情境式系統配置代理人於M2M之應用

I-lung Tsai, Wan-rong Jih, Yen-Ling Kuo* and Jane Yung-jen Hsu†
Department of Computer Science and Information Engineering and
Intel-NTU Connected Context Computing Center
National Taiwan University
Taipei, Taiwan
{*r99944030, wrjih, yjhsu*†}*@csie.ntu.edu.tw, a33kuo@gmail.com**

***Abstract*—Machine-to-Machine (M2M) 是近年來興起的科技議題，主要訴求讓機器可以自動溝通、協調和設定，而不需「人」的介入， 然而，最迫切且重要的是 "如何讓系統可以根據使用者的需求，來重新調整機器的運作方式"，因此 Reconfiguration 和 Adaptation 的問題最爲關鍵。本論文提出了情境式系統配置代理人的架構，包含三種提供給使用者們的文件：Profile、Objective 與 Scenario，分別用來描述電子設備資訊、系統功能及使用者的需求；以及五種代理人來分析環境狀態，並自動重新設定電子設備的運作，且我們利用基因演算法來實作 Adaptive Control，讓系統自動調整電器以回應環境變化。本論文最後以小型研討室的環境爲例，來展示各種使用者情境變化下，我們所設計的架構可以因應隨時變動的環境和使用者的需求來調整，達到 92% 正確的情境轉換，且 10 秒內就可以找出最適的電器控制。**

***Keywords*-M2M; reconfiguration; adaptation;**

I. Introduction

機器對機器 (Machine-to-machine, M2M) 是近年來最流行的趨勢，主要訴求讓任何機器都能透過網路通訊、交換資料而不需人爲介入。近幾年來，許多無線感測網路技術也慢慢轉向 M2M 發展，讓不同的 M2M 應用彼此能交換訊息、整合、和分散管理 [1]。市面上有許多 M2M 的應用，像是節能綠屋和家庭自動化，這些應用主要是收集有人類活動區域的環境資料，經過分析且爲了某些特定目的去回應人們的需求。

然而，目前沒有一套可以整合這些智慧型服務應用的系統，這些應用在開發的時候是分散而且獨立的，不同的系統之間不會相互傳遞訊息，所以要讓這些系統同時配合環境需求、並立即調系統狀態是不容易的。因此，在開發這些智慧型服務應用時會有兩個關鍵的問題：

- Reconfiguration: 我們希望將一套系統搬移到另一個環境的時候，一樣順暢自然地運作，而不需要再重新根據該建築環境重新設定，譬如在一般家庭裡是節能和舒適度調整的應用，在辦公或開會的環境則考慮研討室的應用。
- Adaptation: 當人們有不同的情境需求，可以整合這些智慧型服務應用，找出面面俱到的控制策略，譬如說在同一個房間底下，小孩子會喜歡涼爽的環境，老人可能喜歡溫暖一點的環境。

因此，本篇論文的主要目的是創造一個可以自動重新配置電子設備的系統，降低這些硬體設定和軟體整合的花費，讓人們可以根據不同情境來選擇他們需要的智慧型服務。

II. Background and Related Work

A. Frameworks

2005 年開始，愈來愈多人關注到 M2M 應用對感測網路的穩定需求、可調整性以及可開發性。Gilman Tolle [2] 提出了「SNMS」(Sensor Network Management System)，目標是最佳化管理資源及合併詢問系統，讓開發人員可以直接獲取感測器的資料和事件紀錄來偵錯，然而這套系統不適用於非程式設計專家。Chien-Liang Fok [3] 等人提出了「Agilla」平台，不需事先安裝 M2M 應用，只要透過代理人將工作表格中的資訊傳播出去，但缺點是每個代理人的資源有限，較無法應用於大型的動態感測網路。Manish Kushwaha [4] 等人則提出了「OASiS」平台，提供更簡單抽像的設計方式，將 Object-Centric、Ambient-Aware、Service-Oriented 整合在一起，開發人員可以透過他們所設計的系統介面直接控制建築物裡感測器的運作。2011 年，Stephen Dawson-Haggerty [5] 等人提出「sMAP」(Simple Measuring and Actuation Profile)，讓開發人員可以透過 Internet 來存取感測網路的資料，直接在 Internet 設計智慧型服務應用。他們爲每一個電子設備設計了資料庫，以方便開發人員在設計 M2M 應用時可以檢索和存取。如同上述所提及的 M2M 架構，我們也希望提供更簡易的調整方式，因此選擇使用代理人系統處理電子設備與環境的資訊。代理人可以分散管理和決策所需的工作，它們所負責的部分均獨立而不受其它代理人影響，當某些代理人停止的時候，我們仍希望其它工作可以繼續，雖然會降低部分服務品質，卻能保證系統可以不間斷運行。

