科技部資訊安全技術研發專案計畫

『系統測試計畫書』

System Testing Plan Document

兼具效能與安全之即時多媒體與巨量資料雲端服務系統

MOST 106-2221-E-027-007

**國立臺北科技大學軟體發展研究中心**

2018/4/5

目錄

[版次變更記錄 1](#_Toc511058035)

[1. 簡介 2](#_Toc511058036)

[1.1 計劃書目的 2](#_Toc511058037)

[1.2 測試目的 2](#_Toc511058038)

[1.3 接受準則 2](#_Toc511058039)

[2. 測試環境 2](#_Toc511058040)

[2.1 硬體規格 2](#_Toc511058041)

[2.2 軟體規格 3](#_Toc511058042)

[2.3 測試資料來源 3](#_Toc511058043)

[3. 測試時程、程序與責任 3](#_Toc511058044)

[3.1 測試時程 3](#_Toc511058045)

[3.2 測試程序 3](#_Toc511058046)

[3.3 測試人員責任 7](#_Toc511058047)

[4. 測試案例 7](#_Toc511058048)

[4.1 整合測試案例 7](#_Toc511058049)

[4.2 接受測試案例 13](#_Toc511058050)

[5. 測試結果與分析 23](#_Toc511058051)

[5.1 整合測試案例 23](#_Toc511058052)

[5.2 接受測試案例 23](#_Toc511058053)

[Appendix A： 追朔表 Traceability 24](#_Toc511058054)

[A.1. 子系統 vs. 測試案例 24](#_Toc511058055)

[A.2. 需求 vs. 測試案例 24](#_Toc511058056)

**版次變更記錄**

|  |  |  |
| --- | --- | --- |
| **Revision** | **Description** | **Date** |
| 1 | 簡介、測試環境、測試時程、測試程序、測試人員責任 | 2018/4/11 |

1. 簡介

1.1 計劃書目的

(1) 執行子計畫系統測試的細部活動之準備與規劃

(2) 與各子計畫負責人溝通，規劃測試的工作時程

(3) 訂定測試計畫所需的資源

(4) 訂定系統執行測試所需的測試工具與環境

1.2 測試目的

(1) 於交付成品前提前發現錯誤

(2) 進行整合測試檢查各子計畫整合後是否能正常運作

(3) 進行接受測試檢查系統是否符合功能需求

1.3 接受準則

本測試計劃需要滿足下面的測試接受準則：

(1) 所有測試程序需依照本測試計劃所訂定的規劃進行，且所有測試結果需能符合預期測試結果。

(2) 以測試案例為單位，當測試未通過時，需要進行單元測試，其接受的準則如第一項中所制定的相同。

(3) 各子計畫皆通過單元測試之後，需進行整合測試，驗證各子計畫系統功能於整合後是否能夠正常執行，其接受準則如第一項中所制定的相同

(4) 系統完成並通過整合測試之後，需進行接收測試，驗證系統功能是否皆符合需求，其接受準則如第一項中所制定的相同

2. 測試環境

2.1 硬體規格

測試將在下列硬體配置上執行:

測試用個人主機:

* CPU: Intel(R) Core(TM) i7-6700 CPU@ 3.40GHz.

 RAM: 8GB.

 Hard Driver: At least 200 GB free spaces available.

 Monitor: Intel(R) HD Graphics 530.

 Network Card:

Intel(R) I211 Gigabit Network Connection.

IOT node:

工研院自主研發cc2538 chipset sensor node

Temperature sensor(SHT21):

Datasheet: <http://www.farnell.com/datasheets/1780639.pdf>

其他: TTL轉USB模組

2.2 軟體規格

測試需要使用下列軟體:

Operating System: Ubuntu (instance contiki 3.0 supplied by contiki)、contiki and Microsoft Windows 10 pro

Browser :FireFox(56以前版本) and copper(firfox 附加

元件)皆安裝在Ubuntu

Virtual machine : VMWare player

其他 : AccessPort(安裝在windows)、coap npm(Ubuntu)、python2.7(windows)

.

