國 立 交 通 大 學

資訊科學與工程研究所

碩 士 論 文

適用於家庭自動化的

通用隨插即用感測與促動器基礎架構

UPnP Compatible Sensor/Actuator Infrastructure for

Home Automation

研 究 生：劉育志

指導教授：邵家健 博士

中 華 民 國 九 十 六 年 七 月

適用於家庭自動化的通用隨插即用感測與促動器基礎架構

UPnP Compatible Sensor/Actuator Infrastructure for Home Automation

研 究 生：劉育志　　Student：Yu-Chih Liu

指導教授：邵家健　　Advisor：John Kar-Kin Zao

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute of Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

July 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年七月

適用於家庭自動化的通用隨插即用感測與促動器基礎架構

學生：劉育志　　指導教授：邵家健

國立交通大學資訊科學與工程研究所　碩士班

摘要

NCTU Pervasive Embedded System (PES)實驗室的重點計劃「老年人居家照護」旨在透過逐步增添智能家電來促使居住環境進化，從而協助老年人在自己常居的環境中安享他們的老年生活。實現這類「智能環境」的第一步在於建構一套能隨時容許感測器與促動器組合互動的服務平台。為此，我在論文研究中設計了一個具有下列三個特性的感測與促動器基礎架構：第一、這個基礎架構要具有機能組合性，有能自動發現裝置的機能，以及遠端控制使這些裝置能組合互動的機能；第二、提供操作上的適應性，開發高層的回饋控制系統，對於裝置偶而的增減或系統的改變做出適當的反應。第三、解決結構上的異質性並提供標準模組。

我們提出兩種方法來支持這個基礎架構：一、遍佈式感測器與促動器 UPnP 代理伺服器；二、階層式的軟體方法來開發 UPnP 抽象化裝置與服務界面。藉由 UPnP 代理伺服器讓所有的感測器與促動器在 UPnP 網路上互動；使用階層式 UPnP 裝置與服務界面的軟體開發方法，將原本裝置的功能產生標準界面，再將標準界面組合成上層的虛擬裝置，使用感測器偵測環境的改變，來控制促動器做調整來完成高層回饋控制系統。

最後我們將這個基礎架構實作在交通大學電資大樓智能環境實驗室中，用來做室內燈光回饋控制，也有很好的控制效果。其中產生的標準模組：UPnP 調光器與 UPnP 光感測器，也已經被其他的研究所使用，將發展更多的家庭自動化回饋控制系統。

UPnP Compatible Sensor/Actuator Infrastructure for Home Automation

Student: Yu-Chin Liu Advisor: John Kar-Kin Zao
Institute of Computer Science and Engineering
National Chiao Tung University

ABSTRACT

In fall 2005, an elder-care project named Kannon was launched in NCTU Pervasive Embedded System (PES) Lab. This project aims at enabling elders to live comfortably in their familiar environments by gradually transforming those environments through acquisition of smart appliances. To achieve this goal, my colleagues and I developed an infrastructure for sensors and actuators to work together. Such an infrastructure is expected to have the following three essential properties: (1) the infrastructure should enable on-line devices to discover one another and use their services; (2) the infrastructure should be able to adapt to occasional loss of devices and changes of system configuration; (3) the infrastructure should support modular application programming that enables interoperability among heterogeneous devices and incremental deployment of target systems.

In this thesis, I described a two-step approach to establish this infrastructure: In the first step, my colleagues and I developed an UPnP proxy for wireless sensors and actuators; in the second step, I devised a hierarchical programming model for UPnP devices and services. The UPnP proxy converts different communication/programming interfaces of various sensors and actuators into standard UPnP ones, and thus foster interoperability among those devices. The hierarchical programming model, on the other hand, allows incremental changes of program modules without affecting the structural and functional integrity of the system.

To demonstrate the usefulness of my technology, I built a simple feedback system to control the indoor luminance level of NCTU MIRC Smart Environment Lab. That demonstration system worked well! Furthermore, several of its modules including the UPnP-compatible light sensors and dimmers are useful in other automatic lighting control systems.

誌　　謝

首先要感謝的是邵家健老師，整個論文的題目、方向、方法都是和老師不斷的討論才決定的，也從老師那裡學到解決研究問題的方法，論文寫作與口頭報告的技巧，有老師的指引與建議，這篇論文才能完成，也讓我的研究所時期收穫良多。在研究中，感謝楊明曉學長、李勝焜學弟、陳璟蔚學弟、陳威佑學弟完成了最初 UPnP 代理伺服器的雛型。感謝黃國晉學長、張哲維學長在我研究產生迷惘時給予適時的建議。最後感謝其他所有 620 實驗室的成員：朱書玄、黃淩軒、李明龍、林哲民、郭芳伯、曾輔國，你們是我最好的伙伴，也謝謝家人默默的支持與照顧，我才得以完成碩士學位。

目錄

圖目錄

表格目錄

第一章　綜述

1.1　問題陳述

NCTU Pervasive Embedded System (PES)實驗室的重點計劃之一就是「老年人居家照護」，希望老年人能在自己熟悉的環境中生活，而不是搬到一個雖然完善但是陌生的環境，透過逐步的購買適合自己的智能家電來讓環境進化。而感測器與促動器將是家庭自動化環境首要加入的裝置，衍生的這個研究課題在於發展適用於家庭自動化的感測器與促動器互動基礎架構，這樣的基礎架構主要是建構在通用隨插即用(Universal Plug and Play™, UPnP)的技術上，並要求這樣的架構能滿足以下的三個特性：

1. 機能的組合性(Functional Composeablity)—從傳統家庭轉變到自動化家庭是一種漸進的過程，不像是公司行號，消費者並不會一次購買整個系統，而會個別購買所需要的智能家電，所以這個基礎架構要有能力去加入各種不同的新裝置，並讓新裝置找到其他裝置的機能，使用他們的服務。如果可能的話，可以讓這些裝置的機能組合來完成使用者期望的任務。簡言之，這個基礎架構要有自動發現裝置的機能，以及遠端控制使這些裝置能組合互動的機能。
2. 操作上的適應性(Operational Adaptability) —不只是裝置和其上的服務會從智能環境中加入或移除，也得對環境上的改變馬上做出回應。為了達到這個需求，這個基礎架構要提供一個高層回饋控制系統，對於裝置偶而的增減或系統的改變做出適當的反應。
3. 結構上的異質性與模組化(Structural Heterogeneity and Modularity)—很顯然地，這個基礎架構必須能容納各式各樣的裝置，範圍從簡單的感測器與促動器到複雜的子系統。除此之外，整個智能環境要使用模組化的方式來佈置，如此，高層的設計就可以重覆使用多元裝置的模組，這樣模組化的設計也會讓系統維護更有效率，當有部份的裝置動態增減時，也不會影響整個系統。

1.2 研究方法

綜合的分析以上的問題，為了滿足機能的組合性，所以我們選用 UPnP 來當我們裝置間溝通的協定，這是由 UPnP 論壇所訂定的網路通訊協定，目的是要讓家中和公司的網路環境內所有的裝置都能夠輕易且緊密地連結在一起。

不過事實上有許多的裝置並沒有通用於 UPnP，像是感測器與促動器，可能是製造商沒有提供軟體支援，或者硬體上不能負荷，我們使用 UPnP 代理伺服器來將之轉換成 UPnP 裝置，當感測器與促動器加入或移除時，在 UPnP 網路上，對應的裝置也要跟著加入或移除來解決結構上的異質性。另外，我們使用 UPnP 論壇制定的標準界面，對於論壇並沒有定義標準界面，像是感測器的部份，由我們提供自己設計的標準界面，來為這些感測器及促動器產生標準模組。

此外我們提供軟體開發方法，先定義了感測器與促動器互動基礎架構的功能模塊，再創造了階層式 UPnP 裝置與服務界面，將原本裝置的功能產生標準界面，再將標準界面組合成上層的虛擬裝置，使用感測器偵測環境的改變，來控制促動器做調整來滿足操作上的適應性。這樣高層的設計就可以重覆使用多元裝置的模組，模組化的設計也會讓系統維護更有效率，當有部份的裝置動態增減時，也不會影響整個系統。

1.3 論文大綱

在本論文之後的部份，將在第二章介紹一些背景知識，還有相關的研究。在第三章中主要是本研究的精華技術，定義了感測器與促動器互動基礎架構的功能模塊，並詳述階層式UPnP裝置與服務界面，以及UPnP代理伺服器的設計原理。第四章以感測器與促動器互動基礎架構的操作程序來讓讀者更瞭解這個系統。第五章中透過室內光控實驗，來證明理論的可行性。第六章總結本研究的成果並提出未來仍可研究的課題。

本研究的內容已整理成一篇論文，題目是Ubiquitous e-Helpers: An UPnP-based Home Automation Platform，已經被IEEE SMC 2007 Conference[1]所採用，本碩士論文中的部份內容與圖表會取自那篇論文。

第二章　背景知識

2.1　感測器與促動器

2.1.1 週邊感測器與促動器裝置

感測器是一種反應物理刺激的裝置，像是熱、光、聲音、壓力、移動、流動等等的，然後產生出對應的電子訊號。而促動器是一種裝置將電子控制訊號轉換成實體的動作，促動器可能用來做流量控制閥、幫浦、定位驅動器、馬達、開關、繼電器或計量器。在本研究最後的室內燈光回饋控制中，使用的促動器為調光器，接收 RS-485 的控制訊號來改變亮度。這些感測器或促動器常常會做成周邊裝置方便人們的使用。

2.1.2 無線感測網路

相對於週邊感測器與促動器裝置，由於科技的發達，所以產生了由無線來佈置感促器與促動器的技術：無線感測網路。

無線感測網路的發展，最早是美國加州柏克萊大學(UC Berkeley)，David Culler 教授主持的一項研究計劃，稱之為「智慧灰塵(Smart Dust)」。這項計劃是由美國國防部研究計劃單位(DARPA)所支助，原先的構想是應用在軍事上，例如：戰場監控。

無線感測網路有別於一般的無線網路平台，其上的網路節點皆是由許多微小的微型監測器(Node)所組成，所使用的硬體架構，分為主要幾個部份：運算單元(Processing unit)、通訊界面 (Transceiver) 、感測器 (Sensor)、電力來源(Power Unit)。

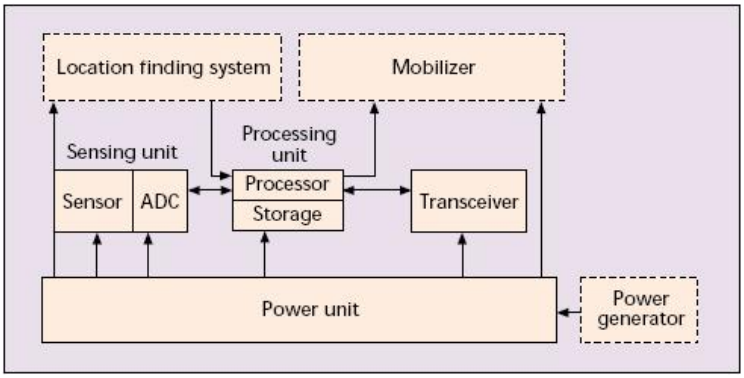

圖 1: 微型監測器(Node)基本組件

由於微型監測器的硬體資源相當有限，所以軟體作業系統上面，必須很有效率的利用有限資源去處理各種狀況事件，像是UC Berkeley所開發的TinyOS[2]，在其上運行程式來控制硬體，從感測器偵測環境的改變，處理收集到的數據，並將處理過後的資料以無線傳輸的方式送到資料收集中心或基地台(Base Station)。

