|  |  |
| --- | --- |
|  | **MINISTRY OF EDUCATION AND** **TRAINING** |

**FPT UNIVERSITY**

Capstone Project Document

**Implementation of Surveillance Camera using Raspberry kit**

|  |  |
| --- | --- |
| **Group 5** | |
| **Group members** | Đặng Ái Trinh – SE60944 (Leader)  Nguyễn Chí Nghĩa – SE61203  Lê Long Hồ – SE61291 (Dropped out)  Nguyễn Hồng Lâm – SE60867 (Dropped) |
| **Supervisor** | Mr. Bùi Đại Trí |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** | SCWR |

-Ho Chi Minh City, 10th May 2016 -

Contents

[**List of Tables** 4](#_Toc458232463)

[**List of Figures** 5](#_Toc458232464)

[**A. Report No. 1 Introduction** 7](#_Toc458232465)

[1. Project Information 7](#_Toc458232466)

[2. Introduction 7](#_Toc458232467)

[3. Current Situation 7](#_Toc458232468)

[4. Problem Definition 8](#_Toc458232469)

[5. Proposed Solution 8](#_Toc458232470)

[5.1. Feature functions 9](#_Toc458232471)

[5.2. Advantage and Disadvantage 9](#_Toc458232472)

[6. Functional Requirements 9](#_Toc458232473)

[7. Role and Responsibility 10](#_Toc458232474)

[**B. Report No. 2 Software Project Management Plant** 11](#_Toc458232475)

[1. Problem Definition 11](#_Toc458232476)

[1.1. Name of this Capstone Project 11](#_Toc458232477)

[1.2. Problem Abstract 11](#_Toc458232478)

[1.3. Project Overview 11](#_Toc458232479)

[2. Project organization 15](#_Toc458232480)

[2.1. Software Process Model 15](#_Toc458232481)

[2.2. Roles and responsibilities 16](#_Toc458232482)

[2.3. Tools and Techniques 17](#_Toc458232483)

[3. Project Management Plan 18](#_Toc458232484)

[3.1. Software development life cycle 18](#_Toc458232485)

[3.2. Increment Detail 19](#_Toc458232486)

[3.3. All Meeting Minutes 23](#_Toc458232487)

[4. Coding Convention 23](#_Toc458232488)

[4.1. General Naming Rules 23](#_Toc458232489)

[4.2. Variable Names 23](#_Toc458232490)

[4.3. File Names 23](#_Toc458232491)

[4.4. Function Names 23](#_Toc458232492)

[4.5. Comments 23](#_Toc458232493)

[4.6. Lines 23](#_Toc458232494)

[**C. Report No. 3 Software Requirement Specification** 24](#_Toc458232495)

[1. User Requirement Specification 24](#_Toc458232496)

[2. System Requirement Specification 25](#_Toc458232497)

[2.1. External Interface Requirement 25](#_Toc458232498)

# List of Tables

[Table 0‑1: Definitions, Acronyms and Abbreviations 6](#_Toc458633285)

[Table A‑1: Roles and Responsibilities 10](#_Toc458633286)

[Table B‑1: Hardware Requirements for Developer 14](#_Toc458633287)

[Table B‑2: Hardware Requirements for System 14](#_Toc458633288)

[Table B‑3: Hardware Requirements for User 14](#_Toc458633289)

[Table B‑4: Software Requirement for Developer 14](#_Toc458633290)

[Table B‑5: Software Requirements for System 15](#_Toc458633291)

[Table B‑6: Software Requirements for User 15](#_Toc458633292)

[Table B‑7: Roles and Responsibilities 17](#_Toc458633293)

[Table B‑8: Tools and Techniques 17](#_Toc458633294)

[Table B‑9: Software Development Life Cycle 19](#_Toc458633295)

[Table B‑10: Sprint 1: Requirement Analysis 20](#_Toc458633296)

[Table B‑11: Sprint 2: Background Preparing 20](#_Toc458633297)

[Table B‑12: Sprint 3: SMS implement and image uploading 21](#_Toc458633298)

[Table B‑13: Sprint 4: Detect Algorithm and power supply circuit implementing 21](#_Toc458633299)

[Table B‑14: Sprint 5: Completing 22](#_Toc458633300)

[Table C‑1: Raspberry Pi 3 Specification 28](#_Toc458633301)

[Table C‑2: Camera Pi Specification 30](#_Toc458633302)

# List of Figures

[Figure A‑1: Example of Surveillance Camera 7](#_Toc458633308)

[Figure A‑2: Example of Surveillance Team 8](#_Toc458633309)

[Figure B‑1: System Boundaries 12](#_Toc458633310)

[Figure B‑2: Example of Building Surveillance System 13](#_Toc458633311)

[Figure B‑3: Example of Multi Cameras 13](#_Toc458633312)

[Figure B‑4: Example of Scrum Model 16](file:///C:\Users\NghiaNC\Desktop\tmp_Report\SCWR_Final_v1.1.docx#_Toc458633313)

[Figure C‑1: Raspberry Pi 3 Model B 27](#_Toc458633314)

[Figure C‑2: Camera Pi 29](#_Toc458633315)

[Figure C‑3: SIM900A GSM/GPRS Module 30](#_Toc458633316)

[Figure C‑4: SIM900A Pins Map 31](#_Toc458633317)

[Figure C‑5: Adapter 9V – 0.6A 32](#_Toc458633318)

[Figure C‑6: Power Supply circuit example 32](#_Toc458633319)

[Figure C‑7: 3D Simulation of Alarm circuit 33](#_Toc458633320)

[Figure C‑8: Python Shell 33](#_Toc458633321)

[Figure C‑9: Terminal 34](#_Toc458633322)

[Figure C‑10: System Overview Use Case 35](#_Toc458633323)

[Figure C‑11: User Overview Use Case 35](#_Toc458633324)

[Figure C‑12: User Receive Alarm Messages Use Case 36](#_Toc458633325)

[Figure C‑13: User View Photos Use Case 37](#_Toc458633326)

[Figure C‑14: User Send Monitor Message Use Case 38](#_Toc458633327)

[Figure C‑15: GSM Overview Use Case 39](#_Toc458633328)

[Figure C‑16: GSM Send Alarm Messages Use Case 40](#_Toc458633329)

[Figure C‑17: GSM Receive Monitor Messages Use Case 41](#_Toc458633330)

[Figure C‑18: <Camera Pi> Overview Use Case 42](#_Toc458633331)

[Figure C‑19: <Camera Pi> Transfer Video Use Case 42](#_Toc458633332)

[Figure C‑21: <Raspberry> Overview Use Case 43](#_Toc458633333)

[Figure C‑22: <Raspberry> Upload Photos Use Case 44](#_Toc458633334)

[Figure C‑23: <Raspberry> Detect Human Use Case 45](#_Toc458633335)

[Figure C‑24: <Raspberry> Manage Alarm Use Case 46](#_Toc458633336)

**Definitions, Acronyms, and Abbreviations**

|  |  |
| --- | --- |
| **Name** | **Definition** |
| SCWR | Implementation of Surveillance Camera using Raspberry kit |
| CCTV | Closed Circuit Television |
| GSM | Global System for Mobile Communications |
| SMS | Short Message Service |
| MMS | Media Message Service |
| IP | Internet Protocol |
| Wi-Fi | Wireless Fidelity |
| IE | Internet Explorer |
| BA | Business Analyst |
| AT Command | AT is the abbreviation of ATtention. Every command line starts with "AT" or "at" |

Table 0‑1: Definitions, Acronyms and Abbreviations

# Report No. 1 Introduction

## Project Information

* Project name: **Implementation of Surveillance Camera using Raspberry kit**
* Project Code: **SCWR**
* Product Type: **Device**
* Start Date: **10/05/2016**
* End Date: **01/08/2016**

## Introduction

In current life, we are usually away from home for working, traveling, etc. So everyone also wants their home to be safe and secured. In the past, we often used CCTV to record, but there are a few of them had remote access or alert. Nowadays, most of home surveillance camera systems allow you to monitor what is going on in your home with live stream or recorded videos. Besides that, some of them also have alarm system or they can send notifications to you to alert. Based on our research and analysis, we have built project deals with the Design and Implementation of Surveillance Camera using Raspberry kit.

SCWR provides a greater benefit to user. This system is connected to the internet to allow user to remote monitoring. We build The SCWR, which help users know what is happening, even when you are away. Messages are sent directly to your mobile device whenever cameras detect motion and transmits, also intruder images will be uploaded via Wi-Fi to a static IP, which is viewed using a web browser, even from mobile device. Besides, our system also makes alert sound at home.

This document describes our working process in 4 months includes our perspective in the system, component designs or detailed core workflows. We developed a good solution that provides a user security application for homes.

## Current Situation

Nowadays, there are many kinds of security. But the most popular model of security camera is surveillance camera.



Figure A‑1: Example of Surveillance Camera

The Figure A-1 is an example of surveillance camera. In fact, user often setup a camera or even a system for their asset. The size of system is based on their necessary. For example, personal home often need one or two cameras, a shop is 4 to 8 ones but a company, building or organization usually need a huge system with hundreds.

The current system not only includes camera but they also have a sub-system with many screens to monitor what happened before camera lens. Therefore, surveillance system also need a team to monitor and react on time.



