

# 支援存取 Web services 的 Jini 家用閘道器之設計

## The Design of Jini Technology-Enabled Residential Gateway Supporting Web services Access

林文璋

林郁全、沈文智

\*張晉源

樹德科技大學資訊管理研究所 樹德科技大學資訊工程系 樹德科技大學資訊工程系

\*E-mail:[cychang@mail.stu.edu.tw](mailto:cychang@mail.stu.edu.tw)

### 摘要

家用閘道器(Residential Gateway, RG)已被許多市場分析機構預測將成為下一波重要的資訊家電產品。它的主要功能是設置於家庭與各種建築物中，作為連結網際網路以獲取外部資訊服務使用。本文探討在一個符合 OSGi 規範[4][9]的家用閘道器上，如何設計支援 Jini 裝置透過此閘道器向外部取得各種 Web services。

本文在設計上將 Jini 架構[14][16]所必需使用的服務協定視為家用閘道器中 JES[13][15]的一項 custom service。具備此種功能的家用閘道器，不但可以讓家庭網路內部的各種資訊家電裝置與網際網路相連結，內部的網路還可以支援 Jini 裝置的使用，讓 Jini 裝置可以透過本文所設計的家用閘道器取得外部的各種 Web services。

### Abstract

Many market investigation companies forecast that Residential Gateway(RG) would be a killer application of IA in the near future. The RG, set up in a building, is applied to connect internal network with the Internet. In this paper, how to design an OSGi-compliant RG supporting Jini technology-enabled devices to access a variety of Web services on the Internet is described.

In design, the service protocols for Jini architecture are regarded as one of the customer services on the Java Embedded Server(JES). The RG deployed the design in this paper, could not only support the Jini-enabled devices to access any services on internal network, but they could also get a variety of services from the external networks.

Keywords: Residential Gateway, OSGi, Jini, JES.

### 一、前言

隨著網際網路的快速成長及新興資訊家電裝置的出現，家庭網路(Home Local Area Network，

Home LAN)所相關的資訊家電裝置已被視為資訊產業下一波市場主力產品。此外，資訊設備的使用已逐漸由以 PC 為中心擴展到以網際網路為範圍的應用架構。所以，使用者也逐漸以簡單易用(easy to use)的資訊家電裝置取代 PC，以連結網際網路而取得所需要的各項資訊與服務。因此，具備與其他裝置交換訊息網路功能的資訊家電產品將會對家庭網路的發展產生的影響。

家用閘道器設置於建築物的網路或家庭網路的接取點處，作為建築物內部網路與外部網際網路間連接的橋樑。根據 Cahners In-Stat 的市場預估統計[10]，由於家庭寬頻上網市場的蓬勃發展，使得家庭寬頻上網的關鍵裝置-家用閘道器的需求也逐漸增強。此外，由於近年來擁有多部 PC 的家庭愈來愈多，設備的單獨運作已經無法滿足使用的需求，再加上各廠家自行設計的資訊家電產品缺乏共同依循的標準，以致於造成相當多產品間互通性的問題[5][6]。因此，設備或裝置間的相互溝通能力必將成為未來資訊家電發展的重點[14]。傳統的 client/server 和多層的網路架構將會被整個網路間隨插即用式(plug-and-play)的裝置連結方式所取代。藉由具備支援 Jini 技術的家用閘道器，內部網路的裝置除了可以和其它的裝置相互連結以外，也能取得網際網路上各種的 web services。例如：使用者將 Jini 數位相機加入 Jini community 時，它可以將儲存於記憶體中的數位相片透過 Jini 印表機直接列印相片。然而在印表機本身含有一些基本的影像特效服務，如：加上佈景、圖片等，當使用者需更多佈景影像或特效服務時，它將可以透過本文所設計的家用閘道器內部服務模組連結網際網路向外部 web services 取得所需的影像特效服務。

為滿足上述需求的功能，本文設計一種支援 Jini 架構且符合 OSGi 的新型家用閘道器。本文設計納入與 Jini 架構相關的服務模組(例如：搜尋服務模組、服務代理模組、網路服務萃取模組和使用介面模組等)，以讓內部網路的 Jini 裝置可透過閘道器上所嵌入的服務模組，從網際網路上的 web services 取得 Jini 裝置所需求的服務。此外，本文設計的閘道器也具備嵌入式的服務模組，以協助使