近年來，也將促動器(Actuator)加在微型監測器上，將之佈署於家中，可以讓人們在遠方或在家裡經由網際網路做許多家事。

無線感測網路是未來重大技術之一，將大量運用於商業、軍事、醫藥、安全和生態學等領域，已有許多研究機構參與。本研究也將使用無線感測網路，將之應用到家庭自動化上，來改善人們的生活。

2.1.3 週邊感測器與促動器裝置和無線感測網路的比較

在這裡比較一下「週邊感測器與促動器裝置」和「無線感測網路」特性上的不同之處，使得我們的設計上會有不同：

1. 在安裝上，週邊感測器與促動器裝置通常都是有線的單一的裝置，可能會有前端控制器，佈置時前端控制器與感測器或促動器通常會放置於不同的地方，所以環境的資訊是不一定同步的，而且在佈置後不易改變。而無線感測網路則較具有動態性，感測器或促動器是安裝在微型監測器(Node)上的，所以環境的資訊是同步的。

2. 在主動與被動性上，無線感測網路因為基地台要收集整個網路的資訊，或發出控制命令，所以會知道整個網路的拓撲結構，這些資訊可以主動的提供給連接的伺服器；相對的，週邊感測器與促動器裝置就沒有固定的佈署方式，所以是屬於比較被動的。

由以上的分析，可以知道構成無線感測網路的微型監測器與感測器或促動器的環境資訊是同步的，但由於有較高的動態性，所以要提供使用者快速更新環境資訊的方式，在安裝後連接的伺服器也可以主動偵測微型監測器的加入或移除。在週邊感測器與促動器裝置與前端控制器的環境資訊是不一定同步的，而且在佈置後不易改變，所以可以讓使用者手動加入這些資訊，並告知連接伺服器新裝置的加入。

2.2 通用隨插即用技術 (Universal Plug and Play™ UPnP)

通用隨插即用技術(Universal Plug and Play, UPnP) [3] 是一種遍佈式的點對點網路架構，在其上的連結的裝置可以是任何廠商的智能家電、無線裝置或個人電腦。UPnP的設計理念在於讓人容易使用、有彈性的，並提供標準模組為基礎，以TCP/IP與Web技術所組成的分散式開放網路，來讓連到這個網路的裝置可以被控制並傳送資料。

UPnP論壇 [4] 是一個工業界的組織，目的在於為眾多不同的廠商的產品制定簡易而且建全的連接管道，企圖發展標準的UPnP相關協定與定義標準的裝置XML描述來讓裝置與裝置能有意義的溝通，同時也為順從這些標準的裝置做審核，來發放符合UPnP標準的商標。

在下面的章節，會介紹與「適用於家庭自動化的通用隨插即用感測與促動器基礎架構」相關的 UPnP 背景知識。

2.2.1 UPnP網路的基本組件

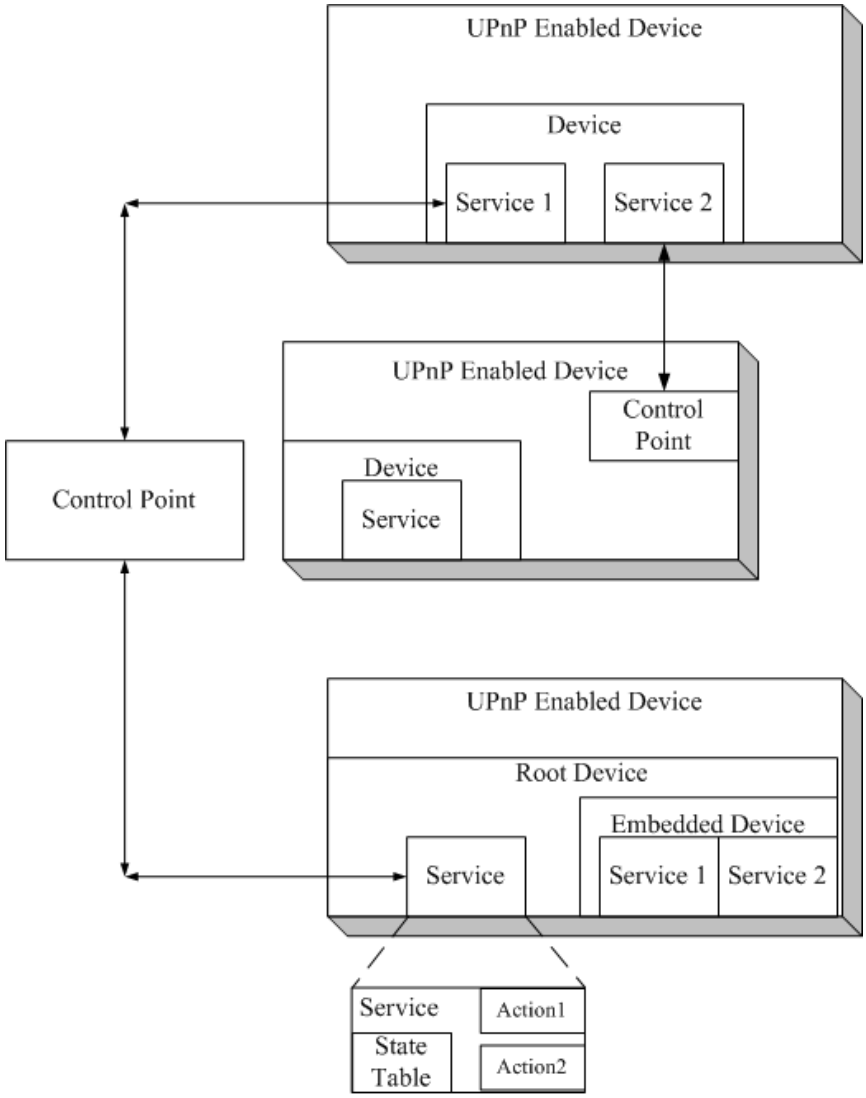

圖 2: UPnP 控制點(Control Point), 裝置(Device), 與服務(Service)

UPnP基本的組成有裝置(Device)，服務(Service)與控制點(Control Point)。圖 2是這三個基本組件之間的關係，有些彼此之間會是巢狀關係。

❖ 服務(Services)

服務是 UPnP 中最小的控制單位，服務提供功能(Action)還有一組狀態變數(State Variable)來記錄目前此服務的情況，其中功能(Action)很像程式中的函式(Function)，可以設計來完一個特定的任務並傳遞參數。

❖ 裝置(Devices)

裝置是包含裝置以及嵌入式裝置(Embedded Device)的設備。

❖ 控制點(Control Points)

控制點可以控制 UPnP 網路上找到的裝置，在找到裝置後，控制點主要做的事情有：

- 取得裝置描述文件(Device Description)，從中取得相關服務的簡易列表。
- 取得有興趣的服務描述文件(Service Description)。
- 傳送功能(Action)來控制服務。
- 向有興趣的服務做訂閱的動作，每當訂閱的服務的狀態變數(State Variable)改變時，會送回一個事件(Event)。

在「適用於家庭自動化的通用隨插即用感測與促動器基礎架構」中就是由低層、中層、高層 UPnP 裝置所組成的。其中高層 UPnP 裝置就包含了一個控制點用來控制中層 UPnP 裝置。

這裡的圖文部份取自Understanding Universal Plug and Play[5]。

2.2.2 UPnP物件模型

一個UPnP裝置會有如同圖 3、圖 4的物件模型，UPnP使用XML描述文件(Description)來描述一個實體裝置，分成兩個邏輯部份：

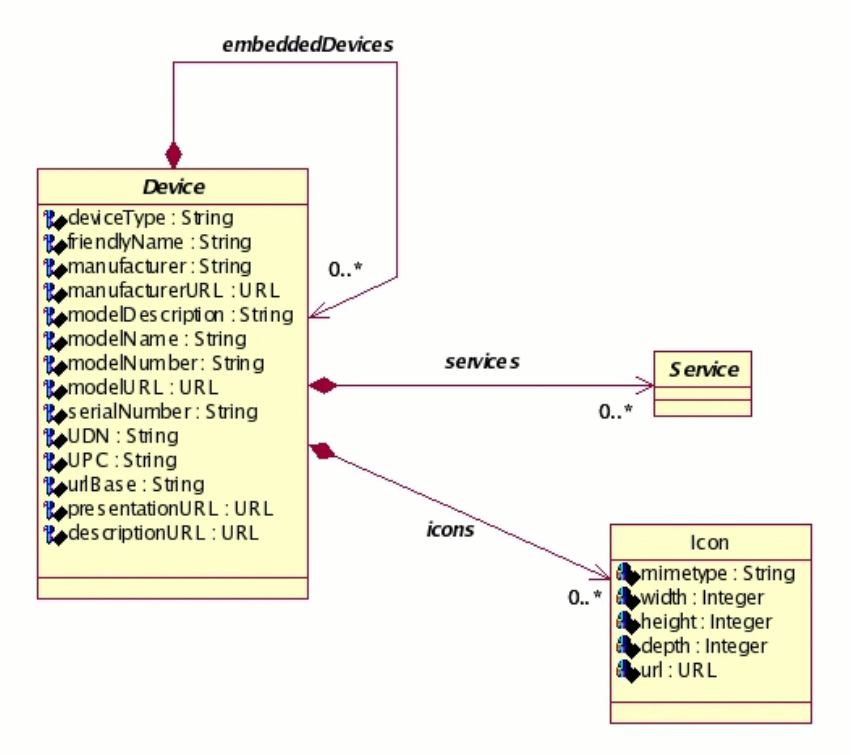

圖 3: UPnP 裝置(Device)物件模型

1. 裝置描述文件(Device Description)用來描述實體的裝置與其中邏輯的組成，可以參考圖 3，可以看出一個裝置(Device)可以包含多個服務(Service)，也可以擁有多個嵌入式裝置(Embedded Device)。

