基於複合式架構建構具高強健性的智慧家庭服務管理系統

Robust Service Management for Smart Home Environments: A Hybrid Approach

張惟誠, 廖峻鋒
國立政治大學資訊科學系
Wei-Chen Chang and Chun-Feng Liao
Department of Computer Science
National Chengchi University
103753019@nccu.edu.tw, cfliao@nccu.edu.tw

摘　要

智慧家庭環境是一個典型的分散式系統，在此類環境中的智慧服務大都由一至多個節點組成，例如一個冷氣恆溫系統需要冷氣機、溫度感測器和邏輯判斷節點。然而，只要服務其中一個節點故障，整個服務就無法正常運作。由於居住在家庭中的大都是不具技術能力的使用者，故理想的智慧家庭服務，即使在有節點故障的狀況下，也應能在短時間內盡可能自動偵測與排除錯誤，使服務的運作不被中斷。本論文主要目的在於提出一個智慧家庭的強健服務管理系統，基於創新的複合式架構，結合點對點與集中式錯誤偵測機制的特色，能在短時間內偵測到節點失效，進而恢復由於軟體所造成的節點故障或尋找待用節點，使得服務能繼續運行。

關鍵字: 智慧家庭、錯誤偵測、錯誤回復

一、前言

智慧家庭是指利用各式資通訊技術，提供使用者安全、舒適、便利及環保的居住環境，此類環境通常擁有許多感測器、智慧家電等硬體元件，以及佈署在硬體元件上的軟體元件，這些元件分散在家庭環境中，彼此透過網路互相合作提供服務。在智慧家庭中，最基本的軟、硬體單元稱為元件 (Component)，服務 (Service) 則是由數個元件所組成的。例如一個冷氣恆溫服務需要一個感測器、一個邏輯控制元件以及一台冷氣機才能運作。然而，如何將這些分散在環境中的元件組合在一起並形成服務，是個相當複雜的問題。此外，由於智慧家庭一般沒有具專業知識的管理員，一旦系統發生錯誤，使用者往往無法自行解決，再者，在家庭環境中的服務，常常與使用者的生命安全有關，例如居家照護和健康管理等等，若這類的服務在執行過程中，有元件失效 (fail) 而導致服務運作不正常或中斷，很可能對使用者造成危害。因此，在智慧家庭中，需要有服務管理系統，用來處理各式元件的尋找與發現、服務組成、以及錯誤偵測與錯誤回復等問題。

過去的相關研究主要分為兩類：集中式 (Centralized) 和點對點 (Peer-to-Peer)。在集中式架構下，系統將所有受管軟硬體元件的狀態集中在伺服器上，例如 PerSAM(Pervasive Service Application Model)/PSMP(Pervasive Service Management Protocol) [6]；反之，點對點架構並不依賴伺服器，而是每個節點透過 IP 單播或群播的方式，自行維護區域網路內其他節點的狀態，例如，RRCP(Rotating Roll-Call-Based Protocol) [4]。然而，上述兩種方法皆有其缺點：集中式架構會發生單一節點失效的問題，一旦伺服器 (如圖 1(a) 中的 A) 失效，整個系統就必須重啟；而點對點架構則是在節點數量上升到一定程度時，每一輪的 Gossip 耗時過久，如圖 1(b)，A F 完成一輪 Gossip 需時約 6t，也就是說，A F 要確認彼此之間的運作狀況，需要約 6t 的時間；但在圖 1(a) 的集中式架構中，僅需約 t 的時間就可以使 A 知道 B F 的運作狀況。在智慧家庭環境中，高延遲的問題很可能導致意外發生時來不及處理。

根據上述理由，我們以 PerSAM 為基礎，提出了一個新的複合式架構：Manager Node 之間採用點對點，而 Manager Node 對 Worker Node 則

* 本論文部份研究受科技部研究計畫經費補助，編號: 104-2221-E-004-001、104-2627-E-002-006、與 104-3115-E-004-001。

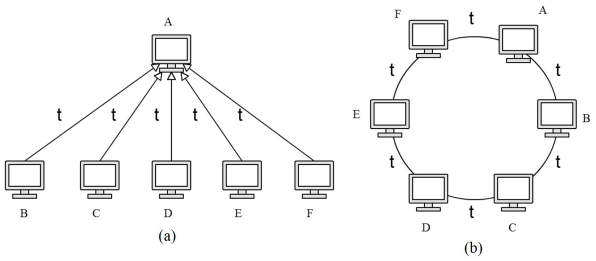

圖 1: 不同架構比較 (a) 集中式 (b) 點對點

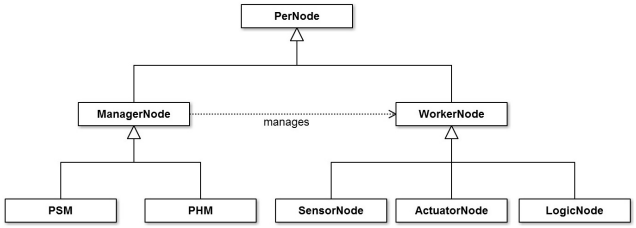

圖 2: PerNode 分類

採用集中式，這篇論文中將使用此複合式架構建構智慧家庭的服務管理系統，期望此系統能夠在長時間的應用場景下，持續不斷的給予使用者服務，達到所謂的高強健性。在下面，首先將簡單回顧相關研究領域與文獻，接下來簡介 PerSAM 架構、PSMP 和 RRCP 的運行機制，之後詳細介紹本論文設計的服務管理系統，最後說明系統評估結果與結論。

