嵌入式系統 Final Project Report

智慧家庭藍圖

B09901060 王泓予 、B09901149 林暄皓

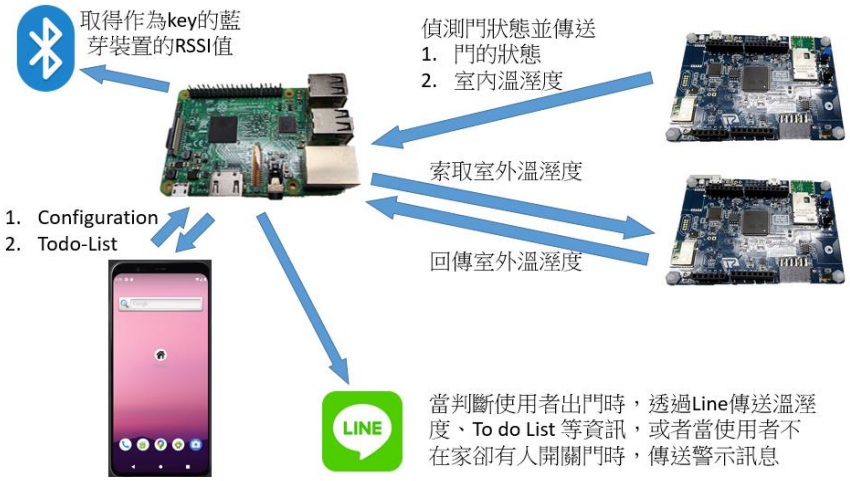
一、 題目簡介

透過安裝在門上的感測器，判斷使用者狀態，在其出門時透過 Line 發送訊息提醒今日天氣資訊及待辦事項。並在使用者不在家時，偵測是否有外人闖入家中，如果發現可疑情況則發送警告訊息給使用者。

二、 動機

隨著科技進步，越來越多人追求生活的便利，於是我們希望有個類似生活助理的功能，幫助我們處理生活比較瑣碎的功能，並且可以在不同的情況下滿足我們的需求。

三、 系統架構



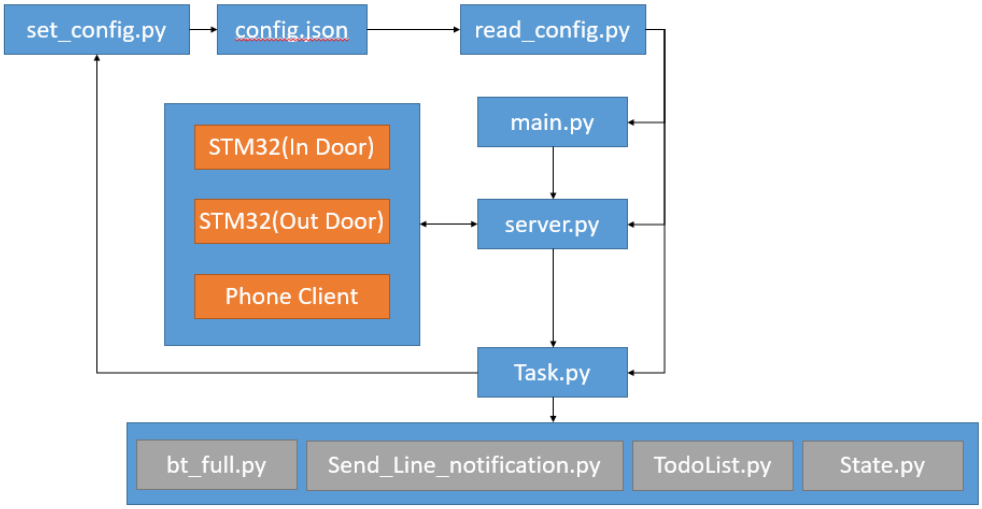
圖一、系統基本架構

socket server	Raspberry pi
socket client	門上的感測器(STM32)、室外的溫濕度計(STM32)、手機 app

當裝於門上的 STM32 感測到開關門後會透過 socket 傳送現在的溫溼度及開關門狀態給 Rpi，而 Rpi 會執行藍芽掃描尋找作為 key 的藍芽裝置的 RSSI 值，並綜合考慮開關門狀態、使用者原本狀態、現在 RSSI 值的狀態決定要執行甚麼任務。如果需要傳送天氣資訊就透過 socket 向室外的 STM32 發送訊息要求溫溼度資訊，讓其回傳溫溼度資訊，最後透過 Line 傳送訊息給使用者。