用者自動搜尋、管理及登錄內部網路上的所有 Jini 裝置。同時，使用者也可以透過閘道器服務模組，直接控制家內部網路中的任何 Jini 裝置。

在閘道器實作平台的選擇上，本研究採用 Linux 作業系統與 Java 相關技術架構出基本的發展環境。在最初的實作版本中，本研究以 PC 平台架構出家用閘道器的雛型。家用閘道器中的嵌入式服務模組主要是遵循 Java 相關技術規範，如：Jini、RMI、servlet 和 OSGi 等所設計[12] [9]，並搭配由 Sun Microsystems 所提出的 JES (Java Embedded Server) Service Framework 架構組合而成[13][15]。在本文第二節中，我們將討論家用閘道器相關技術與研究背景。在第三節中將說明本文所設計家用閘道器之應用架構和此閘道器的功能特色。第四節中則對本文家用閘道器的基本架構模組逐一說明，並詳述各個嵌入式服務模組在閘道器上所具備的功能和執行的方法。第五節則說明本閘道器嵌入式功能模組之設計。最後，第六節中則簡要說明本研究的心得與未來工作的執行方向。

## 二、相關研究背景

目前市場上所發表的家用閘道器產品[11]，底層的硬體平台大多採用 embedded system 的解決方案，而作業系統大多採用 Linux。它所支援的介面一般包括 HomePNA、Ethernet、Powerline 和 WirelessLAN 等。在系統服務的功能上則支援如：HTTP、LOG、Proxy 及 DHCP 等多項服務[7]。近期國內對於家用閘道器的研究主要有交通大學資訊科學系針對無線家庭網路環境所提出的住宅型閘道器[2]。它運用 Jini 的技術提供一個管理服務器。此管理服務器使用一個簡單的介面幫助系統自動搜尋、登錄與管理具有 Jini 功能的家電。另一項研究是由交通大學電機與控制工程系以 Linux 為基礎嵌入符合 OSGi 規範的網路服務閘道器[1]。它的特色在於利用 Java 技術實作其 Framework 的架構，以具備跨平台的運算能力。此一研究並以 X10 裝置展示閘道器所連接的裝置具有 plug-and-play 的功能。

本研究主要著重於如何讓內部網路中的 Jini 裝置可以直接透過家用閘道器存取網際網路上的各項服務，並提供使用者可以直接下載網際網路上的資訊給 Jini 裝置直接驅動使用。因為 Jini 裝置本身兼具有 discovery 與 lookup 的功能，因此，在家用閘道器中支援 Jini 的技術，除了讓內部網路的 Jini 裝置可以透過家用閘道器與網際網路溝通外，也提供使用者一個簡單的 Jini 裝置控制管理的介面。

## 三、家用閘道器應用架構

如圖 1 所示，在一般家庭中，家用閘道器設置於寬頻網際網路與家庭網路之間，扮演著橋樑與出入口的角色[7][8]。它除了可以長時間開機(always on)作為連接寬頻網路運作的網路資料交換器外，同時也提供傳輸介面讓家庭網路內部的各項資訊家電透過該裝置的介面快速、方便地進行網路資料的分享、轉換和移轉等應用。

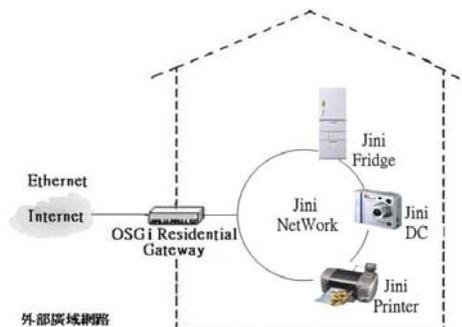


圖 1 本文家用閘道器應用架構

## 四、家用閘道器嵌入式功能模組之說明

### 4.1 組成元件

家用閘道器所提供的 services 包括 system services 與 custom services 兩種。為強化閘道器的功能，閘道器的設計者並須提供自型開發的 custom services 提供使用者應用，以區別產品的應用市場。本文在 custom services 服務模組的實作上除了支援由 JES 所提供的內嵌式基本服務功能模組外，均使用 Java 相關技術(Jini、RMI 和 servlet)分別研發搜尋服務模組(Lookup Service Module)、服務代理模組(Service Agent Module)、使用者介面模組 (User Interface Module) 和網路服務粹取模組(Web services Retrieval Parser Module)等四個服務模組。所有的服務模組都以 bundles 的形式架構在家用閘道器的 JES 層上作為 custom services 使用。

### 4.2 功能模組之介紹

本文家用閘道器主要功能是讓家庭內部的資訊家電能夠存取網際網路上 web services 所提供的各式資源。要達成此項功能，在家用閘道器中必須要嵌入數個基本服務功能模組，各個功能模組在系統架構上扮演著支援 Jini 裝置不同角色。本設計中將這些模組區分為上述的四種，模組間的架構與關係如下圖 2 所示。