二、相關研究

過去關於智慧環境的研究，大多著重在原型 (prototype) 的實作。例如 Dey 利用 Context Toolkit [2] 區隔底層服務與上層的應用；Román 等人提出的 Gaia [8]，將空間及其中的資源視為一種作業系統。透過這些架構，可以降低開發的難度。而後，服務管理議題開始受到重視，許多研究人員嘗試將這些議題委託給底層實作平台提供的服務管理機制。例如 SOCAM [3] 及 CoBra [1] 分別透過 OSGi(Open Sevice Gateway Initiatives) 和 JADE(Java Agent Development Framework) 進行服務管理。不過這些平台僅提供簡易的管理機制，並沒有支援進階的服務組合和服務修復等功能。

近年來在智慧家庭領域中，與容錯 (fault tolerant) 相關的技術漸漸被提出。例如 Kapitanova 等人的 SMART [5] 利用機器學習的方式偵測智慧家庭中的元件錯誤；Silas 等人提出一種集中式的架構 FTSSF [9]，透過資料庫及 log 來做到錯誤偵測及錯誤回復的功能。然而，SMART 不適合非處理感測器類型的失效問題；而一旦資料庫伺服器無法使用，屬於集中式架構的 FTSSF 便無法進行偵錯。PerSAM/PSMP 和 RRCP 相對於 SMART 和 FTSSF 來說，擁有較完整的服務管理機制，但 PSMP 假設 Manager Node 永遠不會失效，這在真實環境中並不會發生；RRCP 則未敘述如何做到錯誤回復的功能以及如何整合 Manager Node 和 Worker Node；此外，由於 PerSAM 為訊息導向 (Message-Oriented) 架構，需要訊息導向中介軟體 (Message-Oriented Middleware, MOM) 才能使服務正常運作，但 PSMP 和 RRCP 皆未考量到 MOM 失效的問題。本論文除了設計出兼顧集中式與點對點優點的架構外，也將針對 PSMP 及 RRCP 較欠缺的整合和 MOM 失效問題進行研究。

三、系統模型

本研究是基於 PerSAM 架構，藉由改良 PSMP 及 RRCP 設計與實作新的複合式機制，因此，在深入論述改良的機制與方法前，首先要介紹相關的背景技術。

3.1 PerSAM 架構

PerSAM 是針對訊息導向智慧家庭系統所設計的服務模型。典型的智慧家庭系統由為數眾多的資訊家電、感測器與軟體服務單元所構成，在 PerSAM 中，最基本的軟體單元稱為 PerNode 或簡稱為 Node。如圖 2所示，Node 分為兩種類型：Manager Node 和 Worker Node。Worker Node 可以根據行為再細分成三種：Sensor Node 會從環境中獲取相關的資訊，例如溫度感測器可以取得當前環境的溫度資訊；Actuator Node 負責執行家電的功能，例如冷氣機；Logic Node 則會將 Sensor Node 取得的資料根據特定邏輯轉換成指令，並讓 Acutaor Node 執行服務。

Manager Node 負責服務管理機制的執行及服務維護、修復等工作，又可分為 Pervasive Service Manager(PSM) 與 Pervasive Host Manager(PHM) 兩種：PSM 主要功能是進行服務組成，並確保所有所需要特定功能的 Worker Node 都正常運作，並在必要時初始化節點的修復工作；PHM 負責管理同一台機器上 Node 的生命週期，包括將 Node 載入到記憶體中和關閉 Node。PerSAM 中 Node 的生命週期如圖 3所示，Node 被安裝 (install) 在稱為 Pervasive Host 的機器後，進入 INSTALLED 狀態，在此狀態下的 Node 可被載入至記憶體 (load)，進入 DORMANT 狀態，DORMANT 狀態的 Node 可以被 discovery protocol 發現，但

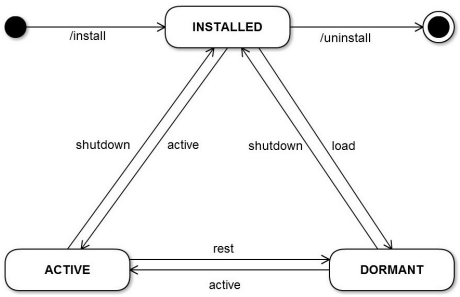

圖 3: PerNode 狀態圖

不進行任何訊息處理的程序。當 Node 被啟動 (active) 後，會進入 ACTIVE 狀態，此時，Node 才可以發送、接收以及處理訊息。同樣的，在 DORMANT 和 ACTIVE 狀態下的 Node 可以被關閉回到 INSTALLED 狀態，或者透過睡眠 (rest) 的指令從 ACTIVE 狀態回到 DORMANT 狀態。

