Analysis Models for Safehome System

Team 07 (Choonghwi Lee, Wookjae Byun)

Table of Contents

1. Introduction

This document contains Analysis mode level contents of Safehome Project. In analysis model process, it creates many possible use cases, and those use cases can be specified with analyzing its primary and secondary actor, goal of that use case, preconditions of that user cases, triggers, scenario(it is same as use case procedure), and possible exceptions and priority, open issues. Also, visualize of use cases can be use “swimlane diagram”. Swimlane diagram show each parts of use case scenario as box, and arrows between box shows flow between those part of scenarios.

2. Use cases and its diagrams

This requirements contain technical system functionalities and essential background of system running(running environment, security features, development environments).

2.1 Use cases - On / Off (Use case #1~#2)

2.1.1 Use case diagram

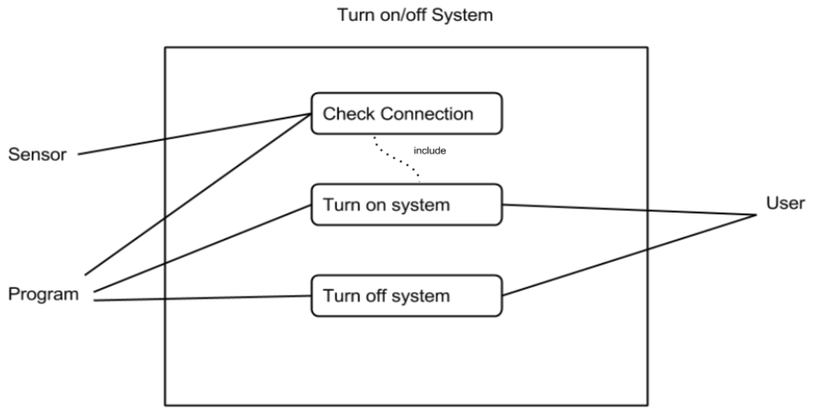

Figure 1: Use case diagram for Use case #1~#2

2.1.2 Use case 1: Turn on System

Name	Turn on system
Actor	**Primary : User** Secondary : Program, Sensors
Goal of Usage	Turn on all system and make the system available.
Preconditions	1. PC is running and Safehome system is installed on PC. 2. All sensors and cameras should running.
Trigger	1. User runs the Safehome program.

Scenario	1. User runs the Safehome program. 2. Require password from user. 3. Program checks that all sensors are running and connected through wireless network. If some sensors are not running or not connected, show alert message on PC. 4. Turn on control panel. 5. Notify for disconnected sensors to user if disconnected sensors exist.
Exceptions	1. (At 1) When the Safehome program is already running: Send alert message and do not run new the Safehome program. 2. (At 2) User enter incorrect password: Print error message and user select (Retry input password) operation or (Exit program) operation.
Priority	High
Open Issues	-

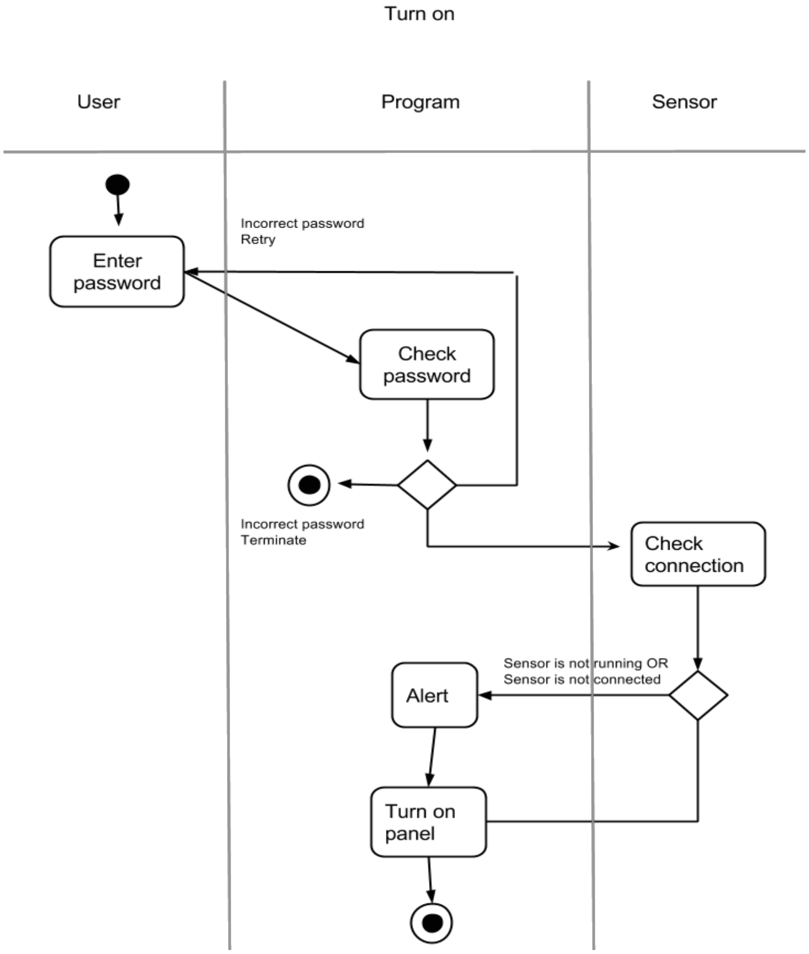

Figure 2: Swimlane diagram for Use case #1

2.1.3 Use case 2: Turn off System

Name	Turn off system
Actor	**Primary : User** Secondary : Program
Goal of Usage	Turn off the Safehome system.
Preconditions	1. Program is running. 2. Safehome system is ‘disarm’ state.
Trigger	1. User terminate the Safehome program.
Scenario	1. User turns off the Safehome program. 2. Stop receiving interactions from users. 3. Terminate Safehome program.
Exceptions	1. (On 3) Some process is executing on the system : Print error message and wait until all processes are terminated. (e.g. Program is still sending log or recorded videos to storage server)
Priority	High
Open Issues	-

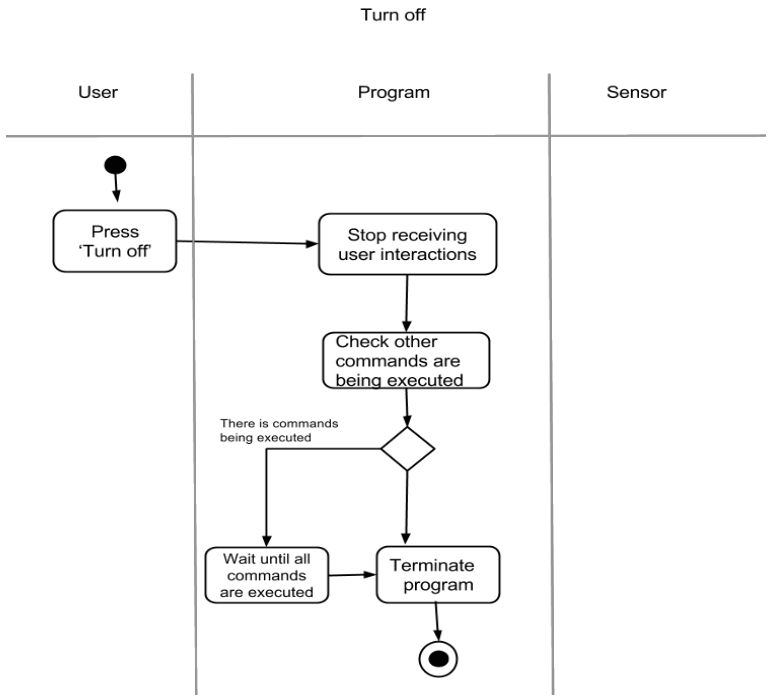
Turn off
User
Program
Sensor
Press ‘Turn off’
Stop receiving user interactions
Check other commands are being executed
There is commands being executed
Wait until all commands are executed
Terminate program

Figure 3: Swimlane diagram for Use case #2

2.2 Use cases - Setting (Use case #3~#11)

2.2.1 Use case diagram

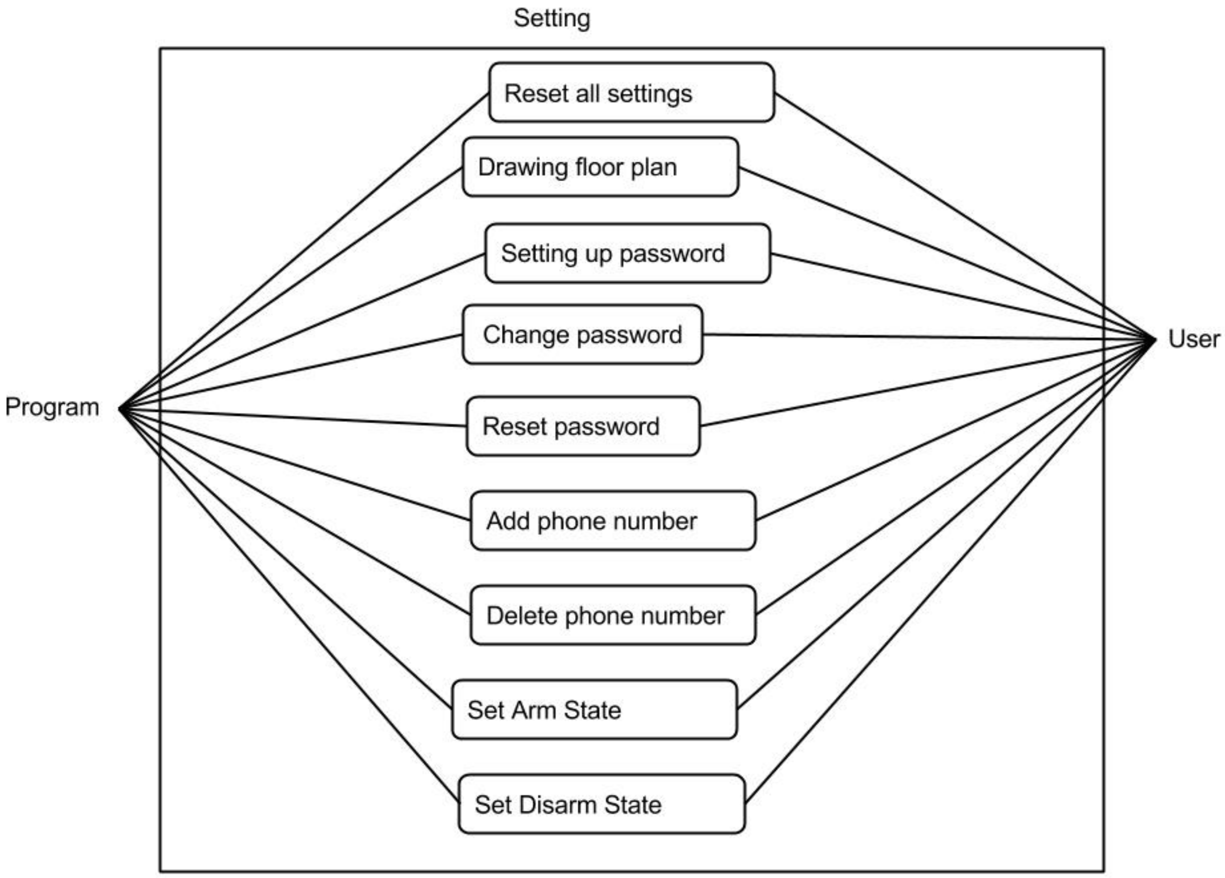

Figure 4: Use case diagram for Use case #3~#11

2.2.2 Use case 3: Drawing floor plan

Name	Drawing floor plan
Actor	**Primary : User** Secondary : Program
Goal of Usage	Set floor plan for home.
Preconditions	1. Program is running.
Trigger	1. System is initially installed 2. User selects ‘reset floor plan’