手機 app 連上 server 之後則可以對 Rpi 進行 configuration 設置或者使用 To Do List 的功能，Rpi 收到指令後就會進行相對應的修改並回傳修改後的狀態給 app 讓其顯示在 UI 介面上。

四、 程式架構



圖二、程式架構

有關於 Rpi 的設置參數儲存於 config.json，使用者可以藉由 set_config.py 對其進行設置，而 read_config.py 可以讓程式取得 config.json 的內容。

一開始，Rpi 開啟 main.py 來開始 project，先利用 read_config.py 進行 initialize，並且確認 configuration 內容是否完整，接著呼叫 server.py 在設定的 ip 以及 port 開啟 server，server.py 會 listen 是否有設備連接，每當有新的 client 連接時，依照他第一次傳的訊息判斷身分，並開啟新的 Thread 繼續收聽訊息，根據每次收到的訊息決定要進行 Task.py 中哪項任務，如有需要，再去呼叫其他檔案的函式。

五、 使用說明

STM32

1. import https://github.com/Howard-149/ES_Final-Project.git 至 Mbed Studio
2. 在 mbed_app.json 中輸入網路帳號密碼
3. 用附上的 target.json 取代原本 mbed-os/targets/targets.json
4. 將 STM32_1.txt 或 STM32_2.txt 的內容分別複製進 source/main.cpp
   (STM32_1.txt 為門上感測器主程式，STM32_2.txt 為室外溫溼度感測器主程式)
5. 在 main.cpp 中輸入 ip address 及 port number 之後燒入對應的 STM32 中即可

Rpi

1. 將 Rpi 資料夾下的程式傳到 Rpi
2. 設置 configuration（詳見七、Configuration 設置）
3. 使用 main.py 開啟服務

手機 app

1. 使用 Android studio 開啟 app/SocketClient 資料夾
2. 啟動手機開發者模式並且開啟 USB 偵錯模式
3. 使用傳輸線連接手機電腦，點擊上方運行以進行燒錄

六、 工作流程

1. 設置 configuration(ip, port, user_phone_key, user_Line_key, rssi_bluetooth_threshold)， 其中 ip 以及 port 只能使用 set_config.py 設置，而其他參數可以在開啟 server 之後，透過手機 app 進行設置。
   設置 user_phone_key 時 Rpi 會利用藍芽掃描附近裝置並回傳，透過選取自己的裝置註冊自己的手機裝置作為 key 。設置 rssi_bluetooth_threshold 時會請使用者根據提示訊息分別將註冊為 key 的手機輪流放在房間內與門口，Rpi 會將房間內與門附近的 rssi 值做比例分配決定 rssi_bluetooth_threshold。
2. 透過 STM32 的加速度計，根據不同的加速度大小及方向偵測開關門並測量溫溼度，透過 wifi socket 通知 rpi，在接獲開關門訊息之後，Rpi 會透過 socket 向戶外的 STM32 要求室外溫溼度資訊。
3. 使用者可以透過手機 app 設定 Todo List，app 透過 socket 將資訊傳給 Rpi，Rpi 將資訊存於 todoList.json 內，於使用者出門時透過 line notify 傳訊息提醒使用者
4. Rpi 接到開關門資訊後透過藍芽掃描決定現在主人是否在屋內，進而判斷是否需要使用 Line notify 傳送溫溼度及 Todo List、警告訊息，或是不須傳送任何訊息給使用者。

七、 Configuration 設置

方法一：直接在 Rpi 上面進行 Configuration 的設置

1. 利用 set_config.py 進行 configuration

```
example command:
              python set_config.py
                    --ip [IP ADDRESS]
                    --port [PORT NUMBER]
                    --set_user_phone_key
                    --user_line_key [LINE NOTIFY TOKEN]
                    --set_bluetooth_threshold
```

2. 完成上述指令後執行 main.py 即可

方法二：使用手機 app 進行 Configuration 的設置

1. 利用 set_config.py 設置 ip address 跟 port

```
example command:
                  python set_config.py
                      --ip [IP ADDRESS]
                      --port [PORT NUMBER]
```

2. 執行 main.py
3. 手機 app 連上 server 後進行其他 configuration 的設置

八、 程式介紹

STM32

1. **裝於門上的感測器(STM32_1.txt):**

作為門上感測器的 STM32 的主程式，透過加速度計大小及方向偵測開關門。具體作法為一開始設置好後先進行歸零，讓三軸加速度的值在不受外力的情況下皆為零。接著若持續超過 1 秒 x 方向加速度 y 方向加速度平方和大於五時認定門有在移動，這時透過剛開始移動時 y 方向加速度的正負號同時並透過 socket 傳送溫、濕度及門開關資訊給 Rpi。