B. Environment Control

室內環境控制可以包含氣候控制和電力管理，其中氣候控制主要著重於氣溫、溼度、通風等層面，應用於溫室的部分居多。1996 年 Marc Fountain [6] 等人開始使用 "數值範圍" 而非 "單點溫度" 來判斷室內環境的條件，之後 Hartmut Pohlheim [7] 等人使用演化策略來對溫室環境進行最佳化控制，設計了許多相關的環境規格與單位標準，透過分析現有的資料來調整環境；J.M Aaslyng [8] 等人則提出了「IntelliGrow」系統，目標在發展系統本身結構和降低系統耗能影響，且最佳化使用自然資源。2011 年 Jan Corfixen Sorensen [9] 提出了一個使用 multi-agent 控制室內環境的方法，將每個控制模組當成是一支 agent，其中選出最可信任的一方作

爲 negotiator，選擇最適當、可接受的數值作爲控制設定點，可預先解決大部份的交互作用問題，將額外損耗降得更低。本篇論文同樣使用類似演化策略的基因演算法來解決環境控制的問題，且利用數值範圍當成使用者的情境目標。而電力管理的部分主要在研究室內環境與電力消耗的關係，現行的電力測量標準包含 ASHARE 以及 Degree-Days (DD) [10]。DD 常用以評估暖氣和冷氣的電力消耗跟溫度的關係，現今大部份的研究者已經運用 DD 的分析法來調查氣候變化對電器的影響，因此本篇論文也使用 DD 評估舒適度與用電量的關係，最後用一套代理人系統找出最適當的電器控制策略。

C. Smart Home

智慧家庭主要的技術可分成可程化控制和最佳化控制，可程化控制較有彈性且兼具包容性，像是 PCS [11]，這項技術稱作 Powerline Carrier System，能透過電力線傳輸新的原始碼，到一般家庭裡已經布建好的可編程開關或插座上。這些訊號能夠傳達命令到對應的位址上或位置上，因此能隨時遠端控制特定的電子設備。最佳化控制應用的目的在於讓居民更簡易地使用系統，而系統可以自動做到最好，比如 2011 年由 Tony Fadell 開發的 Nest [1]，它是一組室內恆溫控制系統，可以自己學習使用者在什麼時候偏好什麼溫度，而且沒人在家時還能自動降低功率。

III. Scenario-based Framework

我們對 Reconfiguration 和 Adaptation 的定義如下：

- Reconfiguration
  針對不同的智慧型服務應用，可以自動管理並重新設定感測網路中的節點參數。當環境中有新的感測器 (sensor) 進入、離開或故障時，不需要中斷系統，即可有效且快速地將之整合到已有的智慧型服務應用中。
- Adaptation
  針對不同的情境，可以自動找出家中電器 (actuator) 最佳的運作方式。當使用者添加新的情境或是安裝新的家電，不需要中斷系統就可以調整電器到最適合的狀態。

要達成上述的目標，我們提出了一個架構，包含三種文件來描述使用者們的需求，以及一套處理流程。

A. Descriptive Documents for Users

我們定義三種不同使用者：安裝人員、開發人員、和居家用户，並利用三種文件 Profile、Objective、Scenario 來描述三個使用者的特性：

1) Profile: 安裝人員熟悉硬體，負責佈建和管理電子設備，因此我們用 Profile 記錄了所有電子設備的規格和資訊，依照不同屬性加以分類，譬如冷氣可以改善溫度、日光燈可以改善亮度等等。Profile 包含感測器節點的資訊、sensor 和 actuator 的規格。前者讓系統知道有哪些 sensor 可以被重新調整，以及哪些 actuator 可以被控制，後者則分別描述這些電子設備的特性。

[1] http://www.nest.com/

Table I
Example for the Node Profile

Node ID	0001
Room	living room
Sensor	[temperature:digital.6, light:analog.4]
Actuator	[fan:digital.8]