Scenario	1. If reset (trigger #2), remove previous floor plan. 2. User inputs room structures. 3. User inputs place of all sensors. 4. Program decide and show ‘Safe zone’ from input room structures and place of all cameras. 5. User press ‘save’ button to save current floor plan. 6. Save current floor plan in computer.
Exceptions	-
Priority	High
Open Issues	-

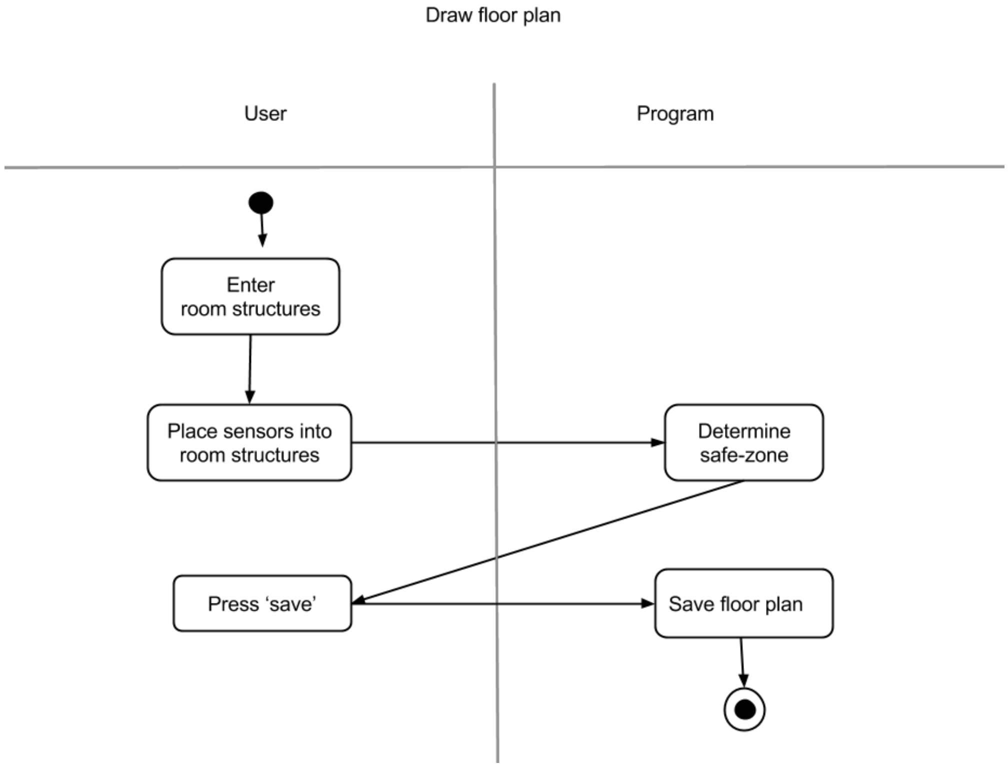

Figure 5: Swimlane diagram for Use case #3

2.2.3 Use case 4: Setting up password

Name	Setting up password
Actor	**Primary : User** Secondary : Program
Goal of Usage	Setup password for authorization.
Preconditions	1. Program is running.
Trigger	1. System initially installed. 2. Password is reseted.
Scenario	1. User enter user's new password. 2. User re-enter user's new password. 3. Program saves new password in system.
Exceptions	1. (On 2) Two password inputs are not equal each other : Print error message and retry setting up password procedure.
Priority	High
Open Issues	-

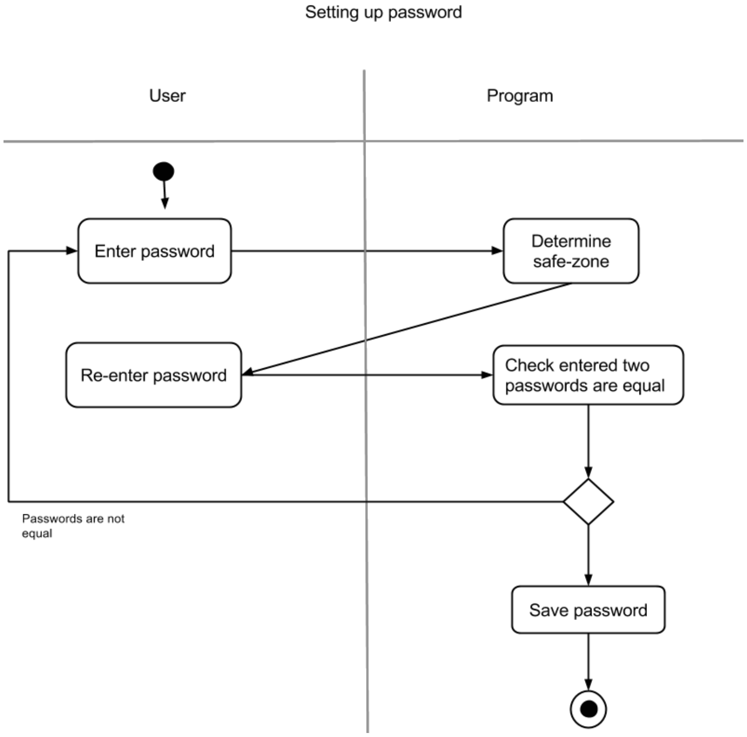

Figure 6: Swimlane diagram for Use case #4

2.2.4 Use case 5: Change password

Name	Change password
Actor	**Primary : User** Secondary : Program
Goal of Usage	Change password to new password.
Preconditions	1. System is booted up.
Trigger	1. User selected ‘change password’.
Scenario	1. User inputs old password. 2. User inputs new password.
Exceptions	1. (On 1) If old password is not correct : Print error message and retry input old password. 2. (On 2) If new password is same as old password : Print error message and retry input new password.
Priority	Medium
Open Issues	-

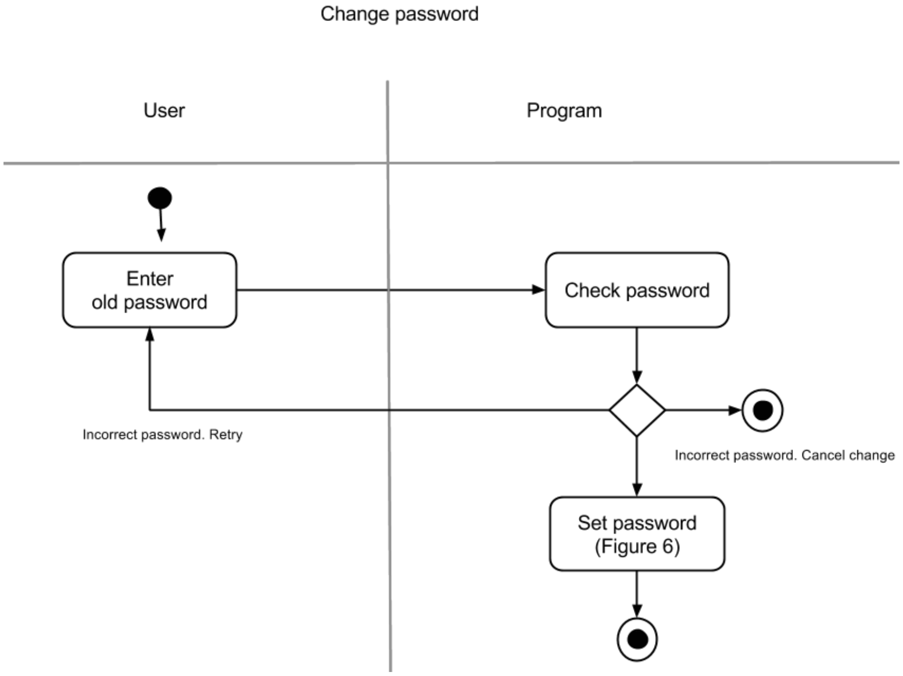

Figure 7: Swimlane diagram for Use case #5

2.2.5 Use case 6: Reset password

Name	Reset password
Actor	**Primary : User** Secondary : Program
Goal of Usage	Reset password to password-unset state.
Preconditions	1. Program is running
Trigger	1. User selected “reset password”.
Scenario	1. User select one phone number from phone number list. 2. Program sends SMS message to selected phone number with validation code. 3. User inputs validation code to program. 4. Program delete registered password.
Exceptions	1. (On 3) User entered incorrect validation code. Print error message and wait for correct validation code. 2. (On 3) User enters validation code after 3 minutes. Print error message and terminate resetting password.
Priority	Medium
Open Issues	-

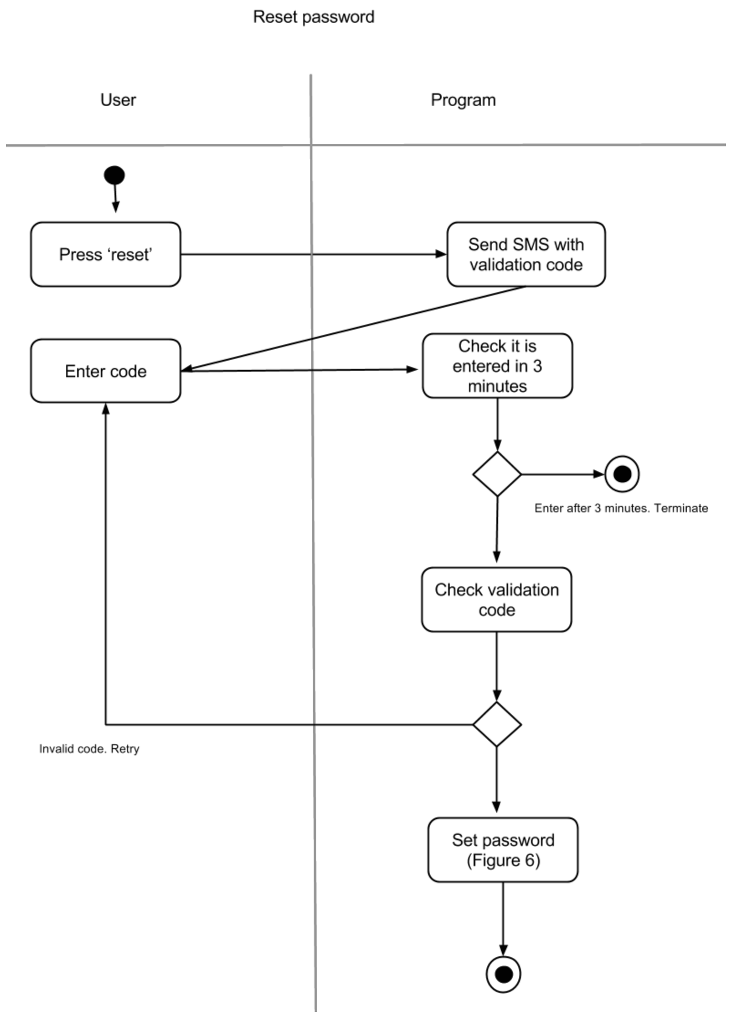

Figure 8: Swimlane diagram for Use case #6

2.2.6 Use case 7: Add phone number

Name	Add phone number
Actor	**Primary : User** Secondary : Program
Goal of Usage	Add phone number for urgent alert notification. Multiple phone numbers are available. At least one phone number must be registered.
Preconditions	1. Program is running.
Trigger	1. System is initially installed. 2. User selects 'Add phone numbers'.