3.2 PSMP

PSMP 是以 PerSAM 為基礎，並對 UPnP(Universal Plug and Play) [10] 中的 SSDP(Simple Service Discovery Protocol) 做延伸的服務管理機制。SSDP 的主要功能包含尋找特定裝置以及即時得知裝置是否可用。SSDP 的原理是透過路由器在 IP 層級所支援的群播 (Multicast) 來實現，SSDP 定義，只要對 239.255.255.250:1900 發送訊息，就會透過群播被區域網路中的 UPnP 裝置接收到。SSDP 是 HTTP(Hypertext Transfer Protocol) 的延伸，其在 HTTP 另行定義了 M-Search 和 Notify 封包，如圖 4和圖 5所示，若有裝置想要尋找特定類型的 M-Search；Notify 訊息包含了裝置的類型和位址等資訊，當一個裝置加入 UPnP 網路時，會發出 Notify 訊息至群播位址，此訊息的 NTS 會被設為"ssdp:alive"，收到的裝置會比對裝置類型 (ST)，若發現有其需要的服務，則根據 Location 標頭取得描述檔，並依據需求做進一步的控制；若有裝置離開 UPnP 網路，會發出 NTS 被設為"ssdp:bye-bye" 的 Notify 訊息，讓網路中其他裝置知道目前無法使用離開的裝置。

當 PSM 要啟動一個服務時，須先送出"psmp:discover" 訊息以尋找符合資格的 Node(圖 6之步驟 1)，"psmp:discover" 為延伸 SSDP 定義出的方法，相較於 SSDP，定義了新的標頭：CRITERIA，CRITERIA 標頭包含許多 key-value pair，這些 key-value pair 可以描述額外的資訊，例如時間和地點，在 Service Discovery 的過程中，Node 需要完全符合 CRITERIA 標頭

```
NOTIFY * HTTP/1.1
Host: 239.255.255.250:1900
Cache-Control: max-age=1800
Location: http://140.119.163.199:4004/description.xml
Server: Microsoft-Windows-NT/5.1 UPnP/1.0 UPnP-Device-Host/1.0
NTS: ssdp:alive
ST: urn:schemas-upnp-org:device:InternetGatewayDevice:1
USN: uuid:e5af0b89-f454-4bd5-83ab-1248eaaced75::upnp:rootdevice
```

圖 4: SSDP Notify 訊息範例

```
M-SEARCH * HTTP/1.1
Host: 239.255.255.250:1900
Man: "ssdp:discover"
MX: 3
ST: urn:schemas-upnp-org:device:TV:1
```

圖 5: SSDP M-Search 訊息範例

才算符合資格，"psmp:discover" 封包內容如圖所示。當符合資格的 Node 收到此訊息後，必須發送回應訊息 (Response message) 給該 PSM(圖 6之步驟 2)，此外，PHM 收到"psmp:discover" 訊息時，會尋找位在同一機器是否有符合資格且狀態為 INSTALLED 的 Node，如果有的話，會將該 Node 載入至記憶體，使之從 INSTALLED 狀態轉變為 DORMANT 狀態 (圖 6之步驟 4)，如此一來，PSM 便可將該 Node 加入服務成員中 (圖 6之步驟 6)，最後，PSM 將會啟動所有的 Node，讓服務開始運行 (圖 6之步驟 7、8)。

PSMP 的錯誤偵測與回復機制是建立在 PSM 與 PHM 永遠不會失效的假設下，如圖 7所示，在 ACTIVE 狀態下的 Node 必須每隔一小段時間就送出"psmp:heartbeat" 訊息給 PSM(圖 7之步驟 1、2)，讓 PSM 知道自己正常運作中，若 PSM 一段時間沒有收到某個 Node 的 heartbeat 訊息，PSM 會將"psmp:suspect" 訊息送給與該 Node 同一台機器的 PHM(圖 7之步驟 3)，一般來說，Node 沒有如期送出 heartbeat 訊息，可能是 Node 停止運作或者是 Node 速度太慢，不管是哪一種情況，PHM 都會將該 Node 關閉 (圖 7之步驟 4)，並代替該 Node 向群播位址發出"ssdp:bye-bye" 訊息 (圖 7之步驟 5)。PSM 收到 Node 的 Leave announcement 後，進入錯誤回復階段，在此階段，PSM 會先試著檢查是否有符合資格且在 DORMANT 狀態的 Node，如果有，PSM 會啟動該 Node 來回復這個服務；如果沒有，PSM 會發出"ssdp:discover" 訊息，重新開始服務組成及啟動程序。