Figure A‑2: Example of Surveillance Team

We can recognize that the number of member in monitoring team will be decided by the size of the system. It means we will spend much money for hiring staff to monitor or maintenance cost. In Figure A-2, there are many people to control a system: monitoring, analyzing, etc.

So we have found out the problem is about the cost and we also focus and solve it in our system SCWR.

## Problem Definition

Although security camera or CCTV systems are used widely but they also have some weakness:

* Users have to pay money for hiring security to monitor the camera.
* Users can’t control the system when they’re going out or traveling.
* They have to store a huge amount of data.

System can record a robbery or crime in progress, but they can't do anything to actually stop the crime.

## Proposed Solution

Our proposed solution is to build a Surveillance Camera system named “SCWR” based on Raspberry kit which increases the security and convenience. The system can overcome the weakness of available CCTV systems and give users more utilities and convenience.

### Feature functions

* Raspberry Pi 3:
  + This system need a core processor which has enough power to run algorithm of detection and controls peripherals.
  + Raspberry Pi 3 has more power than Raspberry Pi 2.
  + Besides that, it also has Wi-Fi so we can connect to the internet but don’t need to use Ethernet.
* Algorithm of detecting:
  + Detect moving: First of all, we need to detect and delimiting the object which is moving in camera area.
  + Detect size: Second, after detect which object is moving, we need to estimate its size and compare with defined parameters.
* GSM 900:
  + Use SMS to send alarm message instead of using Wi-Fi to save battery. On the other hand, if we use Wi-Fi to send SMS, we’ll need to build an Android/iOS application to receive message.
* Alarm system:
  + SCWR use IC555 to make police alarm circuit. This circuit is used to alert for neighbors and police.

### Advantage and Disadvantage

The advantages and disadvantages of the proposed solution:

* Advantages:
  + Easy to setup.
  + User can monitor from anywhere via internet.
  + User don’t need to hire a security to monitor 24/7.
  + Instead of using Wi-Fi or 3G all of day for receiving alarm message, the message will send via SMS. So user can save your battery to do what they want.
* Disadvantages:
  + The accuracy of detection is not high. Cause “Fake Alarm will be occurred regularly.
  + System doesn’t have any GUI or application for user to interact.
  + System doesn’t have recording function to review what was happened.

## Functional Requirements

Function requirements of the system are listed as below:

* System component:
  + Detect moving.
  + Estimate size of object.
  + Turn on/ off alarm system automatically.
  + Upload image to server via Wi-Fi.
  + Send alarm message to user’s mobile via SMS.
* User component:
  + Receive alarm message.
  + View intruder image
  + Turn on/off alarm via SMS.

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Full Name | Role | Position | Contact |
| 1 | Bùi Đại Trí | Project Manager | Supervisor | Daitribk@gmail.com |
| 2 | Đặng Ái Trinh | Developer, Tester | Leader | Trinhdase60944@fpt.edu.vn |
| 3 | Nguyễn Chí Nghĩa | Developer, Tester | Member | [Nghiancse61203@fpt.edu.vn](mailto:phucnhse60749@fpt.edu.vn) |
| 4 | Lê Long Hồ | Developer, Tester | Member | Hollse61291@fpt.edu.vn |
| 5 | Nguyễn Hồng lâm | Developer, Tester | Member | Lamnhse60867@fpt.edu.vn |

Table A‑1: Roles and Responsibilities

# Report No. 2 Software Project Management Plant

## Problem Definition

### Name of this Capstone Project

* Official name: Implementation of Surveillance Camera using Raspberry Kit
* Vietnamese name: Xây dựng camera giám sát dùng Raspberry
* Abbreviation: SCWR

### Problem Abstract

Currently, not only Vietnamese but also all people in the world often travel around their country or go abroad. But when they’re on their strip, they don’t feel comfortable about their home, their asset. Therefore, the demands of using surveillance system are more and more increase.

We provide a system more convenient than others which are available on the market. Our system can detect intruders who illegally entry your home, your room and use alert function immediately. SCWR is different from other because it can make the intruder scare and get out of your home and others doesn’t have. Besides that, SCWR can send alarm message with intruder photos ’s link and the user can monitor from anywhere.

In SCWR, we used SMS instead of Android/iOS application because of saving battery. If user want to receive alarm message, they have to connect their phone to internet because an Android/iOS application need Wi-Fi or 3G to communication with Raspberry.

### Project Overview

#### Current Situation

##### Advantages

* Supporting from open source: Open CV
* The powerful of Raspberry Pi 3
* Ease to connect between Raspberry and Camera Pi
* The convenience for user in monitoring
* Decrease cost for user in hiring staff to monitor
* This system not only a surveillance camera but also it can make the illegal intruder scare
* SCWR use SMS instead of Android/iOS application to save battery

##### Disadvantages

* There is a critical bug about black screen of Raspberry and we need to install OS for many times
* Besides that, there are also many bugs when we configure or communicate hardware.
* The quality of Camera Pi is not really high so it’s affect to the accuracy of detect algorithm.
* Raspberry can be connected to Wi-Fi but there is no alarm function using Wi-Fi.
* The system only can detect human with 60 percent efficiency.

#### The Proposed System

SCWR is a high efficient security camera system and can protect you from burglary, theft and other crimes.

Firstly, when something moves inside the monitored area, we must detect movement in the video stream. Next, the system has ability to detect that moving object is human or not base on their size, it helps to exclude the wrong thieves (such as mouses and cats).

In case moving object is human, we will switch over to still camera mode and take photos over the span of several seconds to capture the movement in high resolution snapshots and then upload these images to Drop box server. Finally, we make an alarm sound, sends an alarm message to users the uploaded images, which can be viewed by using web browser, even from your mobile. So that the user can easier monitor and call police.

#### Boundaries of the system

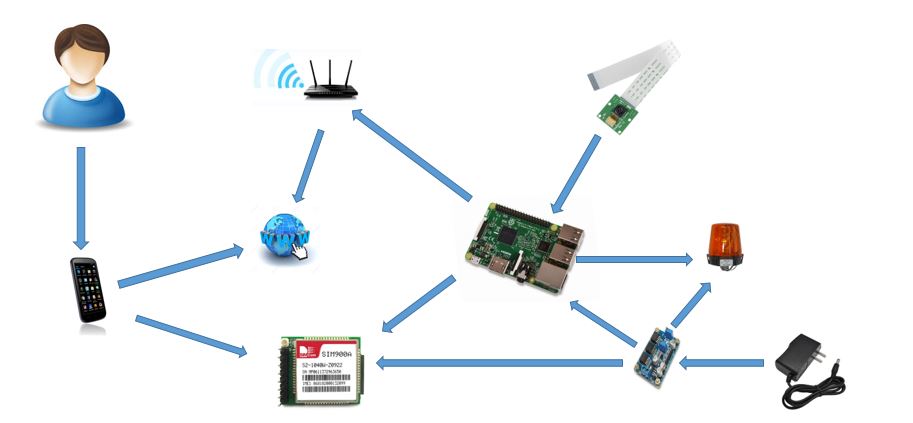


Figure B‑1: System Boundaries

The final system includes:

* Raspberry Pi 3, which is connected to the internet via Wi-Fi, is core processor. The Raspberry interacts with Camera Pi to get the video transfer to calculate in algorithm and require camera to capture image which is uploaded to server.
* GSM Sim900A, which is controlled by Raspberry, is used to communication with mobile of user via SMS. Raspberry will send to user alarm messages with the link of intruder’s images. GSM also receives SMS from user to turn on / off alarm system.
* Alarm system is designed and implemented by IC555s to make police sounds. Alarm circuit is used to alert at the place that system is set up.
* Power supply circuit is used to convert voltage from adapter (9V) to lower level (5V). Power supply uses LM2576 to convert voltage.
* Finally, user need a mobile can send / receive SMS, connect to Wi-Fi, 3G and must have web browser.

#### Future plans

* Current system only can deploy to personal purpose (Homes, bedrooms) or small places. So we will make it is available with larger place (buildings, malls, companies).

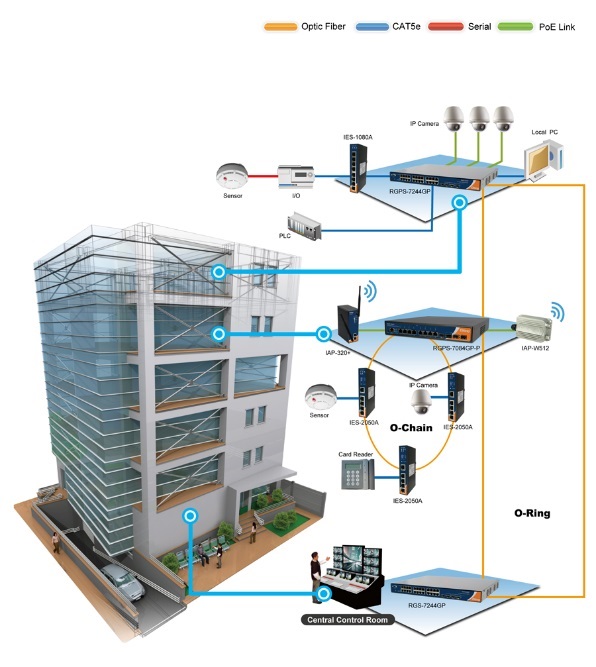


Figure B‑2: Example of Building Surveillance System

* There is only one camera in SCWR. We will make it available for multi-cameras to expand the monitor area.
* Currently, system only alert via SMS. In the future, we will design a mobile application using Wi-Fi, 3G and we have many choices to choose base on each situation.