Scenario	1. User inputs phone number. 2. Program checks phone input number is correct format. 3. Program adds input phone number to phone number list. 4. Program saves list of phone number in computer.
Exceptions	1. (On 2) Input phone number is incorrect format : Print error message and reinput phone number. 2. (On 2) Input phone number is already exists : Print message and terminate.
Priority	High
Open Issues	-

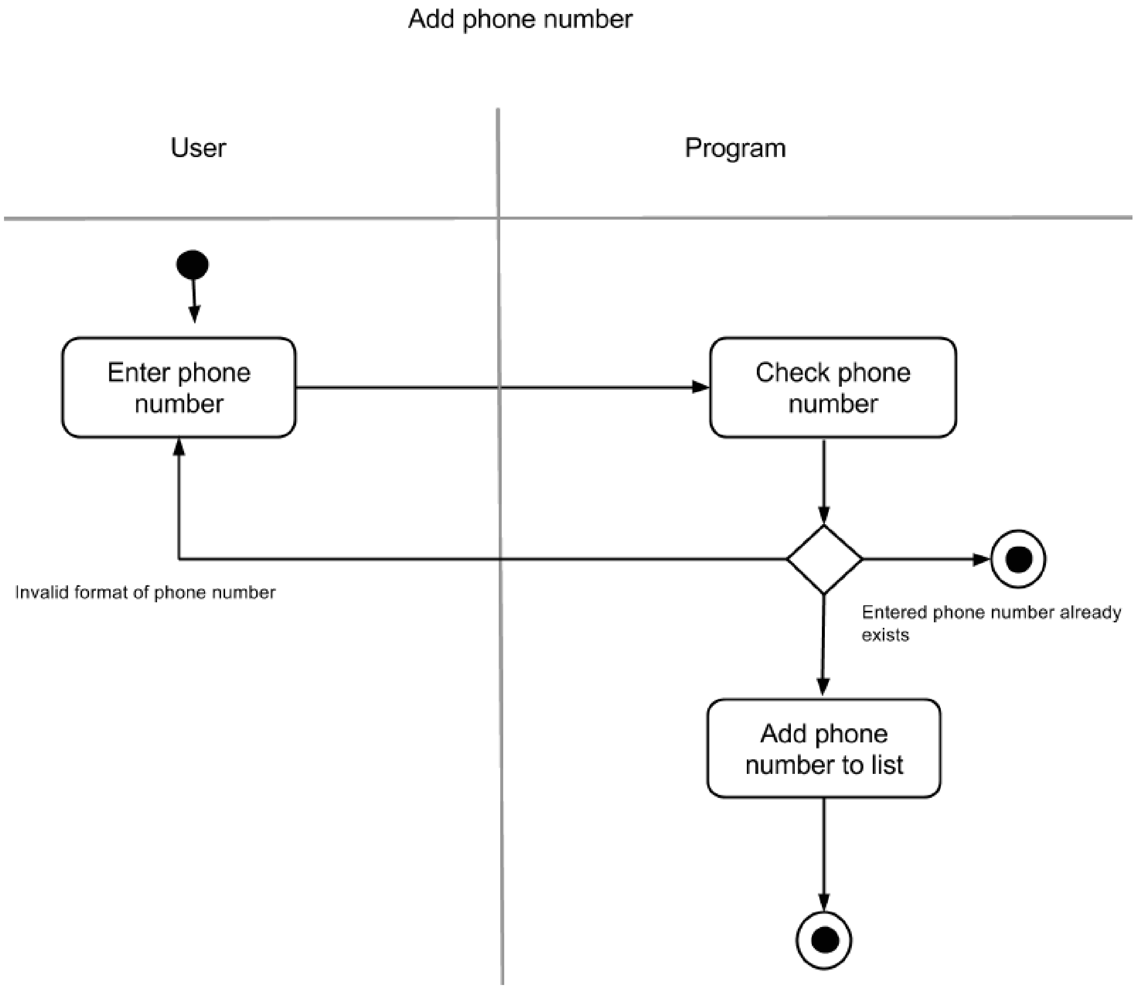

Figure 9: Swimlane diagram for Use case #7

2.2.7 Use case 8: Delete phone number

Name	Delete phone number
Actor	**Primary : User** Secondary : Program
Goal of Usage	Delete phone number from phone number list.
Preconditions	1. Program is running. 2. there are 2 phone numbers at least on phone number list.
Trigger	1. User selects ‘Delete phone numbers’.
Scenario	1. User selects phone number from phone number list. 2. Program delete input phone number from phone number list in computer.
Exceptions	-
Priority	High
Open Issues	-

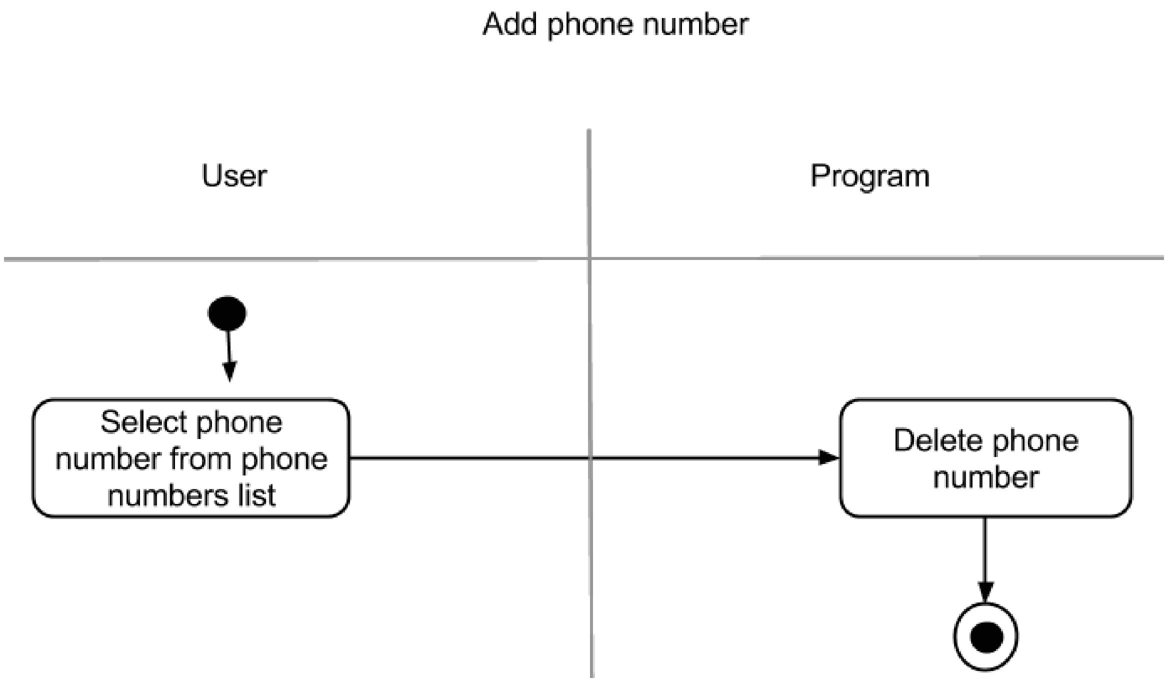

Figure 10: Swimlane diagram for Use case #8

2.2.8 Use case 9: Set Arm state

Name	Set Arm state
Actor	**Primary : User** Secondary : Program
Goal of Usage	Change program state to arm state.
Preconditions	1. Program is running.
Trigger	1. User selected “Change to Arm state”
Scenario	1. User input password. 2. Change program state to arm state
Exceptions	1. (On 1) If password is not correct : Print error message and retry input password. 2. (On 2) If state is already arm state : Print error message.
Priority	Medium
Open Issues	-

2.2.9 Use case 10: Set Disarm state

Name	Set Disarm state
Actor	**Primary : User** Secondary : Program
Goal of Usage	Change program state to disarm state.
Preconditions	1. Program is running.
Trigger	1. User selected “Change to Disarm state”
Scenario	1. User input password. 2. Change program state to disarm state
Exceptions	1. (On 1) If password is not correct : Print error message and retry input password. 2. (On 2) If state is already disarm state : Print error message.