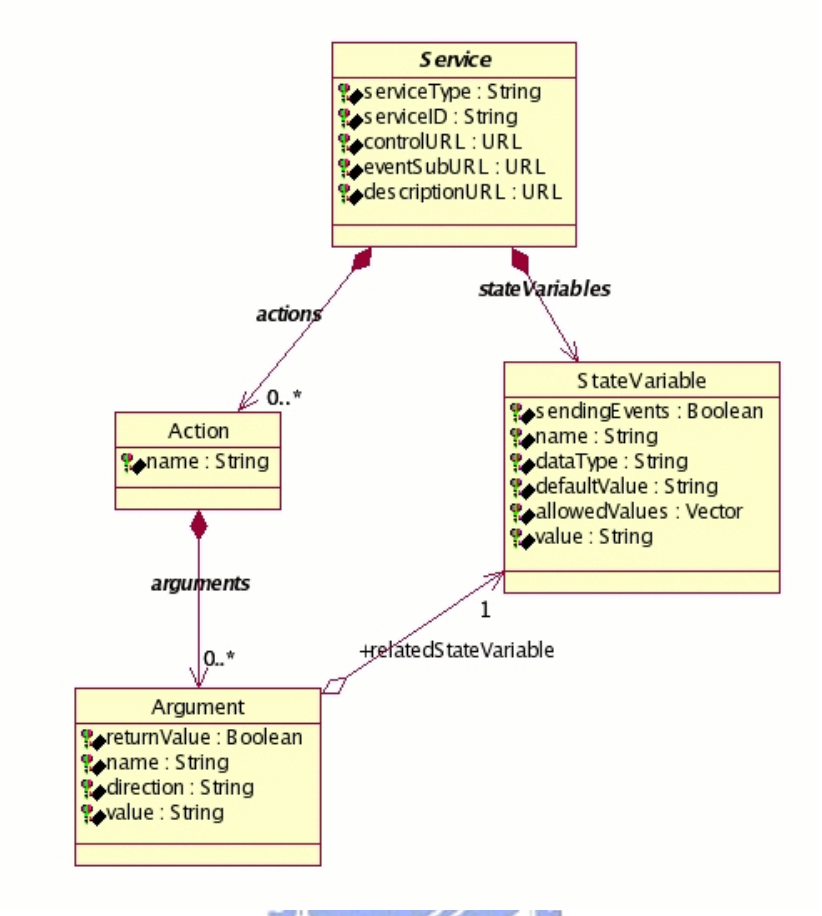

圖 4: UPnP 服務(Service)物件模型，包含功能(Actions)與狀態變數(State Variables)

2. 一個或多個的服務描述文件(Service Description)用來描述這個裝置所提供的功能，可以參考圖 4，可以看出服務(Service)是一群功能(Action)及狀態變數(State Variable)的集合。

在階層式UPnP裝置與服務界面理論中，把原本的實體裝置(Physical Device)產生出低層、中層、高層 UPnP裝置，將在3.1中說明每一層描述文件(Description)的設計與如此設計所帶來的好處。

這裡的圖文部份取自The Journal of Spontaneous Networking, It Just Works: UPnP in the Digital Home [6]。

2.2.3 Intel UPnP軟體開發套件

在本研究中所實作的UPnP裝置與控制點都是使用Intel UPnP SDK [7]：

- 使用 Intel Device Author 來設計服務描述文件(Service Description)。
- 使用 Intel Device Builder 來設計低層、中層、高層 UPnP 裝置與高層 UPnP 裝置中的控制點。
- 使用Device Spy來做為控制點，可以用來控制所有的裝置並陳列出UPnP裝置上所有詳細的資訊，在Device Spy找到裝置後，會產生樹狀圖，包含其上的嵌入式裝置(Embedded Device)與服務(Service)，每一個服務上包含功能(Action)與狀態變數(State Variable)，使用者可以直接的來操作這些功能。圖 5是Device Spy圖示的說明，在3.1 階層式UPnP裝置與服務界面中將使用Device Spy所陳列的詳細資訊來幫助說明。

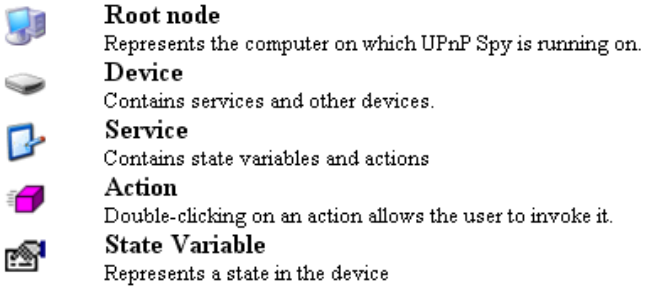

圖 5: Intel Device Spy

2.3 相關研究

2.3.1 UPnP代理伺服器

目前已經有不少適用於感測器的UPnP代理伺服器的相關研究，並將之應用在家庭自動化上，最值得注意的是BOSS[8]，BOSS做了不錯的研究，將感測器分成群組用來做各方面的管理，不過為此提出了一個新的感測器描述文件(Sensor

Description)，如此UPnP控制點就要特別設計過才會解釋感測器描述文件(Sensor Description)；而且感測器描述文件(Sensor Description)是以群組為單位，增減一個感測器時，並無法馬上讓UPnP網路知道。

2.3.2 燈光監測與控制

在我們的研究中，Suet-Fei Li的Wireless Sensor Actuator Network for Light Monitoring and Control Application [9]應該是第一個應用無線感測網路於光控的研究。然而，他大部份的貢獻在於將感測器的讀數和促動器的設定值顯示在個人數位助理(PDA)上。他的研究並沒有提出標準裝置的尋找和服務的管理機制，也沒有高層的系統來讓感測器與促動器直接互動。他的雛型架構並不容易擴展，沒有辦法與其他的系統溝通互動。

第三章　技術

在感測器與促動器互動基礎架構中會有如同[圖 6]的功能模塊，圖中的方框表示的是實體的硬體裝置，圓框表示是產生出來的UPnP 裝置(Device)或者控制點(Control Point)。

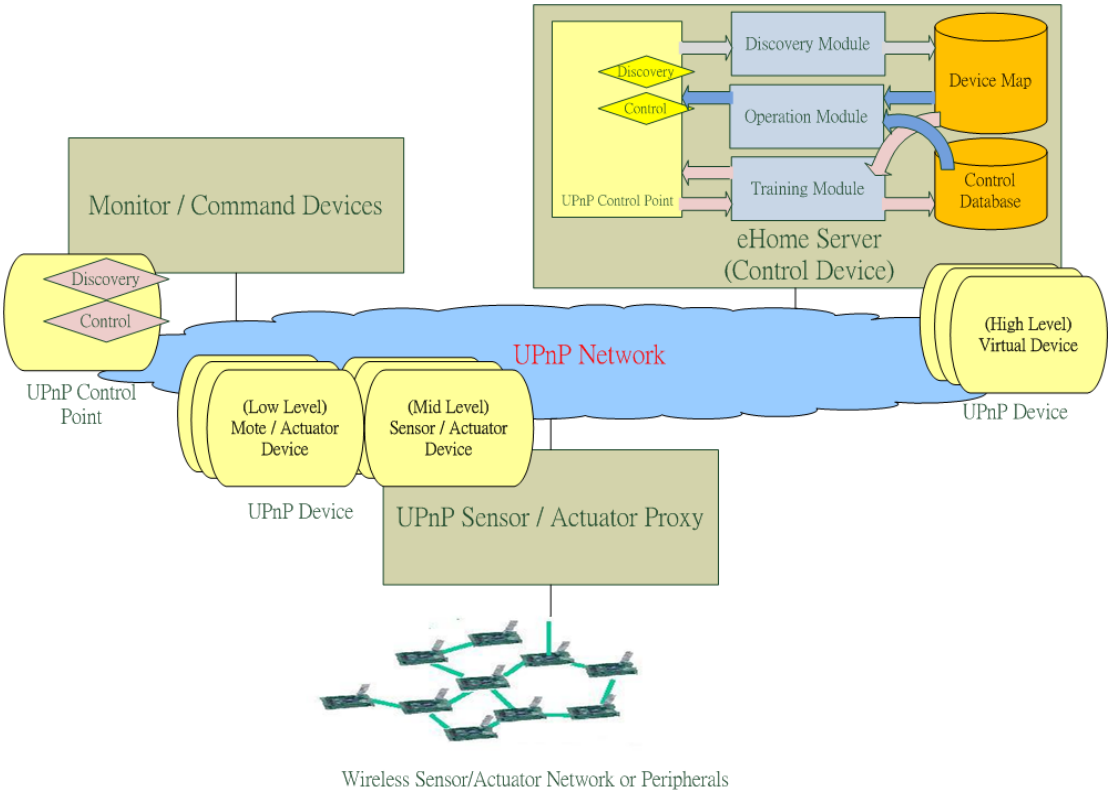

圖 6: 感測與促動器基礎架構功能模塊

在實體的硬體裝置中，基礎的元件就是無線感測網路(Wireless Sensor/Actuator Network)或週邊感測器與促動器裝置(Peripheral)，由於感測器與促動器並沒有足夠的硬體能力直接來運行 UPnP 協定，所以必須藉由 UPnP 代理伺服器將感測器與促動器轉換成低層與中層 UPnP 裝置，讓感測器與促動器有共通的平台來互動。

數位家庭伺服器(ehome Server)是感測器與促動器互動基礎架構的控制中心，依照系統應用的需求來設計高層回饋控制系統，使用 UPnP 感測器與促動器標準模組來完成控制，同時本身也提供 UPnP 的控制界面，我們稱之為高層 UPnP

裝置或虛擬裝置。

只要安裝高層 UPnP 裝置專用的控制點(Control Point)就可以成為監測與控制裝置(Monitor/Command Device)，可以是個人電腦或者個人數位助理，讓使用者可以在舒適的地方來設定高層回饋控制系統的參數，控制感測器與促動器互動基礎架構。

文中所提到的低層、中層、高層UPnP裝置就是階層式UPnP裝置與服務界面，將在3.1做詳細的介紹；UPnP代理伺服器的設計原理和通用結構也將在3.2中做介紹。

3.1 階層式UPnP裝置與服務界面

在未來的數位環境中，所使用的數位家電多半是由不同的廠商所製造的，其中感測器與促動器也是如此，不同的廠商的功能也會有所不同。在此為相同種類的感測器與促動器提供標準模組是很重要的，標準模組的功能是依靠廠商自定的功能所完成的；有了標準模組，上層設計就可以藉由呼叫標準模組來使用不同廠商開發的同種類裝置。

為了能使用到廠商自定的功能，一方面提供高層自動化互動控制所需的標準化模組，我們提出了感測器與促動器的階層式UPnP裝置與服務界面，將之分成三層[圖 7]：

- 低層界面反應廠商自訂裝置的架構與功能。
- 中層界面藉由低層界面提供標準模組的架構與功能。
- 高層界面則提供高層虛擬裝置，利用中層界面達到感測器與促動器回饋控制。

