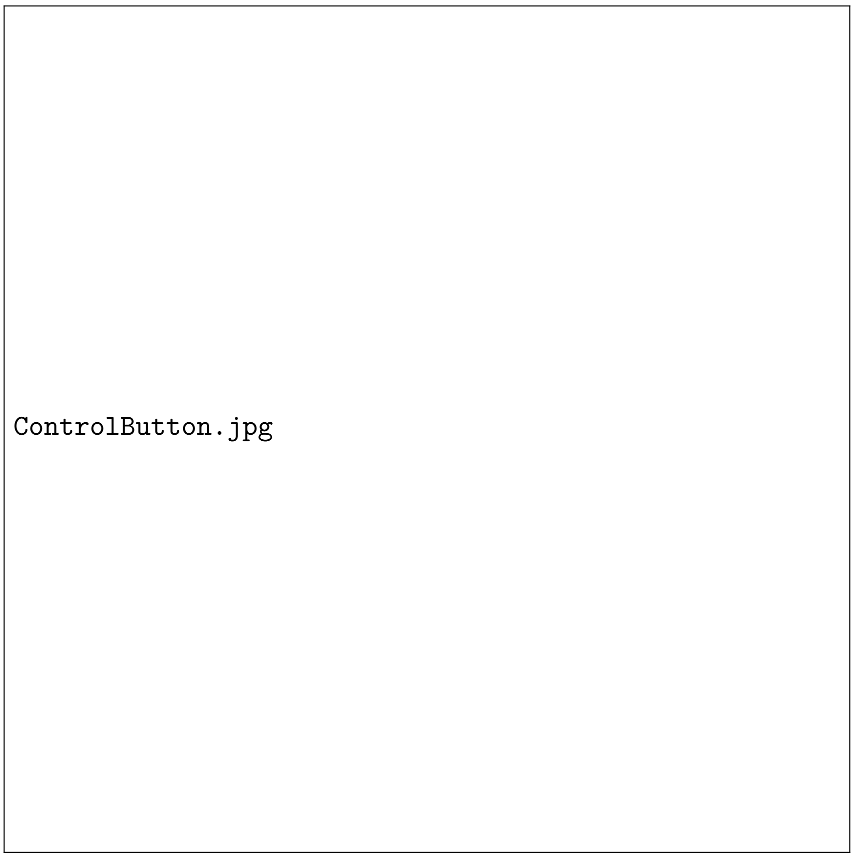

**Figure 20: Moving control**

### 6.2.6 Demonstrate the icon of the map

The robot and map information are required to be displayed on the GUI. It is necessary to use different icons to demonstrate the different map information.