Priority	Medium
Open Issues	-

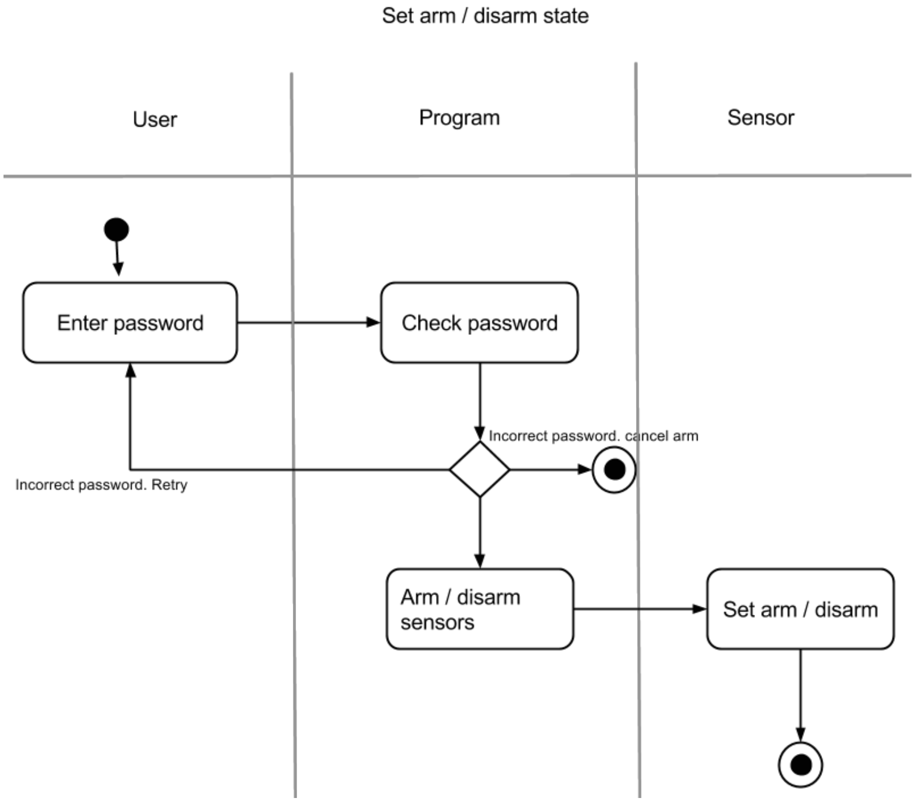

Figure 11: Swimlane diagram for Use case #9,#10

2.2.10 Use case 11: Reset all settings

Name	Reset all settings
Actor	**Primary : User** Secondary : Program
Goal of Usage	Reset all system.
Preconditions	1. Program is running.
Trigger	1. User selected “reset all system”.
Scenario	1. User inputs password. 2. Program deletes all settings (password, floor plan, safe zone, phone numbers). 3. User have to set initial settings through 2.2.2 ~ 2.2.4

Exceptions	1. (On 1) User input password is incorrect : Print error message and requires re-input password.
Priority	High
Open Issues	-

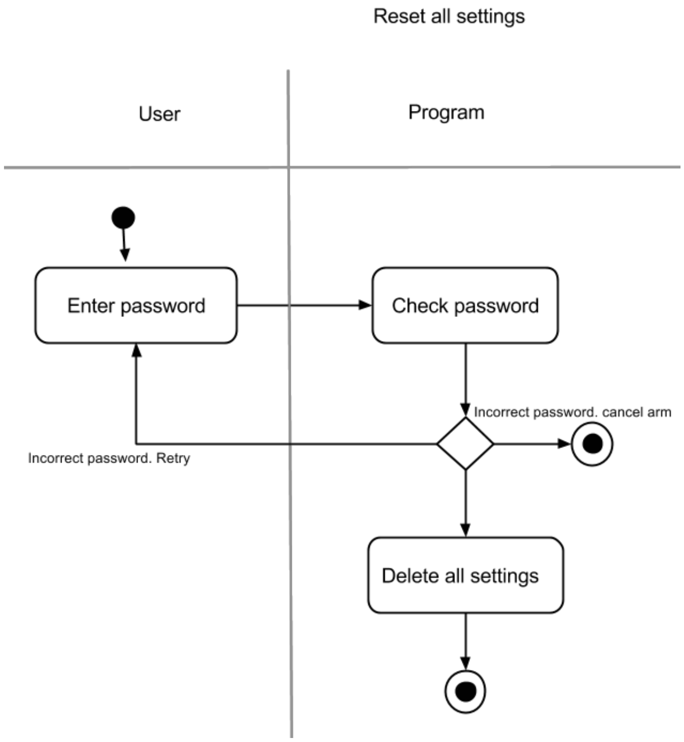

Figure 12: Swimlane diagram for Use case #11

2.3 Use cases - Sensor and reactions (Use case #12~#19)

2.3.1 Use case diagram

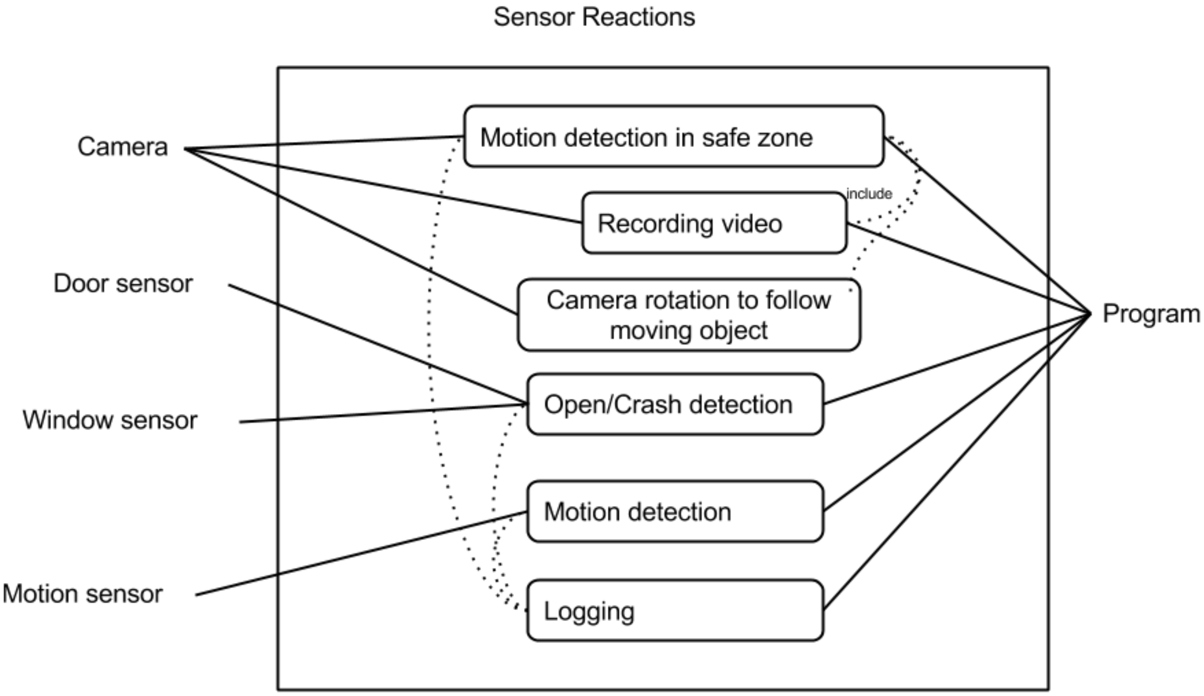

Figure 13: Use case diagram for Use case #12~#19

2.3.2 Use case 12: Motion detection in safe zone

Name	Motion detection in safe zone
Actor	**Primary : Camera** Secondary : Program, Gatecrasher
Goal of Usage	Catch motion detection in safe zone.
Preconditions	1. Camera is running. 2. Door sensor is connected to Program. 3. Program is running. 4. Program state is arm state.
Trigger	Motion of object is detected in safe zone by camera.
Scenario	1. Camera detects motion. 2. Camera alerts changed state to program.

	3. Camera starts recording video. 4. Motion of object is finished. 5. Camera stops recording video. 6. Camera sends recorded video to program.
Exceptions	1. (On 6) Recorded video is too short (e.g. video has just 2 frames so that human cannot recognize) : Program add paddings (some part of video of previous, next few seconds) to recorded video.
Priority	High
Open Issues	None

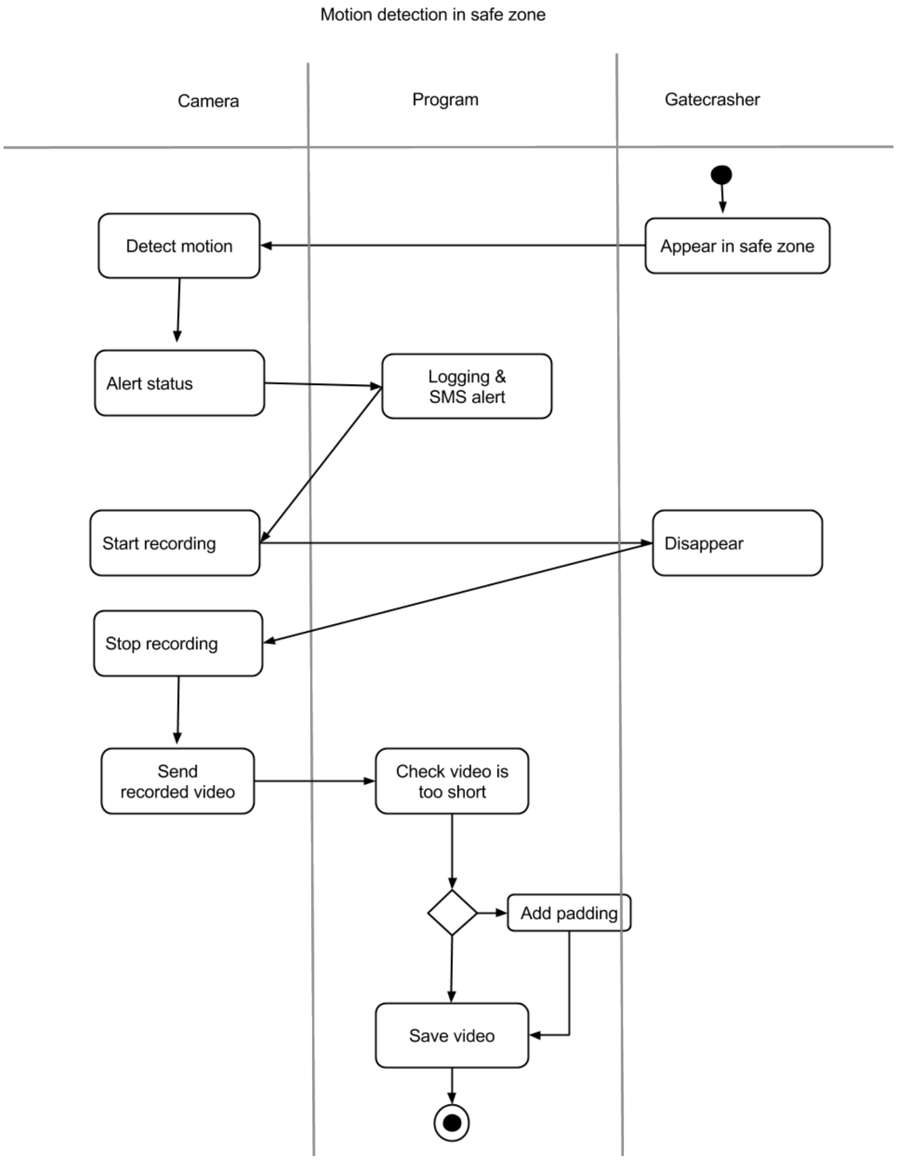

Figure 14: Swimlane diagram for Use case #12

2.3.3 Use case 13: Open / Crash detection of door

Name	Open / Crash detection of door
Actor	**Primary : Door sensor** Secondary : Gatecrasher, Program
Goal of Usage	Catch whether door is open or crashed.
Preconditions	1. Door sensor is running. 2. Door sensor is connected to Program. 3. Program is running. 4. Program state is arm state.
Trigger	1. Door is opened or crashed.
Scenario	1. Door sensor alerts its state('Open' or 'Crush') to program.
Exceptions	-
Priority	High
Open Issues	-

2.3.4 Use case 14: Open / Crash detection of window

Name	Open / Crash detection of door
Actor	**Primary : Window sensor** Secondary : Program
Goal of Usage	Catch whether window is open or crashed.
Preconditions	1. Window sensor is running. 2. Window sensor is connected to Program. 3. Program is running. 4. Program state is arm state.
Trigger	1. Window is opened or crashed.
Scenario	1. Window sensor alerts its state('Open' or 'Crush') to program.