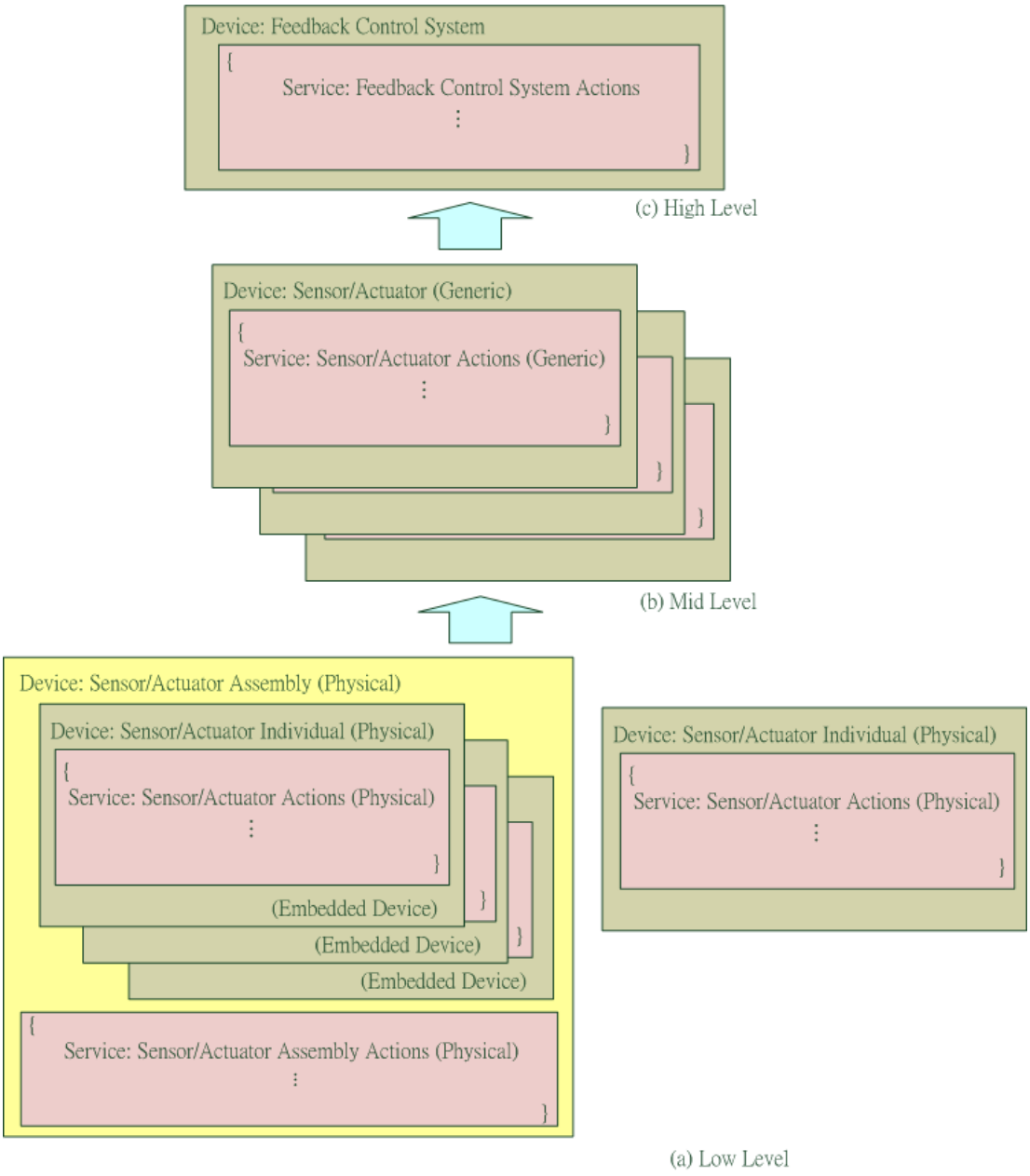

圖 7: 階層式 UPnP 裝置與服務界面

我們將在之後的部份明白的定義每一層的邏輯結構與功能，並使用底層的界面來完成這一層的功能，同時我們會使用Intel Device Spy來提供UPnP裝置的詳細資訊來幫助瞭解，文中所提到的UPnP術語請參考2.2。

3.1.1 低層：廠商自訂裝置與服務界面

低層裝置描述的主體是實體裝置，對象是構成無線感測網路的微型監測器(Node)，或者週邊感測器與促動器裝置(Peripheral)，微型監測器與擁有前端控制器所包含的週邊裝置，會構成感測器或促動器集合(Sensor/Actuator Assembly)，而單獨的週邊裝置，會構成個別感測器或促動器(Sensor/Actuator Individual)，可以參考圖 7a。在低層裝置描述文件(Device Description)中會忠實的呈現最接近實際硬體的架構與功能，加上我們設計的服務界面，提供與環境相關的資訊，例如放置的地點，以及管理的功能。每一個微型監測器(Node)或週邊感測器與促動器裝置(Peripheral)都會產生一個UPnP裝置，有以下的結構：

- ❖ 裝置描述文件(Device Description)反應裝置實體的架構，包含嵌入式裝置(Embedded Device)，以及至少兩個服務(Service)，每一個服務由個別的服務描述文件(Service Description)來描述：
  - Low Level Information Service：用來設定高層回饋控制系統中所需要的環境資訊，例如提供微型監測器的運作狀態(Active Status)或所放置的地點(Location)。這個服務對於無線感測網路的微型監測器尤其重要，在動態的放置這些微型監測器後，可以使用這個服務來設定微型監測器的放置地點(Location)，這些資訊也將同步到 Mid Level Information Service 中。
  - System Management Service：在微型監測器上用來幫助管理無線感測網路，在週邊感測器與促動器裝置上用來設定相關的控制參數。
  - 如果是個別感測器或促動器(Sensor/Actuator Individual)，就在在此提供廠商自訂的感測器或促動器功能服務。

❖ 在感測器或促動器集合(Sensor/Actuator Assembly)上的感測器、促動器都是屬於嵌入式裝置(Embedded Device)，所以要在上述裝置描述文件(Device Description)的裝置列表(Device List)中，要反應個別感測器與促動器的種類及其上的服務，提供廠商自訂的功能，也是由服務描述文件(Service Description)來描述：

- Vender-Specified Control/Sense Service：所提供的功能(Action)可能和中層的很相似，但最明顯的是不同廠牌的感測器就會有不同解析度的讀數；相同的，不同廠牌的促動器就會輸入不同解析度的控制值。

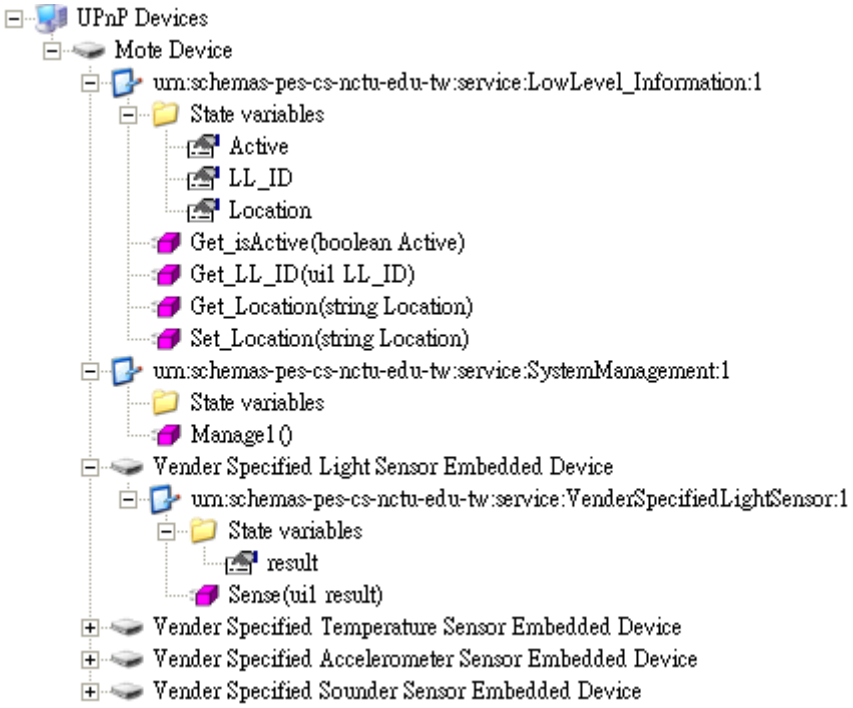

圖 8: 低層範例-UPnP Berkeley Mote

3.1.2 中層：通用裝置與服務界面

在這一層我們主要是為高層回饋控制系統界面提供感測器與促動器標準化模組，裝置描述文件(Device Description)所描述的主體為感測器或促動器 [圖7b]，所以我們直接脫去低層中感測器或促動器集合(Sensor/Actuator Assembly)

這層外殼。在促動器的部份，有些裝置描述文件(Device Description)可以使用UPnP 論壇所提供的標準描述文件(Standardized Description)；而感測器的部份，因為UPnP論壇並無標準，我們會為用到的感測器定義標準描述文件，都提供了二個服務：

- Mid Level Information Service：

  用來提供家高層回饋控制系統中所需要的環境資訊。由2.1.3週邊感測器與促動器裝置和無線感測網路的比較的結果，這裡的服務設計在無線感測網路上和週邊感測器與促動器裝置會有不同的實作方法：在無線感測網路的微型監測器上會將低層所設定的環境資訊(Active Status、Location)等同步到中層個別的感測器與促動器上；週邊感測器與促動器裝置則直接使用在UPnP代理伺服器的裝置驅動模組(Device Driver Module)安裝使所設定的環境資訊。

- Standardized Control/Sense Service：

  讓相同種類的感測器或促動器提供同樣的刻度。感測器的部份，在感測服務中使用感測功能(Sense Action)時，會先從低層感測功能得到感測值，透過轉換方程式(Transformation Function)轉換成標準的刻度，再將標準的刻度從中層感測功能送出。促動器的部份，在控制服務的控制功能(Control Action)中送入的控制值，會透過轉換方程式轉換成低層的刻度再送入低層控制功能中。

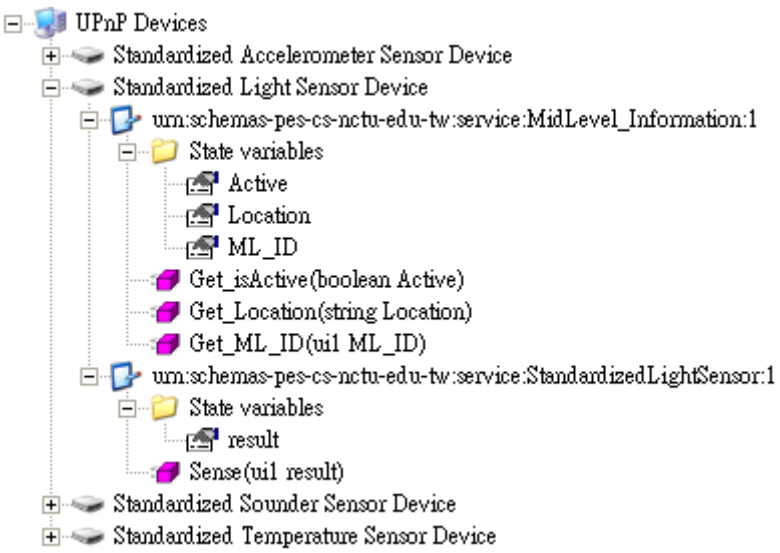

圖 9: 中層範例-UPnP 標準感測器

3.1.3 高層：虛擬裝置與服務界面

在高層我們就可以用開發者的眼光來發展高層回饋控制系統(Feedback Control System)[圖 7c]，只要知道有那些中層的感測器與促動器，從中挑選需要的感測器與促動器便可以開始開發系統，也透過模組化的方式，使系統易於開發和維護，並提供模組的可利用性。

裝置描述文件(Device Description)描述的主體就是高層回饋控制系統，來實現家庭自動化感測器與促動器互動控制，其中提供兩個服務，每一個服務一樣由個別的服務描述文件(Service Description)來描述：