2. **置於室外的溫溼度測計(STM32_2.txt):**

放在室外擔任溫濕度計的 STM32 的主程式，透過 socket 與 Rpi 連接後等待 Rpi 要求傳送溫溼度資訊，收到要求後就回傳溫溼度資訊並繼續等待 Rpi 下一次要求。

Rpi

1. **server.py:**

建立 socket server，我們使用 multi thread 與 client 建立連接及接收每個 client 傳送過來的 data。具體作法為先用一個 thread 專門處理 accept new client 的工作，確保不會有 client 嘗試連接 server 時因為在處理其他程序而失敗。待 client 與 server 建立連接後，因為不同 client 可能會同時傳訊息與 server 溝通，所以對於每個 client 我們都開一個新的 thread 專門處理與其溝通的工作。接收到訊息後再根據收到的訊息執行不同的 Task，為避免一個 Task 執行太久而漏聽某些訊息，所以執行 Task 時也使用與處理接收訊息不同的 thread 來執行。

2. **bt_full.py:**

裡面包含藍芽掃描周遭裝置，及根據指定 mac address 的裝置的 rssi 值判斷現在這個裝置是 inside the room、at the door 或 can't detect key 的狀態。rssi 值取得方法為與指定的 mac address 的裝置連線，透過 API 要求回傳 rssi 值。因為 rssi 值常會浮動，所以我們的作法是每 0.2 秒取一次 rssi 值，取 10 次做平均，如果高於設定好的 rssi_bluetooth_threshold 就會回傳 inside the room，反之就回傳 at the door，若找不到裝置則回傳 can't detect key。

3. **main.py:**

開始執行整套系統，根據 config.json 內的 ip、port 等資訊開啟 server。

4. **state.py:**

紀錄門及主人 state 的相關函式

5. **read_config.py:**

讀取 config.json 內的資訊，直接執行會使用 jq 插件將 json 檔案好看地印出。

6. **set_config.py:**

透過 command line 設置 IP address、 port 、user phone key、user line key 及 rssi bluetoooth threshold 等資訊。

參數	設置方法	說明
ip	--ip [IP Address]	設置 socket server ip 位置
port	--port [Port Number]	設置 socket server port
user_phone_key	--user_phone_key [MAC Address]	手動設置 key 的 MAC address
	--set_user_phone_key	設置 key 的 MAC address
user_line_key	--user_line_key [Line Key]	設置 Line Notify 的 Key
rssi_threshold	--set_bluetooth_threshold	設置 key 的 rssi 門檻

7. send_Line_notification.py:

透過 Line 傳送訊息相關函式，使用 Line notify 的 API 進行訊息推送。

（一開始我們使用 ifttt 服務，但由於限制較多，於是改為直接使用 Line notify）

8. Task.py:

在接收資訊後判斷要執行哪些動作，若是需要使用到戶外的溫溼度資訊時，使用 mutex 確保傳送訊息的 thread 會等待室外的溫溼度資訊更新後才傳送最新的溫濕度資訊。

9. TodoList.py:

處理與 To Do List 相關的功能，可以對 todoList.json 中的資料進行新增、刪除、修改等操作。

10. config.json:

儲存 Rpi 運行時所需要的參數，包含 IP、port、藍芽裝置 MAC Address、Line key、RSSI threshold 等資訊。

[socket info]

ip	開啟 server 的 ip 位置
port	開啟 server 的 port

[user]

user_name	認證為 key 的藍芽裝置名稱
user_phone_key	認證為 key 的藍芽裝置 MAC Address
user_line_key	使用 Line notify 網站生成的 key

[bluetooth]

rssi_threshold	Rpi 量測 key rssi 值時的判斷依據

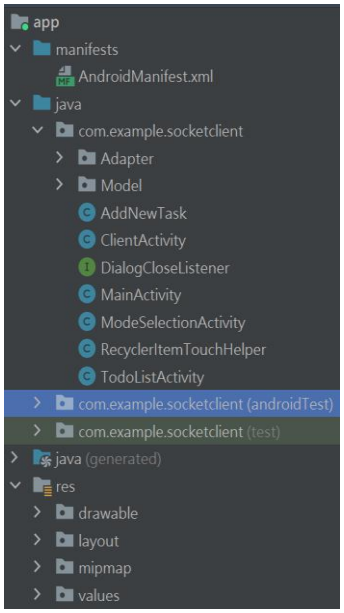
11. todoList.json:

儲存 To Do List 資訊，每項資料有 id、status、task 的屬性。

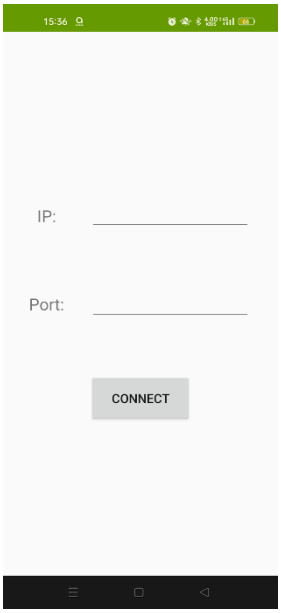
id	辨識不同待辦事項的數字
status	是否已經完成
task	待辦事項的內容

App

圖三、app 圖示

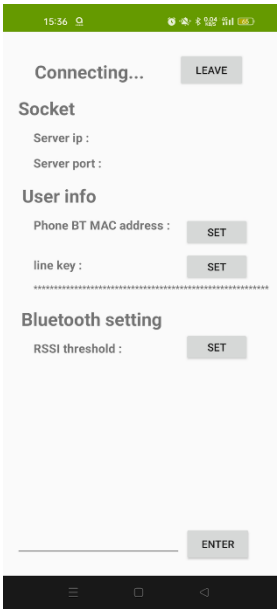


圖四、app 程式

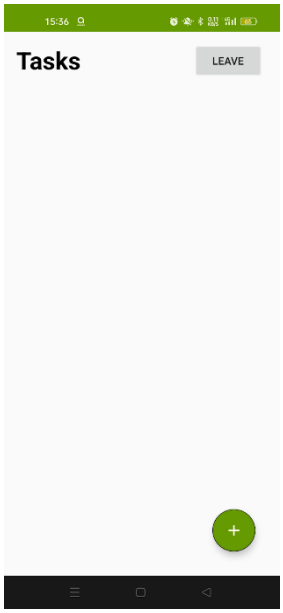


圖五、連線畫面

圖六、選擇功能介面



圖七、Configuration 介面



圖八、To Do List 介面

1. MainActivity.java

圖五顯示程式一開始進入的介面，使用者可以輸入 ip 以及 port 以連接特定 server，MainActivity.java 會將 ip、port 儲存於 bundle，讓下一個 activity 存取

2. ModeSelectionActivity.java

圖六顯示選擇功能的介面，在使用者點選功能之後，會將畫面導向對應的功能介面並且傳遞要連線的 server ip 以及 port 資訊。

3. ClientActivity.java

圖七顯示 Configuration 的介面(未與 server 連線前)，與 server 連線後畫面上會顯示 Rpi 目前的 Configuration，使用者可以透過 app 來對其進行設置或修改。

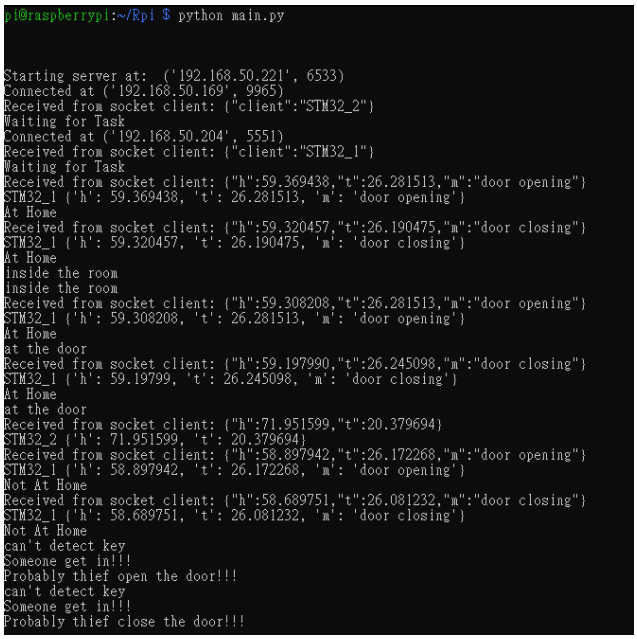
UI Thread	處理畫面 render
Socket Thread	與 server 進行溝通

4. TodoListActivity.java

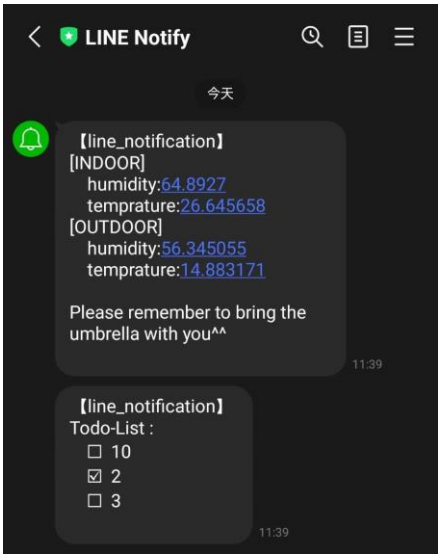
圖八顯示 To Do List 的介面(未與 server 連線前)，與 server 連線後畫面上會顯示使用者目前設置的待辦事項，使用者可以透過 app 來對其進行修改。