Exceptions	-
Priority	High
Open Issues	-

2.3.5 Use case 15: Motion detection from motion sensor

Name	Motion detection from motion sensor
Actor	**Primary : Motion sensor** Secondary : Program
Goal of Usage	Catch motion from motion sensor.
Preconditions	1. Motion sensor is running. 2. Motion sensor is connected to Program. 3. Program is running. 4. Program state is arm state.
Trigger	1. Motion is detected in range of motion sensor detection.
Scenario	1. Motion sensor alerts its state('Motion detected') and additional information(speed/direction) to Program.
Exceptions	-
Priority	High
Open Issues	-

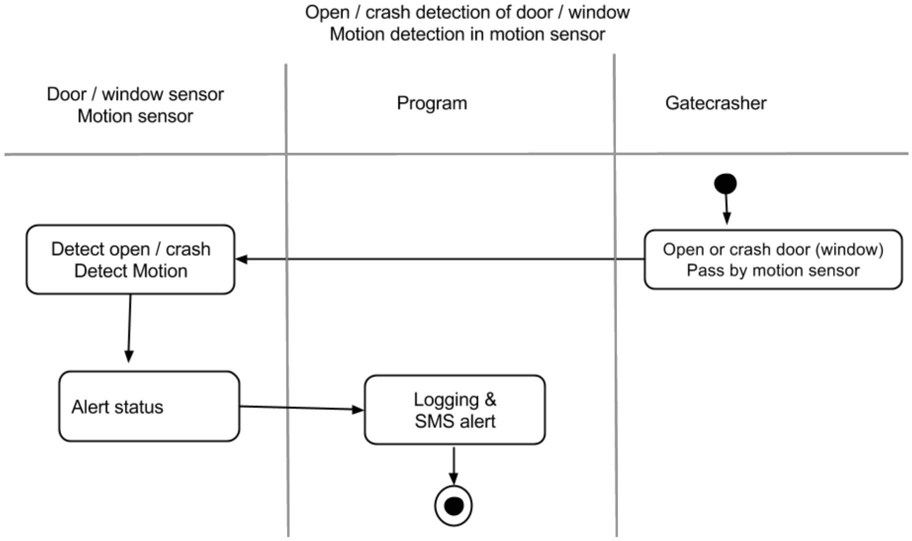

Figure 15: Swimlane diagram for Use case #13,14,15

2.3.6 Use case 16: Camera rotation to follow moving object

Name	Camera rotation to follow moving object
Actor	**Primary : Camera** Secondary : Gatecrasher
Goal of Usage	Rotate camera view to follow detected moving object.
Preconditions	1. Motion sensor is running. 2. Program is running. 3. Program state is arm state.
Trigger	1. Moving object is moving around edge of camera view.
Scenario	1. While target object is moving in sight of camera, camera rotates to proper direction which moving object is positioned as near as center of camera view. 2. Movement of objects is finished. 3. Camera rotates to initial position.
Exceptions	1. (On 1) Camera reached to rotation angle limit : Camera stop tracing that moving object.
Priority	High
Open Issues	-

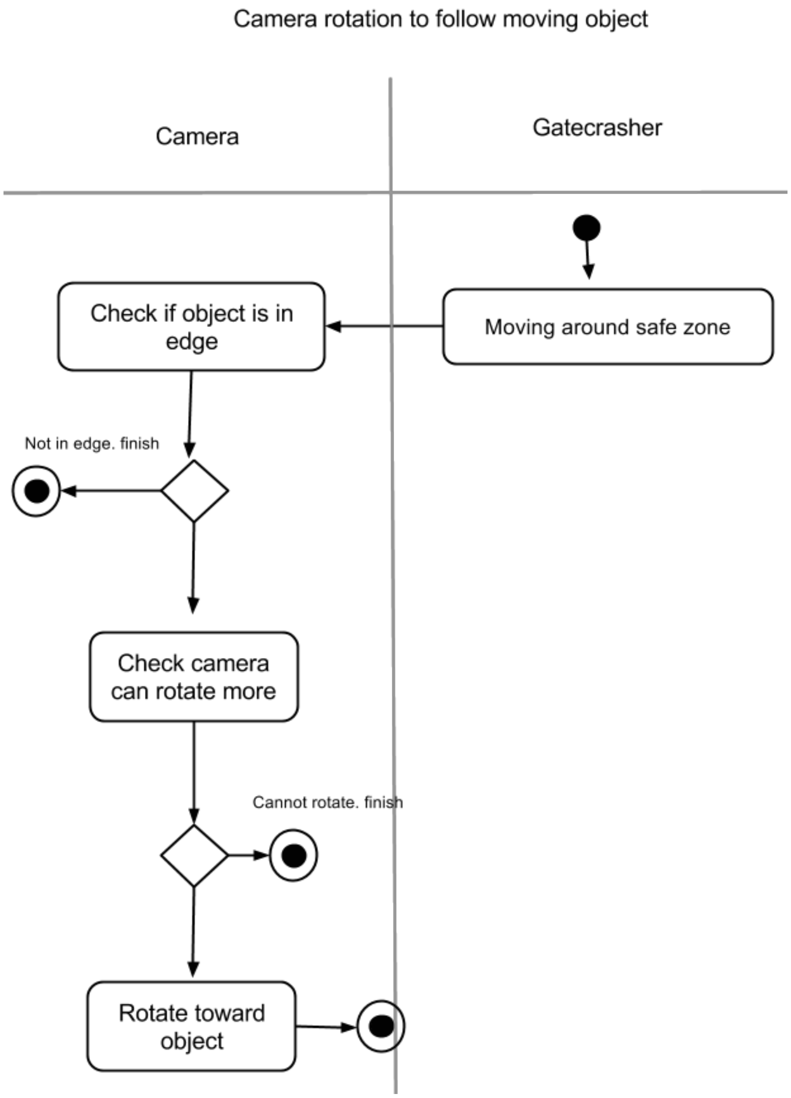

Figure 16: Swimlane diagram for Use case #16

2.3.7 Use case 17: Logging security actions

Name	Logging security actions
Actor	**Primary : Program** Secondary : Storage server
Goal of Usage	Save all logs from security actions from sensors to both computer and server.
Preconditions	1. Program is running. 2. Remote storage service is provided (by Safehome provider company).

Trigger	1. Program received alert of state changes from sensors.
Scenario	1. Write the log with information (time, sensor, prev state, next state). 2. If alert is from camera, started time and finished time of video is added to current log. 3. Save current log to computer. 4. Upload current log to remote storage service.
Exceptions	1. (On 3) There is no more storage for saving log on computer : Remove oldest logs or videos until enough space is left.
Priority	High
Open Issues	-

2.3.8 Use case 18: Saving recorded video

Name	Saving recorded video
Actor	**Primary : Program** Secondary : Storage server
Goal of Usage	Save video sent from camera to both computer and server.
Preconditions	1. Program is running 2. Remote storage service is provided (by Safehome provider company)
Trigger	1. Program received alert of motion detection from camera.
Scenario	1. Program receive recorded video from camera. 2. Add additional informations(started time/finished time) to recorded video. 3. Save current video to computer. 4. Upload current video to storage service.
Exceptions	1. (On 3) There is no more storage for saving video on computer : Remove oldest logs or videos until enough space is left.
Priority	High
Open Issues	-

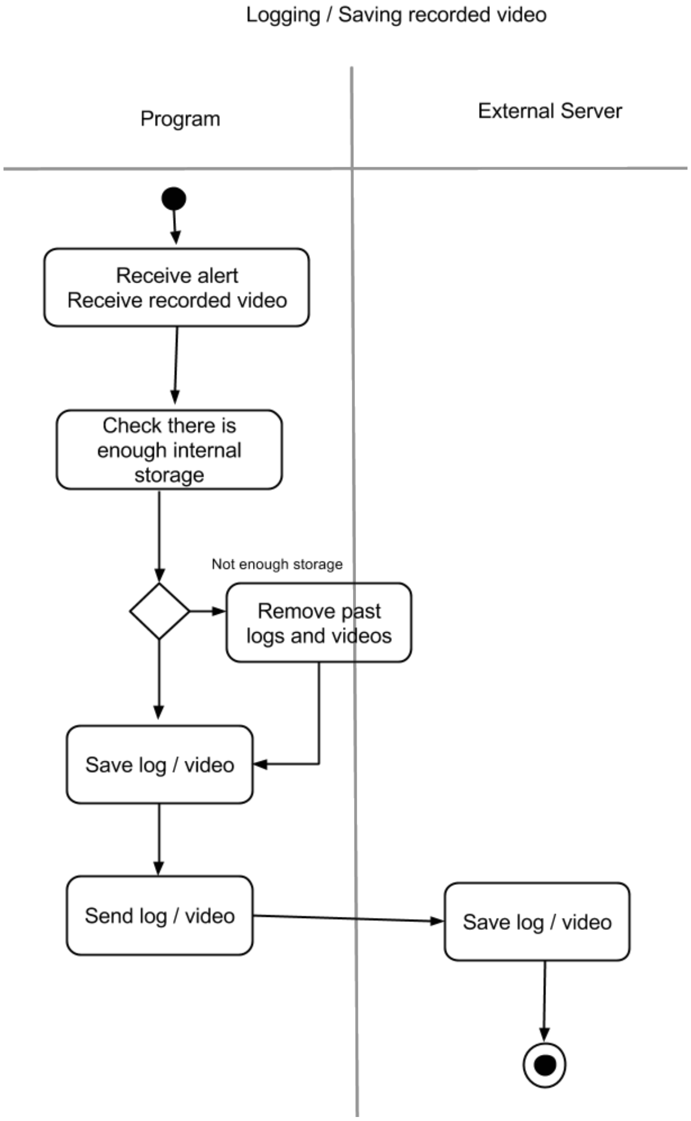

Figure 17: Swimlane diagram for Use case #17,#18

2.3.9 Use case 19: Send urgent SMS

Name	Send urgent SMS
Actor	**Primary : Program** Secondary : User
Goal of Usage	Alert critical state to user through SMS message
Preconditions	1. Program is running. 2. Program is arm state.
Trigger	1. One of sensors sends alert to Program.
Scenario	1. Send SMS message to all phone number on phone number list.
Exceptions	-
Priority	High
Open Issues	-