- ❖ High Level Information Service：設定這個系統的環境參數，例如這個系統服務的地點。
- ❖ Sensor Actuator Collaboration Service：提供讓系統自動控制維持在一定參數的服務，是高層回饋控制系統的核心服務。

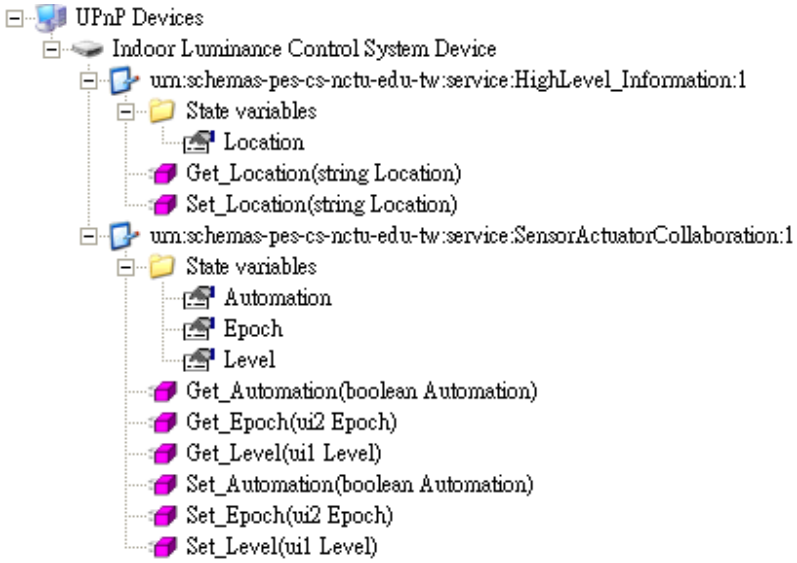

圖 10: 高層範例-UPnP 室內燈光回饋控制系統

以大部份的開發者而言，發展高層回饋控制系統多半會想利用通用的模組(中層 UPnP 裝置)來開發，如此就可以透過通用模組來使用到各種廠商製造的同種類裝置；當然，也可以使用低層 UPnP 裝置，因為低層 UPnP 裝置提供最接近實體裝置的架構與功能，可以用來做較精細的控制。

3.2 UPnP代理伺服器

家庭自動化的基礎架構最重要的就是要將感測器與促動器加入，但事實上這些實體裝置多半並沒有通用於 UPnP，可能是製造商沒有提供軟體支援，或者硬體上不能負荷，所以我們想要透過 UPnP 代理伺服器將之轉換成 UPnP 裝置，並希望有以下的特性：

- ❖ 隨時能增加部署(Incremental Deployment)

  在家庭自動化的基礎架構中加入新的實體裝置時，UPnP 代理伺服器要能即時的產生對應的 UPnP 裝置來擴展這個基礎架構。

- ❖ 動態操作(Dynamic Operation)

  在實體裝置移除或失去回應時，UPnP 代理伺服器要能讓對應的 UPnP 裝置暫時失去能力或者回復成一般的實體裝置。

❖ 多樣化的資訊傳輸方式(Various Data Transportation)

由於 UPnP 控制協定本身只能做參數的傳遞，我們希望能提供不同的資料傳輸方式，來提供串流的傳輸。像是 UPnP AV 架構運作流程中，控制點只負責伺服器與呈現器的協調及傳輸協定設定。當設定完成後，呈現器與伺服器會依照設定的結果建立連線，並將資料傳送到呈現器中播放

為了滿足以上的這些特性，我們要讓 UPnP 代理伺服器能提供這兩種操作程序：

1. UPnP 裝置初始化 (UPnP Device Initialization)
2. UPnP 裝置無回應與移除 (UPnP Device Deactivation and Retirement)

由以上這些分析，進而設計出通用UPnP代理伺服器的結構，操作程序也將在第四章做更詳細的介紹。

在此另外要聲明的是，UPnP 代理伺服器部份是由我和同實驗室的楊明曉學長所共同開發的，他會為開發更有彈性的 UPnP 代理伺服器繼續研究，所以本章節的部份內容會出現在他將來的研究論文中。

3.2.1 通用UPnP代理伺服器

UPnP 代理伺服器的目的是將實體裝置轉換成UPnP裝置，會有如同[圖 11]的通用結構。

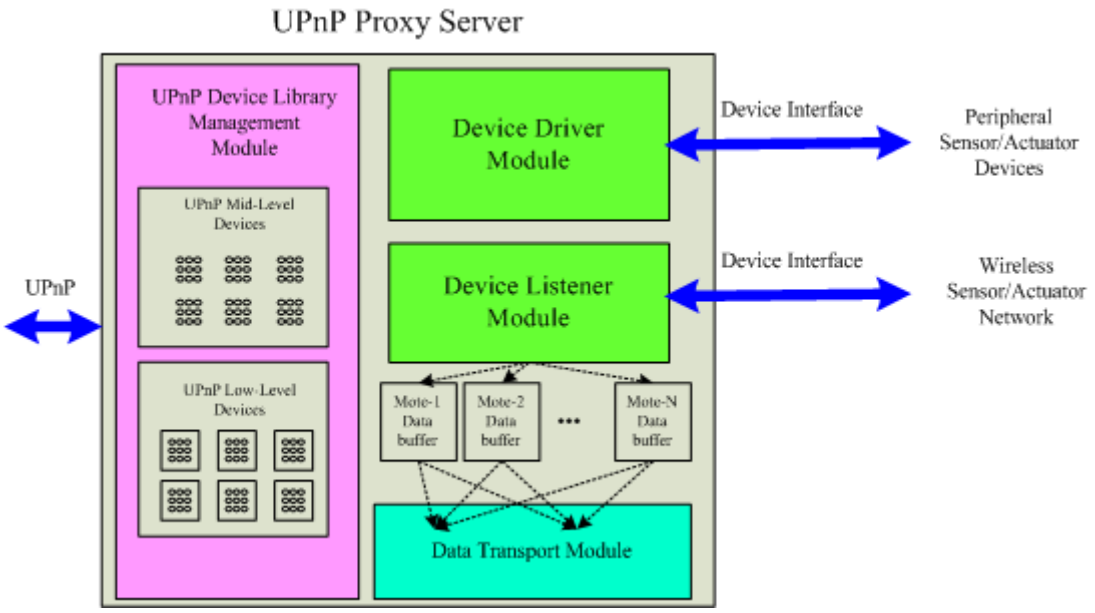

圖 11: 通用 UPnP 代理伺服器結構

UPnP 代理伺服器轉換的對象是週邊感測器與促動器裝置(Peripheral Sensor/Actuator Device)或者無線感測網路(Wireless Sensor/Actuator Network)，在轉換前統稱實體裝置(Physical Device)，也不是所有的實體裝置都能轉換成 UPnP 裝置，至少要提供與 UPnP 代理伺服器之間的裝置界面(Device Interface)，才能開發驅動程式來控制這些實體裝置。

在UPnP代理伺服器中有裝置驅動模組(Device Driver Module)與裝置收聽模組(Device Listener Module)，分別為不同的轉換對像服務。無線感測網路因為可以將資訊主動提供給UPnP代理伺服器，所以可以使用裝置收聽模組(Device Listener Module)來發現新的微型監測器的出現；而週邊感測器與促動器裝置和UPnP代理伺服器連接時，就必需要由使用者來操作裝置驅動模組(Device Driver Module)，如同2.1.3 週邊感測器與促動器裝置和無線感測網路的比較所介紹的，週邊感測器與促動器裝置較被動而且沒有無線感測網路有較高的動態性，所以使用者會在安裝週邊感測器與促動器裝置驅動程式的同時給予Information Service中的環境參數。

在UPnP裝置庫管理模組(UPnP Device Library Management Module)中會管理

UPnP裝置程式的存放,這些UPnP裝置程式都是依據3.1 階層式UPnP裝置與服務界面所設計出的低層、中層UPnP裝置。

在資料傳輸模組(Data Transport Module)中，則用來提供 UPnP 控制協定之外的資料傳輸方式。

3.2.2 適用於Berkeley Mote的UPnP代理伺服器

在本章節，我們為Berkeley Mote所構成的無線感測網路設計了Berkeley Mote UPnP代理伺服器，下圖[圖 12]為Berkeley Mote UPnP代理伺服器的結構。

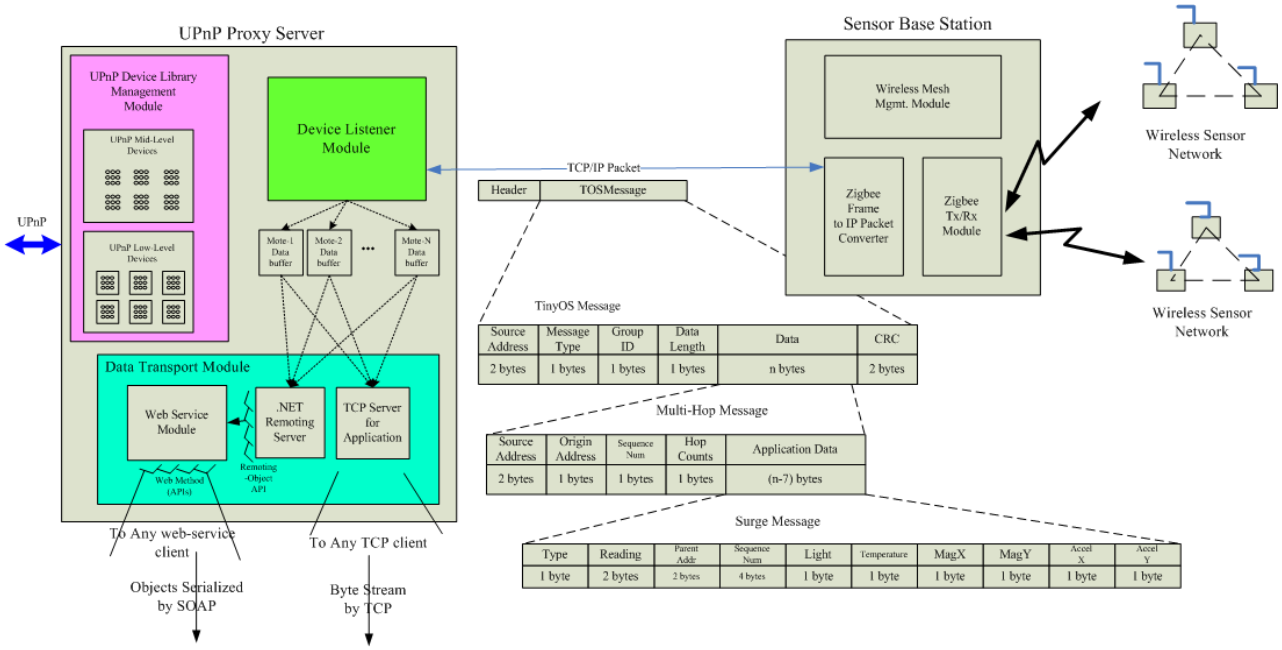

圖 12: Berkeley Mote UPnP 代理伺服器

Berkeley Mote UPnP 代理伺服器能順利運行的關鍵在於三點：

1. 無線感測網路與 UPnP 代理伺服器之間的界面有固定的溝通格式。

我們從Deciphering TinyOS Serial Packets [10]中瞭解到TinyOS有定義符合Surge應用程式的封包資料結構[圖 13]，這個封包資料結構成為無線感測網路與UPnP UPnP代理伺服器之間的溝通格式。如此，裝置收聽模組(Device Listener Module)便可藉由這些資訊可以解析(Deciphering)出基地台收集傳送給UPnP代理伺服器的資訊，也可以知道無線感測網路中是否有Mote的加入來自動產生

