B. Report No.2 Software Project Management Plan

1. Problem Definition

1.1. Name of this Capstone Project

- Design and implement the model of smart garden (DIMOSG)

1.2. Problem Abstract

- With gardening, tracking information about temperature, humidity, light, PH degree and soil moisture is need to do usually. Gardener have to combine many indexes and have a better plan to process it by themselves. That is easily when you have a small garden with one kind of tree, but when you have variety trees in garden, you won’t have enough time to handling each case. Besides that, gardener have another problem when they working on large garden. It also takes too much time and cost for tracking indexes by manually from different areas in garden. Changes in the weather is also a difficult problem, for example gardener doesn’t allow flower to bloom too early so they limit water for the tree but it’s rain and they don’t have any immediately solutions.

- So we have a suggestion for Smart Garden Model to solve above problems. We provide a system to measure indexes of trees from sensors in automatically and regularly to save time and costs for gardener. We also provide an immediately response system with changing of the weather to limit gardeners’ losses. On the other hand, we have other advantages of this system to help gardener who can control their garden from anywhere, anytime on mobile application. Gardening is lighter with a detail schedule which is planned out daily, weekly, or monthly. A webserver is used for processing received indexes from microcontroller. Based on that, it give a response to control actuators through microcontroller or remind a plan for gardener.

1.3. Project Overview

1.3.1 Current Situation

- There are the problems encountered in this project.

+ Knowledge of garden: lack of information about trees and their indexes. One tree has many variety indexes in different unit such as light is lux, soil moisture is present or temperature is degree. It is complex to combine those indexes together to compute range standard suitable values for each area or each tree.

+ Endurance in severe weather: the system use many kind of sensors which are communicate with environment (such as water, humidity, or chemical fertilizers) directly. With our budget as a student, we just doing this project with some cheaper sensors so maybe it won’t have a long durable time.

+ The variety of model: gardener can plant kind of tree in an area or split them in each potted plant. There are 2 device relate with soil is pH sensors and soil moisture. We need a solution to get indexes from 2 device in a same soil. If there are many potted plant and many kind of trees, so the system must equipped sensors for each kind. That make cost rise.

+ Lack of efficacy device: there are less of pH sensors for soil in the market. We just found a sensor which measure pH degree by manually. Because we can not plug sensor in the soil 24/24, it is make sensor damage earlier. In addition, this sensor has a lower price than other pH sensor so it can give not true value.

+ Lack of energy and connectivity: There are individually module sensors which are used in this system. They don’t have power bank, battery or power recharge themselves so we need to power for them. Beside that there are connect to microcontroller via wired. That make complex to deploy this system in the garden with many electric wires.

1.3.2 The Proposed System

- According to the technology researches in first week, we decide to focus on how to processing sensors’ indexes, managing devices, giving protocol module with sever and user (gardener) with server. We also pay attention how to make a plan for each tree base on received indexes, how to control a life cycle of tree such as when are they need more light for growing or delay bloom flower with warm water v.v. …). Above solutions will help gardener not take too much time and cost in gardening.

- According above purpose, we chosen a microcontroller which is…. This microcontroller have…. We can learn more about… We want to challenge…. We want to make a deep research on….

Đủ khả năng xử lý các thông tin nhận được từ các sensor, tích hợp bộ RF core, có nhiều chuẩn giao tiếp

- We also chosen some cheaper sensors which just get approximate values for sending to webserver. The important thing is we can processing all indexes and combine them together to have a better plan or response automatically.

- There are four parts in this model:

**1.3.2.1 Peripheral Block**

- This block includes one microcontroller which receive indexes from sensors and transmit it to main controller automatically. The microcontroller control actuators according to main controller’ commands based on processed indexes.

**1.3.2.2 Main Controller**

- This block will connect with peripheral block via RF to receive indexes and transmit to web service via a Wi-Fi module. There are two modes automatically and manually.

*Automatically:* Main controller ask to receive indexes from sensors with period time, transmit to web service for tracking and combining. We set up some directly situations based on sensors’ indexes with changing of the weather such as when it rain too much, it make the soil moisture higher than identified index, main controller will pull roof to cover the trees.

*Manually:* Main controller will receive manual command via web service to control sensors and actuators bases on user’ requirement such as measure an index from individual sensor anytime or control an actuator when user want.