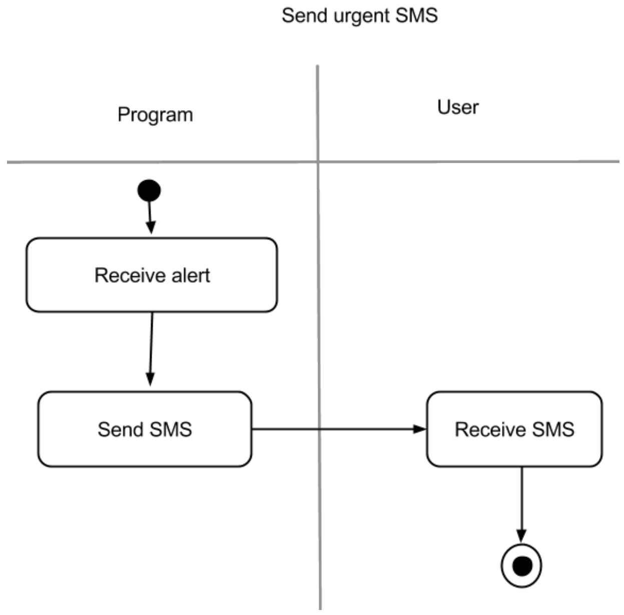

Figure 18: Swimlane diagram for Use case #19

2.4 Use cases - Checking (Use case #20~#22)

2.4.1 Use case diagram

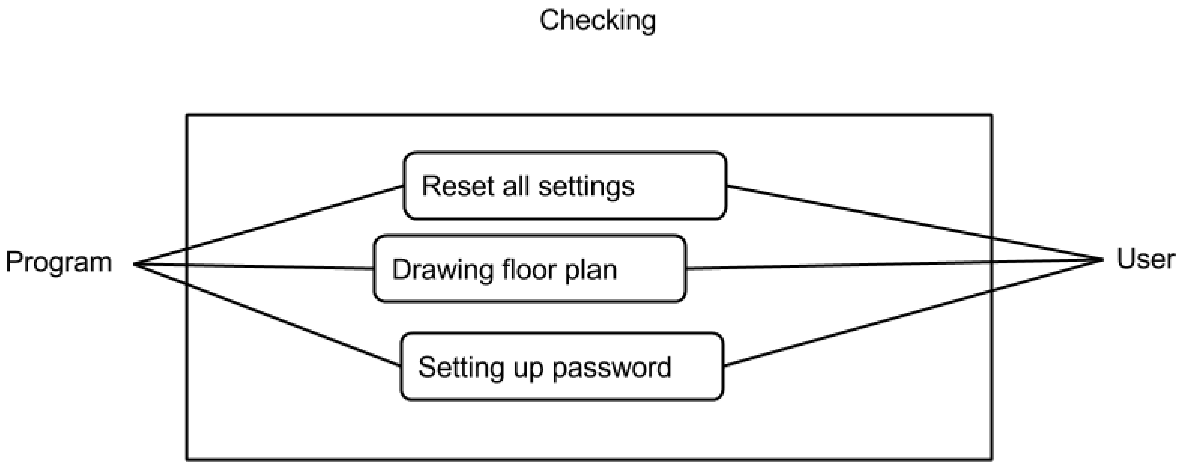

Figure 19: Use case diagram for Use case #20~#22

2.4.2 Use case 20: Floor plan check

Name	Floor plan check
Actor	**Primary : User** Secondary : Program
Goal of Usage	Show state through floor plan
Preconditions	1. Program is running. 2. Floor plan is already set
Trigger	1. User selected “Floor plan check”.
Scenario	1. Show floor plan and sensor states on program. (Caught alert=Red, Uncaught alert=Blue)
Exceptions	-
Priority	Medium
Open Issues	-

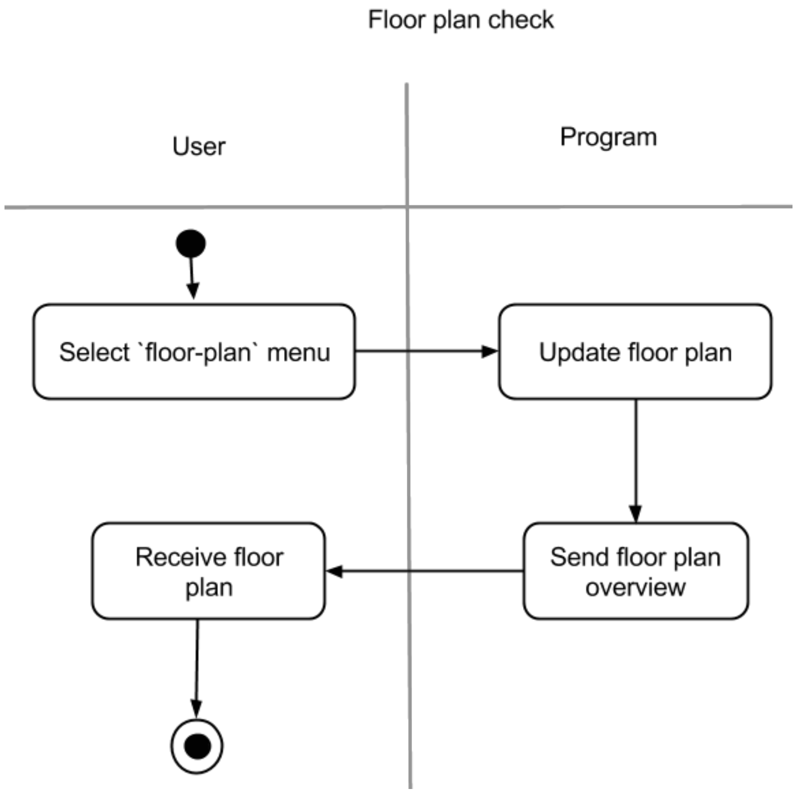

Figure 20: Swimlane diagram for Use case #20

2.4.3 Use case 21: Log check

Name	Log check
Actor	**Primary : User** Secondary : Program
Goal of Usage	Check and read stored logs
Preconditions	1. Program is running.
Trigger	1. User selected “Log check”. 2. User select some log from log list. 3. User input password.
Scenario	Show selected log details.
Exceptions	1. (On 2) When there is no log in log list : Print error message. 2. (On 3) Input password is incorrect : Print error message and re-input password.
Priority	Medium
Open Issues	-

2.4.4 Use case 22: Video check

Name	Video check
Actor	**Primary : User** Secondary : Program
Goal of Usage	Check and read stored videos
Preconditions	1. Program is running.
Trigger	1. User selected “Video check”. 2. User select some video from video list. 3. User input password.
Scenario	Show selected video details.
Exceptions	1. (On 2) When there is no video in video list : Print error message.

	2. (On 3) Input password is incorrect : Print error message and re-input password.
Priority	Medium
Open Issues	-

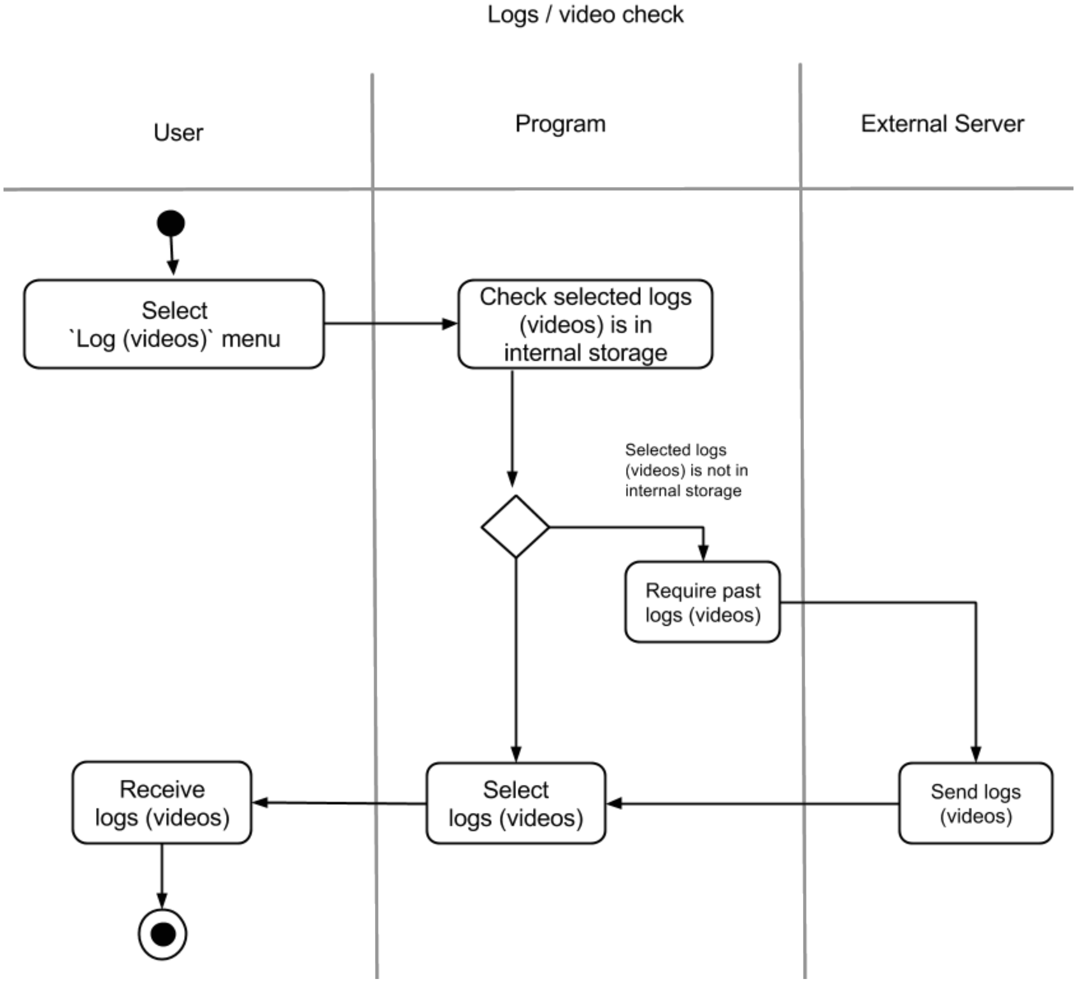

Figure 21: Swimlane diagram for Use case #21,#22

2.5 Use cases - Web access (Use case #23)

2.5.1 Use case diagram

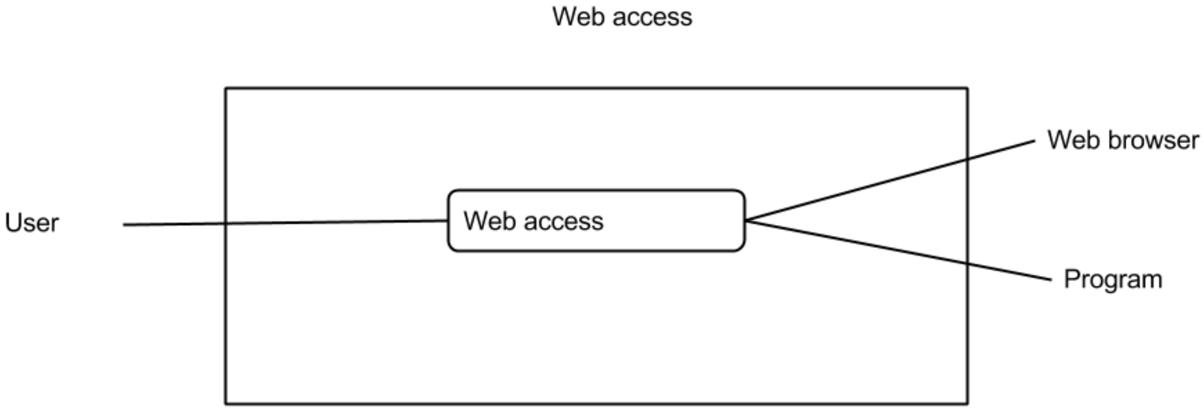

Figure 22: Use case diagram for Use case #23

2.5.2 Use case 23: Web access

Name	Web access
Actor	**Primary : User** Secondary : Browser, Program
Goal of Usage	Access program through web browser in remote place
Preconditions	1. Program is running. 2. User have ID and Password on Safehome Service.
Trigger	1. User access Safehome Manager Homepage(Which is provided by Safehome provider). 2. Trigger from one of use case 4~11, use case 20~22
Scenario	1. User put ID and Password on login page of Safehome Manager Homepage. 2. Safehome Manager Homepage show exactly same view of Program. 3. Scenario from one of use case 4~11, use case 20~22 can be executed by use case selection.