Figure B‑3: Example of Multi Cameras

* We also maintain the algorithm and make it better (more accuracy, faster, more optimal). Besides that, we will change to another camera with more highly functions.

#### Development Environment

##### Hardware requirements

* For developer

|  |  |  |
| --- | --- | --- |
| **Hardware** | **Minimum Requirements** | **Recommended** |
| Internet Connection | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| Operating System | Windows 7 | Windows 7 or later |
| Computer Processor | Intel® Xeon ® 1.4GHz | Intel® Xeon ® Quad Core  (12M Cache, 2.50 GHz) |
| Computer Memory | 1GB RAM | 2GB or more |

Table B‑1: Hardware Requirements for Developer

* For system

|  |  |  |
| --- | --- | --- |
| **Hardware** | **Minimum Requirements** | **Recommended** |
| Internet Connection | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| Operating System | Raspbian Jessie | Raspbian Jessie |
| Computer Processor | Raspberry Pi 2 | Raspberry Pi 3 |
| Peripherals | GSM Sim 900A | GSM Sim 900A |
| Camera Pi | Camera Pi NoIR |
| Alarm system | Two IC555s circuit |
| Speaker 0.5W | Speaker 1W |

Table B‑2: Hardware Requirements for System

* For user

|  |  |  |
| --- | --- | --- |
| **Hardware** | **Minimum Requirements** | **Recommended** |
| Mobile Connection | Connected to GSM | Mobifone, Vinafone, Viettel, etc. |
| Internet Connection | Wi-Fi (4 Mbps)/2G/3G | Wi-Fi (8 Mbps) or 3G |
| Browser | Default browser | Internet Explorer, Chrome, Safari, etc. |

Table B‑3: Hardware Requirements for User

##### Software Requirements

* For developer

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| Operating System | Windows 7 or later / Linux | Operating system and platform for remote desktop |
| Raspbian Jessie | Operating system and platform for executing |
| IDE | Python Shell | Programing tool |
|  | Terminal | Execution |
| Web browser | Default Web Browser of Raspberry | Searching tool for knowledge, asking expert, solutions. |

Table B‑4: Software Requirement for Developer

* For System

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| Operating System | Raspbian Jessie | Operating system and platform for executing |
| IDE | Terminal | Execution |

Table B‑5: Software Requirements for System

* For User mobile

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| Operating System | Android or iOS | Operating system and platform of user mobile |
| IDE | SMS manager | To send monitoring messages and receive alarm messages |
|  | IE, Chrome, Safari, Opera | Web browser to view photos on Drop box |

Table B‑6: Software Requirements for User

## Project organization

### Software Process Model

#### Overall Description

Scrum is an iterative and incremental [agile software development](https://en.wikipedia.org/wiki/Agile_software_development) framework for managing product development. The term is named for the scrum (or scrummage) formation in rugby, which is used to restart the game after an event that causes play to stop, such as an infringement. Scrum involves:

* Initial appointment of a project manager called the "scrum master."
* Definition and prioritization of tasks to be done.
* Planning sessions for each task.
* Daily meetings among teams.
* Identification and evaluation of potential project risks and process pitfalls.
* Execution of projects in brief, high-intensity, frequent work sessions.
* Reviews of progress and evaluations of completed projects.
* Openness to constructive criticism and ideas for improvement.

#### Reasons for choosing

We choose this model because of following reasons:

* Scrum is easier to deliver a quality product in a scheduled time.
* Scrum allows us to change priorities and requirements quickly.
* Scrum has daily meeting that help us to measure individual productivity. Then we can improve the productivity of members.
* Issues can be defined and resolved rapidly because of daily meeting.

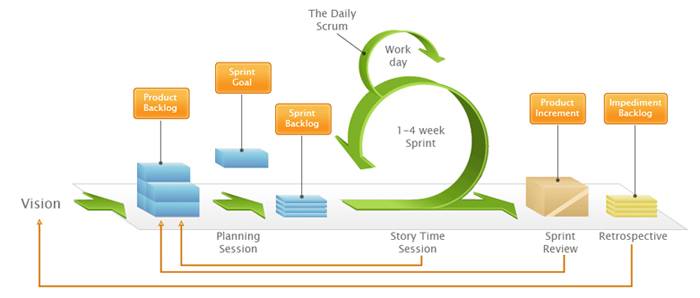


Figure B‑4: Example of Scrum Model

Reference: <http://www.netspecglobal.com/approach.html>

For more information:

<https://en.wikipedia.org/wiki/Scrum_(software_development)>

### Roles and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in group** | **Responsibilities** |
| **1** | Bui Dai Tri | Supervisor, Project Manager | * Specify user requirements * Giving out technique and business analysis support * Control the development process |
| **2** | Dang Ai Trinh | Team Leader, BA, Developer, Tester | * Clarifying requirements * Creating plan * Managing process * Reviewing documents * Coding |
| **3** | Nguyen Chi Nghia | BA, Developer, Tester | * Clarifying requirements * Configuring Hardware * Writing documents * Coding * Testing |
| **4** | Le Long Ho | Developer, Tester | * Writing documents * Coding * Testing |
| **5** | Nguyen Hong Lam | Developer, Tester | * Writing documents * Coding * Testing |

Table B‑7: Roles and Responsibilities

### Tools and Techniques

|  |  |
| --- | --- |
| **Tool / Technique** | **Name / Version** |
| Environment | Raspbian Jessie, Ubuntu 15.0 |
| Programming language | Python 2, python 3, AT Commands for GSM |
| Circuit design | Proteus 8 Professional v8.3 SP2 (Build 19906) |
| Diagram Design | Software Ideas Modeler 10.10.6002.15505 64-bit |
| Management | Trello, Slack, Source Tree, GitHub |
| Support | Remote Desktop Control, Putty, Arduino IDE 1.6.9 |

Table B‑8: Tools and Techniques

## Project Management Plan

### Software development life cycle

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Phase** | **Description** | **Deliverables** | **Resource needed** | **Dependencies**  **and Constrains** | **Risk** |
| Sprint 1:  Requirement Analysis | - Collect requirement  - Research the same system  - Research tools and technique  - Research and choose hardware  - Identify and clarify requirement  - Create introduction report  - Create Software Requirement Specification report | - Introduction report  - Software Requirement Specification | - 30 man-days | - Need to wait for supervisor ’s confirmation | - Missing requirement  - Unclear scope of project  - Lack of member share of understand |
| Sprint 2:  Background Preparing | - Create project management plan report  - Architecture design for the system  - Detail design using top-down  - Break down using top-down  - Choose architecture style  - Study documents  - Configure hardware devices | - Software project management plan  - Hardware communication  - Component instruction  - Configure instruction | - 64 man-days | - Depend on “Requirement Analysis” | - Lack of Experience.  - Not fulfil requirement  - Hardware incompatible  - Lack of understanding |
| Sprint 3:  SMS implement and image uploading | - Upload image to the internet (Dropbox server)  - Send / receive SMS via GSM  - View photos from SMS  - System detail design  - Design and implement alarm circuit | - GSM instruction  - Image upload instruction  - Software design document  - Alarm board | - 64 mans-day | Depend on  - Hardware configuration  - Hardware researching  - Delay time for making board | - Lack of Experience.  - Hardware incompatible  - Lack of understanding  - Time for fixing and making board too long |
| Sprint 4:  Detect Algorithm and power supply circuit implementing | - Create System Implementation & Test report  - Design and implement power supply circuit  - Implement detect motion algorithm  - Implement detect size algorithm  - Integrate and testing | - System Implementation & Testing report  - Power supply board  - Algorithm of detect motion and size  - Test report | - 168 mans-day | Depend on  - Hardware configuration  - Delay time for making board | - Lack of Experience.  - Overlap resource  - Time for fixing and making board too long  - Hardware incompatible  - Camera went wrong |
| Sprint 5:  Completing | - Integration test  - Update documents  - Update test reports  - Create installation guide  - Create User guide  - Implement model for system.  - Prepare slide for presentation  - Print documents  - Make CDs | - Installation guide  - User guide  - Integration test report  - Model of system  - Documents (full)  - CDs  - Presentation slide | - 64 mans-day | - Finish testing | - Bugs of integration testing  - Time delay for making shield of system |

Table B‑9: Software Development Life Cycle

### Increment Detail

#### Sprint 1: Requirement Analysis

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| Collect requirement | Clarify requirement, identify scope and verify with supervisor | TrinhDA, NghiaNC, HoLL, LamNH |
| Write report | Write introduction report | HoLL |
| Buy hardware devices | Choose and buy necessary main hardware design for system | NghiaNC |
| Slide for reporting | Make slide to report what team have done in each week | NghiaNC |

Table B‑10: Sprint 1: Requirement Analysis

#### Sprint 2: Background Preparing

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| Prepare hardware | Install OS for Raspberry, configure camera, GSM | NghiaNC |
| Configure virtual environment | Install Ubuntu and Open CV on Linux | HoLL |
| Create project management plan | Create project plan with detail task  Choose development model | TrinhDA |
| Study Open CV | Study Open CV documents and demo | HoLL, LamNH |
| Study Math lab | Study Math lab documents and demo | LamNH |
| Interface GSM and Raspberry | Configure GSM, interface with Raspberry to send/receive SMS | NghiaNC |
| Write report | Write week reports | NghiaNC, Ho LL |
| Make Slide | Make slide for reporting | NghiaNC |