2.3 測試資料來源

測試的活動需要參考下列資料來源:

1. 子計畫四 IOT-需求規格書。

3. 測試時程、程序與責任

3.1 測試時程

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Predecessor tasks** | **Effort** | **Finish Date** |
| (1) 準備測試計劃 | 測試畫書撰寫完畢 | 2 days | 2018/4/20 |
| (2) 設計整合測試案例 | Task 1 | 2 days | 2018/4/25 |
| (3) 準備整合測試案例 | Task 2 | 3 days | 2018/4/25 |
| (4) 建立整合測試環境 | Task 3 | 3 days | 2018/4/28 |
| (5) 進行整合測試 | Task 4 | 4 days | 2018/5/10 |
| (6) 撰寫整合測試結果報告 | Task 5 | 2 days | 2018/5/10 |
| (7) 設計接受測試 | Task 6 | 3 days | 2018/6/14 |
| (8) 準備接受測試案例 | Task 7 | 3 days | 2018/6/14 |
| (9) 建立接受測試環境 | Task 8 | 3 days | 2018/6/15 |
| (10) 進行接受測試 | Task 9 | 4 days | 2018/6/30 |
| (11) 撰寫接受測試結果報告 | Task 10 | 5 days | 2018/6/30 |

3.2 測試程序

3.2.1 測試方法

(1) 需求規格測試

開發人員依照多媒體雲端計畫書撰寫需求規格書，並由各計畫負責教授檢閱所寫需求規格是否正確且完整。

(2) 架構設計測試

開發人員需盡可能參與討論詳細架構設計，並驗證架構設計是否可行，原則上於每周一次的Scrum meeting與每周兩次的daily scrum進行討論。

(3) 整合測試

1. 子計畫四:
   1. 將IOT node的資料送達border router顯示
   2. 透過border router 下達COAP指令來修改IOT node 的內部資料
2. 子計畫四對於子計畫(DATA):
   1. 能將子計畫四中的各個IOT node所讀取的資訊透過https傳送到子計畫(DATA)的資料庫
   2. IOT mote 與 溫溼度感測器(SHT21)的串接與讀取溫度與濕度