Exceptions	1. (On 1) Input ID is invalid : print error message and re-input ID. 2. (On 1) Input password is invalid : print error message and re-input password.
Priority	Medium
Open Issues	Concurrency Problem

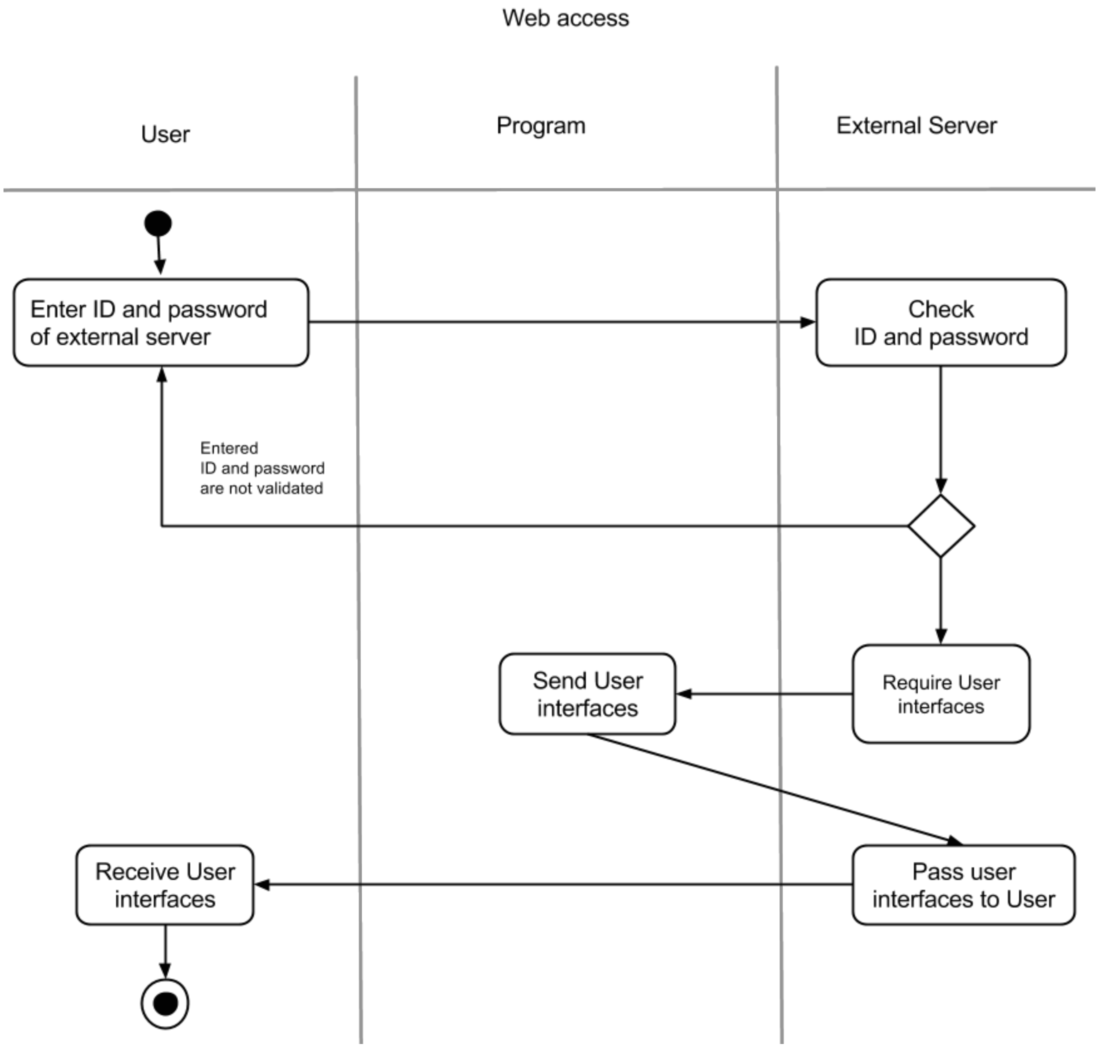

Figure 23: Swimlane diagram for Use case #23

Appendix A. Traceability matrix

Use Cases	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
sensor.common.run	V																						
sensor.common.stop		V																					
sensor.common.self_check	V																						
sensor.common.alert												V	V	V	V								
sensor.camera.state												V											
sensor.camera.record												V				V							
sensor.camera.save_video												V				V		V					
sensor.camera.rotate																V							
sensor.door.state													V										
sensor.door.breaking													V										
sensor.window.state														V									
sensor.window.breaking														V									
sensor.motion.state															V								
sensor.manager.check	V											V	V	V	V	V							
sensor.manager.alert												V	V	V	V	V							
system.state.arm									V														
system.state.disarm										V													
system.monitor.camera												V				V							

system.monitor.sensor													V	V	V								
system.monitor.log																					V		
system.monitor.video																						V	
system.web.login																							V
system.web.arm																							V
system.web.disarm																							V
system.web.camera																							V
system.core.set_password			V	V																			
system.core.reset_password				V	V																		
system.core.reset_setting											V												
system.core.reset_all											V												
system.core.log																		V					
system.core.upload																		V	V				
system.core.download																					V	V	

Appendix B. Who-did-what list

Task	Choonghwi Lee	Wookjae Byun
Table of contents		V
Use case 1~11	V	
Use case 12~23		V
Use case diagrams	V	
Swimlane diagrams	V	
Revision of SRS		V
Appendix A.		V
Appendix B.		V
Appendix C.		V
Appendix D.		V
Appendix E.		V

Appendix C. Figure index

Appendix D. Meeting logs

#1) 2015/5/3 Sun, N1, #109
Meeting during 20:30~22:30
Short revision of SRS(Changed name : secure.* -> sensor.*, panel.* -> system.*)
Traceability table content decision : function name & use cases
Write scenarios to use cases.

#2) 2015/5/4 Mon, E11, #411
Meeting during 14:30~15:30
Specified use-cases for pre-written scenario
Divide roles to each members (Lee: case 1 ~ case 10, Byun: case 11 ~ case 21)
Next time : decide documentation format and implement contents.

#3) 2015/5/6 Wed, N1, #114
Meeting during 17:00~25:00
Main purpose : Analysis model documentation finish
Revised each members' work. (Works divided at last time)
Revised SRS. (Some difference between SRS document and Analysis model document, so reduced difference)
Finished documents.

Appendix E. Revision history

Version	Changed contents	Date
1.0	Analysis Model Document Completion	2015/5/6

Software Requirement Specification

Team 07 (Choonghwi Lee, Wookjae Byun)

Table of Contents

1. Introduction

1.1 Purpose

This Safehome service helps customer to keep their house more safe by cameras, sensors, and control system. Users can surveil their home, even though they are not in their home. If someone not invited enter users home, then alarm bell go off and security agents are come their home. Notifications are also sent to users when alarm bell went off.

1.2 User Scenario

Initailly, user installs safehome system to their house. User should initialize settings during installing safehome system.
Initial setting starts from designing *floor plan* by user. *Floor plan* is drawed by setting all places of doors, windows, and sensors. Then user can set *safe zone*, by adding some group of sensors to each *safe zone*. Then initial setting is finished.

User can manage or activate/deactivate all systems by *control system.* By *floor plan* GUI with *safe zone* expression, user can figure out overall current system state. More specifically, user can monitor many places inside of house by each sensor state and image of camera view.
User can *arm* safehome system to protect house at absent state to manage emergency cases, and *disarm* safehome system when returned.

By managing functions of system, user can check previous logs and videos of specific time from system storage. By checking stored logs and videos, user can figure out when thieves entered to house and what thieves look like.

And user can access safehome system from remote places by mobile applications through web browser, and user can use exactly same functions from control system.

1.3 SRS overview

This document contains functionalities of each hardwares(sensors, cameras, control panel) and system(grouped as *functional requirements*), and background to run safehome system such as running environment, security level, development environments(grouped as *nonfunctional requirements*). Appendix contains glossaries, index, meeting logs, and others.

In this document, we designed functions to cover

2. Specific Requirements

This requirements contain technical system functionalities and essential background of system running(running environment, security features, development environments).

2.1 Functional requirements (System features)

2.1.1 Sensor feature (sensor.*)

2.1.1.1 Overall description

Safehome system provides home security features. Followings are functional features to be provided.

a. Surveillance features
    - Safehome system surveils inside of house via cameras(CCTV), door sensors, motion sensors.
    - Each camera records when there is some not-invited person or people in sight of camera.
    - Door sensors check whether door is opened, and detect abnormal trial of opening door.
    - Window sensors check whether window is opened, and detect abnormal trial of opening window.
    - Motion sensors detect passing of some object or person through sensor.

b. Notification features
    - Safehome system notifies an emergency state to user
    - Safehome system notifies via SMS service, Push notification of Smart-Phone or ARS service.
    - Safehome system can notifies to multiple users at once.