#### 4.2.1 搜尋服務模組(Lookup Service Module)

搜尋服務模組主要是提供 Lookup、Discovery 以及 Join 等協定功能。搜尋服務模組是一個集中註冊服務的地方，其功能主要是執行網路尋找服務、管理服務的使用期限和服務註冊等。此外，搜尋服務模組中存放著許多服務項目，每個服務項目都有一個 RMI 的遠端物件，每個物件都提供一個遠端介面。這些介面提供用戶端可引用的函式，而用戶端程式引用這些函式時，就如同呼叫當地的函式一樣的方便。

本模組在註冊服務項目方面採用 group 的形式作為分類。group 主要分為 device group 和 service group。如此的分類目的在於方便管理與維護。當內部網路中的 Jini 裝置要向外部 web services 取得它所需要的服務時，這時的 Jini 裝置則扮演著 client 的角色，服務代理模組則扮演著 service provider。

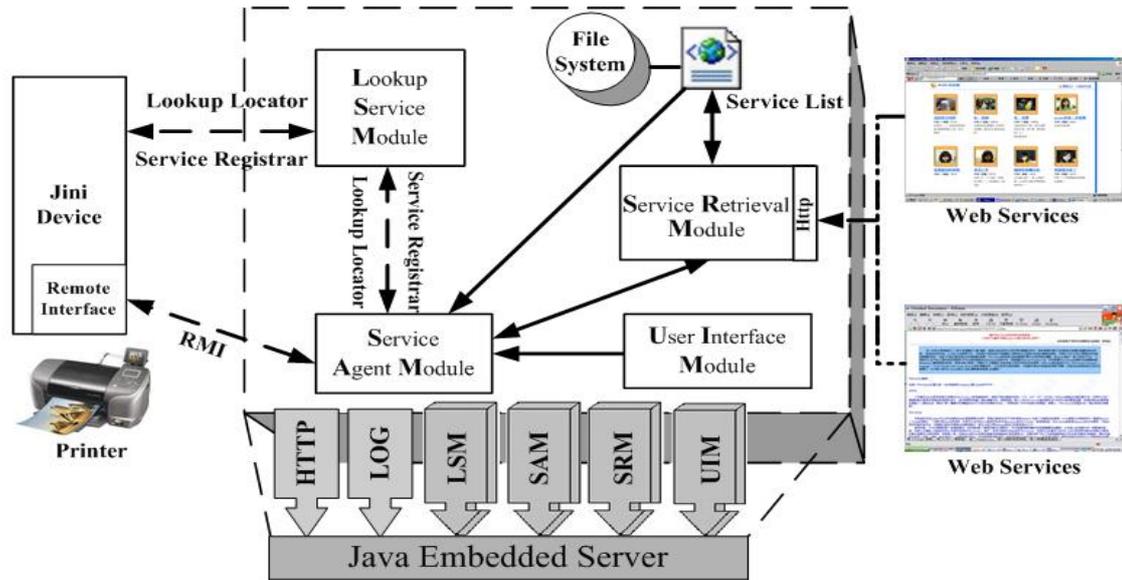


圖 2 本文家用閘道器功能模組架構圖

因此，當 Jini 裝置向搜尋模組發出請求 lookup service 時，所取得到的服務是從 service group 中下載而來的，事實上該服務是由服務代理模組所提供。另一方面，當使用者要以 web 畫面進行監視裝置狀態時，Jini 裝置則扮演著 service provider 的角色。Jini 裝置會將與遠端介面及相關的應用程式註冊至搜尋服務模組裡的 device group 裡，並透過服務代理模組轉變為 servlet 物件。服務代理模組在這裡則扮演著 client 的角色，它向搜尋模組取得從 service group 中的服務，並將該服務傳遞至使用者介面模組。使用者介面模組則以 servlet 技術實作出讓使用者能夠以 web 畫面進行監視裝置狀態的應用程式畫面。

#### 4.2.2 網路服務萃取模組(web services Retrieval Parser Module)

網路服務萃取模組透過服務代理模組的通知，且藉由 HTTP 通訊協定連接至外部 web services，以萃取 web services 所提供的資源（如圖片等）。所萃取的資源以檔案串流型態（streaming type）儲存，再將該串流回傳至服務代理模組，最後透過服務代理模組將該服務傳遞至 Jini 裝置上。另一方面，本模組透過語法剖析程序會自動萃取被指定 web services 所提供網頁中符合的標籤 tag(如 <img 開頭的 tag)。