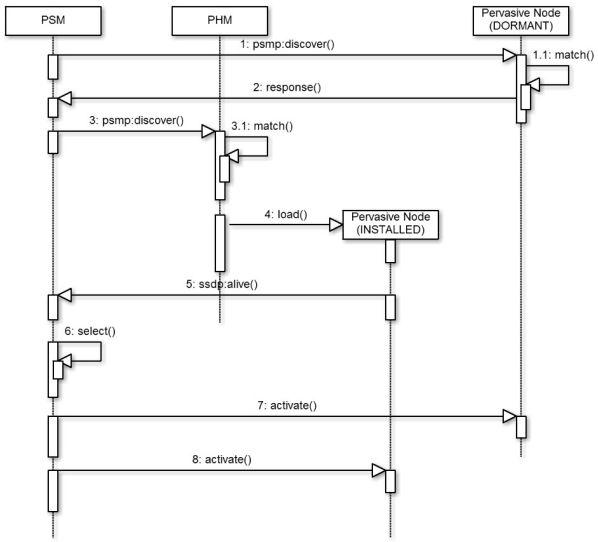

圖 6: PSMP 服務組成與啟動流程

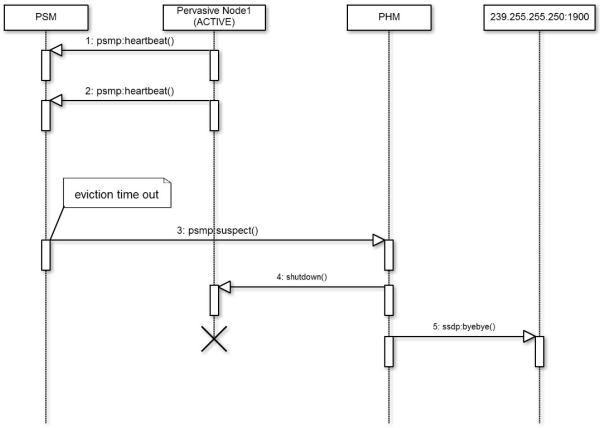

圖 7: PSMP 錯誤偵測與回復流程

3.2 RRCP

RRCP 是點對點式的錯誤偵測與錯誤回復機制，當 RRCP 啟動時，Node 間必須透過 Leader Eleciton 選出 Leader，以進行錯誤偵測流程，Leader 會發送 Check message(CHK 訊息) 給其他 Node(圖 8之步驟 1、3、5)，當 Node 收到 CHK 訊息時，必須在一定時間內回傳確認訊息 (Acknowledgement, ACK) 給 Leader，如果 Leader 等待一段時間後沒有收到回應，則 Leader 會標記該 Node 為懷疑 (Suspect)，每個 Node 皆會擁有一份懷疑清單 (Suspect list)，而這份懷疑清單會在 Leader 交接時一併更新，當 Leader 完成檢查程序後，必須確認懷疑名單上是否有 Node 被標記超過 n/2 次，n 為系統中正常運作

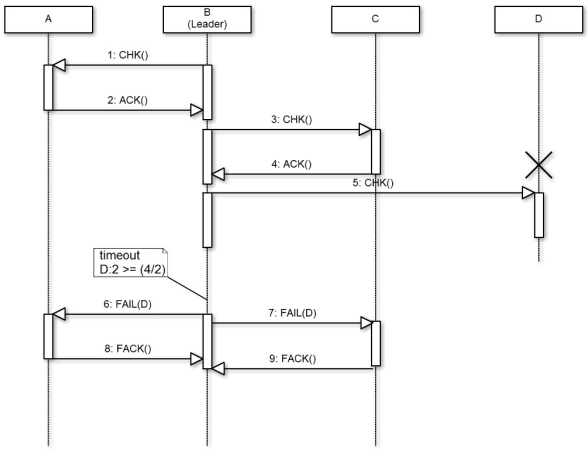

圖 8: RRCP 錯誤偵測與回復流程

的 Node 總數，如果有，Leader 會宣布此 Node 失效，並將此失效資訊透過 FAIL(Failure) 訊息通知系統中其他的 Node(圖 8之步驟 6、7)，收到 FAIL 訊息的 Node 必須回傳 FACK(Failure Acknowledgement) 訊息給 Leader。Leader 完成檢查程序後，會隨機選擇一個 Node 做為下一輪的 Leader，並繼續進行錯誤偵測流程。偵測到 Node 失效的 Leader 必須負責回復該 Node 的運作，當 PSM 從失效中回復後，必須重新檢查自己原本管理的服務；而 PHM 從失效回復後，須重新檢查在同一台機器正在運作的 Node。

四、複合式服務管理系統架構

PSMP 完整的定義了一個智慧家庭服務管理機制，然而，PSMP 最大的問題在於 PSMP 假設 Manager Node(PSM 和 PHM) 永遠不會失效，這在真實環境中不可能發生；而 RRCP 雖透過 Gossip 機制，解決了單點失效的問題，但 RRCP 僅針對 Manager Node 的錯誤偵測，並沒有敘述如何整合 Manager Node 和 Worker Node 之間的偵錯、回復機制，也沒有對於 Node 的回復細節加以說明。例如，若有一個 PHM 失效，位在不同機器的 Leader 要如何將該 PHM 從錯誤中回復，又或者，在 PHM 失效的狀況下，系統是否無法回復。最後，PSMP 及 RRCP 皆未考量到在真實環境中，MOM 也有可能會失效。