Table B‑11: Sprint 2: Background Preparing

#### Sprint 3: SMS implement and image uploading

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| Upload image | Study and demo how to upload image from Raspberry to server on the internet | TrinhDA |
| Send SMS via GSM | After upload image successfully, send link of these images via SMS and view. | NghiaNC, LamNH |
| Study distance | Study and demo application of distance sensor | HoLL |
| Capture image | Demo capture image with time delay by using Camera Pi | LamNH |
| Design alarm circuit | Design schematics and PCB of alarm circuit | NghiaNC, HoLL |
| Implement alarm board | Make alarm board and test | NghiaNC |
| Write report | Write week reports | NghiaNC, HoLL |
| Make Slide | Make slide for reporting | NghiaNC |

Table B‑12: Sprint 3: SMS implement and image uploading

#### Sprint 4: Detect Algorithm and power supply circuit implementing

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| Design power supply circuit | Design schematics and PCD of power supply circuit | NghiaNC |
| Implement power supply circuit | Make power supply board and test | NghiaNC |
| Design and implement integrated board | Design PCB and make integrated board | NghiaNC |
| Design test cases | Design test case to prepare for testing | TrinhDA, NghiaNC |
| Integrated system and testing | Integrated all components, test and fix bug | TrinhDA, NghiaNC |
| Fix bugs | Fix bugs after testing | TrinhDA, NghiaNC |
| Write report | Write week reports | NghiaNC |
| Make Slide | Make slide for reporting | NghiaNC |

Table B‑13: Sprint 4: Detect Algorithm and power supply circuit implementing

#### Sprint 5: Completing

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| - Create installation guide | Create guide of installation. | NghiaNC |
| - Create User guide | Create guide for user how to use the system | TrinhDA |
| Alpha Test | Test all system again to make sure it works well  Fix bugs. | TrinhDA, NghiaNC |
| Write documents | Integrate and update all reports to final document | TrinhDA, NghiaNC |

Table B‑14: Sprint 5: Completing

### All Meeting Minutes

All meeting reports are located at:

<https://github.com/cp-scwr/SCWR/tree/master/Meeting_Minutes>

## Coding Convention

### General Naming Rules

* Names to avoid:

Never use the character 'l' (lowercase letter el), 'O' (uppercase letter oh), or 'I' (uppercase letter eye) as single character variable names. Because in some font, we could make a confusion between ‘I’ and ‘L’ (lowercase) or ‘O’ and number 0.

* Constants:

Constants are usually defined on a module level and written in all capital letters with underscores separating words.

Examples include MAX\_OVERFLOW and TOTAL.

### Variable Names

* Global Variables:

Modules that are designed for use via from M import \* should use the \_\_all\_\_ mechanism to prevent exporting global, or use the older convention of prefixing such global with an underscore

### File Names

Name of image files will be followed the format bellows:

* + Image\_DDMMYYYY\_hhmmss

### Function Names

Function names wills be lowercase, with words separated by underscores as necessary to improve readability. “mixedCase” is allowed only in contexts where that's already the prevailing style (e.g. threading.py), to retain backwards compatibility.

Reference: <https://www.python.org/dev/peps/pep0008/#function-names>

### Comments

* Inline comments:

Inline comments will be separated by at least two spaces from the statement. They should start with a # and a single space.

* Block comments:

Each line of a block comment starts with a # and a single space

### Lines

* Maximum line length

The Python standard library is conservative and requires limiting lines to 79 characters (and docstrings/comments to 72). So we decided the limitation all lines to a maximum of 79 characters.

Reference: <https://www.python.org/dev/peps/pep0008/#maximum-line-length>

* Blank lines

Surround top-level function and class definitions with two blank lines. Method definitions inside a class are surrounded by a single blank line. Use blank lines in functions, sparingly, to indicate logical sections

Reference: <https://www.python.org/dev/peps/pep0008/#blanklines>

# Report No. 3 Software Requirement Specification

## User Requirement Specification

### User Requirement

User is a person who can interact with system via SMS by using their mobile. There are some functions user can use:

* Receive alarm messages
* Send monitor messages
* View photos

### System Requirement

System includes:

* Raspberry Pi 3: which is the main processor, can process the algorithm of detecting human, control the camera, alarm system and GSM
* GSM: which is connected to Raspberry, helps User to communicate with Raspberry via SMS.
* Alarm system: a board can make police sound to alarm.
* Camera Pi: which is connected to Raspberry, streams video to Raspberry.

## System Requirement Specification

### External Interface Requirement

#### User interface

* User interface must be simple and easy to use.
* Reuse default interface of mobile producers.

##### Send and Receive SMS

* This UI is the message screen on user’s phone.

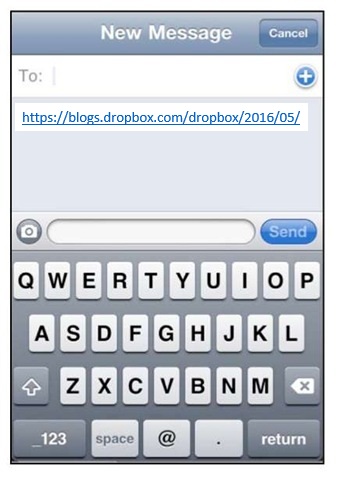
****

Figure C‑1: SMS Manager Interface

##### View Intruder Photos

* Browser screen on user’s phone will display photos which are uploaded to drop-box server before.

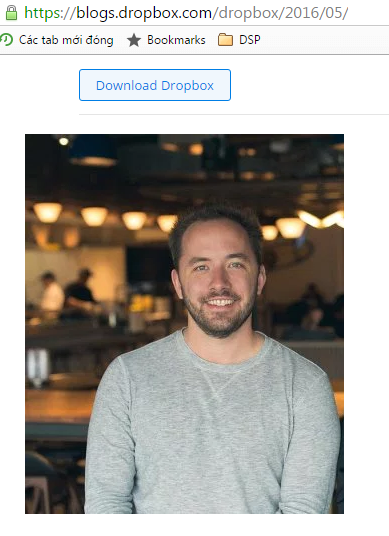
****

Figure C‑2: View Photos Interface

#### Hardware interface

##### Raspberry Pi 3



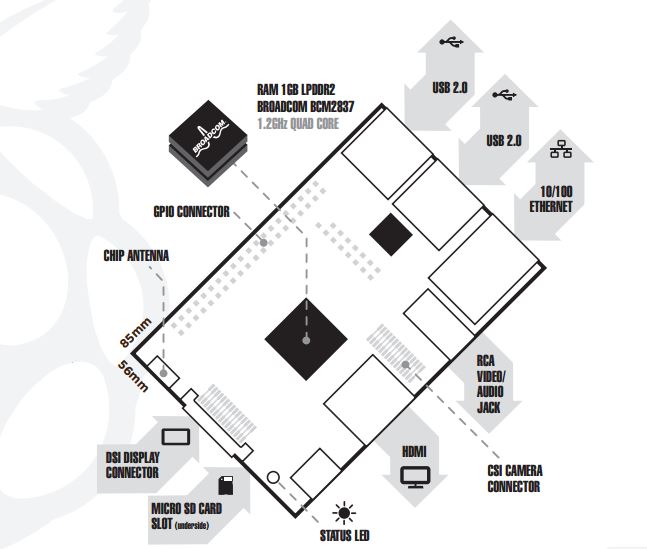


Figure C‑3: Raspberry Pi 3 Model B

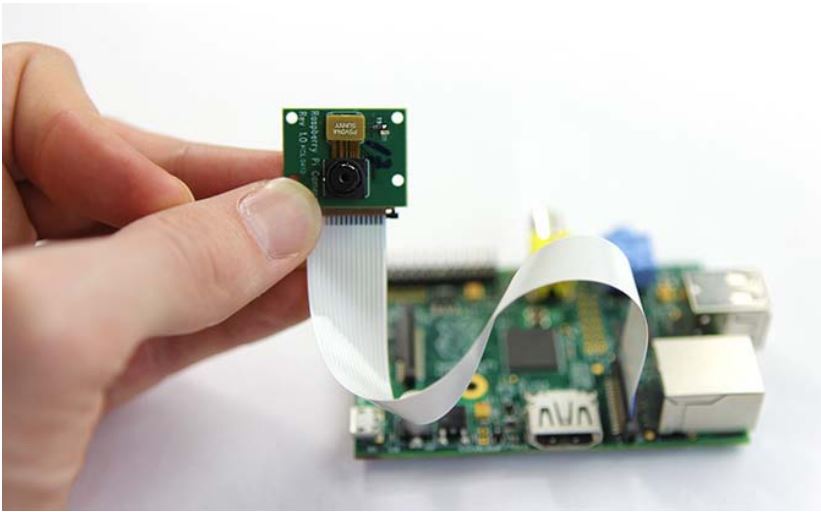
The Raspberry Pi 3 Model B is the third generation Raspberry Pi. This powerful credit-card sized single board computer can be used for many applications and supersedes the original Raspberry Pi Model B+ and Raspberry Pi 2 Model B. Whilst maintaining the popular board format the Raspberry Pi 3 Model B brings you a more powerful processer, 10x faster than the first generation Raspberry Pi. Additionally, it adds wireless LAN & Bluetooth connectivity making it the ideal solution for powerful connected designs.