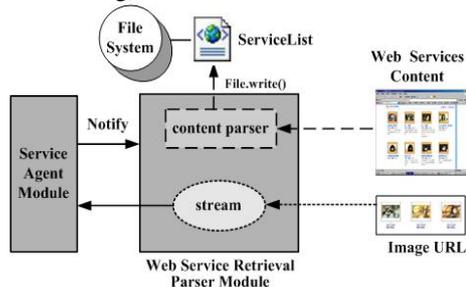


圖 3 網路服務萃取模組架構

在本文的假設應用中，圖 3 顯示產生符合本架構中 Jini 列印裝置所需服務的圖片連結位址資料。

#### 4.2.3 服務代理模組 (Service Agent Module)

本模組主要扮演 Jini 裝置與網路服務萃取模組間相互溝通的橋樑。它將與 Jini 裝置溝通相關的 remote interface 和 object(這裡的 remote interface 和 object 指的就是 service object) 以 discovery/join 的方式註冊到搜尋服務模組的 service group 中，如圖 4 所示。

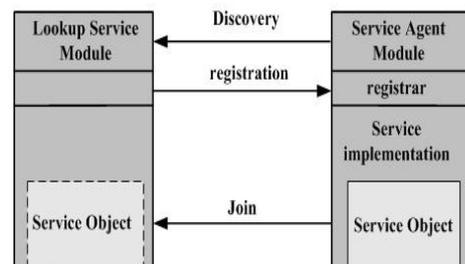


圖 4 物件與屬性註冊至搜尋服務模組

當內部網路中的 Jini 裝置查詢到該 service object 時，搜尋服務模組會將該 service object 下載至 Jini 裝置上。Jini 裝置藉由 service object 所提供的遠端介面可連結結至服務代理模組，並將上述範例中的圖片服務清單回傳至 Jini 裝置上以提供後續使用。此外，當使用者透過使用者介面模組要求得知目前內部網路控制節點的狀態時，服務代理模組則扮演呼叫代理者的角色。它會將所有內部網路裝置的狀態資訊傳送給使用者介面服務模組，並隨即以網頁畫面呈現給使用者。相對地，當使用者送出控制某個裝置狀態的訊息時，使用者介面服務模組會透過本模組將該控制訊息轉而通知所指定的裝置，並下達指令以完成所要控制的動作。

#### 4.2.4 使用者介面模組(User Interface Module)

使用者介面模組的功能主要使用 servlet 技術實作出讓使用者以 web 畫面進行監視裝置狀態回報作業的應用程式畫面。此模組的設計目的是讓使用者不必事先確認出目前所連接的是哪種類型的網路 (Jini community)。透過本服務模組，用戶端可以得知目前內部所連接的 Jini 網路中所有控制節點的狀態資訊。

#### 4.3 Jini 列印裝置 (Jini 裝置-Printer)

Jini 列印裝置在於提供一個 view-interface 給內部網路的使用者去設定及使用。當裝置啟動並連接到網路時，它便會啟動裝置內部的 Jini service 去搜尋網路上的 lookup service。利用 lookup 協定去取得從服務代理模組註冊至搜尋服務模組的服務物件屬性後，可經由服務代理模組所提供的遠端物件取得 ServiceList(ServiceList 指的是裝置可選取的服務列表，主要是儲存服務的 ID 與服務相對應的 URL)。此外，當採用 Jini 技術時，Remote Method Invocation(RMI)是不可或缺的一環。基於 RMI 本身具有高度地抽象性，使得 developer 在開發 client 和 service provider 之間的通訊程式時十分便利，因此，本研究在實作這些模組時都撰寫成具備有 RMI 功能的 bundles。在本研究的 Jini 列印裝置中，目前是以應用程式來模擬呈現出 Jini 裝置，並利用 RMI 功能與服務代理模組作為訊息溝通使用。

#### 4.4 功能模組運作流程

本文設計閘道器服務模組之目的主要用來符合下列兩種情況：

1. Jini 裝置能夠藉由閘道器向網際網路上的 web services 取得 Jini 裝置本身要求的服務，並將萃取回來的服務做一個有效的管理。當進行該動作時，本閘道器服務模組運作流程如下：

服務代理模組會將與 Jini 裝置溝通相關的遠端介面和相關應用程式(遠端介面和相關應用程式指的就是 service object)註冊至搜尋服務模組的 service group 裡。Jini 裝置會向搜尋服務模組 lookup 且下載 service group 裡的 service object，Jini 裝置會透過該 service object 所提供的遠端介面與服務代理模組做溝通，服務代理模組這時會讀取儲存在 FileSystem 裡的 ServiceList 檔，並將服務清單回傳給 Jini 裝置，讓 Jini 裝置可選取它所要的服務。當 Jini 裝置選取它所需要的服務後，Jini 裝置同樣會透過由搜尋服務模組所下載的 service object 傳遞該要求訊息(device\_id 與 service\_id)至服務代理模組。服務代理模組接受到訊息後，會透過網路萃取模組向外部的 web services 取得服務，最後將該服務回傳至 Jini 裝置上所使用。