**1.3.2.3 Web service**

Web service will combine received sensors’ indexes together to give an automatically actuators control plan in a day, week or month based on condition and period of plant growing. Any received indexes will archive in database. Web service will stored in cloud, user can manage and control the system from any where anytime.

**1.3.2.4 Mobile Application**

User manage the system through an application on mobile. They can get all information from database via web service. This application allow user to control the other parts such as request measure indexes from sensors, limit volume of water for tree or set up determined plan daily, weekly….

1.3.3 Boundaries of the System

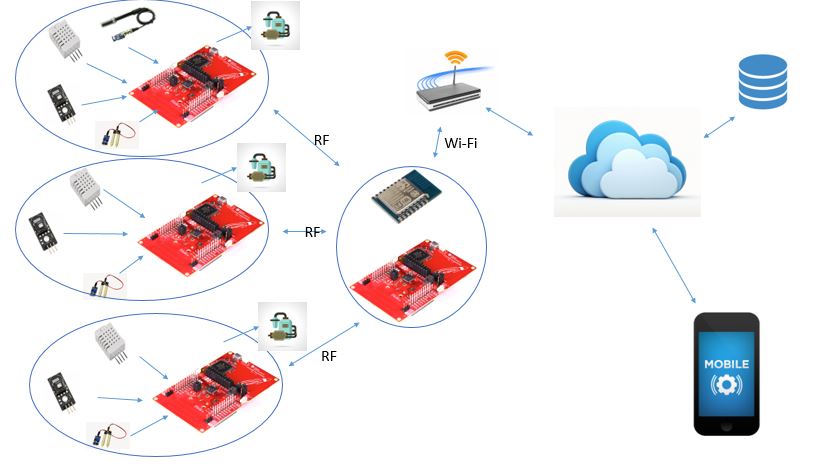
****

Figure 1: Boundaries of the System

1.3.4 Future plans

- Current system can deploy to any kind of garden, but the system just get some indexes of trees. The gardener have to tracking more indexes such as wind, rainfall, or fertilizer level moisture… We design the system to scale to bigger model. We can integrated more sensors for tree in this system. That will help tracking more detail in periods of growing plant. The gardener will have a best plan to take care and decrease loss in gardening.

1.3.4 Development Environment

1.3.4.1 Hardware requirements

- For Sub Controller

|  |  |  |
| --- | --- | --- |
| Devices | Minimum Requirements | Recommended |
| Controller | | |
| Microcontroller | ARM Cortex | CC1310 Launchpad |
| Sensors | | |
| Light sensor | Using photo resistor LM393 | Using photodiode BH-1750 |
| Temperature and Humility | DHT 11, DHT 22 |  |
| Soil Moisture | Kit using LM393 Comparison |  |
| pH | Kit with UART |  |
| Component | | |
| Water pump | Aquarium Pumps for demo |  |
| The traction motor |  |  |
| Nebulizer |  |  |

- For Main Controller

|  |  |  |
| --- | --- | --- |
| Devices | Minimum Requirements | Recommended |
| Microcontroller | Arduino or ARM Cortex | CC1310 Launchpad (RF interface) |
| Module Wi-Fi | Wi-Fi transceiver ESP8266 V7 | Wi-Fi transceiver ESP8266 V7 |

- For Mobile Application

|  |  |  |
| --- | --- | --- |
| Devices | Minimum Requirements | Recommended |
| Android Mobile |  |  |

1. CC1310 Launchpad

2. Light Sensor BH1750 FVI

3. Temperature and Humility Sensor DHT22

4. Soil Humility Sensor LM393

5. PH Sensor

6. Water pumps

7. The traction motor

8. Nebulizer

9. ESP8266 V7

PA\\*EFORMAT 1

1.3.4.2 Software requirements

|  |  |  |
| --- | --- | --- |
| Software | Name / Version | Description |
| Project Management | | |
| Real-time messaging | Slack Cautionary Tale 2.0.6 | Slack brings all communication together in one place |
| Task Plan | Trello.com |  |
| Source Control | SourceTree 1.8.3.0 |  |
| Development | | |
| Environment | Java |  |
| Modeling tool | Star UML/ Software Ideas Modeler |  |
| DBMS | Microsoft SQL Server 2012/ MySQL |  |
| IDE | Android Studio  IAR Workbench  Code Composer Studio  Spring Tool Suite |  |