UI Thread	處理畫面 render
Socket Thread	與 server 進行溝通

九、 成果



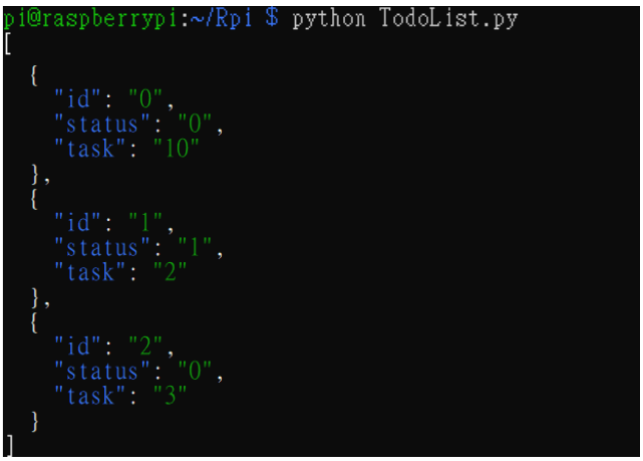
圖九、不同情況時 Rpi 顯示的訊息



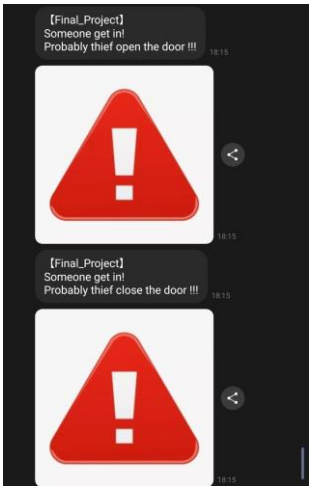
圖十、使用者 Line 收到的訊息

圖九為模擬三種不同情形開關門時，Rpi 上輸出的訊息。可以看到在 server 先與兩台 STM32 建立連線後，第一次開關門是原本主人在家，開關門時手機仍在房間內，所以會先輸出 At Home 表示主人原本的狀態，再輸出 RSSI 的結果為 inside the room。由於我們使用 multi thread 且掃描 RSSI 需要一些時間所以開門時的 inside the room 在關門的 At Home 出現後才輸出。第二次開關門則是主人帶著手機要出門，所以結果為 At Home 及 at the door，最後一次則是小偷趁主人不在時偷偷闖入，所以結果為 Not At Home 與 can't detect key。

圖十為使用者出門時透過 Line 收到的通知訊息，包含現在的室內外溫溼度及使用者自己設定的 Todo List。



圖十一、todoList.json 設置完的內容



圖十二、使用者 Line 收到的警示訊息

圖十一為使用者設定的 To do List 在 Rpi 的 todoList.json 裡面被存取的樣子。

圖十二為小偷闖入時手機上會收到的通知訊息

十、 心得

林暄皓：

原本這次的 Final Project 在一開始提案的時候許多細節及可以新增的功能都沒有非常完整的想法，但在進行的過程中許多不錯的點子慢慢地出現。而我們也試著把可行的想法慢慢完成，最後做出了這次的 Final Project。我非常感謝我的組員幫忙寫出了一個這麼酷的 app，雖然中間有遇到一些困難，但我們還是順利的完成這個 app，讓我們的成果更加豐富。

王泓予：

在完成這次 Final Project 中我學到很多，也對於一些上課教的內容更為熟悉，在進行的過程中，我們也遇到了一些困難，像是將 project 改為 Multi-Thread 時的 debug，或者是撰寫 app 時 java 的 socket 使用問題，而在尋找解答的時候，我們也對這些概念更加了解，最後終於成功做出這麼一份 project。

十一、 參考資料

藍芽:

https://github.com/ewenchou/bluetooth-proximity.git

app:

https://www.youtube.com/@PenguinCoders
https://github.com/MU-PING/Android-studio-socket

十二、 Demo 影片連結:

https://youtube.com/playlist?list=PLEo7jpswi7buv0_PzJ9YszB951DKhUjs7

十三、 Github 連結

https://github.com/Howard-149/ES_Final-Project.git