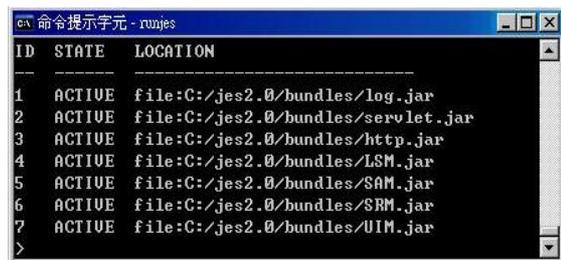
2. 使用者可透過使用者介面模組直接控制家庭內部任何註冊至搜尋服務模組的 Jini 裝置。當進行該動作時，本閘道器服務模組運作流程如下：

Jini 裝置會將本身的狀態資訊及溝通介面等

相關物件與屬性(service object)註冊至搜尋服務模組的 device group。服務代理模組會向搜尋服務模組查詢且下載 device group 中的 service object，透過該 service object 所提供的遠端介面將該服務傳遞至使用者介面模組。讓使用者能夠透過 web 畫面得知目前家庭內部 Jini 網路所有 Jini 裝置的狀態資訊。另一方面，當使用者從使用者介面模組所實作出的 web 畫面傳送控制裝置的訊息時，使用者介面模組會將該要求訊息傳遞至服務代理模組，隨後並將該訊息傳遞至 Jini 裝置上變更目前的狀態值。

### 五、功能模組之設計

本研究在嵌入式功能模組的設計部份主要關鍵技術在於納入 Jini 技術後如何將 web services 上的資源下載至內部網路中讓 Jini 裝置存取及如何透過功能模組所提供的 web 畫面進行 Jini 裝置控管等功能。本家用閘道器的功能包含搜尋服務功能、網路服務萃取功能、和服務代理等功能。在技術使用方面，除使用 Jini 技術外，使用各種 Java 技術 (RMI、servlet) 和 Web 技術，且遵循 OSGi 規範去實作出本研究家用閘道器嵌入式服務模組，這些架構在 Service Framework 上的 custom services 都是以 bundle 的形式執行測試，範例測試畫面如圖 5 所示。



ID	STATE	LOCATION
1	ACTIVE	file:C:/jes2.0/bundles/log.jar
2	ACTIVE	file:C:/jes2.0/bundles/servlet.jar
3	ACTIVE	file:C:/jes2.0/bundles/http.jar
4	ACTIVE	file:C:/jes2.0/bundles/LSM.jar
5	ACTIVE	file:C:/jes2.0/bundles/SAM.jar
6	ACTIVE	file:C:/jes2.0/bundles/SRM.jar
7	ACTIVE	file:C:/jes2.0/bundles/UIM.jar

圖 5 bundle 執行畫面

最初的開發平台上是以 PC 實作出本研究的基本雛型架構。雛型開發時所使用的套件如下：

- +OS：RedHat Linux 7.2。
- +JDK：1.4.1。
- +Jini：1.2.1。
- +Service Framework：JES2.0。

事實上，Jini 只是用來規定彼此通訊的介面。即是於實作當中，除定義 Jini API 中的 Discovery 部份之外，大多都只是 Java Interface，在 client 與 service provider 之間完全沒有限定必須使用何種方法來溝通。目前由 Sun 實作的 lookup service 所提供的通訊介面是使用 RMI 的方式，所以本文在 client 與 service provider 之間的溝通也採用相同 RMI 的方式。

本研究在功能模組設計可分為 org.stu.osg.SRM、org.stu.osg.GUI、org.stu.osg.SAM、org.stu.osg.DEV 以及 org.stu.osg.LSM 等五個 packages，本節將介紹這些 packages 中所包含應用程式的功能定義。圖 6 中顯示出各種服務模組相互間通訊的架構。

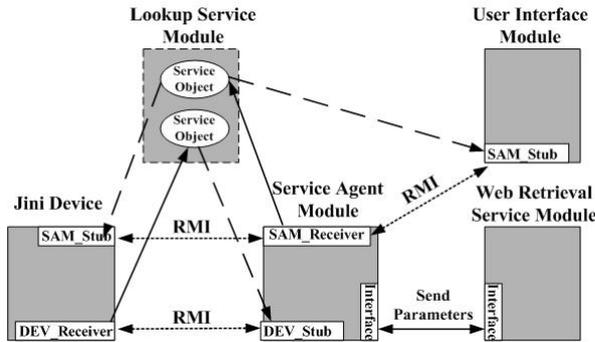


圖 6 本文開道器功能模組溝通架構

### 5.1 網路萃取服務模組之設計

網路萃取服務模組主要負責與外部 web services 和內部 FileSystem 溝通。在網路萃取服務模組的 service bundle 設計方面，本研究中將 org.stu.osg.SRM package 命名為 SRM.jar。此 jar 檔包含 SRM\_Interface 與 SRM\_Impl 應用程式。在 SRM\_Interface 中介面程式分別定義 parserWebContent、getServiceList 和 getWebResources 三個 methods，其功能分別說明如下：