(4) 接收測試

系統完成所有整合測試後，進行接收測試，用以驗證系統的功能符合需求規格書上的需求，同時輔以追朔表進行需求追蹤，確保需求皆有對應到接收測試案例。

(5) 問題追蹤、變更管理

1. 使用github進行程式版本控管，並進行追蹤。
2. 定時更新當前進度的程式碼。

3.2.2整合測試程序

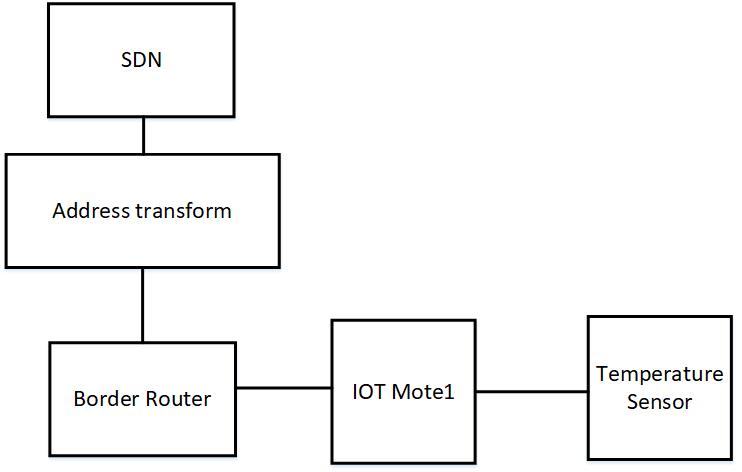


圖:本系統測試程序(IOT\_IT\_01~ IOT\_IT\_14)，依序整合如下

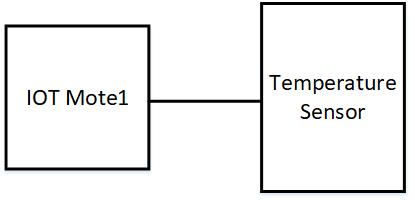


圖:首先整合IOT\_IT\_13、IOT\_IT\_14