本論文為了解決上述問題，以 PerSAM 為基礎設計出 HSM(Hybrid Service Management) 架構，在 Manager Node 的部分，使用點對點架構的錯誤偵測與回復；在 Worker Node 則是用集中式架構，透過 Manager Node 進行管理。此外，HSM 架構亦考量到 MOM 失效的問題。以下分成

```
psm1: 192.168.4.102:/home/hsm/psm1.jar
psm2: 192.168.4.103:/home/hsm/psm2.jar
```

圖 9: psm-config 範例

三個小節詳細說明本論文之架構。

4.1 系統初始化

本節主要目的，是說明在服務管理系統剛啟動，以及因為嚴重錯誤導致系統必須重新啟動時，使用者該如何操作或設定系統。

在 HSM 中，每一台機器 (device) 都有一個 PHM，PHM 會在所處 device 的電源開啟後自動被載入，如果無法被載入，則此 device 無法加入網路，視為失效 (failed)，此處的失效為硬體失效，使用者可手動重啟失效之 device，若重啟後 PHM 仍無法被載入，則需聯絡 device 之製造廠商協助排除問題。

系統中有一個特別的 Node 叫做 Bootstrapping Node，以下簡稱 BN。使用者透過 BN 將管理服務的 PSM 自由分配安裝在任一台擁有 PHM 的 device。在 BN 中，有個目錄是/init/，目錄底下的 psm-config.ini 會在使用者分配 PSM 時，記錄所有 PSM 所在 device 的 IP 及目錄路徑，psm-config.ini 的範例如圖 9。使用者可以透過 BN 啟動服務管理系統，在啟動時，BN 會發送訊息要求其他 device 上的 PHM 開啟這些服務的 PSM，在這同時，BN 也會將此設定檔的內容傳送給所有的 PHM，PHM 在開啟 device 上的 PSM 時也會將此設定檔傳送給這些 PSM，以便後續的錯誤偵測流程進行。此外，當系統需要重新啟動時，BN 會主動發出訊息，提醒使用者手動重啟系統。

4.2 錯誤偵測流程

如圖 10，HSM 的錯誤偵測分兩層進行。首先，在 Pervasive System 的部分，最初，使用 Leader Election 選擇一個 PSM 當 Leader，Leader 會發送 CHK 封包給網路中的 Manager Node，封包中必須包含當 Leader 的 PSM 所管理的 Worker Node 資訊，此舉是為了在回復 PSM 時，讓 PSM 繼續管理原本的 Worker Node。收到 CHK 封包的 Manager Node 必須在一定時間內回傳 ACK 封包給 Leader，如果是 PHM 收到 CHK 封包中附上自己管理的 Worker Node 存在與否資訊 (Available information)，這裡指的存在與否指的是 Node 是否正常運作；如果是 PSM 收到 CHK 封包，則封包內不需要額外附上其他資訊。若 Leader 在一定時間內沒有收到該 Node 發出的 ACK 封包，

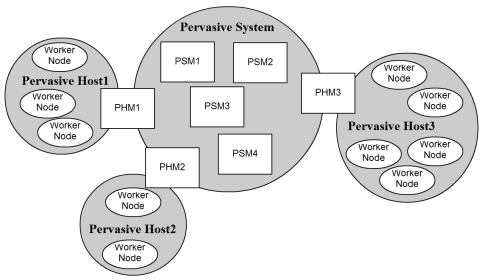

圖 10: HSM 錯誤偵測與回復架構圖

則 Leader 會將此 Node 標記為懷疑 (Suspect)，並將此 Node 放入懷疑名單 (Suspect List) 中。當 Leader 對所有 Manager Node 都發送過一次 CHK 封包後，Leader 須將 Leader 的權限以及 Suspect List 透過 LDR 封包交接給下一個 PSM，新的 Leader 依照上述的流程繼續進行錯誤偵測，如圖 11。

Leader 沒有收到 ACK 封包並不代表 node 失效，有時候是因為網路延遲或者 device 速度較慢，因此，為了提升可靠度，假設系統中有 n 個運作中的 PSM，當上述過程中 Leader 發現 suspect list 內有 Node 被標記次數在 n/2 以上，Leader 才會宣告此 Node 失效，並將此 Node 移出 suspect list，而後，Leader 替此 Node 發送”ssdp:bye-bye”訊息至群播位址，接著啟動錯誤回復機制。此外，當 Leader 與未回傳 ACK 封包的 Node 在同一台 device 上時，由於不會因為網路延遲而沒有回應，此時，Leader 可以直接宣告該 Node 失效，而不用將它放到 suspect list 中。