1. parserWebContent: 指定 web services 所提供的網頁做內容標籤之分析，萃取網頁中符合要求的標籤，如 <img 開頭的標籤，最後將符合要求標籤的字串 (Service\_URL) 寫入至 FileSystem 裡的 ServiceList 檔，讓 ServiceList 隨時保持更新。
2. getServiceList: 此 method 主要是用來讀取 FileSystem 裡的 ServiceList 檔案內容，輸入條件值為 null，回傳資料型態為 Hashtable。
3. getWebResources: 根據使用者介面模組替 Jini 裝置選取服務傳遞來的參數向外部 web services 抓取資源，此方法為 getWebResources(Hashtable Device\_properties, String Resources\_URL)，輸入資料型態為 Hashtable 與 String，回傳資料型態為 BufferedInputStream。

如圖 7 所示，在 Service Framework 的運作過程中，本服務模組中的 Service bundle 會 export org.stu.osg.SRM. package 和註冊 SRM\_Interface 服務至 Service Framework 上。另外，SRM\_Impl 該應用程式主要是實作 SRM\_Interface 定義的介面，向外部 web services 萃取資源以及 ServiceList 檔的取得及更新都是由該 SRM\_Impl 應用程式所執行。

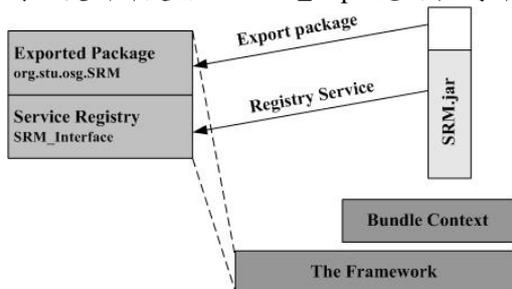


圖 7 export package 和註冊 SRM\_Interface

### 5.2 服務代理模組之設計

服務代理模組主要負責接受 Jini 裝置與使用者介面模組的訊息。當 Jini 裝置要求服務時，本模組將與網路萃取服務模組連繫。因此，在整體架構上本模組扮演著三者相互溝通的代理者。在本研究中 org.stu.osg.SAM package 命名為 SAM.jar，該 jar 檔包含 SAM\_Interface、SAM\_Impl 與 SAM\_DEV\_App 等應用程式。SAM\_Interface 遠端介面程式分別定義下列幾個 methods：

1. connectTo: 此 method 主要是讓 client 端的 Jini 裝置可以連線至本模組，並將 Jini 裝置的所有狀態和屬性值以 HashMap 型態儲存起來，使得讓服務代理模組可得知目前有哪些 Jini 裝置已連線，並可知道每個 Jini 裝置的狀態和屬性值。輸入條件值為 null，回傳資料型態為 boolean。
2. getServiceList: 主要透過網路服務萃取模組所提供的 SRM\_Interface 去呼叫 SRM\_Impl 應用程式裡的 getServiceList method，將讀取 FileSystem 裡的 ServiceList 檔案內容回傳至本模組。輸入資料型態為 null，回傳資料型態為 Hashtable。
3. getWebResources: 同樣是透過網路服務萃取模組所提供的 SRM\_Interface 去呼叫 SRM\_Impl 應用程式裡的 getWebResources method，將所取得的資料回傳至本模組。方法為 getWebResources ( Hashtable Device\_properties, String service\_URL)，輸入資料型態為 Hashtable 與 String，回傳資料型態為 BufferedInputStream。
4. setWebResources: 主要將從網路服務萃取模組所回傳的資料透過該 method 傳遞至 Jini 裝置上。方法為 setWebResources (Hashtable Device\_properties, BufferedInputStream input)，輸入資料型態為 Hashtable 與 BufferedInputStream，回傳資料型態為 boolean。
5. getDevicesStatus: 當 Jini 裝置連線至服務代理模組時，本模組會將 Jini 裝置的所有狀態和屬性值以 HashMap 型態儲存起來，因此當使用者介面模組透過本模組所提供的遠端介面來呼叫該 method 時，則會回傳所有 Jini 裝置的狀態和屬性值。輸入條件值為 null，回傳資料型態為 HashMap。
6. setDevicesStatus: 接受由使用者介面模組所傳遞過來的控制狀態訊息，將該控制狀態訊息回傳至所指定的 Jini 裝置，變換 Jini 裝置的狀態值。方法為 setDevices( Hashtable devices\_properties )，輸入資料型態為 Hashtable，回傳資料型態為 boolean。