|  |  |
| --- | --- |
| Specifications | |
| Processor | + Broadcom BCM2387 chipset.  + 1.2GHz Quad-Core ARM Cortex-A53.  + 802.11 b/g/n Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE). |
| GPU | + Dual Core Video-Core IV® Multimedia   Co-Processor. Provides Open GL ES 2.0,  hardware-accelerated OpenVG, and 1080p30  H.264 high-profile decode.  + Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs   with texture filtering and DMA infrastructure. |
| Memory | + 1GB LPDDR2 |
| Operating System | + Boots from Micro SD card, running a version of   the Linux operating system or Windows 10 IoT. |
| Dimensions | + 85 x 56 x 17mm |
| Power | + Micro USB Socket 5.1V, 2.5A |
| Connectors | |
| Ethernet | + 10/100 BaseT Ethernet socket |
| Video Output | + HDMI (rev 1.3 & 1.4)  + Composite RCA (PAL and NTSC) |
| Audio Output | + Audio Output 3.5mm jack, HDMI  + USB 4 x USB 2.0 Connector. |
| GPIO Connector | + 40-pins 2.54 mm (100 mil) expansion header:   2x20 strip.  + Providing 27 GPIO pins as well as +3.3 V, +5 V   and GND supply lines. |
| Camera Connector | + 15-pins MIPI Camera Serial Interface (CSI-2) |
| Display Connector | + Display Serial Interface (DSI) 15 ways flat flex cable connector with two data lanes and a clock lane. |
| Memory Card Slot | + Push/pull Micro SDIO |

Table C‑1: Raspberry Pi 3 Specification

##### Camera Pi

Featuring the same 5 mega pixel image sensor as the standard   
Raspberry Pi camera with the infrared cut-off filter removed to increase IR light sensitivity. The Pi NoIR is compatible with Raspberry Pi Model A & B and it also provides high definition, high sensitivity, low crosstalk and low noise image capture in a small and lightweight design. The camera module connects to the Raspberry Pi board via the CSI connector which is capable of extremely high data rates and it exclusively carries pixel data to the BCM2835 processor.



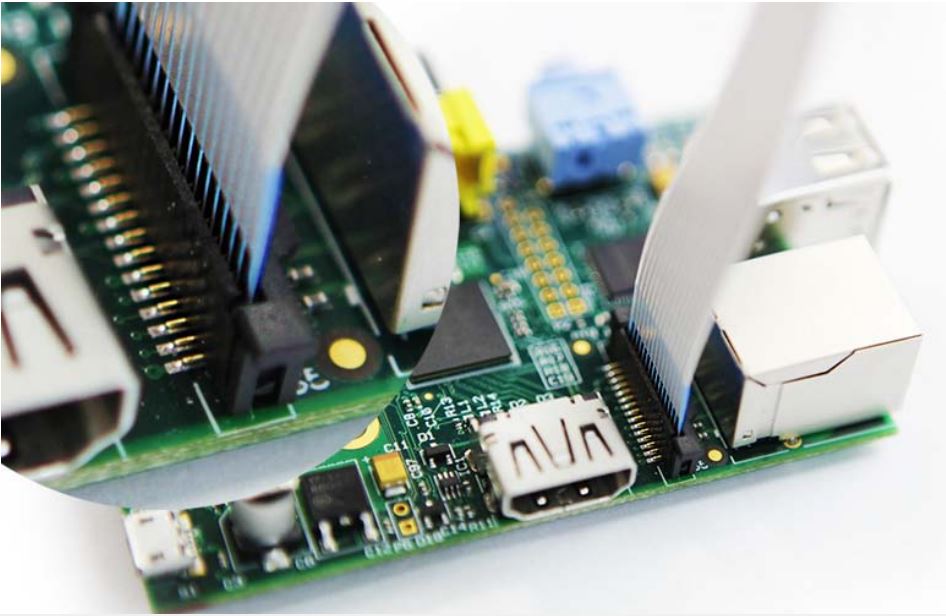


Figure C‑4: Camera Pi

|  |  |
| --- | --- |
| Specifications | |
| Image Sensor | + Omnivision 5647 CMOS image sensor in a fixed-focus  module with IR blocking filter removed |
| Resolution  Still picture resolution | + 5-megapixels  + 2592 x 1944 |
| Max image transfer rate | + 1080P: 30fps (encode and decode)  + 720p: 60fps |
| Connection to Raspberry Pi | + 15 Pin ribbon cable, to the dedicated 15-pin MIPI   Camera Serial Interface (CSI-2) |
| Image control functions | + Automatic exposure controlling + Automatic white balance + Automatic band filter + Automatic 50/60 Hz luminance detection + Automatic black level calibration |
| Temp range | + Operating: -30° to 70°  + Stable image: 0° to 50° |
| Lens size | + 1/4” |
| Dimensions | + 20 x 25 x 10mm |
| Weight | 3g |

Table C‑2: Camera Pi Specification

##### GSM Sim900A



Figure C‑5: SIM900A GSM/GPRS Module

The SIM900 is a complete Quad-band GSM/GPRS solution in a  
SMT module which can be embedded in the customer applications. Featuring an industry-standard interface, the SIM900 delivers GSM/GPRS 850/900/1800/1900MHz performance for voice, SMS, Data, and Fax in a small form factor and with low power consumption. With a tiny configuration of 24mm x 24mm x 3 mm, SIM900 can fit almost all the space requirements in your M2M application, especially for slim and compact demand of design.

* SIM900 is designed with a very powerful single-chip processor integrating AMR926EJ-S core
* Quad - band GSM/GPRS module with a size of 24mmx24mmx3mm
* SMT type suit for customer application
* An embedded Powerful TCP/IP protocol stack
* Based upon mature and field-proven platform, backed up by our support service, from definition to design and production

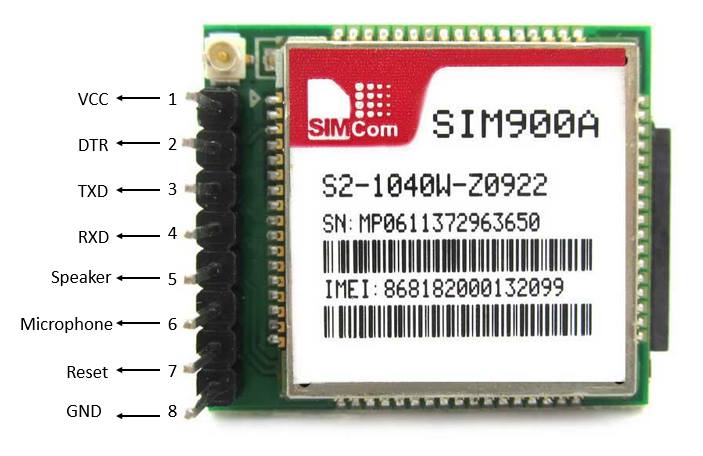


Figure C‑6: SIM900A Pins Map

##### Power Supply

* Adapter 9V - 0.6A

We chose the value of power for alarm circuit is 9V and we also have an adapter 9V already.



Figure C‑7: Adapter 9V – 0.6A

* Power supply 9V to 5V

Our alarm circuit uses power supply with 9V but Raspberry and GSM only work with 5V. So, we have to convert from 9V to 5V. We use LM2576 to regulate the voltage. We will talk more detail about it in next parts.



Figure C‑8: Power Supply circuit example

##### Alarm Circuit

SCWR have alert function and we decided to use police sound with two IC555s. This alarm circuit between 5V – 12V, the values of power supply is based on your output. We will talk more detail about it in next parts.