Node profile 記錄了每個感測器節點的資訊，如 Table I，除了 ID、所在的房間，還有連接在節點上面的其他電子設備。Sensor 欄位的 “temperature:digital.6” 代表溫度感測器裝置在該節點第 6 個 digital 位置，Actuator 也是同樣格式。

Table II
Example for the Sensor Profile

Sensor ID	Name	Type	Sampling Period	Valid
0001	temperature	digital	100ms	[-40,125]°C
0002	humidity	digital	100ms	[0,100]%
0005	motion	digital	1ms	0,1

Sensor profile 記錄了 ID 和所在的位置，如 Table II，其餘的是它的規格，像它是屬於 digital 或 analog 的，以及最短傳輸週期、最大的感測範圍等等。

Table III
Example for the Actuator Profile

Actuator ID	Name	Voltage	Power
0002	fan	110V	[50,90]W
0004	lamp	110V	[11,15]W
0005	curtain	110V	60W

Actuator profile 記錄了 ID 和所在位置，以及這些電器的規格，像是電壓和耗電功率等等，如 Table III。

2) Objective: 開發人員必須考慮居家用户的需求和情境，來設計完整的運作方式。我們以 Objective 來定義使用者的需求和偏好，以及系統控制 actuator 時必需考慮的各種屬性，包含百分比及者二進制的形式。如 Table IV 描述的，百分比形式代表的是系統在運作時應該多注重這項屬性，也就是權重值，包含 Climate、 Lighting、 Ventilation 等，Climate 代表室內溫濕度，Lighting 代表光線的亮度，而 Ventilation 則是二氧化碳濃度。這幾項和居民的舒適度非常相關，都用百分比表示是因爲有明確的量測方法，且代表該 context 對電器決策有多大影響；二進制形式則代表是否考慮這個屬性，如 Privacy 只和當下的使用情境相關，它們沒有特別定義的量測方法，因此可以視爲是一種限制。

此外，爲了連結 Profile 和 Objective 這兩種文件，我們利用「關係表」來表述每個 actuator 的特性，以及它們應該被歸屬到什麼樣的 Objective 底下，如 Table V。「關係表」是讓系統決定是否啓動 actuator 的評分標準，舉例來說，風扇需要達成的條件有 “Climate”，這代表在評量風扇是否開啓時會考慮「溫度」和「濕度」等因素。

Table IV
EXAMPLE FOR THE OBJECTIVES

Objective	Expression	Index	Valid
Climate	%	THI	0∼40
Lighting	%	illumination index	0∼1500
Ventilation	%	carbon index	0∼10000
Privacy	binary		

Table V
SEMINAR RELATION TABLE

Actuator	Climate	Lighting	Ventilation	Privacy
fan	√		√	
lamp		√		
curtain		√		√

3) Scenario: 居家用户爲了不同需求和偏好創造許多 Scenario，而情境式系統配置代理人再根據現在的情境、狀態來自動切換這些 Scenario，並調整內部的運作方式。Scenario 用來定義居家用户的偏好和需求，如 Table VI，使用者偏好可以對應到 Objective 底下的屬性。比如 “Empty” 對 Climate、Lighting、Ventilation 的權重各是 15%、0%、15%，因此系統完全不控制 Lighting，同時只會少量地關注 Climate 和 Ventilation。而當 Scenario 有勾選二進制屬性的時候，系統會參考「關係表」來啓動 actuators。比如 “Quiet Work” 需要 Privacy，而且如 Table V 所示，整個「關係表」和 Privecy 相關的電器只有 curtain，這時系統會特別讓 curtain 關閉。

Table VI
SEMINAR SCENARIOS

Scenario	Climate	Lighting	Ventilation	Privacy
Empty	15%	0%	15%	
Quiet Work	60%	60%	60%	√
Presentation	95%	20%	95%	
Discussion	90%	85%	65%	

因此，每個 Scenario 在決策電器控制時可以和「關係表」作結合。如 Figure 1 所示，每個 Scenario 將權重值或 binary 數值填入對應的「關係表」，因此 Scenario 改變權重也跟著改變，最終會影響到電器決策的結果。

B. Process Flow

Figure 2 說明了處理的流程，每一個區塊都會有一支代理人負責處理：

- Start
  這個階段代表系統開始運作，會直接讀取環境中的電子設備資料和當下 Scenario 的資料。
- Configuration
  這個階段代表設定電子設備的運作方式，可以由安裝人員手動設定，或由 Configuration 代理人解析 Start 階段獲得的資料來自動設定。
- Communication