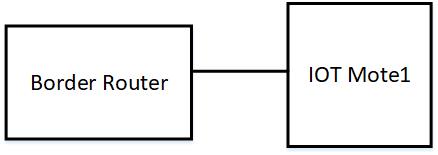


圖:再整合IOT\_IT\_01、IOT\_IT\_02、IOT\_IT\_03、IOT\_IT\_04

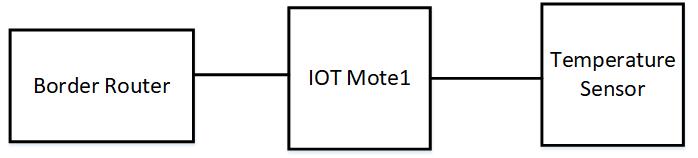


圖:再整合IOT\_IT\_05、IOT\_IT\_06、IOT\_IT\_07、IOT\_IT\_08

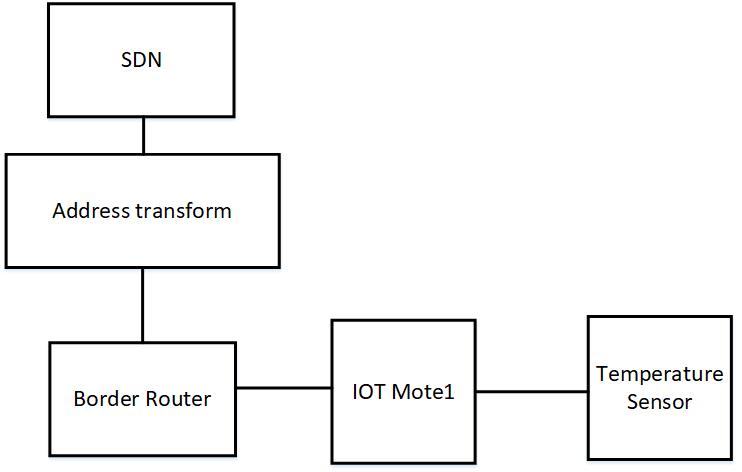
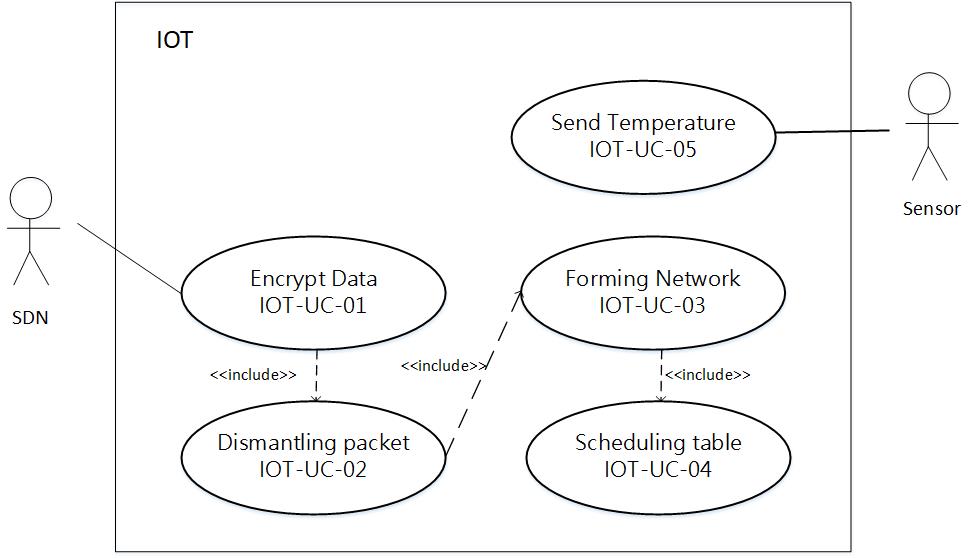
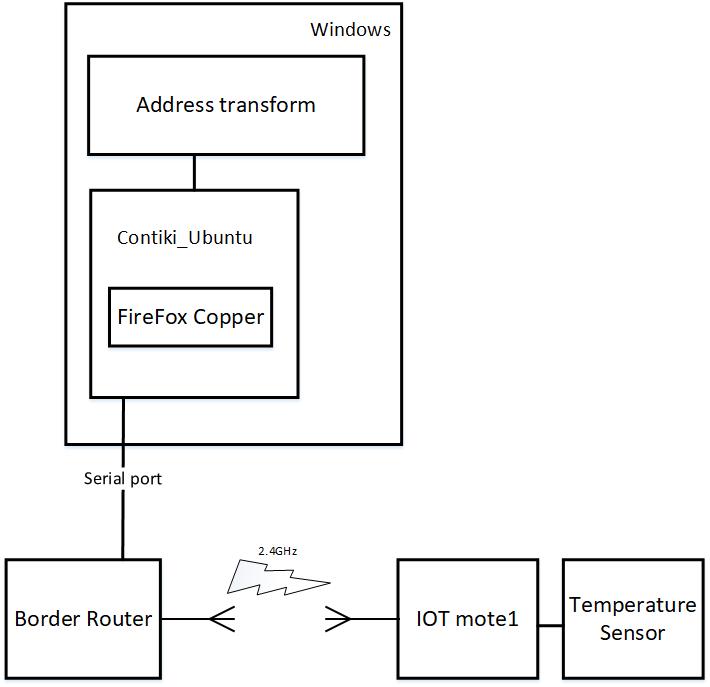