本開道器中服務代理模組分別執行著 device 的 RMI client - DEV\_Stub(SAM\_DEV\_App) 與 SAM 的 RMI Server - SAM\_Receiver(SAM\_Impl) 具備

RMI 功能的通訊程式。DEV\_Stub 主要是建立與 Jini 裝置的連線。它將服務代理模組的訊息傳遞至 Jini 裝置上和接受 Jini 裝置所傳遞的訊息。SAM\_Receiver 則建立與使用者介面模組的連線，主要處理彼此間的訊息傳遞。然而與網路萃取服務的溝通方面，主要是藉由從 Service Framework 中 import osg.stu.org.SRM 的 package，藉由網路萃取服務模組的 Interface 轉而呼叫 SRM\_App 應用程式來完成所有的動作，相關動作流程如圖 8 所示。

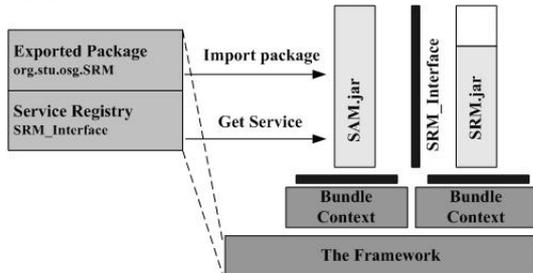


圖 8 import package 和取得服務

### 5.3 搜尋服務模組之設計

在搜尋服務模組方面，主要是將服務代理模組和 Jini 裝置所提供的遠端介面以相關的應用程式分別註冊至本模組的 service group 與 device group 中，此模組的執行畫面如圖 9 所示。



圖 9 搜尋服務模組執行畫面

### 5.4 使用者介面模組之設計

使用者介面模組主要是讓使用者可以透過該模組控管內部網路中所有的 Jini 裝置。在本研究中 org.stu.osg.GUI package 命名為 UIM.jar。該 jar 檔包含 GUI\_App 與 GUI\_Servlet 應用程式。GUI\_App 應用程式除負責從服務代理模組取得所有 Jini 裝置的狀態與屬性值外，也負責接受從 GUI\_Servlet 所傳送的控制訊息，再將此控制訊息轉而傳送至服務代理模組。至於 GUI\_Servlet 則是從 GUI\_App 所接受到訊息以圖形介面化的方式呈現在 web 畫面，以讓使用者可直接看到目前家庭內部所有 Jini 裝置的狀態。

本研究中，此模組藉由遠端介面 SAM\_Interface 執行著 SAM 的 RMI client - SAM\_Stub(GUI\_App)，它主要與服務代理模組的 SAM\_Receiver(SAM 的 RMI server)做連線，負責處理彼此間的溝通訊息。

### 5.5 Jini 裝置之設計

本研究的 Jini 裝置目前主要是以 Java 撰寫應用程式來模擬。每個 Jini 裝置都賦予自身狀態(例

如：裝置目前是開機或關機的狀態等)和屬性值(例如：裝置的版本等)的 properties 檔案，以用來儲存 Jini 裝置的所有狀態。此一檔案的內容包含 device\_ID、device\_NAME、device\_STATUS、device\_VERSION 和 device\_Remote object 等屬性。此部份包含了 DEV\_Interface、DEV\_Properties 和 DEV\_Impl 等應用程式，其中 DEV\_Interface 是個遠端介面，分別定義 SET\_DeviceProperties 以及 GetService 兩個 methods，其功能說明如下：

1. SET\_DevicePropertie：主要是用來變更 Jini 裝置目前的狀態。使用者可利用使用者介面模組下達控制指令，經由服務代理模組會將該指令傳至 Jini 裝置，以達到控制的目的。此方法為 SET\_DeviceProperties(Hashtable DeviceProperties)，輸入資料型態為 Hashtable。
2. GetService：透過服務代理模組呼叫網路萃取模組的 method，以將 Jini 裝置所指定的服務給萃取回來回傳到服務代理模組。隨後，再依照 Jini 裝置的類型轉換成符合 Jini 裝置可讀取的資料型態回傳至 Jini 裝置。方法為 GetService(Hashtable DeviceProperties)，輸入資料型態為 Hashtable，回傳資料型態為 BufferedInputStream。