1. Spring Tool Suite
2. Android Studio
3. IAR Workbench
4. Source Tree
5. Slake
6. Trello
7. Microsoft SQL Server/ MySQL
8. Star UML/ Software Ideas Modeler

2. Project organization

2.1 Software Process Model

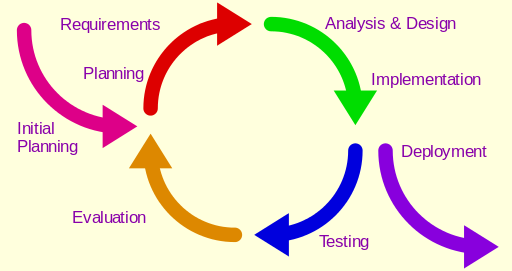


Figure 2: Incremental Model

For more information:

* <https://en.wikipedia.org/wiki/Iterative_and_incremental_development>

Scrum: dễ thay đổi, dễ update liên tục requirement

2.2 Roles and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| No | Full name | Role in Group | Responsibilities |
| 1 | Nguyễn Đức Lợi | Supervisor  Project manager | * Defining business * Supporting in technique issues * Controlling the development process |
| 2 | Phan Thành Sang | Team leader  Business Analysis  Developer  Tester | * Managing process * Clarifying requirements * Preparing documents and reports * Creating task plan * Reviewing documents and reports * Committing all individual works. * Design and implement hardware. * Design and implement Database. * Researching components, document for implementing. * Supporting each other. |
| 3 | Phạm Hoàng Chinh | Developer  Tester | * Implement document and reports * Review documents and reports * Committing all individual works * Researching components, document for implementing. * Implement chart. |
| 4 | Huỳnh Hữu Nghị | Developer  Tester | * Implement document and reports * Review documents and reports * Design and implement mobile application. |
| 5 | Lê Văn Pháp | Developer  Tester | * Implement document and reports * Review documents and reports * Design and implement web server. |

Table 2: Roles and responsibilities

MAT 1

2.3 Tools and Techniques

|  |  |  |
| --- | --- | --- |
| No | Tools/Techniques | Name/Version |
| 1 | For Hardware | IAR Workbench 7.4 |
| 2 | For Hardware | PCB Editor 16.6 |
| 3 | For Software | Spring Tool Suite |
| 4 | For Software | Android Studio 2.1 |

Table 3: Tools and Techniques

3. Project Management Plan

3.1 Software development life cycle

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Phase | Description | Deliverables | Resource needed | Dependencies  and Constrains | Risk |
| Increment 0:  Analysis & Research | - Collect requirement  - Research the same system  - Research tools and technique  - Research hardware (sensor, MCU…)  - Identify and clarify requirement  - Define test phase  - Create introduction report | - Introduction report  - Research report | - 7 man-days | - N/A | - Missing requirement  - Unclear scope of project  - Lack of member share of understand |
| Increment 1:  Research  Planning  Design | - Create Software project management plan report  - Create Software Requirement Specification report  - Define requirement  - Define performance measures  - Research giant.  - Research hardware, sensor  - Connect sensor with MCU.  - Research RF connection.  - Research connect MCU to Internet  - Research Web server  - Research mobile application  - Design database  - Design test case | - Software project management plan  - Software Requirement Specification  - Connect and transfer data with hardware.  - Connect and transfer data with internet  - Connect and transfer data with mobile application  - Test report  - Performance measures report | - 14 man-days | - Depend on introduction report, research reports, hardware device | - Lack of Experience.  - Late for shipping hardware  - Measure is not exactly  - Test case cannot cover all cases |
| Increment 2  Implementation | - Create Software Design Description report  - Implement transfer data from sensor to MCU and MCU to server  - Implement control with algorism in web server and sent to mobile application  - Implement Web API and hardware get API, controlling water supplying system.  - Testing hardware transfer  - Design user interface  - Design model of demo | - Software Design Description report  - Mobile application  - Web server and Web API  - Test report  - Performance measures report | - 21 mans-day | - Depend on introduction report, research reports, hardware | - Lack of Experience.  - Measure is not exactly  - Hardware work not good  - Test case cannot cover all cases |
| Increment 3  Implementation (communication in system)  Testing | - Create System Implementation & Test report  - Implement schematic of board  - Implement hardware connects together  - Implement transfer data with wireless communication  - Implement transfer data from mobile application to server  - Implement control water supplying  - Implement model for demo  - Test transfer data  - Test control signal from server to MCU | - Create System Implementation & Test report  - Board for hardware  - Web server for control and transfer and save data  - Mobile Application for control and show information  - Test report  - Performance measures report | - 21 mans-day | - Depend on introduction report, research reports, hardware | - Lack of Experience.  - Measure is not exactly  - Hardware is not exactly  - Test case cannot cover all cases |
| Increment 4  Deployment | - Create installation guide  - Create User guide  - Integration test  - Implement modeling garden. | - Installation guide  - User guide  - Integration test report  - Modeling garden | - 7 mans-day | - All functions and reports are completed | - Lack of experience  - Hard to grow plant  - Depend on weather |