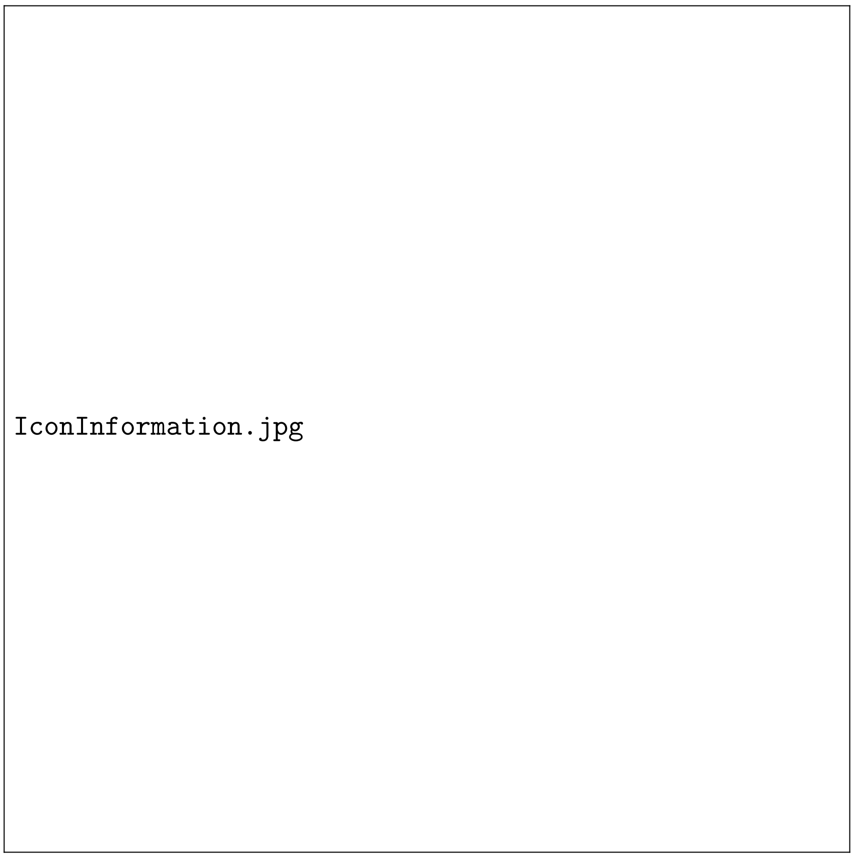

**Figure 21: Icon of the map**

### 6.2.7 Demonstrate information of the GUI on the log information board

The Log Information board displays the information of the GUI. For example, the Log Information board demonstrates the version and the developers of the robot. If the client changes the control mode, the board displays which mode it is now. The board also is required to indicate the connection information.

**Figure 22: The logging information**

### 6.2.8 Control the speed of the robot

The GUI includes a speed bar and one specific number to control the speed of the robot. The client is allowed to change the speed of the robot by change the number of the bar. The larger number means the faster the robot is. If the number is zero, it means the robot stops immediately.

**Figure 23: The speed controlling**

### 6.2.9 The location of the robot

The GUI is required to display the specific location of the robot. The map is made up of a lot of grids. The location of the robot is able to display by coordinate, which is more accurate. When the robot moves, the coordinate will changes immediately.

RobotCoordinate.jpg

**Figure 24: The coordinate of the robot**

# Chapter 7

# Resource Estimates

The resource of the project is estimated according to the Project Description and the client's requirements in every week's meeting. All hardware and software which are used in the project are demonstrated below.

## 7.1 Hardware

### 7.1.1 Robot

Name: NXT Robot includes all components of the robot
Function: The robot is used to search the map and find all the obstacles, hidden walls and "no-go" zone. All operations are required to finish by controlling the robot.

### 7.1.2 Host

Name: PC
Function: The PC is required to demonstrate the searching information of the robot. PC is also used to control the robot on Manual Mode. The programs are uploaded to the robot from the PC.

### 7.1.3 Connection

Name: Blue-tooth and USB
Function: The Blue-tooth and USB are all used to connect with the robot. The USB is limited by the length of the cable. The Blue-tooth is limited by the uploading speed. In this project, the USB is used to upload the program and the Blue-tooth is used to implement the manual control.

## 7.2 Software

### 7.2.1 Operation Environment

Name: Mac, Linux and Windows
Function: This project is not providing the working environment. Any system is able to develop the programs for the robot.

### 7.2.2 Developing language

Name: Java, latex
Function: The codes are allowed to write in Java language, which is convenient to be read

by any developers. The documents are required to write by Latex, then generating the pdf documents.

### 7.2.3 Developing tool

Name: Eclipse
Function: Eclipse is a better tool to make the program for this project. The Eclipse is used to write the operation commands for the robot and draw the GUI for this project.

### 7.2.4 Robot Software

Name: leJOS 0.9.1
Function: The leJOS is used to control the robot and developing tool is required to support the leJOS 0.9.1.

### 7.2.5 Testing tool

Name: JUnit
Function: The JUnit is used to debug the code.

# Chapter 8

# Definitions, Acronyms, and Abbreviations

## 8.1 Acronyms and Abbreviation

| **Acronyms/Abbreviation** | **Description** |
|---|---|
| API | Application Programmable Interface |
| BT | Blue-tooth |
| GUI | Graphical User Interface |
| PC | Personal Computer |
| SRS | Software Requirements Specifications |
| SPMP | Software Project Management Plan |
| UML | Unified Modelling Language |
| USB | Universal Serial Bus |
| XML | eXtensible Markup Language |

**Table 1: Acronyms/Abbreviations**

## 8.2 Definitions

1. API: A set of classes and interfaces which are used to make the program for the robot include PC API and lejos API.

2. BT: A device to implement the wireless connection.

3. GUI: A interface which is generated on the PC is used to send the commands to the robot.

4. PC: A device which is used to make the program for the robot and build the GUI.

5. UML: A modelling language is used to build the diagrams for the project.

6. USB: A device to implement the wired connection.

7. XML: The data structure is used to store the map information.