在 Worker Node(Pervasive Host) 的錯誤偵測部分，Worker Node 必須定期發送 heartbeat 訊息給 PHM，當 PHM 一段時間沒有收到某個 Worker Node 的 heartbeat 訊息時，代表此 Worker Node 失效，PHM 會在回復 Leader 的 ACK 封包中附上 Worker Node 失效資訊。此舉和 Worker Node 向 PSM 發送 heartbeat 不同的地方在於，Worker Node 和 PHM 在同一台機器上，因此不會耗費網路頻寬，亦不需要太多的傳輸時間。

4.3 錯誤回復流程

由於 HSM 架構有三種不同的成員，此外，本論文亦考量到 MOM 之失效問題，以下將分四小節討論不同情況下的錯誤發生與回復。

4.3.1 Failure recovery of worker node

當 Leader 收到 PHM 傳來的 Available information 時，若有 unavailable worker node，則

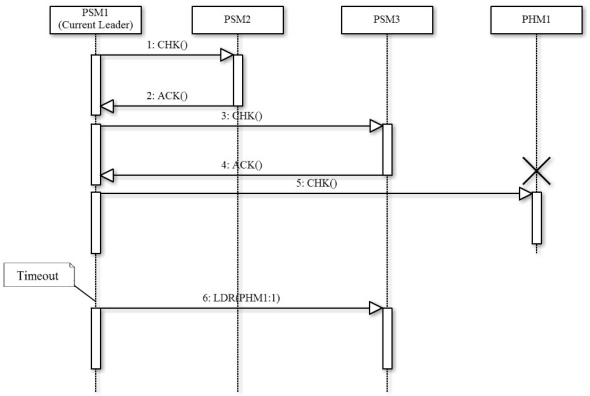

圖 11: HSM 錯誤偵測機制

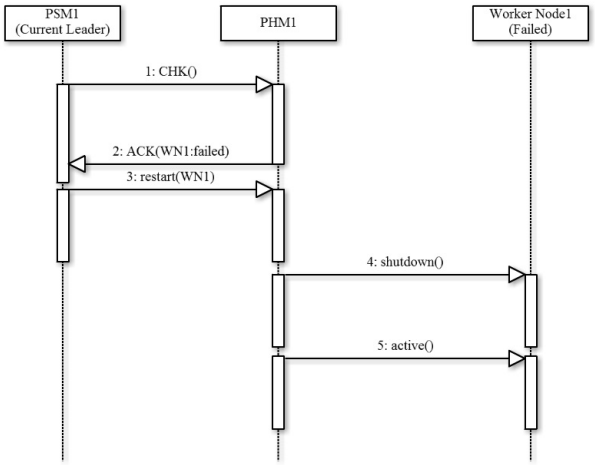

圖 12: HSM 架構下 Worker Node 錯誤回復機制

Leader 會比較這些 Node 是否在自己管理的服務成員中，如果是，Leader 會先要求 PHM 重新啟動這些 Worker Node 以回復服務的運作，若無法重新啟動失效的 Node，則 Leader 會重新進行服務組成的程序。如圖 12，Leader(PSM1) 從 PHM1 給的資訊得知 Worker Node1 unavailable(圖 12之步驟 2)，由於 PSM1 中需要 Worker Node1 才能組成完整的服務，因此 Leader 要求 PHM1 重新啟動 Worker Node1(圖 12之步驟 3)，PHM1 會先將 Worker Node1 關閉後再重新啟動它 (圖 12之步驟 4、5)。

4.3.2 Failure recovery of PHM

雖然 PHM 失效時不會影響 Worker Node 的功能，但若沒有 PHM 回傳 available information，有可能會發生 PSM 管理的服務中斷卻沒辦法回

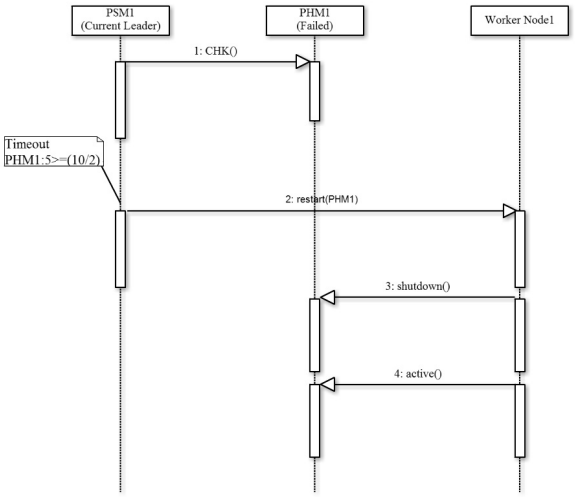

圖 13: HSM 架構下 PHM 錯誤回復機制

復。當 Leader 宣告 PHM 失效時，Leader 會要求與該 PHM 位於同一台機器上的 Worker Node 重新啟動此 PHM。如圖 13，在一包含 10 個 PSM 的智慧家庭系統中，由於 PHM1 已經被標記了 9 次，目前的 Leader(PSM1) 標記第 10 次後，宣告 PHM1 失效，Leader 通知與 PHM1 位在同一台機器上的 Worker Node1，要求 Worker Node1 重新啟動 PHM1(圖 13之步驟 3)，由於 Worker Node1 與 PHM1 在同一台機器上，因此，Worker Node1 可以將 PHM1 關閉後 (圖 13之步驟 4)，重新啟動 PHM1(圖 13之步驟 5)。