低層、中層UPnP裝置，以及Mote是否有回應來提供UPnP裝置無回應與移除的操作程序。

TinyOS Message data structure

```
typedef struct TOS_Msg
{
    /* The following fields are transmitted/
received on the radio. */
    uint16_t addr;
    uint8_t type;
    uint8_t group;
    uint8_t length;
    int8_t data[TOSH_DATA_LENGTH];
    uint16_t crc;

    /* The following fields are not actually
transmitted or received
    * on the radio! They are used for
internal accounting only.
    * The reason they are in this structure
is that the AM interface
    * requires them to be part of the
TOS_Msg that is passed to
    * send/receive operations.
    */
    uint16_t strength;
    uint8_t ack;
    uint16_t time;
    uint8_t sendSecurityMode;
    uint8_t receiveSecurityMode;
} TOS_Msg;
```

Multihop Message data structure

```
typedef struct MultihopMsg {
    uint16_t sourceaddr;
    uint16_t originaddr;
    int16_t seqno;
    uint8_t hopcount;
    uint8_t data[(TOSH_DATA_LENGTH - 7)];
}
```

Surge Message data structure

```
typedef struct SurgeMsg {
    uint8_t type;
    uint16_t reading;
    uint16_t parentaddr;
    uint32_t seq_no;
    uint8_t light;
    uint8_t temp;
    uint8_t magx;
    uint8_t magy;
    uint8_t accelx;
    uint8_t accely;
}
```

圖 13: 封包資料結構

2. 裝置收聽模組(Device Listener Module)能藉由使用者所提供的資訊，來知道 Mote 目前支援的 Management Service 和感測器。

為了能知道Mote安裝何種Sensor Board與TinyOS Surge應用程式支援的感測器，我們使用封包中Surge Message未定義的READING欄位來讓使用者在應用程式中提供這些資訊，我們將READING欄位[表 1]做了以下的定義來使用。

Support Management Service	Sensor Board Type	Support Sensor
2 bits	4 bits	10 bits

表 1: Surge Message 中的 **READING** 欄位定義 (2 byes)

表 1中每一個欄位的數值意義[表 2]：

Field	**Code**	**Description**
Support Management Service	00	No Support
	01 ~ 11	Management Service 1 ~ Management Service 3
Sensor Board Type	0000	No Support
	0001 ~ 1111	Type 1 ~ Type 15
Support Sensor	00 0000 0000	None – No sensor
	00 0000 0001 ~ 11 1111 1111	Each bit represents one sensor, set 1 to indicate support. 00 0000 0001 → ambient light sensor 00 0000 0010 → photo-sensitive sensor 00 0000 0100 → photo resistor sensor 00 0000 1000 → temperature sensor 00 0001 0000 → magnetic field sensor 00 0010 0000 → acceleration sensor 00 0100 0000 → sounder sensor 00 1000 0000 → tone sensor 01 0000 0000 → barometric pressure sensor 10 0000 0000 → humidity sensor

表 2: **READING** 欄位資料表示法

裝置收聽模組(Device Listener Module)藉由解析Surge Message中的READING欄位來瞭解Mote安裝何種Sensor Board與TinyOS Surge應用程式支援的感測器。以下我們用5.1 室內燈光回饋控制中UPnP代理伺服器所收到的READING欄位為例子:

Support Management Service	Sensor Board Type	Support Sensor
2 bits	4 bits	10 bits
00	0001	00 0011 1100

表 3: **READING** 欄位範列

表示不支援Management Service，0001 的Sensor Board樣式是MTS310[圖 14]，雖然硬體支援更多的感測器，但TinyOS Surge應用程式支援的感測器只有photo resistor感測器、temperature感測器、magnetic field感測器、acceleration 感測器，再配合Surge Message中的感測器欄位[圖 13]，就可以知道個別感測器的讀數。

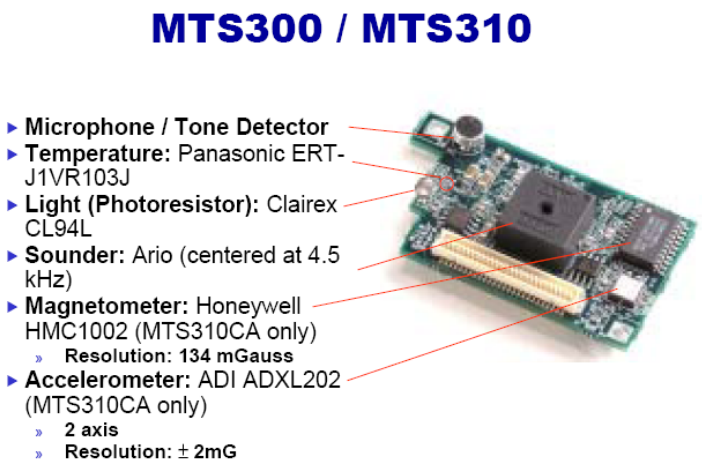

圖 14: MTS300/MTS310 Sensor Board

3. UPnP 裝置庫管理模組(UPnP Device Library Management Module)裡支援的對應的 UPnP Mote 與感測器裝置。

UPnP 代理伺服器在裝置收聽模組(Device Listener Module)分析完 READING 欄位的資訊後，會產生低層 UPnP Mote 裝置，加入支援的 Management Service，也將 TinyOS Surge 應用程式支援的感測器加入成為嵌入式裝置(Embedded Device)，個別支援的感測器也會產生對應的中層 UPnP 標準感測器裝置。

除了以上的設計，我們也提供了資料傳輸模組(Data Transport Module)來支援

Web Service 與 TCP 連線。

所以只要使用者滿足以下的條件：無線感測網路上的 Mote 運行 TinyOS，開發的應用程式資料結構使用 Surge Message，並在 Reading 欄位提供應用程式支援的 Management Service 與感測器資訊，加上 UPnP 裝置庫管理模組裡有支援的 UPnP 裝置，便可以透過我們的 Berkeley Mote UPnP 代理伺服器來自動產生 UPnP 裝置。

第四章　操作程序

在這個章節以使用者如何部署一個「適用於家庭自動化的感測與促動器基礎架構」為主軸，以及系統在背後所進行的運作流程，來深入瞭解：

- UPnP 代理伺服器何時產生低層、中層、高層 UPnP 裝置。
- 階層式 UPnP 裝置與服務界面理論中低層、中層、高層 UPnP 裝置的互動關係。

「適用於家庭自動化的感測與促動器基礎架構」在背後的支持在於兩個基礎部份，各提供了一些操作程序：

- UPnP 代理伺服器
  1. UPnP 裝置初始化 (UPnP Device Initialization)
  2. UPnP 裝置無回應與移除 (UPnP Device Deactivation and Retirement)
- 高層回饋控制系統 (Feedback Control System)
  1. UPnP 裝置發現 (UPnP Device Discovery)
  2. UPnP 裝置測試 (UPnP Device Training)
  3. UPnP 裝置控制 (UPnP Device Control)

在使用者部署完成後，只要這兩個部份同時運作就可以讓這個系統持續運行，UPnP 代理伺服器產生或移除 UPnP 裝置，透過使用者給予 UPnP 裝置環境的參數，最後由高層回饋控制系統 (Feedback Control System)，其實就是高層 UPnP 裝置，由監測與控制裝置(Monitor/Command Device)設定好參數後啟動來完成家庭自動化回饋控制系統。

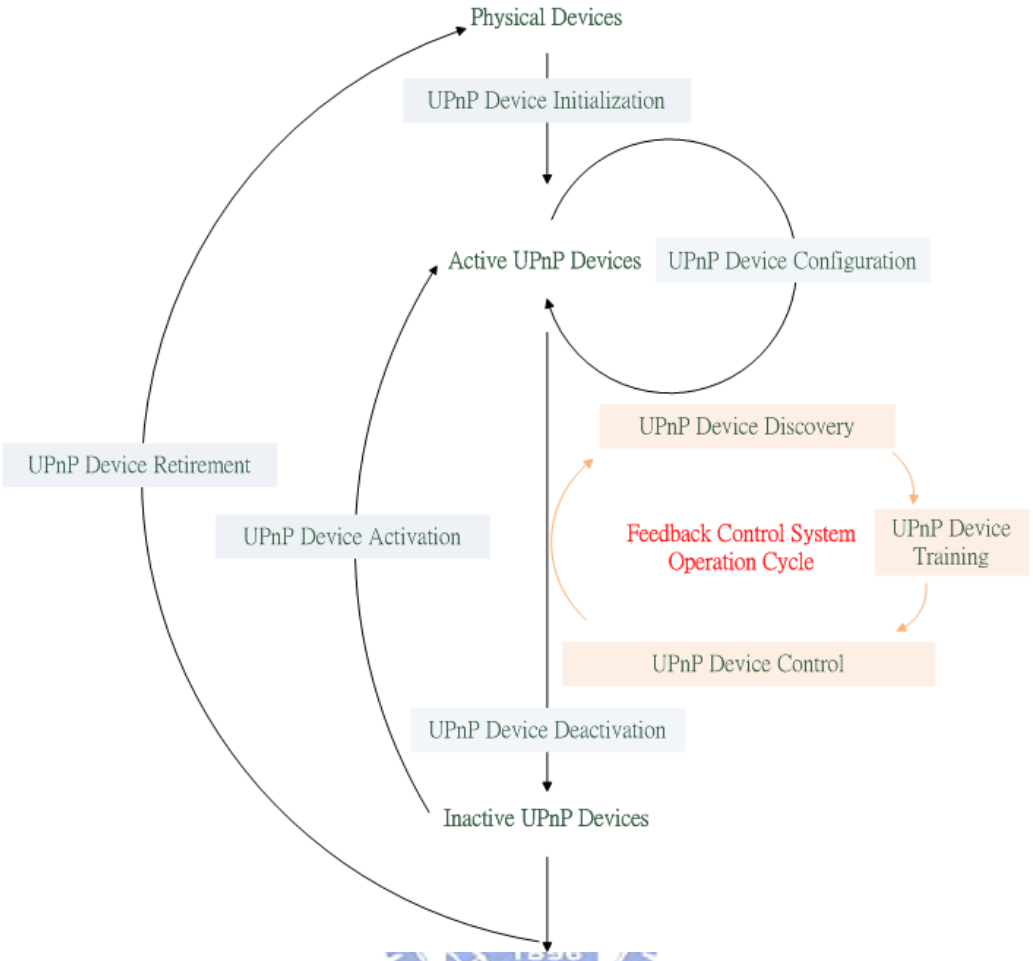

圖 15: 感測器與促動器生命週期

在之後的部份我們以實體裝置加入到這個自動化基礎架構後的生命週期 [圖 15]為順序，為大家介紹操作程序的細節。

4.1 UPnP裝置初始化

使用者在開始使要將圖 6: 感測與促動器基礎架構功能模塊中的所有元件部署好：

1. 啟動 UPnP 代理伺服器。
2. 從監測與控制裝置(Monitor/Command Device)設定好數位家庭伺服器(ehome Server)上高層回饋控制系統(Feedback Control System)的相關參數並啟動。
3. 接著就可以開始部署無線感測網路或週邊感測器與促動器裝置，我們統稱