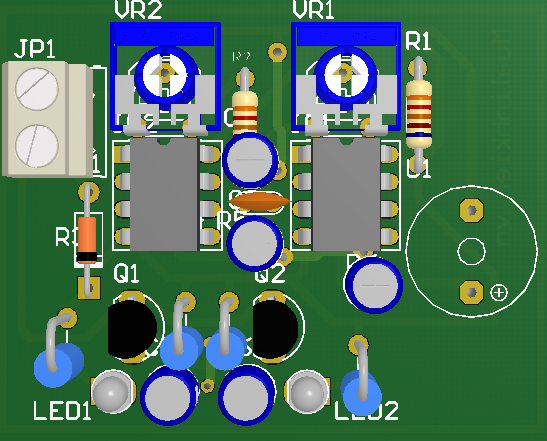


Figure C‑9: 3D Simulation of Alarm circuit

#### Software Interface

##### Python Shell

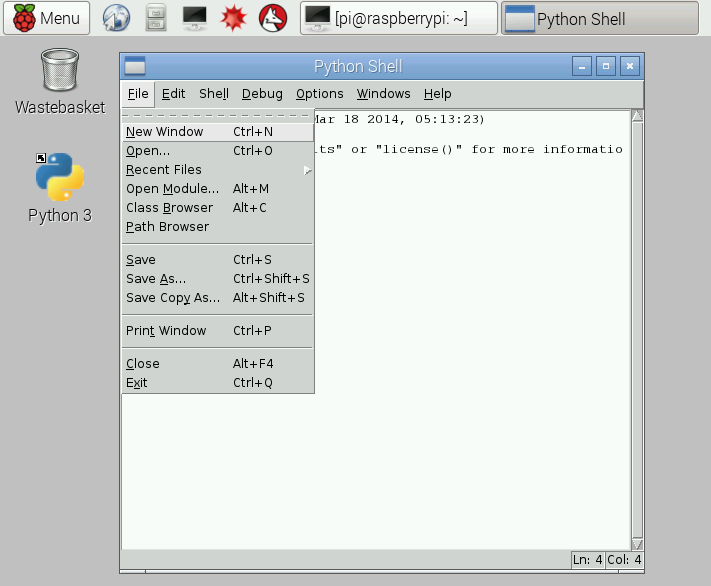


Figure C‑10: Python Shell

* Python Shell for programming python language
* Python Shell version 3.5.2

##### Terminal

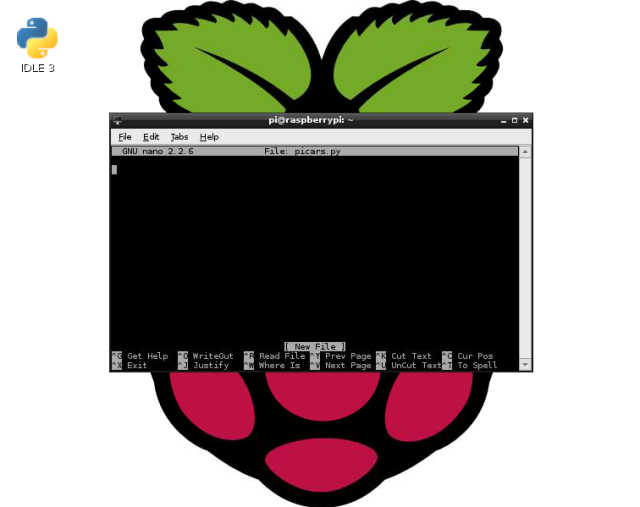


Figure C‑11: Terminal

* Terminal is a default software of Raspbian OS. Terminal is used to command what OS will do. Terminal is similar with CMD of Windows.
* In this case, Terminal is use to compile code and execute.

#### Communication Protocol

### System Overview Use Case

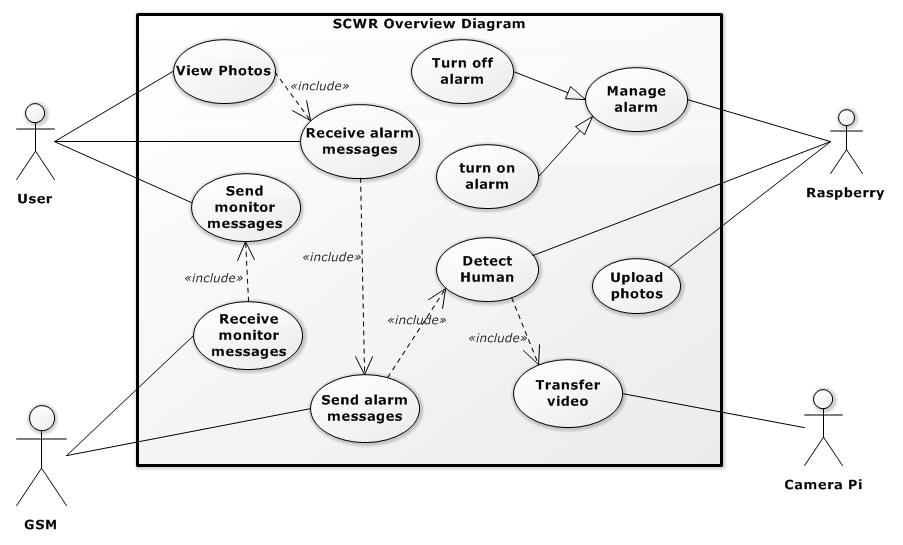


Figure C‑12: System Overview Use Case

### List of Use Case

#### <User> Overview Use Case

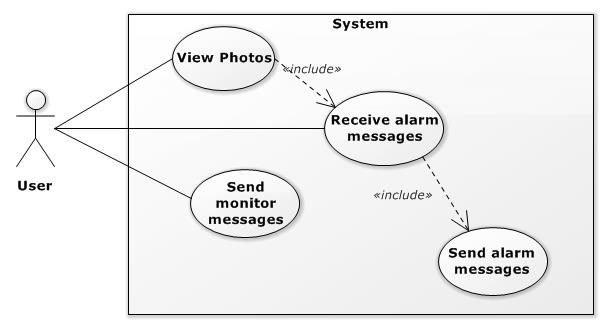


Figure C‑13: User Overview Use Case

##### <User> Receive Alarm Messages Use Case

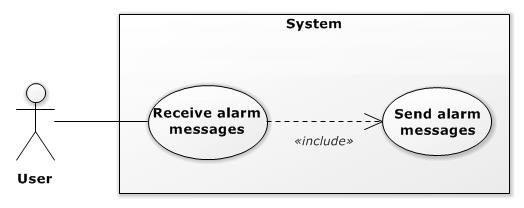


Figure C‑14: User Receive Alarm Messages Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC01** | | | |
| **Use Case No.** | **UC01** | **Use Case Version** | 2.0 |
| **Use Case Name** | Receive Alarm Messages | | |
| **Author** | Nguyen Chi Nghia | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - User  **Summary:**  - User receive alarm messages with the link of intruder ‘s images inside.  **Goal:**  - User has already had the link.  **Triggers:**  - GSM sends alarm messages.  **Preconditions:**  - GSM has already sent alarm messages.  **Post Conditions:**  - Success: New SMS received on User ‘s mobile.  - Fail: No SMS received.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access SMS manager on mobile | Display SMS manager | | 2 | Touch on “New incoming message” | Display New message content |   **Alternative Scenario:**  **-** N/A  **Exceptions:**  - N/A  **Relationships:**  - Send Alarm Messages  **Business Rules:** N/A | | | |

##### <User> View Photos Use Case

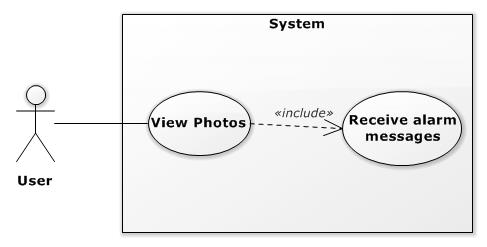


Figure C‑15: User View Photos Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC02** | | | |
| **Use Case No.** | **UC02** | **Use Case Version** | 2.0 |
| **Use Case Name** | View Photos | | |
| **Author** | Nguyen Chi Nghia | | |
| **Date** | 08/06/2016 | **Priority** | Low |
| **Actor:**  - User  **Summary:**  - User accesses the link of intruder ‘s images to view.  **Goal:**  - User has already seen who intruders are.  **Triggers:**  - User touch on the link.  **Preconditions:**  - GSM has already sent alarm messages. Messages arrived to user ‘s mobile successfully.  - User connected to the internet (Wi-Fi/3G)  **Post Conditions:**  - Success: User can see photos of intruder on Drop box  - Fail: No photo shown.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access SMS manager on mobile | Display SMS manager | | 2 | Touch on “New incoming message” | Display New message content | | 3 | Touch on image ‘s link | Redirect to image ‘s address and show image |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access SMS manager on mobile | Display SMS manager | | 2 | Touch on “New incoming message” | Display New message content | | 3 | Copy image ‘s link | “Copied” message box is shown | | 4 | Paste to browser access link | Redirect to image ‘s address and show image |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | No network connected | No Network display screen | | 2 | Server is down | Error page is displayed |   **Relationships:**  - Receive Alarm Messages  **Business Rules:**  - N/A | | | |

##### <User> Send Monitor Messages Use Case

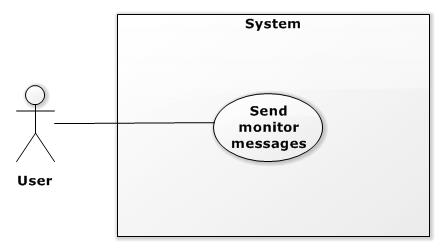


Figure C‑16: User Send Monitor Message Use Case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USE CASE – UC03** | | | | |
| **Use Case No.** | **UC03** | **Use Case Version** | | 2.0 |
| **Use Case Name** | Send Monitor Messages | | | |
| **Author** | Nguyen Chi Nghia | | | |
| **Date** | 08/06/2016 | **Priority** | Normal | |
| **Actor:**  - User  **Summary:**  - After view intruder photo. In case the intruders are relative, user can use this case to turn off alarm.  **Goal:**  - User send command to turn off alarm via SMS.  **Triggers:**  - Intruder has found  **Preconditions:**  - Alarm is still running.  **Post Conditions:**  - Success: “SMS sent” is shown.  - Fail: Failed to send SMS.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User touch on reply message | Display Message Editor screen | | 2 | Write “Off” | Display “Off” on screen | | 3 | Touch on “send” button | “SMS sent” message box is shown |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Access to SMS manager | Display SMS manager screen | | 2 | Touch on “New Message” | Redirect to Message Editor screen | | 3 | Write “Off” | Display “Off” on screen | | 4 | Touch on “send” button | “SMS sent” message box is shown |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Network error | “Failed to send message” is shown | | 2 | User doesn’t have enough money | “Failed to send message” is shown |   **Relationships:**  - NA  **Business Rules:**  - N/A | | | | |