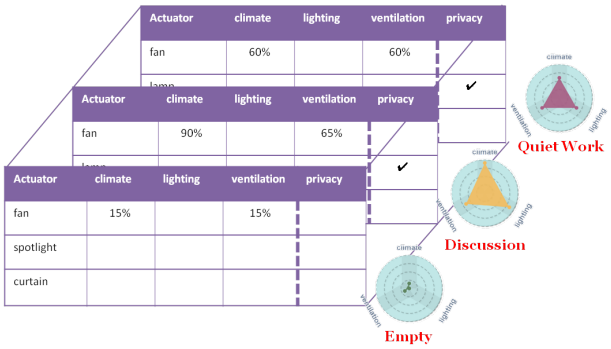

Figure 1. Profile, Objective and Scenario

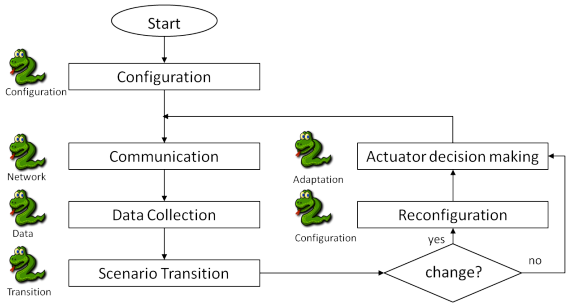

Figure 2. Flow Diagram

這個階段代表設定網路的溝通方式，Network 代理人會自動解析 Internet 訊號或無線網路送過來的資料，再轉交給下一階段使用。

- Data Collection
  Data 代理人會接收各個節點送回來的資料，像是溫度、濕度，這些資料會根據每個 sensor 定義在 Profile 中的 location 來決定需要反應的 actuator。
- Scenario Transition

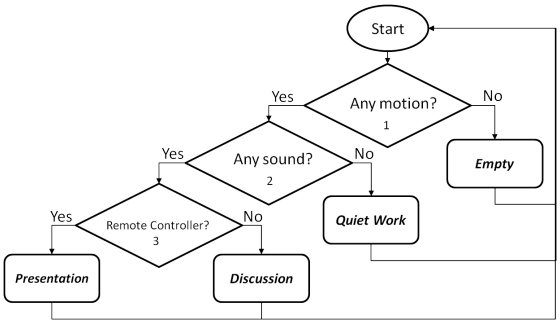

Figure 3. Transition Flow

Transition 的目標是自動判別 Figure 3 所設定的條件，如果滿足便會切換 Scenario。這些條件會用邏輯表示式存在資料庫中，Transition 只是一個判斷 True or False 的代理人，它會將這些邏輯表示式抓出來，並利用 sensor 的資料進行比對。比如判斷式1代表如果房間裡有任何一個 motion sensor 的數

值回傳爲 True，會自動將其分類到其他結果，否則爲 Empty。

- Reconfiguration
  Configuration 代理人獲得情境轉換的訊息後，會先讀出 Profile 和 Objective 並查詢關係表，根據目前的使用者需求找出 Scenario 的參數表。將這些資訊組合成完整的配置指令後送出，節點接著會擷取、解析包含在這些指令底下的參數，重新設定節點和 sensor 的運作方式。如 Figure 4 所示，這些指令集支援了節點、sensor 的參數決定和 actuator 的啓動、關閉。

Command	Example	Description
<ID> ***reset***	0001 ***reset***	0001回復到初始狀態
<ID> ***sleep***	0001 ***sleep***	0001進入一個idle的迴圈
<ID> ***insert*** <pin>:<type>	0001 ***insert*** 4:motion	0001的pin 4設為 motion sensor
<ID> ***set_period*** <pin>:<value>	0001 ***set_period*** 4:1000	0001的pin 4回傳資料的週期為 1000 ms
<ID> ***write*** <pin>:<value>	0001 ***write*** 8:1	0001的pin 8 輸入1為 High，0為Low

Figure 4. Command Set

- Actuator Decision Making
  Scenario 切換後，我們需要另一個 Adaptation 代理人找出最適合的 actuator 來達成目標。

IV. Adaptive Control

爲了解決舒適度和電力取捨的問題，我們選擇使用基因演算法 (Genetic Algorithm, GA) 來分析當下的環境資料。在一定的時間內，我們的系統可以找出最好的 actuator 設定。基因演算法可以帶來許多的好處，像是：