當任一台機器中，所有 PHM 與 Worker Node 皆失效，則需要使用者手動重新啟動此機器，在此狀況下，PHM 被加入 Suspect List 的次數會異常的高，此時，Leader 會通知 BN，讓 BN 通知使用者。

4.3.3 Failure recovery of PSM

PSM 的失效代表無人管理該服務，一旦此服務的 Worker Node 失效而導致服務中斷，則系統及使用者皆不會發現。當 PSM 被視為失效時，Leader 必須根據設定檔內容找到此 PSM 位在哪一台機器上，並要求此機器的 PHM 關閉失效的 PSM，接著重新啟動它，如圖 14，Leader(PSM1) 由 suspect list 的資訊將 PSM2 視為失效，Leader 根據設定檔找到 PSM2 與 PHM1 在同一台機器上，因此要求 PHM1 回復 PSM2(圖 14之步驟 2、3、4)。此外，若所有的 PSM 都失效，表示沒有人當 Leader，在此狀況下會導致 PHM 長時間沒有收到 CHK 訊息，此時，PHM 會自動重啟位在同

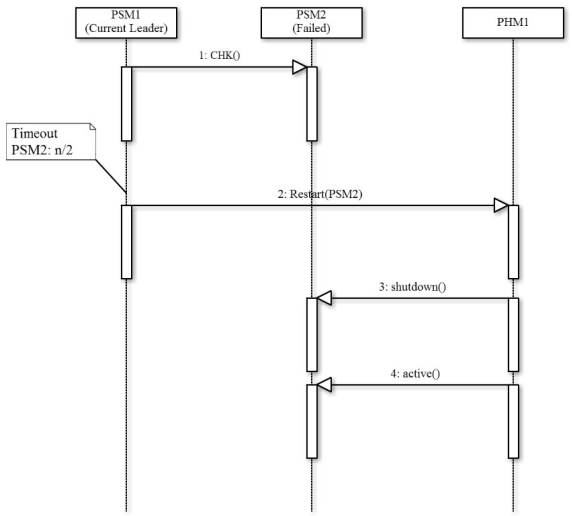

圖 14: HSM 架構下 PHM 錯誤回復機制

一台機器上的所有 PSM，使得錯誤偵測與回復流程得以繼續進行。

4.3.4 Failure recovery of MOM

實務上，當 MOM 失效時，所有有訂閱 MOM 的 Node 皆會收到例外通知，收到例外通知的 PSM 會主動通知 Leader，Leader 會要求與 MOM 在同一台機器的 PHM 重新啟動 MOM。

五、實驗與討論

為了驗證系統的強健性，我們進行了一連串的實驗來比較 PSMP 和 HSM 的錯誤回復能力，實驗程式的開發語言使用 Java，開發環境為 Eclipse，MOM 使用以 MQTT 為標準的 Mosquitto [7]，並以同一區域網路的 3 台 Intel Core i5 的個人電腦做為實驗環境，相同語言與相同開發環境讓實驗能夠更公正的顯示兩系統比較的結果。

我們佈署了 3 個 PHM、10 個 PSM 以及 50 個 Worker Node，並以 0%~100% 的錯誤率，隨機使所有 Node 失效，兩個系統皆以 10% 錯誤率為分界，每個錯誤率進行 100 次實驗。其中，當一個 Node 失效時，有 10% 和 40% 的機率發生硬體錯誤，也就是說，無法透過 PSMP 和 HSM 等軟體回復方法來進行修復，其中，由於通常在系統剛開始啟動以及將要關閉系統時，硬體發生錯誤的機率會比系統穩定運作時高，因此，我們將分別針對上述狀況做討論，在此實驗中，我們假設系統剛開始啟動以及將要關閉系統時，若一個 Node 失效，會有 40% 的機率發生硬體錯誤；而系統穩定運作時，