Table 3: Software Development Life Cycle Detail

3.2 Increment Detail

3.2.1 Increment 0: Analysis & Research

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| - Collect requirement | Meeting with supervisor and get all information about requirement of systems | Sangpt, Chinhph, Phaplv |
| - Research the same system | Find some current systems all over the world and popular ways in local  Find out strength and weakness for each current system. | Chinhph, Phaplv |
| - Research tools and technique | Predict tools and technique for the system based on requirement and current system. | Sangpt |
| - Clarify requirement | Define main functions which are approved by supervisor | Sangpt, Chinhph, Phaplv |
| - Research hardware | Research popular module sensors in local market (how to work, IO, accuracy, pros and cons….) | Chinhph, Phaplv |
| - Define test phase | Define test phase for each part in system. | Sangpt |
| - Create introduction report |  | Sangpt, Chinhph |

Table 4: Increment 0 Detail

3.2.2 Increment 1

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| - Create Software project management plan report | Define phase of project and what is need to do for each part in system | Sangpt |
| - Create Software Requirement Specification report | Identify detail requirements of system. | Sangpt, Chinhph, Phaplv, Nghihh |
| - Research Gantt chart. | Giving plan management for project.  Define model for management and development of project | Sangpt |
| - Research hardware, sensor | Compare sensors based on functions and sensors report in Increment 0.  Choose sensors which approved by supervisor for system | Sangpt, Phaplv, Chinhph, Nghihh |
| - Connect sensors with MCU. | Interfaces: I2C, UART, SPI | Phaplv, Chinhph |
| - Research RF connection. | Communicate between MCUs via RF | Sangpt, Chinhph |
| - Research connect MCU to Internet | Communicate MCU with Internet | Sangpt, Phaplv |
| - Research Web server | Identify tool and technique to create Web service | Phaplv |
| - Research mobile application | Define tools and techniques to create mobile application for user |  |
| - Design database | Store all information of system for tracking and processing. | Phaplv, Nghihh |
| - Design test case | Create test case for each situation in system | Chinhph, Nghihh |

Table 5: Increment 1 Detail

3.2.3 Increment 2

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| - Create Software Design Description report |  | Sangpt, Chinhph, Phaplv, Nghihh |
| - Implement transfer data from sensor to MCU and MCU to server | Transmit indexes from sensors in sub-controller to Main Controller | Sangpt, Chinhph |
| - Implement control with algorism in web server and sent to mobile application | Connect between Mobile Application and Web API | Phaplv, Nghihh |
| - Implement Web API and hardware get API, controlling water supplying system. | Implement connection of Web API with Main Controller | Sangpt, Phaplv |
| - Testing hardware transfer | Do test case based on design in Increment 1 | Nghihh, Chinhph |
| - Design user interface | Design mobile application’ interface | Phaplv, Chinhph |
| - Design model of demo | Identify kind of trees, model of garden for demo | Chinhph, Nghihh |