- 基因演算法可以在任何時間結束運算。
- 基因演算法的 encoding 和 fitness 如果設計良好，可以很快地找到一組不錯的解答。
- 基因演算法可以輕易地讓安裝人員、開發人員和居家用户將 Profile、Objective、Scenario 套在其中一個環節上，比如增加想控制的電器只要增加染色體長度、增加想監測的 context 只要改變 Fitness Function 的變數。

A. Encoding

我們定義所有電器控制的集合稱爲 actuator plan，而每一組 actuator plan 就是基因演算法中的一組染色體，並用浮點數來描述每一個 actuator 的控制，這些浮點數代表 kWh 的資料。舉例來說，如果我們將和 Climate 相關的 actuator 排成一序列，並從 Profile 中讀取相關的資訊，像是 {air-conditioner, fan, dehumidifier}，它的初始數值便可能是 {700, 40, 210} kWh。接著系統依據 Fitness Function 計算出最好的分數，以保證系統可以不斷地自動調整及更新。Table VII 記述了所有我們會使用到的基因演算法參數，其中電器數量是可以動態更新的，不同環境中需要的電器數量會不同，比如家庭、研討室和辦公室就會有不同的電器，而電器數量也是代表染色體的長度。

Table VII
Genetic Algorithm Parameters

GA Parameters	Discription	Out Setting
Max Computation Time	最大的計算時間	10 seconds
Max Rounds	最大的計算回合數	500
Max Standpoint	最大的停止演化回合數	50
Max Selection	最大的替換下一代比率	10%
Population Size	族群個數	50
Number of Actuators	電器數量	8
Sampling for Actuators	取樣交配的數量	3
Crossover Rate	交配機率	90%
Mutation Rate	突變機率	10%

B. Fitness

居家用户會事先定義許多的 Scenario 目標，隱含著不同的要求，當目前的環境條件已經遠離目標，我們當然會希望獲得一個最佳策略去控制家裡的電器，然而我們系統和固定目標的最大不同是可變化的 fitness 公式，也就是說，即使在同樣的環境底下，當使用者有不同的需求時，甚至可以得到不同的控制策略。我們定義 Fitness 的公式如下：

$$\min \sum_{a \in A} \left[\alpha_e \times \phi_e^a + (1 - \alpha_e) \sum_{s \in \{c,l,v\}} \alpha_s \times \phi_s \right] \quad (1)$$

其中：

- A 代表 actuator 的集合，a 代表一個 actuator
- α_e 代表 Energy Saving 的權重，譬如 α_e 在 “Discussion” 下是 0.2
- ϕ_e^a 代表 Energy Saving 與目標的誤差
- α_s 代表 Climate、Lighting、Ventilation 的權重，譬如 α_c: α_l: α_v 在 “Discussion” 下是 0.90: 0.85: 0.65
- ϕ_s 代表 Climate、Lighting、Ventilation 與目標環境的誤差

現在我們用「冷氣」和「電風扇」當作例子，並依據 Fitness Function 來評分。假設現在是夏天，若屏除使用者的需求，只單看每個電器可以提供的舒適度，分數的分布可能會如 Table VIII 所示。

Table VIII
Fitness for Climate

	air-conditioner	fan
comfort measurement (climate)	0.1	0.6
energy measurement	0.7	0.3
average	0.40	0.45

如果不考慮 Scenario 的權重，則可以發現冷氣和電風扇在這個時刻啓動不會有太大的差異。若我們納入情境的考量，使用 Scenario 底下的 Climate 數值作爲權重，比如 “Discussion” 下 α_c 爲 0.9 且 α_e 爲 0.2，再觀察同樣的例子，差異就會明顯許多。

Table IX 表示在 “Discussion” 情境下的結果，因爲風扇調整後的分數很糟，所以它不可能被當成是主要降溫

Table IX
FITNESS FOR CLIMATE : DISCUSSION

	air-conditioner	fan
comfort measurement (climate)	0.1 × 0.9	0.6 × 0.9
energy measurement	0.7 × 0.2	0.3 × 0.2
weighted sum	0.23	0.60