圖:再整合IOT\_IT\_09、IOT\_IT\_10、IOT\_IT\_13、IOT\_IT\_14

3.2.3接受測試程序



3.2.4 測試環境架構圖



3.3 測試人員責任

|  |  |  |
| --- | --- | --- |
| 角色 | 責任 | 參與人員 |
| **Test Design** | 準備測試計畫書  準備測試設計規格 | 陳彥叡  林進偉  李宜倫 |
| **Test Analysis** | 準備測試案例規格  準備測試程序規格  撰寫系統測試報告 | 陳彥叡  林進偉 |
| **Test Execute** | 執行測試項目  檢查批次測試結果  處理測試錯誤報告 | 陳彥叡  李宜倫 |

4. 測試案例

4.1 整合測試案例(內部)

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_01 |
| 測試名稱 | IOT mote成功加入網路 |
| Goal | 確認IOT mote正確加入網路 |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近 |
| 測試期望結果 | 於輸入步驟4所產生的terminal顯示  等待 border router視窗顯示 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_02 |
| 測試名稱 | IOT mote加入網路失敗 |
| Goal | 確認IOT mote沒有加入網路 |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出設為windows 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 3. 將IOT mote放置在離border router訊號範圍外的位置 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 |
| 測試期望結果 | 於terminal顯示 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_03 |
| 測試名稱 | 顯示IOT mote資訊 |
| Goal | 於border router上顯示IOT mote資訊 |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近 5. 等待 border router視窗顯示      1. 開啟新的terminal 輸入coap observe “coap://fd00::212:4b00:615:a677/g/nodeID” |
| 測試期望結果 | 於terminal 顯示[nodeID] id : a677 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_04 |
| 測試名稱 | 顯示IOT mote資訊失敗 |
| Goal | 於border router上無法顯示IOT mote資訊 |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近 5. 等待 border router視窗顯示      1. 開啟新的terminal 輸入錯誤的參數   coap observe “coap://fd00::212:4b00:615:a677/g/node” |
| 測試期望結果 | 於terminal 沒有顯示訊息 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_05 |
| 測試名稱 | 顯示IOT mote所讀取的溫、溼度 |
| Goal | 於border router上顯示IOT mote的溫、溼度資訊 |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近 5. 等待 border router視窗顯示      1. 開啟新的terminal 輸入   coap observe “coap://fd00::212:4b00:615:a677/g/temperature” |
| 測試期望結果 | 於新開的terminal 顯示  temperature : 26  humility : 74  (溫度室溫約18~28C 濕度約50-80%) |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_06 |
| 測試名稱 | 顯示IOT mote所讀取的溫、溼度失敗 |
| Goal | 於border router上無法顯示IOT mote的溫、溼度資訊 |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近 5. 等待 border router視窗顯示      1. 開啟新的terminal 輸入錯誤參數   coap observe “coap://fd00::212:4b00:615:a677/g/temper” |
| 測試期望結果 | 於新開的terminal 沒有顯示訊息 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_07 |
| 測試名稱 | 修改IOT mote內部參數 |
| Goal | 修改遠端mote內部參數 |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近 5. 等待 border router視窗顯示      1. 開啟新的terminal 輸入coap observe “coap://fd00::212:4b00:615:a677/g/freqency” 2. 於terminal 顯示 frequency : 20 3. 開啟firefox copper，於網址輸入   coap://[fd00::212:4b00:615:a677]:5683並按下  Discover按鈕   1. 於網址輸入   coap://[fd00::212:4b00:615:a677]:5683/cloud/temperatur?threshold=10，並按下POST按鈕 |
| 測試期望結果 | 於terminal 顯示  freqency : 10 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_08 |
| 測試名稱 | 修改IOT mote內部參數失敗 |
| Goal | 於border router上顯示IOT mote的溫、溼度資訊 |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近 5. 等待 border router視窗顯示      1. 開啟新的terminal 輸入coap observe “coap://fd00::212:4b00:615:a677/g/frequency” 2. 於terminal 顯示 frequency : 20 3. 開啟firefox copper，於網址輸入   coap://[fd00::212:4b00:615:a677]:5683並按下  Discover按鈕   1. 於網址輸入錯誤參數thres=10   coap://[fd00::212:4b00:615:a677]:5683/cloud/temperatur?thres=10 |
| 測試期望結果 | 於terminal 顯示  frequency : 20 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_09 |
| 測試名稱 | 將訊息成功轉送至address transform模組 |
| Goal | 準備將資訊送往SDN |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近等待 border router視窗顯示      1. 於window內開啟新terminal，輸入 cd ../coap-observer 2. 修改coap-observer資料夾內singlemote.python的host參數為fd00::212:4b00:615:a677 3. python singlemote.python |
| 測試期望結果 | 於log資料夾內檔案顯示  rootId:a69b  moteId:a677  Temperature:26  Humility:70  Date: 1525495200.0 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_10 |
| 測試名稱 | 將訊息轉送至address transform模組失敗 |
| Goal | 準備將資訊送往SDN |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近等待 border router視窗顯示      1. 於window內開啟新terminal，輸入 cd ../coap-observer 2. 未修改coap-observer資料夾內singlemote.python的host參數為 3. python singlemote.python |
| 測試期望結果 | 於log資料夾內檔案顯示 error |

* 1. 整合測試案例(外部)