實體裝置(Physical Device)。

在使用者為無線感測網路中加入微型監測器時，其實在基礎架構的背後就會透過UPnP代理伺服器中的裝置收聽模組(Device Listener Module)自動生成低層與中層 UPnP 裝置，此時 Information Service 中的參數都是初始的狀態，但已經是有效的(Active)UPnP 裝置。

4.2 UPnP裝置設定

對於高層回饋控制系統(Feedback Control System)而言，知道基礎架構中感測器與促動器的環境資訊是很重要的，但這些資訊是無法透過機器自動偵測的，像是將某個裝置放置在某個地點(Location)，因為地點(Location)的概念是只有使用者才曉得的，所以要藉由裝置設定來完成。

在週邊感測器與促動器裝置安裝好時，使用者會從 UPnP 代理伺服器中去設定裝置驅動模組(Device Driver Module)來安裝低層與中層 UPnP 裝置，在安裝的同時會直接給予 Information Service 中的環境參數，來產生有效的(Active)UPnP 裝置。

而微型監測器因為放置是動態的，所以微型監測器所產生的 UPnP 裝置需要讓使用者放置好後，使用控制點對低層 UPnP 裝置的 Low Level Information Service 設定環境資訊，在低層 UPnP 裝置設定完後，會將資訊同步到中層 UPnP 裝置中。

4.3 高層回饋控制系統操作週期

此時使用者的工作就告一段落了，可以開始享受高層回饋控制系統帶來的便利性，進入高層回饋控制系統操作週期，其實就是開發者所發展的高層或虛擬UPnP裝置，為了避免混淆，在第四章中統一使用高層回饋控制系統這個詞來說明。

高層回饋控制系統會從有效的(Active)UPnP裝置去尋找適合的感測器與促動器

來自動控制環境，其中核心的Sensor Actuator Collaboration Service要經過良好的設計才能讓感測器與促動器有良好的互動，為此我們提出模組化的方式來完成，可以對照圖 16：

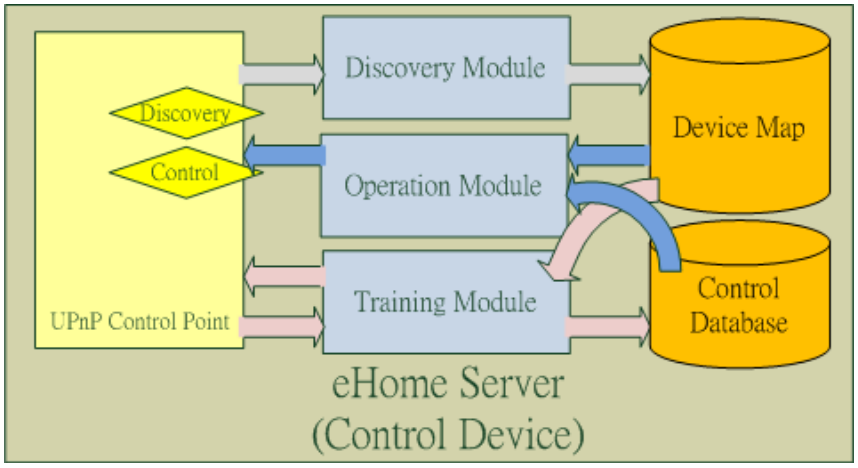

圖 16: 高層回饋控制系統模組化設計

- 控制點(Control Point)用來控制中層 UPnP 感測器或促動器裝置。
- *發現模組(Discovery Module)*利用控制點來發現系統要使用的感測器與促動器存入裝置庫(Device Map)中。
- *測試模組(Training Module)*控制裝置庫中的促動器並讀取感測器的變化，來測試所有感測器與促動器的互動性，將決定的控制方法(Control Method)存入控制資料庫(Control Database)中。
- *操作模組(Operation Module)*接受 UPnP 的控制協定，利用控制點執行決定的控制方法來完成高層回饋控制系統。

後文中的操作程序將使用以上的模組來完成。

4.3.1 UPnP裝置發現

UPnP 裝置發現(UPnP Device Discovery)主要是由*發現模組(Discovery Module)*所完成的，高層回饋控制系統藉由控制點來尋找所需要的中層 UPnP 裝置，可

以使用 UPnP 發現協定中的收尋目標標籤(Search Target Tag)來尋找特定的裝置類型或服務類型。

在室內燈光回饋控制的例子中，只需要用到中層的調光器(Dimmerable Light)與光感測器(Light Sensor)，所以高層回饋控制系統內部的*發現模組(Discovery Module)*只要指定 UPnP 控制點在使用 UPnP 發現協定中的收尋目標標籤為以下裝置類型就夠了：

- `urn:schemas-upnp-org:device:DimmableLight:1`
- `urn:schemas-peslab-cs-nctu-edu-tw:device:StandizededLightSensor:1`

由於感測器裝置在 UPnP 論壇並沒有標準存在，所以我們使用自訂的命名空間(Namespace)，*發現模組(Discovery Module)*所找到的裝置便會存放在裝置庫(Device Map)中。

讀者可能會質疑為什麼在此處並不過濾 UPnP 裝置的環境參數與檢查裝置是否為有效的(Active)狀態，這就要探討 UPnP 發現協定了，高層回饋控制系統的*發現模組(Discovery Module)*在找到裝置的時機只有在這兩個時間點：

1. 高層回饋控制系統剛開啟時，控制點初次啟動會廣播 Discovery Request，來主動尋找需要的 UPnP 裝置。
2. 個別 UPnP 裝置在啟動後會發出 Presence Announcement，讓高層回饋控制系統中的控制點來發現。

如果在此處就過濾 UPnP 裝置的環境參數與檢查是否為有效(Active)狀態，就會有 UPnP 裝置被過濾掉，如果之後這些 UPnP 裝置改變狀態，就必需要等到高層回饋控制系統重新開啟，或者 UPnP 裝置重新啟動才會再經過*發現模組(Discovery Module)*再次過濾，所以我們把這個部份保留到每次使用服務(Service)時檢查，雖然會稍微費時，但會增加系統的即時性。

4.3.2 UPnP裝置測試

UPnP 裝置測式(UPnP Device Training)主要是由*測試模組(Training Module)*所完成的，*測試模組(Training Module)*控制裝置庫中的促動器並讀取感測器的變化，來測試所有感測器與促動器的互動性，包含：

- ❖ 個別促動器對個別感測器影響的比重
- ❖ 多個促動器如何加總產生對感測器的影響

將決定的控制方法(Control Method)存入控制資料庫(Control Database)中。

*測試模組(Training Module)*在一個感測器與促動器的基礎架構中是必需存在的，不過由於這個部份的通用設計需要比較多的控制理論，通用的設計方法還仍待研究。

4.3.3 UPnP裝置控制

UPnP 裝置控制(UPnP Device Control)主要是由*操作模組(Operation Module)*所完成的，主要有兩個部份：

1. 接受使用者來自監測與控制裝置(Monitor/Command Device)上的設定，像是自動控制多久執行一次，自動控制是否啟動，以及想要自動維持的環境參數。
2. 執行 UPnP 裝置測試所決定的控制方法(Control Method)。

當使用者啟動自動控制後，*操作模組(Operation Module)*便會定期執行選定的控制方法將環境維持在所想要的參數。例如在室內燈光回饋控制的例子中，會用事先選定的控制方法，透過控制點根據光感測器的變化來控制特定的調光器，直到光感測器的讀數達到使用者所想要的參數。

4.4 UPnP裝置無回應與移除

最後，如果使用者想要移除裝置，如果是週邊感測器與促動器裝置不再提供服務，使用者就要透過裝置驅動模組(Device Driver Module)將之回復成一般的實體裝置。而在無線感測網路上，只要將不需要的微型監測器直接移除就好，並不會影響高層回饋控制系統操作週期，背後的基礎架構是這樣運作的：

- 當 UPnP 代理伺服器中的裝置驅動模組(Device Driver Module)發現有效的(Active)UPnP 裝置無回應一段時間後，感測器的讀數將不更新，促動器也不再接受控制，會將他的狀態設定成無回應成為無效的(Inactive)UPnP 裝置， 高層回饋控制系統要注意略過這些無效的(Inactive)UPnP 裝置。
- 這些無效的(Inactive)UPnP 裝置可能在過一回後就恢復回應，那就會變回有效的(Active)UPnP 裝置，給適合的高層回饋控制系統來使用。
- 如果這些無效的(Inactive)UPnP 裝置無回應過久超過我們預設的門檻時，UPnP 代理伺服器便會送出 UPnP Device Unavailable Message，來通知這個 UPnP 裝置與其上的服務已經不存在了，回復成一般的實體裝置。

第五章　應用

5.1　室內燈光回饋控制

最後我們將「適用於家庭自動化的通用隨插即用感器與促動器基礎架構」實作在交通大學電資大樓智能環境實驗室(NCTU MIRC Smart Environment Lab) [圖17]中，用來做室內燈光回饋控制，是使用3.1 階層式UPnP裝置與服務界面所提出的理論所實作出來的，也有很好的控制效果。

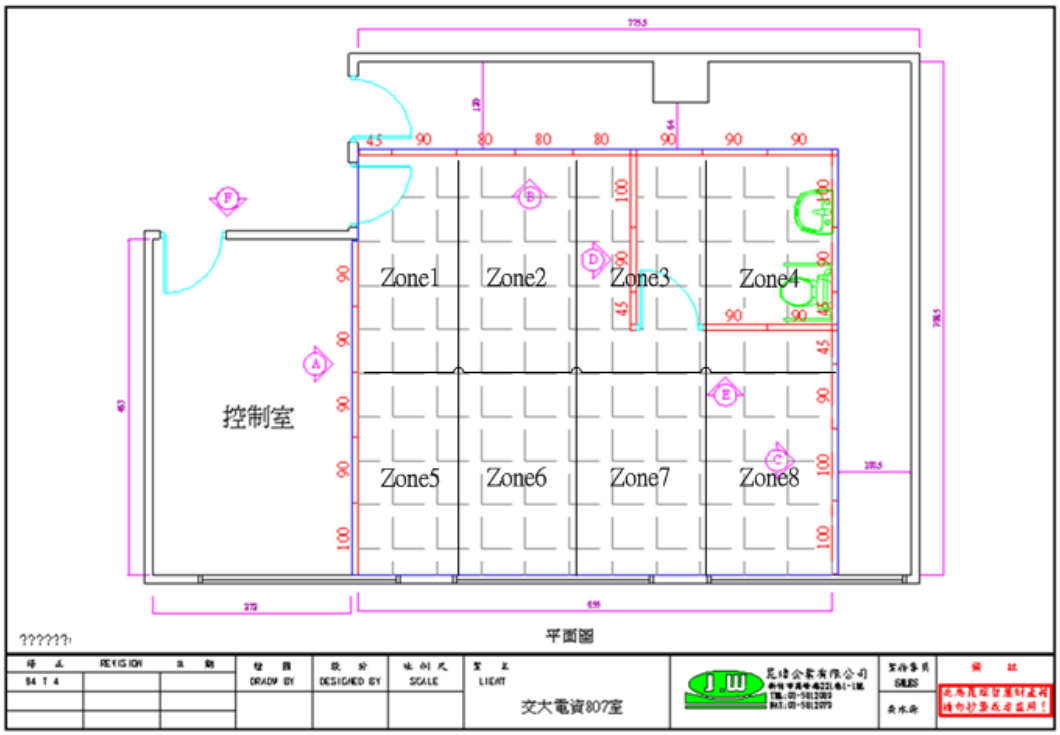

圖 17: 交通大學電資大樓智能環境實驗室

在實驗室中有預先安裝的調光器，分成八個區域(Zone1~Zone8)，使用Lite-Puter公司的前端燈光控制系統(Lighting Control System)，可以接收RS485 的控制訊號來控制這些調光器，我們使用3.2.1 通用UPnP代理伺服器來產生：