的 actuator。但這樣的結果不是絕對的，因爲 fitness 的分數在夏天或者冬天可能是完全相反的。

可變動的 Fitness Function 最大的功效是在於能夠將不同環境和情境同時納入考量，卻只需要同樣一套系統的運作方式去產生電器決策，而基因演算法第三個優點是可以在執行時間內更改 Fitness Function，因爲目標環境和權重值不同，也會讓電器的控制方式大不相同；如果在房間裡裝置了新的電子設備，也可以很快地加入 Fitness Function 的運算，完全不會耽誤系統的運作。

V. EXPERIMENTS

A. Experimental Design

我們使用學校的小型研討室來佈建系統配置代理人，並考量現實會發生的情境，讓代理人可以自行紀錄和切換情境、重新設定電子設備、配合環境和控制研討室裡的電器。

1) Environment:

- Node Program and Network
  我們使用單板微控制器 Arduino Uno 作爲感測器節點，並制定一串接收指令集燒錄在節點上。環境的原始資料會透過 XBee 來傳送，控制 actuator 則是 radio frequency (RF) 或 ZWave，情境切換一類的網路訊號則直接透過 Internet 或者 WiFi 來傳遞。
- Agents
  代理人用 python 語言編寫，每個系統代理人都是無限的執行迴圈。
- User Interface
  使用者介面可以當作代理人、資料庫和各個節點的中介層。我們讓三種使用者透過使用者介面修改可控制的相關變數，比如 Profile、Objective、Scenario 等文件。

2) Seminar Scenario: 研討室可能會用到的 Scenario 有：空房 (Empty)、靜態作業 (Quiet Work)、報告中 (Presentation) 和討論中 (Discussion)，我們對這些 Scenario 有不同預設的屬性數值，如 Table X 描述了上述那些不同 Scenario 所預設的目標。

Table X
SCENARIO GOAL

Scenario	Climate(THI)	Lighting	Ventilation (CO_2)
Empty	11~30	200~350	300~700
Quiet Work	20~26	500~750	300~700
Presentation	16~19	350~500	1300~2000
Discussion	16~19	750~1000	1300~2000

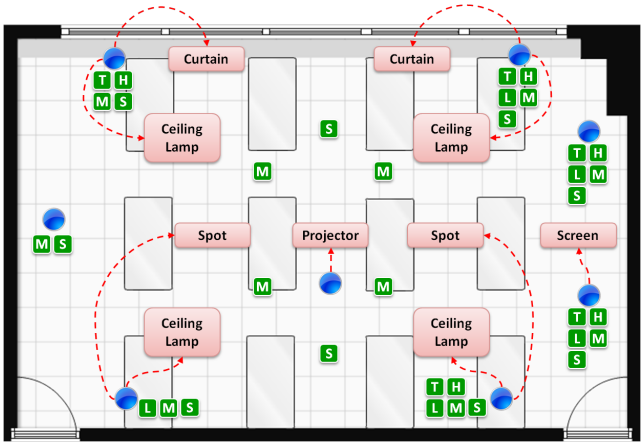

Figure 5. The Floor Plan of Seminar Room

Figure 5 是我們佈建的無線感測網路和可以控制的電器，圓形代表 Arduino 節點；單一英文字母的方框代表 sensor，T 是 temperature、H 是 humidity、L 是 light、M 是 motion、S 是 sound；弧邊方框代表 actuator，Arduino 節點可以直接控制附近的 actuator。

B. Evaluation

Reconfiguration 和 Adaptation 最大的不同在於 Reconfiguration 不是隨時進行的，而是當有情境變化以及電子裝置佈建發生變化的時候，因此要評估 Reconfiguration 必須從 Transition 的判斷精確度來下手，而我們也會計算 Adaptation 所需的時間長短，來驗證這套代理人系統是方便有效率的。

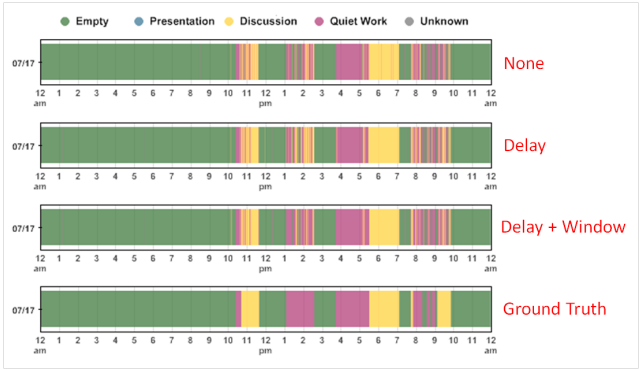

Figure 6. Transition Accurancy

1) Transition Accuracy: Figure 6 是我們在研討室進行一週的 Transition 測試，且取其中一個禮拜二來觀察的結果。我們有三種不同的 Transition 方法，分別是：None、Delay、及 Delay + Window。None 是直接判斷當下 sensor 的數值；Delay 是加上情境延遲時間 (本論文使用 5 秒延遲)；Delay + Window 是加上利用歷史資料提升判斷的準確性。