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_11 |
| 測試名稱 | 啟動溫濕感測器(SHT21)並讀取(成功) |
| Goals | 成功透過IOT mote讀取SHT21的感測溫、濕度 |
| 前置條件 | 1. 將sht21與IOT mote以I2C的方式串接 2. 於SDA、SCL與正電間加上10K歐姆電組 |
| 測試步驟與輸入資料 | 1. 將IOT mote的TX、RX接上TTL轉USB模組，並插上測試電腦且確認模組上LED正常閃爍 2. 於測試電腦開啟AccessPort程式並選定TTL轉USB模組所對應的PORT，接著點取開始監看 |
| 測試期望結果 | 在AccessPort程式內顯示:  Temperature : 26.01 C  Humanity : 73 %  (溫度室溫約18~28C 濕度約50-80%) |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_12 |
| 測試名稱 | 啟動溫濕感測器(SHT21)並讀取(失敗) |
| Goals | IOT mote無法讀取SHT21的感測溫、濕度 |
| 前置條件 | 1. 將sht21與IOT mote未以I2C的方式串接 2. 於SDA、SCL與正電間未加上10K歐姆電組 |
| 測試步驟與輸入資料 | 1. 將IOT mote的TX、RX接上TTL轉USB模組，並插上測試電腦且確認模組上LED正常閃爍 2. 於測試電腦開啟AccessPort程式並選定TTL轉USB模組所對應的PORT，接著點取開始監看 |
| 測試期望結果 | 在AccessPort程式內顯示:  DHT21 don’t open |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_13 |
| 測試名稱 | 將訊息成功轉送SDN |
| Goal | 將資訊送往SDN |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近等待 border router視窗顯示      1. 於window內開啟新terminal，輸入 cd ../coap-observer 2. python collectCLI.python 3. 輸入 add fd00::212:4b00:615:a677 |
| 測試期望結果 | 於瀏覽器輸入  <https://140.124.184.204:8181/api/get_iot/1>  查看是否每20秒更新一次資料 |

|  |  |
| --- | --- |
| Test case ID | IOT\_IT\_14 |
| 測試名稱 | 將訊息轉送SDN失敗 |
| Goal | 將資訊送往SDN |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近等待 border router視窗顯示      1. 於window內開啟新terminal，輸入 cd ../coap-observer 2. python collectCLI.python 3. 輸入 add fd00:212:4b00:615:a677 |
| 測試期望結果 | 於視窗內顯示  Address error |

* 1. 接受測試

|  |  |
| --- | --- |
| Test case ID | IOT\_AT\_01 |
| 測試名稱 | 啟動溫濕感測器(SHT21)並讀取(成功) |
| Goals | 成功透過IOT mote讀取SHT21的感測溫、濕度 |
| 前置條件 | 1. 將sht21與IOT mote以I2C的方式串接 2. 於SDA、SCL與正電間加上10K歐姆電組 |
| 測試步驟與輸入資料 | 1. 將IOT mote的TX、RX接上TTL轉USB模組，並插上測試電腦且確認模組上LED正常閃爍 2. 於測試電腦開啟AccessPort程式並選定TTL轉USB模組所對應的PORT，接著點取開始監看 |
| 測試期望結果 | 在AccessPort程式內顯示:  Temperature : 26.01 C  Humanity : 73 %  (溫度室溫約18~28C 濕度約50-80%) |

|  |  |
| --- | --- |
| Test case ID | IOT\_AT\_02 |
| 測試名稱 | 啟動溫濕感測器(SHT21)並讀取(失敗) |
| Goals | IOT mote無法讀取SHT21的感測溫、濕度 |
| 前置條件 | 1. 將sht21與IOT mote未以I2C的方式串接 2. 於SDA、SCL與正電間未加上10K歐姆電組 |
| 測試步驟與輸入資料 | 1. 將IOT mote的TX、RX接上TTL轉USB模組，並插上測試電腦且確認模組上LED正常閃爍 2. 於測試電腦開啟AccessPort程式並選定TTL轉USB模組所對應的PORT，接著點取開始監看 |
| 測試期望結果 | 在AccessPort程式內顯示:  DHT21 don’t open |