Table 6: Increment 2 Detail

3.2.4 Increment 3

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| - Create System Implementation & Test report |  | Sangpt, Phaplv, Nghihh, Chinhph |
| - Redefine requirement | Rechecking requirement based on results of Increment 2. Find out risks and problem of system.  Fix requirements with supervisor’ agreement. | Chinhph, Nghihh |
| - Redefine performance measures | Check performance measures of sensors. Replace when necessary. | Sangpt, Phaplv |
| - Redesign test case | Redesign test case which suitable with changes. | Chinhph, Nghihh |
| - Implement schematic of board | Design schematic of board in sub and main controller.  Packet sensors with MCU and MCU with module Wi-Fi | Sangpt, Chinhph |
| - Implement hardware connects together | Combine all in one system in real environment for experimentally and testing | Sangpt, Phaplv, Chinhph, Nghihh |
| - Implement transfer data in system | Implement transfer data with Wireless Connection and Mobile Application with Webserver | Phaplv, Nghihh |
| - Implement control actuators | Control actuators in system which combined all parts together | Sangpt, Chinhph |
| - Implement demo model | Create a small model to demo | Chinhph, Nghihh |
| - Test transfer data | Testing transfer data based on test case  Fix bugs. | Sangpt, Phaplv, Chinhph, Nghihh |
| - Test control signal from server to MCU | Test rightness of control signal from server to MCU based on plan or sensors’ indexes. | Sangpt, Phaplv, Chinhph, Nghihh |

Table 7: Increment 3 Detail

3.2.5 Increment 4

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| - Create installation guide | Create guide of installation. | Chinhph, Nghihh |
| - Create User guide | Create guide for user how to control the system | Chinhph, Nghihh |
| - Integration test | Retest all test case in combined system.  Fix bugs. | Chinhph, Nghihh  Sangpt, Phaplv |
| - Implement modeling garden. | Deploy system to demo garden model | Sangpt, Phaplv, Chinhph, Nghihh |

Table 8: Increment 4 Detail

3.3. All Meeting Minutes

<https://github.com/sangphanthanh/CP_SmartGarden/tree/master/Document>

4. Coding Convention

4.1. Local Variables

Declare variables in local scope and initialization should be used instead of declaration and assignment.

<http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#Local_Variables>

4.2. General Naming Rules

Function names, variable names, and filenames should be descriptive; eschew abbreviation. Give as descriptive a name as possible, within reason. Do not worry about saving horizontal space as it is far more important to make your code immediately understandable by a new reader. Do not use abbreviations that are ambiguous or unfamiliar to readers outside your project, and do not abbreviate by deleting letters within a word.

<http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#General_Naming_Rules>

4.3. File Names

Filenames should be all lowercase and can include underscores (\_) or dashes (-). Follow the convention that your project uses. If there is no consistent local pattern to follow, prefer "\_".

<http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#File_Names>

4.4. Variable Names

The names of variables and data members are all lowercase, with underscores between words. Data members of classes (but not structs) additionally have trailing underscores.

<http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#Variable_Names>

4.5. Function Names

Regular functions have mixed case; accessors and mutators match the name of the variable. Functions should start with a capital letter and have a capital letter for each new word. No underscores

<http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#Function_Names>

4.6. Type Names

Type names start with a capital letter and have a capital letter for each new word, with no underscores. The names of all types — classes, structs, typedefs, and enums — have the same naming convention. Type names should start with a capital letter and have a capital letter for each new word. No underscores.

<http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#Type_Names>

4.7. Line Length

Each line of text in your code should be at most 80 characters long. We recognize that this rule is controversial, but so much existing code already adheres to it, and we feel that consistency is important.

[http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#Line\_Length](http://google-styleguide.googlecode.com/svn/trunk/cppguide.html%23Line_Length)

4.8. Class Comments

Every class definition should have an accompanying comment that describes what it is for and how it should be used.

[http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#Class\_Comments](http://google-styleguide.googlecode.com/svn/trunk/cppguide.html%23Class_Comments)

4.9. Function Comments

Declaration comments describe use of the function; comments at the definition of a function describe operation.

[http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#Function\_Comments](http://google-styleguide.googlecode.com/svn/trunk/cppguide.html%23Function_Comments)

4.10. Variable Comments

In general, the actual name of the variable should be descriptive enough to give a good idea of what the variable is used for. In certain cases, more comments are required.

[http://google-styleguide.googlecode.com/svn/trunk/cppguide.html#Variable\_Comments](http://google-styleguide.googlecode.com/svn/trunk/cppguide.html%23Variable_Comments)