#### <GSM> Overview Use Case

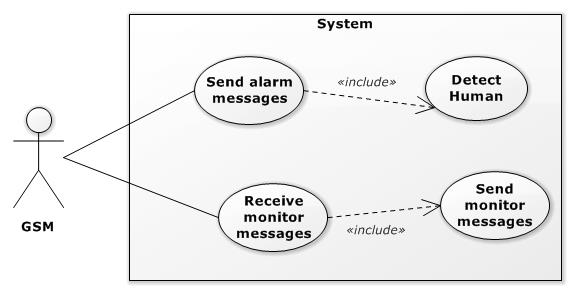


Figure C‑17: GSM Overview Use Case

##### <GSM> Send Alarm Messages Use Case

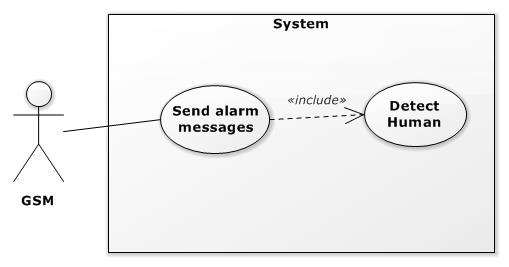


Figure C‑18: GSM Send Alarm Messages Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC04** | | | |
| **Use Case No.** | **UC04** | **Use Case Version** | 2.0 |
| **Use Case Name** | Send Alarm Messages | | |
| **Author** | Nguyen Chi Nghia | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - GSM  **Summary:**  - GSM receive AT Command from Raspberry and then send alarm messages to user  **Goal:**  - GSM send SMS successfully  **Triggers:**  - Raspberry send AT Command to send SMS  **Preconditions:**  - Detect Human return “true”  - Raspberry sends AT Command successfully  **Post Conditions:**  - Success: GSM sends message successfully and return “OK” to Raspberry  - Fail: GSM sends fail.  **Main Success Scenario:**  - N/A  **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Sim doesn’t have enough money | “OK” isn’t shown | | 2 | Network Error | “OK” isn’t shown |   **Relationships:**  - Detect Human  **Business Rules:**  - N/A | | | |

##### <GSM> Receive Monitor Messages Use Case

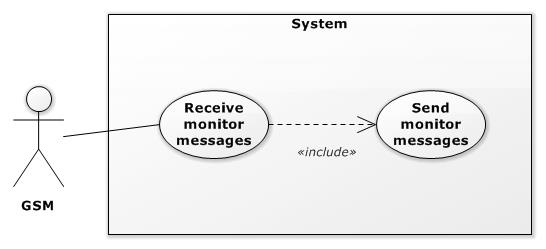


Figure C‑19: GSM Receive Monitor Messages Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC05** | | | |
| **Use Case No.** | **UC05** | **Use Case Version** | 2.0 |
| **Use Case Name** | Receive Monitor Messages | | |
| **Author** | Nguyen Chi Nghia | | |
| **Date** | 08/06/2016 | **Priority** | Normal |
| **Actor:**  - GSM  **Summary:**  - User sends monitor messages and then GSM will receive them. After that it will transfer to Raspberry.  **Goal:**  - GSM receives monitor messages from user to Raspberry can process  **Triggers:**  - User sends monitor messages successfully  **Preconditions:**  - User has already sent messages.  **Post Conditions:**  - Success: “New message content is transferred when Raspberry request.  - Fail: No message shown  **Main Success Scenario:**  - N/A  **Alternative Scenario:**  **-** N/A  **Exceptions:**  **-** N/A  **Relationships:**  - NA | | | |

#### <Camera Pi> Overview Use Case

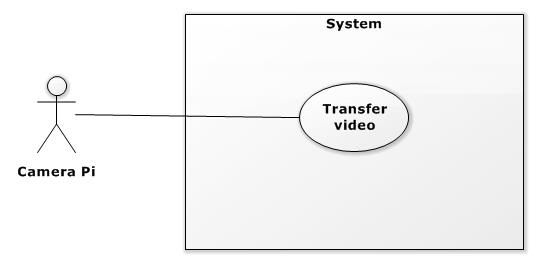


Figure C‑20: <Camera Pi> Overview Use Case

##### <Camera Pi> Transfer Video

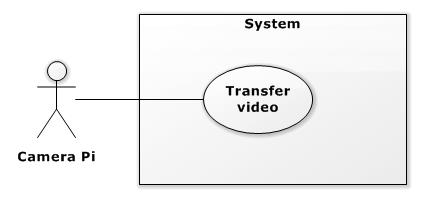


Figure C‑21: <Camera Pi> Transfer Video Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC06** | | | |
| **Use Case No.** | **UC06** | **Use Case Version** | 2.0 |
| **Use Case Name** | Transfer Video | | |
| **Author** | Nguyen Chi Nghia | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - Camera Pi  **Summary:**  - This function is the condition to run detect algorithm  **Goal:**  - Raspberry receives video from Camera Pi  **Triggers:**  - Raspberry sends command “raspistill -tl 500 -t 9999999… “  **Preconditions:**  - Camera Pi has been plugged in Raspberry.  **Post Conditions:**  - Success: Raspberry can view streaming video  - Fail: No video is shown  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | On terminal, write “sudo raspistill -tl 500…” | Display typing command | | 2 | Press “Enter” button | Terminal runs this command, streaming screen will be shown |   **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Wrong command | Display Errors | | 2 | Unplugged Camera | Display Errors |   **Relationships:**  - N/A  **Business Rules:**  **-** N/A | | | |

#### <Raspberry> Overview Use Case

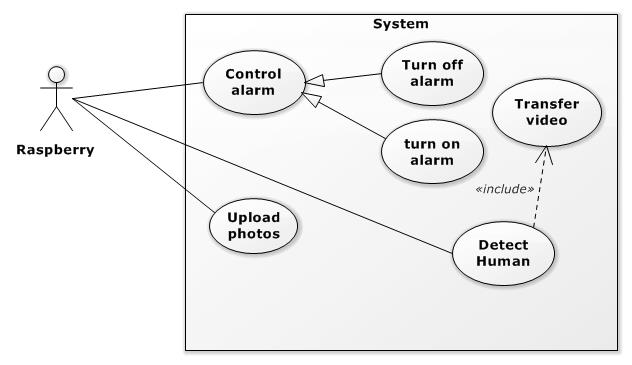


Figure C‑22: <Raspberry> Overview Use Case

##### <Raspberry> Upload Photos Use Case

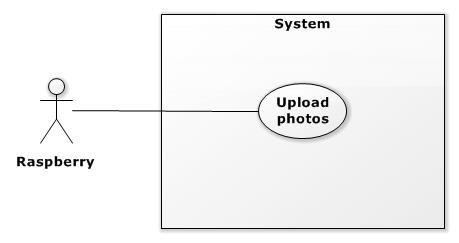


Figure C‑23: <Raspberry> Upload Photos Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC07** | | | |
| **Use Case No.** | **UC07** | **Use Case Version** | 2.0 |
| **Use Case Name** | Upload Photos | | |
| **Author** | Nguyen Chi Nghia | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - Raspberry  **Summary:**  - After camera captures intruders, Raspberry uploads them to server and then send the link to user.  **Goal:**  - Upload photos successfully.  **Triggers:**  - Camera has already captured.  **Preconditions:**  - Raspberry has been connected to the internet.  **Post Conditions:**  - Success: These photos appear in server.  - Fail: No photo appears.  **Main Success Scenario:**  **-** N/A  **Alternative Scenario:**  **-** N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | No network connected | No Network display screen | | 2 | Server is down | Error page is displayed |   **Relationships:**  - N/A  **Business Rules:**  **-** N/A | | | |

##### <Raspberry> Detect Human Use Case

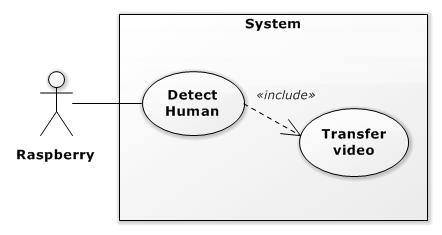


Figure C‑24: <Raspberry> Detect Human Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC08** | | | |
| **Use Case No.** | **UC08** | **Use Case Version** | 2.0 |
| **Use Case Name** | Detect Human | | |
| **Author** | Nguyen Chi Nghia | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - Raspberry  **Summary:**  - Detect any motion object in surveillance area and distinguish does they’re humans  **Goal:**  - Detect any intruder who appears in surveillance area  **Triggers:**  - Camera transfers video successfully  **Preconditions:**  - N/A  **Post Conditions:**  - N/A  **Main Success Scenario:**  **-** N/A  **Alternative Scenario:**  **-** N/A  **Exceptions:**  - N/A  **Relationships:**  - Transfer Video  **Business Rules:**  - N/A | | | |

##### <Raspberry> Control Alarm Use Case

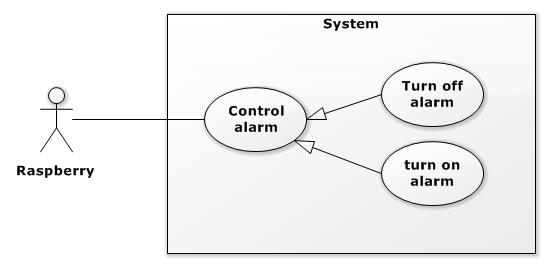


Figure C‑25: <Raspberry> Control Alarm Use Case

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC09** | | | |
| **Use Case No.** | **UC09** | **Use Case Version** | 2.0 |
| **Use Case Name** | Control Alarm | | |
| **Author** | Nguyen Chi Nghia | | |
| **Date** | 08/06/2016 | **Priority** | High |
| **Actor:**  - Raspberry  **Summary:**  - To turn on or off alarm.  **Goal:**  - Alarm begins working when detect finished with intruder found and stop working when receive monitor message from user.  **Triggers:**  - Raspberry output “Low” level on pin No.19  **Preconditions:**  - N/A  **Post Conditions:**  - Success: pin No.19 output “Low” level and police sound appears.  - Fail: No sound heard  **Main Success Scenario:**  - N/A  **Alternative Scenario:**  **-** N/A  **Exceptions:**  - N/A  **Relationships:**  - N/A  **Business Rules:**  - N/A | | | |

## Software System Attribute

### Usability

### Reliability

### Availability

### Security

### Maintainability

### Portability

### Performance

# Report No. 4 Software Design Description

## Design Overview

This document describes the technical and user interface design of SCWR System.  
It includes the architectural design, the detailed design of common functions and  
business functions.