我們手動標記了一整天的情境，紀錄三個方法的 Transition 結果，且使用 window size 來計算情境轉換的

精確度，比如說，若五秒以內的情境轉換和現實情境相同，則當作一次正確，否則爲一次錯誤，找出一整天的正確和錯誤次數，之後再計算出情境正確轉換的比率作爲 Transition Accuracy。我們執行過 0 至 400 秒的 window size，其中挑出最好區別三個方法且表現不差的當作我們的評估標準，最後採用 60 秒當作 window size，依此作依據來判斷我們的 Transition 精確程度：None 的精確程度爲 65.88%，Delay 爲 86.82%，Delay + Window 爲 92%，None 的缺失在於很容易因爲門外一點動靜或聲音就導致判斷錯誤而直接切換情境；Delay 主要是針對 Quiet Work 來進行修正，因爲 Quiet Work 非常容易和 Empty 以及 Discussion 混淆，因此 Delay 效果已經明顯變好；Delay + Window 的表現最好，是因爲 Window 可以避免誤判，比如說，如果過去 10 秒前的資料顯示之前就有這些動靜或聲音，此時轉成其他情境會比直接用當下的 sensor 數值來判斷更準確，所以我們選擇 Delay + Window 來當作我們的 Transition 方式，未來在這一塊可以使用更多精確的方法，增進系統 Transition 的表現。

2) Adaptation Time: Figure 7 記錄一整天 Adaptation 的時間花費。Adaptation Time 是指 Adaptation 代理人收到 Transition 代理人發出的訊號，直到找出最適電器控制的時間差，我們取每個小時的平均和最長（最差）時間來評估，結果發現 Adaptation 均不會超過十秒。另外，凌晨兩至四點的最長時間偏短約兩秒，是因爲情境都是處在 Empty 而沒有切換，我們在情境未切換的時候，會考慮繼承上一次 Adaptation 的結果，因此可以大量縮減 Adaptation 的時間。

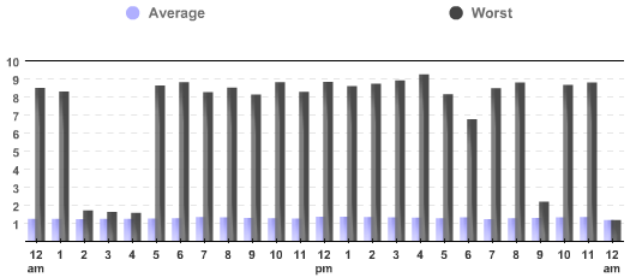

Figure 7. Adaptation Time

Figure 8 針對不同數量的電器，計算 Adaptation 的時間花費。我們模擬了一般家庭的電器環境，並測試 1 到 50 個電器的平均時間花費，每筆資料都是 100 次模擬實驗後計算出平均數值，依此繪出 1 到 50 個電器的時間曲線。且在這個模擬實驗中，我們不讓 GA 繼承上一個計算結果，以表現公平性。實驗結果顯示，我們在擁有 18 個電器以後，這條時間曲線的成長速率會變得平滑且慢，這代表如果我們增加更多電器，也不會對系統造成無限制的負擔，我們也可以從這張圖驗證在擁有 18 個電器的環境下，Adaptation 時間花費會小於 10 秒，而在我們的研討室實驗環境中只使用了 12 個電器，平均的 Adaptation 時間是 7.7 秒。