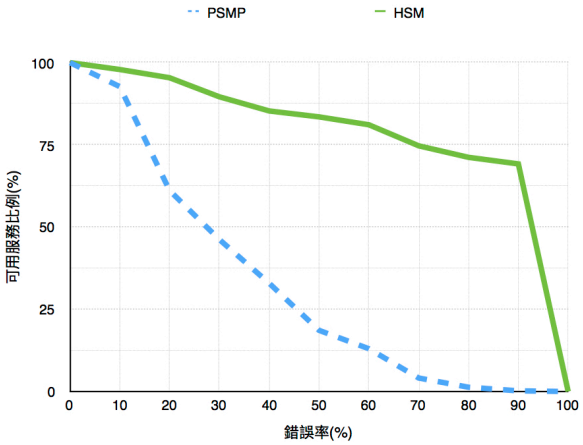

圖 15: 10% 硬體錯誤率下，在不同錯誤率下服務回復的比例

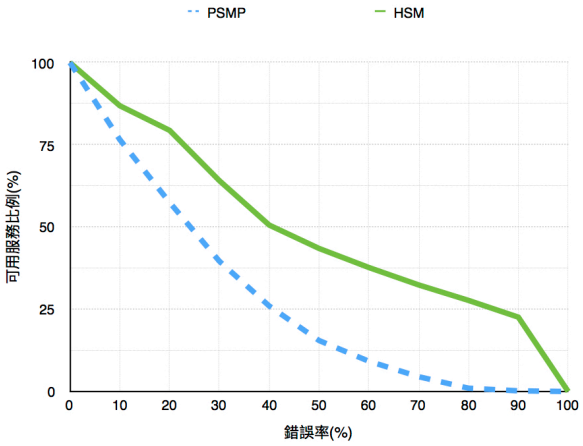

圖 16: 40% 硬體錯誤率下，在不同錯誤率下服務回復的比例

有 10% 的機率在 Node 失效時發生硬體錯誤。

實驗結果如圖 15和圖 16，藍色虛線為 PSMP，綠色實線為 HSM。首先，圖 15為在 10% 硬體錯誤率的實驗結果，可以發現就算在 90% 的錯誤率下，HSM 仍有約 70% 的服務正常運作，PSMP 則是在 30% 的錯誤率開始，系統中正常運作的服務比例就低於一半，原因是 PSMP 並沒有針對 Manager Node 的錯誤做處理，一旦 Manager Node 發生錯誤，若再有 Worker Node 失效，則失效的 Worker Node 便無法回復運作；反之，在 HSM 架構下，只有在有任一台機器中所有 PHM 及 Worker Node 皆失效時，會有服務無法被回復。圖 16則是在 40% 硬體錯誤率的實驗結果，在此條件下，最主要影響 HSM 錯誤回復能力的是 MOM 的硬體錯誤，在 PerSAM 架構下，一旦 MOM 失

效，系統中所有服務皆會無法運作，從圖 16中可看出，由於此實驗硬體錯誤率為前一個實驗的四倍，導致 PSMP 和 HSM 皆受到影響而降低服務回復比例。不過在系統穩定運作的狀況下，如圖15的結果，我們認為這樣的服務回復比例是夠強健的。

六、結論

本論文結合了集中式與點對點的特色，設計出了複合式架構，並以此架構建構智慧家庭服務管理系統，在本論文中，考慮了各種失效情況，並定義了在這些失效情況下，系統將會如何處理。實驗結果顯示，在系統穩定運作的情況下，就算錯誤率高達 90%，我們的系統仍有約 70% 的服務正常運作。然而，本論文仍有研究限制，首先，本論文假設網路永遠不會中斷；第二，本論文聚焦在軟體的 fail-stop 狀況，也就是當元件失效時，該元件會停止運作並停止發送、處理訊息，因此，本論文將不處理如硬體損壞或者拜占庭錯誤 (Byzantine failures) 等狀況；最後，由於重新啟動 Node 必須仰賴 PHM 及 Worker Node，因此，若有一台機器的所有 PHM 及 Worker Node 皆失效，則該機器必須重啟。然而，上述狀況發生機率不高，幾乎可以忽略不考慮。在未來，我們將針對這些限制設計新的機制加以改善。

參考文獻

[1] H. Chen, T. Finin, and A. Joshi, "Semantic web in the context broker architecture," DTIC Document, 2005.

[2] A. K. Dey, "Providing architectural support for building context-aware applications," Georgia Institute of Technology, 2000.

[3] T. Gu, H. K. Pung, and D. Q. Zhang, "A service-oriented middleware for building context-aware services," *Journal of Network and computer applications*, vol. 28, pp. 1-18, 2005.

[4] Y. W. Jong, C. F. Liao, L. C. Fu, and C. Y. Wang, "A Rotating Roll-Call-Based Adaptive Failure Detection and Recovery Protocol for Smart Home Environments," in *Ambient Assistive Health and Wellness Management in the Heart of the City*, 2009, pp. 201-208.

[5] K. Kapitanova, E. Hoque, J. A. Stankovic, K. Whitehouse, and S. H. Son, "Being SMART about failures: assessing repairs in SMART homes," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, 2012, pp. 51-60.

[6] C. F. Liao, Y. W. Jong, and L. C. Fu, "Toward Reliable Service Management in Message-Oriented Pervasive Systems," in *IEEE Transactions on Services Computing*, vol. 4, no. 3, pp. 183-195, July-Sept. 2011.

[7] Mosquitto, http://mosquitto.org/

[8] M. Román, C. Hess, R. Cerqueira, A. Ranganathan, R. H. Campbell, and K. Nahrstedt, "A middleware infrastructure for active spaces," *IEEE pervasive computing*, vol. 1, no. 4,pp. 74-83, 2002.

[9] S. Silas, K. Ezra, and E. B. Rajsingh, "A novel fault tolerant service selection framework for pervasive computing," *Human-centric Computing and Information Sciences*, vol. 2, pp. 1-14, 2012.

[10] *UPnP Device Architecture 2.0*, UPnP Forum, 2014.