- The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.

- The detailed design describes static and dynamic structure for each component and functions. It includes the detailed description of each component and calculations of them, sequence diagrams, flow chart for algorithms

- Document overview:  
• Section 2: gives an overall description of the system architecture design.  
• Section 3: gives component diagrams that describe the connection and  
integration of the system.  
• Section 4: gives the detail design description which includes class diagram,  
class explanation, and sequence diagram to details the application functions.  
• Section 5: describe screens design.  
• Section 6: describe a fully attributed ERD.  
• Section 7: describe algorithms**.**

## System Architectural Design

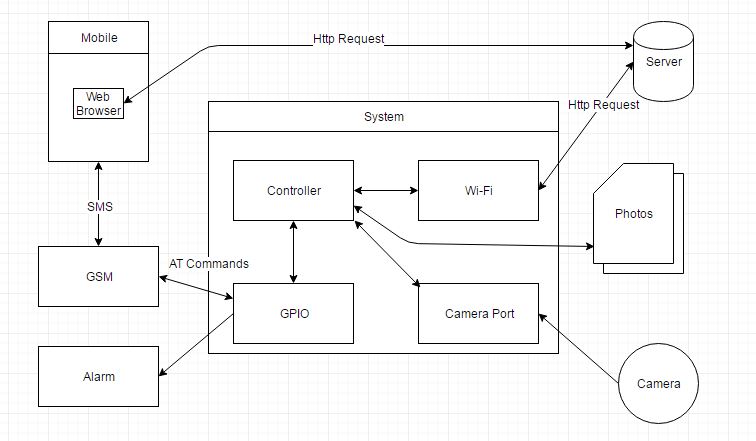


Figure D‑1: System Overview Architectural

## Component Diagram

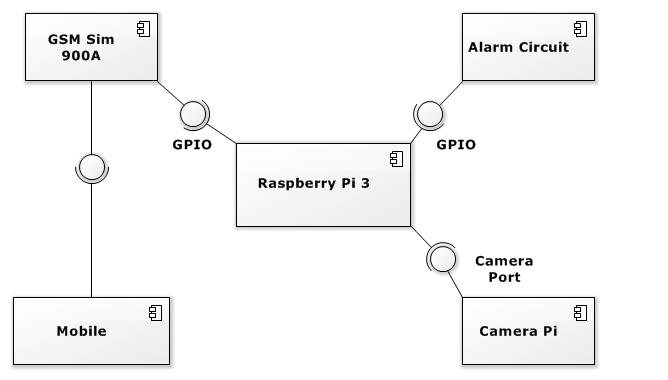


Figure D‑2: Component Diagram

|  |  |
| --- | --- |
| **Component Dictionary: Describe Component** | |
| **Component Name** | **Description** |
| Raspberry Pi 3 | Core processor, control GSM, Alarm, Camera Pi. Process the detect algorithm and upload intruder to server |
| GSM Sim 900A | Sim module: send and receive SMS between Raspberry and User. |
| Alarm Circuit | A circuit, which can make police sound to alarm |
| Camera Pi | A peripheral of Raspberry, stream video to Raspberry. |
| Mobile | User uses this component to send and receive SMS with Raspberry. It’s also used to view photos on server. |

Table D‑1: Component Dictionary

## Detailed Description

### Block Diagram

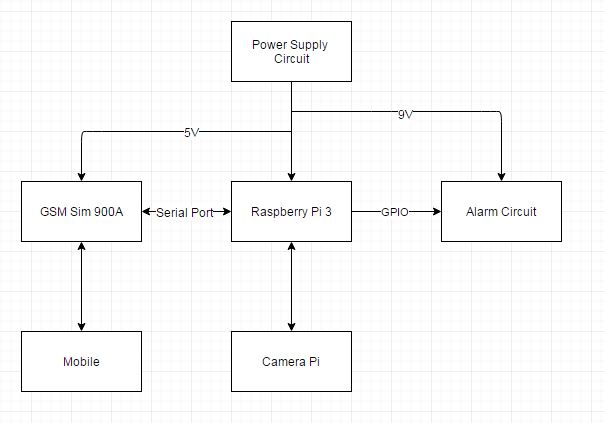


Figure D‑3: Block Diagram

|  |  |
| --- | --- |
| **Block Dictionary: Describe Block** | |
| **Block Name** | **Description** |
| Raspberry Pi 3 | Core processor, control GSM, Alarm, Camera Pi. Process the detect algorithm and upload intruder to server. |
| GSM Sim 900A | Sim module: send and receive SMS between Raspberry and User. GSM 900A is connected with Raspberry via GPIO (TX, RX) and communicate via Serial Port. |
| Alarm Circuit | A circuit, which can make police sound to alarm |
| Camera Pi | A peripheral of Raspberry, stream video to Raspberry. |
| Mobile | User uses this component to send and receive SMS with Raspberry. It’s also used to view photos on server. |

Table D‑2: Block Dictionary

### Interaction Diagram

## Interface

### Component Interface

## Algorithms

### Document Breakdown

#### Definition

#### Define Problem

#### Solution

#### Complexity

#### Flowchart

# Report No.5 System Implementation & Testing

## Introduction

## Test Plan

### Features to be tested

* Detect Motion
* Detect Size
* Turn on/off alarm
* Send/Receive SMS
* View Photos

### Features not to be tested

* Stream video
* Upload photo

## System Testing Test Case

### Raspberry Test Case

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Name** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Result** | **Test Date** | **Note** |
| **1. Motion Testing** | | | | | | | |
| MT\_1 | Motion Test | To test the accuracy of detecting motion algorithm | 1. Run detect motion algorithm  2. Tester keep moving in front of the camera | - Detect screen is shown  - The moving objects have been covered by the border |  | 30/06 – 03/07/2016 |  |
| MT\_2 | Multi Motion Test | To test the accuracy of detecting motion algorithm with multi-people | 1. Run detect motion algorithm  2.There are three or four people moving in front of the camera | - Detect screen is shown  - On detect screen, the border must cover anything moves in streaming video |  | 30/06 – 03/07/2016 |  |
| **2. Size Testing** | | | | | | | |
| ST\_1 | Lower Size Test | To test the accuracy of detecting size algorithm | 1. Set the standard size of object  2. Run detect size algorithm with available images which have size are smaller than the standard | - The processing of algorithm is shown to console  - The algorithm will return false to console. |  | 06 - 10/07/2016 |  |
| ST\_2 | Larger Size Test | 1. Set the standard size of object  2. Run detect size algorithm with available images which have size are larger than the standard | - The processing of algorithm is shown to console  - The algorithm will return true to console. |  |  |  |
| **3. Control Alarm Testing** | | | | | | | |
| CA\_1 | Turn on Alarm | To test Raspberry to turn on alarm | 1. Connect alarm signal pin to pin No.19 of Raspberry  2. Output pin No.19 with High level. | - Alarm works successfully  - The loudness of alarm is acceptable |  | 25 – 30/07/2016 |  |
| CA\_2 | Turn off Alarm | To test Raspberry to turn off alarm | 1. Connect alarm signal pin to pin No.19 of Raspberry  2. Output pin No.19 with Low level. | - Alarm stops working.  - No sounds play. |  |  |
| **4. Upload Photos Testing** | | | | | | | |
| UP\_1 | Upload Photos | To test Raspberry to upload photos | 1. Connect Raspberry to the internet  2. Choose photos directory  3. Run algorithm of uploading photos | - Internet is available  - Photos appear in server  - The link of photos is returned |  | 25-26/06/2016 |  |

Table E‑1: Raspberry Test Case

### GSM Test Case

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Test Case Procedure** | **Expected output** | **Result** | **Test Date** | **Note** |
| Send SMS | To test the sending SMS function of GSM | 1. Connect Raspberry and GSM via Serial Port  2. Open Serial Port  3. Send AT Commands to send SMS | - User receives SMS successfully. |  | 30/06 – 03/07/2016 |  |
| Receive SMS | To test the receiving SMS function of GSM | 1. User send an SMS to GSM  2. Connect Raspberry and GSM via Serial Port  3. Open Serial Port  4. Send AT Commands to read SMS | - SMS content is shown correctly. |  | 30/06 – 03/07/2016 |  |

Table E‑2: GSM Test Case

### User Test Case

### Integration Test Case