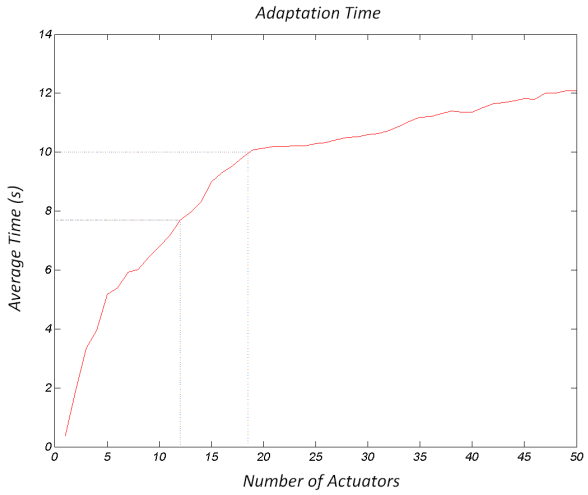

Figure 8. Adaptation Time for Different Number of Actuators

VI. Conclusion

爲了解決 M2M 應用中，電子設備設定 (Reconfiguration) 和操控 actuator 來回應環境 (Adaptation) 的問題，我們提出了三種文件 (Profile、Objective、Scenario) 來描述電子設備與使用者的資訊，代理人可根據這些文件的描述來重新設定、調整 sensor 與 actuator，且本論文利用基因演算法來找出最適策略，其最大的好處是可以不間斷系統執行來達到 Adaptive Control。本系統佈建於學校研討室中，實驗結果顯示我們可以達到 92% Transition 精確程度，以及 10 秒內就可以找出最符合情境的電器控制結果。

Acknowledgment

This work was also supported by National Science Council, National Taiwan University and Intel Corporation under GrantsNSC 100-2911-I-002-001, and 101R7501.

References

[1] N. Wang, N. Zhang, and M. Wang, “Wireless sensors in agriculture and food industry—recent development and future perspective,” *Computers and Electronics in Agriculture*, vol. 50, no. 1, pp. 1 – 14, 2006.

[2] G. Tolle and D. Culler, “Design of an application-cooperative management system for wireless sensor networks,” in *Proceeedings of the 2nd European Workshop on Wireless Sensor Networks (EWSN 2005)*, ser. EWSN ’05, January 2005, pp. 121–132.

[3] C.-L. Fok, G. C. Roman, and C. Lu, “Rapid development and flexible deployment of adaptive wireless sensor network applications,” in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS 2005)*, ser. ICDCS ’05, June 2005, pp. 653–662.

[4] M. Kushwaha, I. Amundson, X. Koutsoukos, S. Neema, and J. Sztipanovits, "Oasis: A programming framework for service-oriented sensor networks," in *Proceedings of the 2nd International Conference on Communication Systems Software and Middleware (COMSWARE 2007)*, ser. COMSWARE '07, January 2007, pp. 1–8.

[5] S. Dawson-Haggerty, X. Jiang, G. Tolle, J. Ortiz, and D. Culler, "smap: A simple measurement and actuation profile for physical information," in *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems (SenSys 2010)*, ser. SenSys '10, 2010, pp. 197–210.

[6] M. Fountain, G. Brager, and R. de Dear, "Expectations of indoor climate control," *Elsevier Energy and Buildings*, vol. 24, no. 3, pp. 179 – 182, 1996.

[7] H. Pohlheim and A. Heiner, "Optimal control of greenhouse climate using real-world weather data and evolutionary algorithms," in *Proceedings of the ACM Conference on Genetic and Evolutionary Computation Conference (GECCO 1999)*, ser. GECCO-99, 1999, pp. 13–17.

[8] J. M. Aaslyng, J. B. Lund, N. Ehler, and E. Rosenqvist, "Intelligrow: A greenhouse component-based climate control system," *Elsevier Environmental Modelling and Software*, vol. 18, no. 7, pp. 657–666, September 2003.

[9] J. C. Sørensen, B. N. Jørgensen, M. Klein, and Y. Demazeau, "An agent-based extensible climate control system for sustainable greenhouse production," in *Proceedings of the 14th International Conference on Agents in Principle, Agents in Practice (PRIMA 2011)*, ser. PRIMA '11, November 2011, pp. 218–233.

[10] Y. Yau and H. Pean, "The climate change impact on air-conditioner system and reliability in malaysia: A review," *Elsevier Renewable and Sustainable Energy Reviews*, vol. 15, no. 9, pp. 4939–4949, 2011.

[11] K. H. Zuberi, "Powerline carrier (plc) communication systems," Master's thesis, Department of Microelectronics and Information Technology, IMIT Royal Institute of Technology, KTH IT-Universitetet, Kista, Stockholm, Sweden, 2003.