|  |  |
| --- | --- |
| Test case ID | IOT\_AT\_03 |
| 測試名稱 | 將訊息成功轉送SDN |
| Goal | 將資訊送往SDN |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近等待 border router視窗顯示      1. 於window內開啟新terminal，輸入 cd ../coap-observer 2. python collectCLI.python 3. 輸入 add fd00::212:4b00:615:a677 |
| 測試期望結果 | 於瀏覽器輸入  <https://140.124.184.204:8181/api/get_iot/1>  查看是否每20秒更新一次資料 |

|  |  |
| --- | --- |
| Test case ID | IOT\_AT\_04 |
| 測試名稱 | 將訊息轉送SDN失敗 |
| Goal | 將資訊送往SDN |
| 前置條件 | 1. 將border router接上TTL轉USB模組，並接上測試電腦，並且將port輸出從windows系統轉為Ubuntu 2. 於Ubuntu terminal 輸入 dmseg尋找border router所對應的port，ttyUSB0 |
| 測試步驟與輸入資料 | 1. 開新terminal輸入cd $(CONTIKI)/tools 2. make tunslip6(使用一次即可) 3. sudo ./tunslip6 -s /dev/ttyUSB0 fd00::1/64 4. 將IOT node接上電源並放置在border router附近等待 border router視窗顯示      1. 於window內開啟新terminal，輸入 cd ../coap-observer 2. python collectCLI.python 3. 輸入 add fd00:212:4b00:615:a677 |
| 測試期望結果 | 於視窗內顯示  Address error |

5.測試結果與分析

5.1 整合測試案例

|  |  |  |
| --- | --- | --- |
| **Test Case #** | **Results(PASS/FAIL)** | **Comment** |
| IOT\_IT\_01 |  |  |
| IOT\_IT\_02 |  |  |
| IOT\_IT\_03 |  |  |
| IOT\_IT\_04 |  |  |
| IOT\_IT\_05 |  |  |
| IOT\_IT\_06 |  |  |
| IOT\_IT\_07 |  |  |
| IOT\_IT\_08 |  |  |
| IOT\_IT\_09 |  |  |
| IOT\_IT\_10 |  |  |
| IOT\_IT\_11 |  |  |
| IOT\_IT\_12 |  |  |
| IOT\_IT\_13 |  |  |
| IOT\_IT\_14 |  |  |

5.2 接受測試案例

|  |  |  |
| --- | --- | --- |
| **Test Case #** | **Results(PASS/FAIL)** | **Comment** |
| IOT\_AT\_01 |  |  |
| IOT\_AT\_02 |  |  |
| IOT\_AT\_03 |  |  |
| IOT\_AT04 |  |  |
|  |  |  |

Appendix A： 追溯表 Traceability

A.1. 系統 vs. 整合測試案例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 子系統  測試案例 | SDN | Address transform | Border router | IOT Mote | Temperature sensor |
| IOT\_IT\_01 |  |  | o | o |  |
| IOT\_IT\_02 |  |  | o | o |  |
| IOT\_IT\_03 |  |  | o | o |  |
| IOT\_IT\_04 |  |  | o | o |  |
| IOT\_IT\_05 |  |  | o | o | o |
| IOT\_IT\_06 |  |  | o | o | o |
| IOT\_IT\_07 |  |  | o | o | o |
| IOT\_IT\_08 |  |  | o | o | o |
| IOT\_IT\_09 |  | o | o | o | o |
| IOT\_IT\_10 |  | o | o | o | o |
| IOT\_IT\_11 |  |  |  | o | o |
| IOT\_IT\_12 |  |  |  | o | o |
| IOT\_IT\_13 | o | o | o | o | o |
| IOT\_IT\_14 | o | o | o | o | o |

A.2. 需求 vs. 接受測試案例

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 需求  測試  案例 | UC-01 | UC-02 | UC-03 | UC-04 | UC-05 |
| IOT\_AT\_01 |  |  |  |  | o |
| IOT\_AT\_02 |  |  |  |  | o |
| IOT\_AT\_03 | o | o | o | o |  |
| IOT\_AT\_04 | o | o | o | o |  |
|  |  |  |  |  |  |