2.1.1.2 Common Features (sensor.common.*)

Sensors have common features and those are described below. Sensors can detect change of states(Cameras and motion sensors have ‘motion undetected’ and ‘motion detected’ states, door sensors and window sensors have ‘open’, ‘closed’, ‘crush’ states). Sensors can alert change of state to system, when Sensors are ‘run’ state.

sensor.common.run	
Assumption	Sensor state is ‘stop’.
Description	Set sensor state to ‘run’ and turn on that sensor.
Exceptions	Concurrency : Concurrent operations by users.

sensor.common.stop	
Assumption	Sensor state is ‘run’.
Description	Set sensor state to ‘stop’ and turn off that sensor.
Exceptions	Concurrency : Concurrent operation by users.

sensor.common.self_check	
Assumption	-
Description	Sensor perform self-checking procedure and return self-check result.
Exceptions	CheckError : self-check procedure cannot be done because of error.

sensor.common.alert	
Assumption	Sensor state is ‘run’. Sensor detect some movements.
Description	Sensor send message to control system.
Exceptions	-

2.1.1.3 Camera (sensor.camera.*)

Cameras surveil safe-zone. Cameras can check whether something is moving in safe-zone. When moving is caught, camera checks that state and record that scene. Recorded videos are saved in storage.

sensor.camera.state	
Assumption	Camera state is ‘run’.
Description	Return camera state to Program. 1. Recording / Not recording 2. Moving / Static
Exceptions	-

sensor.camera.record	
Assumption	Camera state is ‘run’. Camera detect movement.
Description	Camera starts recording until movement disappears.
Exceptions	-

sensor.camera.save_video	
Assumption	Camera is recording video. Camera does not detect any movement.
Description	Camera stops recording and save video file to main system.
Exceptions	VideoTooShort : Recorded video is too short to be recognized by person, such as record has just 2 frame. it should be padded by before and after videos. Connection error : Connection is unlinked by network or disconnection of cable.

sensor.camera.rotate	
Assumption	To rotate camera, one of followings is satisfied. - Camera is recording and target object is moving - Camera finished recording
Description	Rotate camera to target or to initial angle.

Exceptions	CannotRotateMore : Camera cannot rotated more.

2.1.1.4 Door sensors (sensor.door.*)

Door sensors detect whether a door is open or closed, and detect abnormal trial to open or break door.

sensor.door.state	
Assumption	Sensor state is ‘run’.
Description	Return door state to Program. 1. Open / Closed 2. Crashed 3. Whether some object is passing or not
Exceptions	-

sensor.door.breaking	
Assumption	Sensor detect abnormal trial of breaking door.
Description	Send alert message to Program.
Exceptions	SensorCrash : Sensor can be crashed also.

2.1.1.5 Window sensors (sensor.window.*)

Window sensors detect whether a window is open or closed, and detect abnormal trial to open or break window.

sensor.window.state	
Assumption	Sensor state is ‘run’.
Description	Return window state to Program. 1. Open / Closed 2. Crashed 3. Whether some object is passing or not
Exceptions	-

sensor.window.breaking	
Assumption	Sensor detect abnormal trial of breaking window.
Description	Send alert message to Program.
Exceptions	SensorCrash : Sensor can be crashed also.

2.1.1.6 Motion sensors (sensor.motion.*)

sensor.motion.state	
Assumption	Sensor state is ‘run’.
Description	Return state and additional information of motion sensor to Program. 1. Whether movement is detected 2. Movement direction and speed
Exceptions	-

2.1.1.7 Handler (sensor.handler.*)

Manager is supervisor of sensors. Manager manages sensors and decides whether the change is danger when manager detects state change of sensors. If danger state is catched, alert to system.

sensor.handler.check	
Assumption	‘sensor.common.alert’ is called.
Description	If system is arm state, call ‘sensor.handler.alert’.
Exceptions	-

sensor.handler.alert	
Assumption	Called from ‘sensor.handler.check’.
Description	Send alert to main system.
Exceptions	-

2.1.2 Systems and Control panels (system.*)

2.1.2.1 Overall Description

This service provides control panel to manage several options and function usage. Followings are methods related to using control panel

- Using control panel
- Remote access via internet

2.1.2.2 State change (Arm / Disarm) (system.state.*)

There are two state on system. One state is *arm*, which is defensing mode(alert to user is on). Another state is *disarm*, which is normal mode(not alert to user).

system.state.arm	
Assumption	Current state is ‘disarm’.
Description	Change state to ‘arm’ state.
Exceptions	Concurrency : Concurrent operation by users.

system.state.disarm	
Assumption	Current state is ‘arm’.
Description	Change state to ‘disarm’ state.
Exceptions	Concurrency : Concurrent operation by users.

2.1.2.3 Monitoring sensors / logs (system.monitor.*)

By control panel, user can monitor sensors and cameras. Also, user can read previous logs about sensors or watch recorded video from cameras.

system.monitor.camera	
Assumption	Camera state is ‘run’.
Description	Show current camera view on system.
Exceptions	Invalidstate : Camera state is ‘stop’.

system.monitor.sensor	
Assumption	Sensor state is ‘run’.
Description	Show current sensor states on system on Program. (sensor.*.state)
Exceptions	Invalidstate : Sensor state is ‘stop’.

system.monitor.log	
Assumption	-
Description	Show previous logs of seleced sensor on Program.
Exceptions	-

system.monitor.video	
Assumption	-
Description	Show selected recorded video on Program.
Exceptions	-

2.1.2.3 Remote access through web (system.web.*)

User can access to main system from not only fixed panels but also mobile devices through web browsers.

system.web.login	
Assumption	Computer(installed Program) is connected to internet.
Description	First, Input username and password to access web server. Second, Input system password and connect to system in home from web browser.
Exceptions	LoginException : Wrong Password. NetworkException : Network connection has problem. SystemNotAvailable: System from home is off.

system.web.arm

Assumption	User is logged in.
Description	Call ‘system.state.arm’ to arm system.
Exceptions	NetworkException : Network connection has problem.

system.web.disarm	
Assumption	User is logged in.
Description	Call system.state.disarm’ to disarm system.
Exceptions	NetworkException : Network connection has problem.

system.web.camera	
Assumption	User is logged in.
Description	Call ‘system.monitor.camera’ and adjust view to browser environment.
Exceptions	-

2.1.2.4 System core (system.core.*)

Some critical processes such as reset should be controlled by authorized users, so those processes are controlled in system core.

system.core.set_password	
Assumption	User is authorized. User input new password.
Description	Set password to user input.
Exceptions	UnauthorizedException : User is not authorized.

system.core.reset_password	
Assumption	User is authorized.
Description	Reset password to initial one. ex) ‘0000’ or non-set state

Exceptions	UnauthorizedException : User is not authorized.

system.core.reset_setting	
Assumption	User is authorized.
Description	Reset all settings to initial settings.
Exceptions	UnauthorizedException : User is not authorized.

system.core.reset_all	
Assumption	User is authorized.
Description	Reset all settings to initial settings. Remove all data.
Exceptions	UnauthorizedException : User is not authorized.

system.core.log	
Assumption	‘sensor.common.alert’ is called.
Description	Make and save proper logs.
Exceptions	-

system.core.upload	
Assumption	Connected to network. Some video or log passed than storage due.
Description	Upload old video data or logs to provider’s storage. Remove uploaded data.
Exceptions	ConnectionException

system.core.download	
Assumption	Connected to network. User want to read older video or log than locally-stored video or log.

Description	Download some video or log from provider's storage
Exceptions	ConnectionException

2.2 Non-functional requirements

2.2.1 Running environment

1. Platform
   System run with all platform but java is avaliable for that system.
2. Java version
   System run with Jave se 8 and java over 1.7.0.

2.2.2 Security requirement

1. Password managing
   a. Password is stored in file only accessible by system
   b. Password is stored with md5 hashing

2.2.3 Development requirement

1. Programming language - java
2. Programming tools - eclipse (or intellij)
3. Communication - slack
4. CVS - git & bitbucket

2.2.4 Storage requirement

1. Storage is 500GB at least
2. Past video and logs stored in storage
3. Remote storage by provider can be available

2.2.5 Serving web server for remote access

1. Provider serves a web server for customers' homesafe system

Appendix A. Glossaries and Indices

Sensor (P. 4)

Sensor is hardware conponent of Safehome system. Some actions activates sensors and sensor alerts to system. Sensor includes camera.

Camera (P. 5)

Camera is special kind of sensor. Like a cctv, camera is for surveillance of some spaces. It can record video of specific section of time.

Program (P. 5)

In this document and next documents, ‘Program’ means Safehome control program on computer. This works core of Safehome and all functions works during this program is running.

Not-invited person (P. 3)

Not-invited person is person disrelated from house (safehome system set) or owner of that house. If not-invited person activates sensor of Safehome and it is armed system, Emergency Notification is sended to service provider and user and user knows invasion of someone.

System (P. 8)

Central system of Safehome. System manages sensors, logs, stored vidoes. System also connect to provider’s server to grant to remote access

Web server (P. 9)

Web server from service provider. It provides web interface to access system away from home.

Safe-zone (P. 2)

Coverage of Camera.

Floor plan (P. 2)

Blueprint of house. Set by group of sensors.

Appendix B. Traceability matrix

Parts	Initial Setting	Control	Watching Log/Video	Remote Access	Critical Controls
sensor.common.run		V			
sensor.common.stop		V			

sensor.common.self_check		V			
sensor.common.alert		V			
sensor.camera.state		V			
sensor.camera.record		V			
sensor.camera.save_video		V			
sensor.camera.rotate		V			
sensor.door.state		V			
sensor.door.breaking		V			
sensor.window.state		V			
sensor.window.breaking		V			
sensor.motion.state		V			
sensor.manager.check		V			
sensor.manager.alert		V			
system.state.arm	V	V			
system.state.disarm	V	V			
system.monitor.camera			V		
system.monitor.sensor			V		
system.monitor.log			V		
system.monitor.video			V		
system.web.login				V	
system.web.arm				V	
system.web.disarm				V	
system.web.camera				V	
system.core.set_password					V
system.core.reset_password					V
system.core.reset_setting					V

system.core.reset_all					V
system.core.log					V
system.core.upload					V
system.core.download					V

Appendix C. Who-did-what list

Task	Choonghwi Lee	Wookjae Byun
Table of contents	V	V
Functional requirements - Home security features	V	
Functional requirements - Control panels		V
Non-functional requirements	V	V
Appendix A.	V	V
Appendix B.		V
Appendix C.	V	
Appendix D.	V	V
Appendix E.		V

Appendix D. Meeting logs

#1) 2015/4/17 Fri, N1, #114
Meeting during 19:00~23:30
SRS initial structurize(v0.0)
Contruct table of content and assign each works
(function requirement describing of each part)

#2) 2015/4/20 Fri, N1, #102
Meeting during 19:00~20:00
Assign personal jobs. Divided each roles of TOC. Getting attention to functional requirements
We'd like to add an simple images, which can helps customers' understaning.
Roughly decided documentation format.

#3) 2015/4/26 Sun, N1, #114
Meeting during 17:30~23:30
Version 1.0 of SRS. Implemented all places from individual drafts by each other.
Little change about structures.

Appendix E. Revision history

Version	Changed contents	Date
0.0	Initial contents draft.	2015/4/17
0.1	Dividing functional/nonfunctional parts. More specific dividing & grouping.	2015/4/20
1.0	Implement contents.	2015/4/26
1.1	Revised nonfunctional requirements and Appendix A(Glossaries)	2015/5/6