DEV\_Impl 應用程式主要是實作 DEV\_Interface 定義的介面。DEV\_Properties 應用程式所定義的 read() method 主要是用來讀取 Jini 裝置的 properties 檔案內容，並將讀取的內容儲存至 Hashtable 中。輸入資料型態為 null，回傳資料型態為 Hashtable。

在本研究中，Jini 裝置同時執行著 device 的 RMI server - DEV\_Receiver(DEV\_Impl)與 SAM 的 RMI client - SAM\_Stub(DEV\_Impl)。其中 DEV\_Receiver 主要是建立與服務代理模組的連線和接受服務代理模組所傳遞的訊息，SAM\_Stub 則是負責建立與使用者介面模組的連線，並將 Jini 裝置本身的所有狀態與屬性值傳遞至網路萃取服務。

## 六、結論

本研究主要探討 Jini community 與 web services 兩者相互連結後所延伸出的設計議題。此一連結的方式結合到家用開道器中將使未來的資訊家電的應用更為便利。本設計開道器的特色在於除了具備原有整合家庭內部自動化裝置的功能外，它同時也強化開道器服務層面，使內部網路的 Jini 裝置可以透過本文開道器上的嵌入式服務模組存取網際網路上各式 web services，以取得裝置所需的服務。這樣的開道器將不只可以應用於家庭網路中，更可以提供作為建築物與辦公場所網路接取設備使用。相信最後終將使得所有資訊家電產品，在不需要使用者的干涉下，都可以在網路上相互通訊以取得所需要的服務。

在未來的發展中，隨著各種功能的家庭資訊裝置不斷地出現，內部網路的設置需求也逐漸轉為殷

切，但由於家庭控制網路目前尚未完全標準化，以目前應用導入的成本來看，家庭中已經建構完成的網路通常為電力線網路與數據資訊網路兩種。因此，在本研究架構開發未來可供支援 LonWorks 網路與 Ethernet 的連接介面，使得家用網路不需重新佈建即可經由開道器與網際網路相連接，並訂閱各式的網路服務功能。

### 七、誌謝

本研究之經費承蒙國科會(計劃編號 92-2815-C-366 -003 -E)贊助。

### 八、參考文獻

- [1] 陳邦正 “以 OSGi 為基礎之嵌入式 LINUX 網路服務開道器設計與實現”，國立交通大學碩士論文，民國 90 年
- [2] 施宏政 “在無線家庭網路環境下支援彈服務之住宅型開道器設計及實作”，國立交通大學碩士論文，民國 90 年
- [3] 鄧煥友 “以 Jini 為基礎並達成服務同步的代理人系統”，國立中央大學碩士論文，民國 90 年
- [4] D. Marples and P. Kriens, "The Open Services Gateway Initiative: An Introductory Overview," IEEE Commun. Mag., Vol. 39 no. 12, Dec. 2001.
- [5] D. Valtchev and I. Frankov, "Service gateway architecture for a smart home," IEEE Commun. Mag., vol. 40, pp. 126-132, Apr. 2002.
- [6] K.Wacks, "Home systems standards: achievements and challenges." IEEE Commun. Mag., vol. 40, pp. 152-159, Apr. 2002.
- [7] L. Gong, "A software architecture for open service gateways," IEEE Internet Comput., vol. 5, pp. 64 -70, Jan.-Feb. 2001.
- [8] Takeshi Saito, Ichiro Tomoda, Yoshiaki Takabatabke, Junko Ami and Keiichi Teramoto "Home Gateway Architecture And Its Implementation," IEEE Trans on Consumer Electronics, Vol. 46, No. 4, Nov 2000.
- [9] Open Services Gateway Initiative, "OSGi service platform release 2, " Version 1.1, Oct. 2001.
- [10] Cahners In-Stat, <http://www.instat.com>
- [11] PoweriA, <http://www.iantech.com.tw/>
- [12] Remote Method Invocation, <http://java.sun.com/rmi/>
- [13] Sun Microsystems, "The connected home powered by Java Embedded Server software white paper," 2001. <http://www.sun.com/software/embeddedserver/whitepapers/dot.com.home.pdf>
- [14] Sun Microsystems, Jini Network Technology, <http://www.sun.com/jini/>
- [15] Sun Microsystems, Java Embedded Server, <http://www.sun.com/software/embeddedserver/>
- [16] The Community Resources for Jini Technology,

- [17] <http://www.jini.org>  
H. M. Deitel, P. J. Deitel, S. E. Santry "Advanced Java Platform-How To Program-", DEITEL
- [18] Kirk Chen, Li Gong, "Programming Open Service Gateways with Java Embedded Server(TM) Technology", Addison Wesley.