- ❖ 低層裝置－UPnP 燈光控制系統
- ❖ 中層裝置－UPnP 調光器(使用 UPnP 論壇所訂的標準裝置描述文件 DimmableLight:1 Device Template Version 1.01)。

在進行實驗時，我們會佈置無線感測網路，組成的微型監測器是使用Crossbow公司的Micaz，安裝MTS310 Sensor Board[圖 14]，我們是使用上面的Light (Photoresistor)感測器。無線感測網路藉由processing board MIB600 提供Ethernet的界面與3.2.2 適用於Berkeley Mote的UPnP代理伺服器來連接，將無線感測網路中基地台所收集到的資訊送到UPnP代理伺服器上，動態產生：

- ❖ 低層裝置－UPnP Mote
- ❖ 中層裝置－UPnP 光感測器

並依照所放置的區域，使用控制點來設定 UPnP Mote 地點參數。

有了以上的需要的UPnP裝置，就可以使用監測與控制裝置(Monitor/Command Device)，可以是PDA或PC上的控制點來控制數位家庭伺服器(ehome Server)上的室內燈光回饋控制系統[圖 18]。

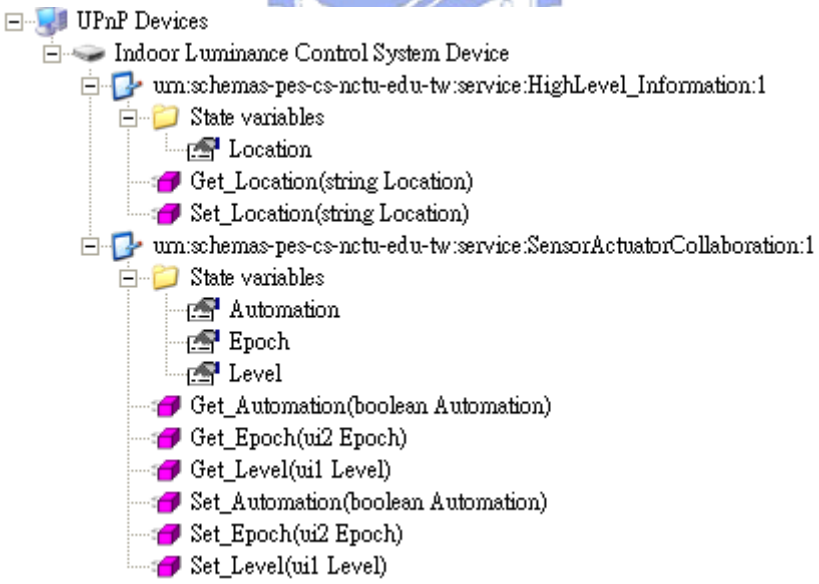

圖 18: 室內燈光回饋控制系統

使用者可以透過 High Level Information Service 來設定室內燈光回饋控制系統所運作的地點 (Zone1~Zone8)，室內燈光回饋控制系統就會使用放在該區域的 UPnP 調光器與 UPnP 光感測器來進行自動控制。

在設定好High Level Information Service後，就要使用Sensor Actuator Collaboration Service來設定自動控制參數，當Automation設定成啟動(True)後，室內燈光回饋控制系統就會在每過Epoch(單位：秒)的時間後，執行圖 19的控制方法來完成自動控制，能在一個區域維持很好的光控效果。

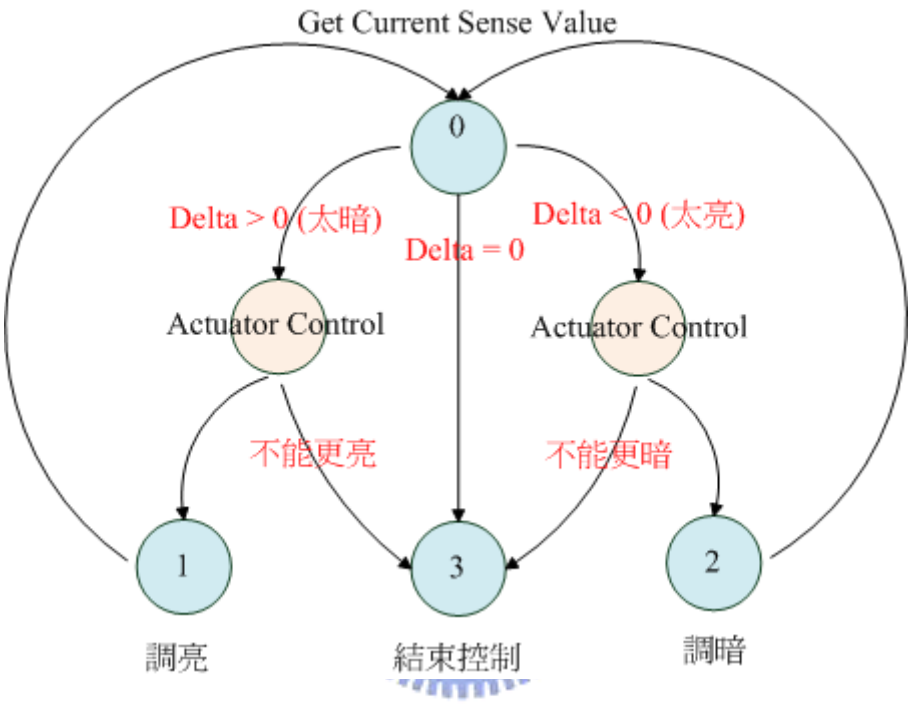

圖 19: 室內燈光控制方法

(Delta = User Wanted Level – Current Sense Value)

第六章　結論與未來方向

6.1　研究成果

在這個研究中我們完成的有:

- ❖ 定義「適用於家庭自動化的通用隨插即用感測與促動器基礎架構」的功能模塊來描述基礎架構中該有的組成。
- ❖ 發展階層式UPnP裝置與服務界面理論，利用這個理論來為感測器與促動器開發UPnP低層與中層裝置，以及高層回饋控制系統，將之分成三層[圖 7]：
  - 低層界面反應廠商自訂裝置的架構與功能。
  - 中層界面藉由低層界面提供標準模組的架構與功能。
  - 高層界面則提供高層虛擬裝置，利用中層界面達到感測器與促動器回饋控制。
- ❖ 設計 UPnP 代理伺服器來將實體裝置轉換成 UPnP 裝置。
- ❖ 發展「適用於家庭自動化的通用隨插即用感測與促動器基礎架構」的操作程序，由 UPnP 代理伺服器與高層回饋控制系統來支持，其中高層回饋控制系統核心的 Sensor Actuator Collaboration Service 也提出模組化的方式來完成。
- ❖ 最後將「適用於家庭自動化的通用隨插即用感測與促動器基礎架構」使用在交通大學電資大樓智能環境實驗室中，用來做室內燈光回饋控制的實驗，也有很好的控制效果，其中使用 UPnP 代理伺服器產生的標準模組：UPnP 調光器與 UPnP 光感測器，也已經被其他的研究所使用，將發展更多的家庭自動化回饋控制系統。

6.2 未來方向

為了讓「適用於家庭自動化的通用隨插即用感測與促動器基礎架構」更完善，以下是我們仍待解決的：

❖ 階層式 UPnP 裝置與服務界面的部份

- 低層與中層轉換方程式(Transformation Function)

  在低層 UPnP 裝置與中層 UPnP 裝置的刻度轉換中，目前是使用線性轉換，是否可以利用第三方準確的裝置做標準，在刻度轉換時能加以校正，在中層 UPnP 裝置提供準確的刻度是可以研究的課題。

- 測試模組(Training Module)

  測試模組(Training Module)在一個感測器與促動器的基礎架構中是必需存在的，不過由於這個部份的通用設計需要比較多的控制理論，通用的設計方法還仍待研究。

- System Management Service

  System Management Service 是在低層 UPnP 裝置中預留的一個服務，未來如何與無線感測網路、週邊感測器與促動器裝置有效整合，並提供一個標準服務描述文件(Standardized Service Description)和通用的設計方法還仍待研究。

❖ UPnP 代理伺服器的部份

- 裝置驅動程式與對應的 UPnP 裝置

  發展更多的驅動程式與對應的 UPnP 裝置以支援更多的感測器與促動器。

- 裝置庫管理系統

  開發良好的使用者介面讓驅動程式與對應的 UPnP 裝置可以方便匯入，並提供良好的管理系統來管理。

參考文獻

[1] 2007 IEEE International Conference on Systems, Man, and Cybernetics, http://www.ieeesmc.org/

[2] TinyOS, http://www.tinyos.net/

[3] UPnP™ Device Architecture, http://www.upnp.org/specs/arch/UPnP-DeviceArchitecture-v1.0-20060720.pdf

[4] UPnP Forum™, http://www.upnp.org/

[5] Microsoft, “Understanding Universal Plug and Play” White Paper

[6] The Journal of Spontaneous Networking, It Just Works: UPnP in the Digital Home by Michael Jeronimo, October 5, 2004, http://www.artima.com/spontaneous/upnp_digihome.html

[7] Intel® Software for UPnP Technology, http://www.intel.com/cd/ids/developer/asmo-na/eng/downloads/upnp/overview/index.htm

[8] Hyungjoo Song, Daeyoung Kim, Kangwoo Lee, Jongwoo Sung, “UPnP-Based Sensor Network Management Architecture,” *Proceedings of 2nd International Conference on Mobile Computing and Ubiquitous Networking*, April 13–15, 2005 in Osaka, Japan.

[9] Suet-Fei Li, “Wireless Sensor Actuator Network for Light Monitoring and Control Application,” *Proceedings of 5th IEEE Consumer Communications and Networking Conference*, January 8-10, 2006 in Las Vegas, Nevada.

[10] Jeff Thorn, Director of Product Development, “Deciphering TinyOS Serial Packets,” Octave Tech Brief #5-01, March 10, 2005