行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※

※ 全球分散式物件網路上的個人資訊助理 ※

※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：☑個別型計畫　□整合型計畫

計畫編號：NSC 89－2213－E－032－005

執行期間：88 年 8 月 1 日至 89 年 7 月 31 日

計畫主持人：鍾興臺

共同主持人：施國琛

本成果報告包括以下應繳交之附件：

□赴國外出差或研習心得報告一份

□赴大陸地區出差或研習心得報告一份

□出席國際學術會議心得報告及發表之論文各一份

□國際合作研究計畫國外研究報告書一份

執行單位：淡江大學資訊工程所

中 華 民 國 89 年 10 月 20 日

行政院國家科學委員會專題研究計畫成果報告

全球分散式物件網路上的個人資訊助理

Personal Information Agents on the Object Web

計畫編號：NSC 89-2213-E-032-005

執行期限：88 年 8 月 1 日至 89 年 7 月 31 日

主持人：鍾興臺　淡江大學資訊工程所

共同主持人：施國琛　淡江大學資訊工程所

一、中文摘要

本計劃主要目的在設計一個符合單一個人資訊需求的自動化軟體助理(Personal Information Agent，簡稱 PIA)，以減輕使用者需費時費力在網際網路上蒐集資訊的困擾，並希望能提供使用者更高品質、更專業化的資訊服務，以因應未來的全球分散式物件網路(Object Web)的需求。

關鍵詞：軟體助理、Object Web、Web 資訊擷取

Abstract

The main purpose of this project is to design a personal information agent to alleviate the problems users need to collect information from the Internet, in addition, to provide users with the high-quality professional information services in respond to the future trend of Object Web.

Keywords: Software Agents, Object Web, Web Information Retrieval

二、緣由與目的

現今的社會型態因為資訊科技的進步下已經有了相當大的改變，以往只有專家與科學家使用的各種網際網路技術現今也被廣泛的利用在生活的各個層面。資訊量成長在軟硬體的發展下也成指數倍數的成長，但是問題也隨著資訊量的大量膨脹而產生，如何在超過人們負荷的大量資訊中尋找真正有用的資訊，加上自動化且方便的處理機制，預先過濾處理龐大的資訊並加以整理，並透過簡易的介面，減少在使用上繁瑣的操作，並建立可以擴充的機制，這是本文所要探討的主要重點。

PIA 是以物件化的介面來封裝改寫傳統分散的軟體功能模組並加上可以動態載入的功能，並以此架構建立與使用者的工作環境模式。其基本功能特性包含：(1)提供使用者熟悉的 Web 瀏覽器介面，來操作 PIA。(2)提供工作編輯器，讓使用者編定所需要自動化的工作；而 PIA 能在使用者訂定的時間執行該工作，並能主動回報各工作的執行結果。(3)以更簡易有效之操作介面，支援外加傳統網頁搜尋，新聞群組搜尋，電子郵件檢查，檔案傳輸查詢等功能。(4)可以藉由其他主機上的 Web 瀏覽器在遠端操作控制 PIA。

為了提供使用者更高品質、更專業化的資訊服務，及因應未來的分散式物件網路(Object Web)的需求，PIA 更進一步支援軟體物件內嵌介面平台(plug-in platform)，允許：(1)不同的服務提供者提供各自的專屬介面，內嵌至 PIA 中。使用者依據個人喜好設定好資訊需求後，PIA 可透過內嵌介面定時至服務者端下載相關資訊，使用者因此能隨時獲得新的、即時的資訊，避免了使用者必須主動尋找資訊、費時費力的不便。(2) CORBA 等分散式計算的需求，PIA 本身提供的是一個自由度相當高的作業平台，只要撰寫符合內嵌的軟體介面，PIA 也可以是分散式計算環境中的一員。(3)發展中介軟體(middlewares)，可依據上內嵌介面的規格，連並不同網站或 CORBA/DCOM 物件，來擷取、過濾、整合資訊，解決在 Web 上資訊整合不易的困擾。由於對使用者而言，此是完全透明的，也避免了使用者操作上額外的負擔。

軟體助理(Software Agent)的技術[3,4,6,9,10]在近幾年慢慢的受到重視，主要起因為軟體助理相較於以往一般軟體必須有更高的對於環境的適應與自我應變能力，也就是說要適應網際網路環境的多變及與其他各種不同網際網路服務與軟體助理協同完成所要處理的工作。而個人化的軟體助理則是完全以單一個人需求與使用習慣的為目標所設計的軟體助理，建立一個可以協助個人在網際網路中更方便來獲取所需資訊的工作平台。

目前一般個人在網際網路上的使用通常屬於主動式的操作習慣，也就是說幾乎所有的動作都必須由使用者下達一連串的指令與動作之後，電腦才能獲得使用者真正所需要的資訊。然而一般個人在網路存取習慣上有相當大的週期性，舉例來說，使用者通常每天都必須檢查電子郵件信箱才能知道有無新郵件，所以使用者必須每天重複打開電子郵件軟體與檢查電子郵件的動作，若使用者希望收到某位重要的人寄來的 E-mail 便立即通知使用者，以目前的一般軟體操作環境而言，除非使用者隨時使電子郵件軟體監視電子郵件信箱，而且使用者有必須在電腦前才能夠獲得通知。如果使用者無法隨時

隨地在電腦之前，幾乎無法立即通知使用者相關事件的發生，如重要電子郵件的寄達、系統管理者監控的伺服器發生問題等等。再者使用者並不能隨時隨地的在自己專屬電腦之前，也不應該限制使用者只能夠在 PIA 所在的電腦才能存取資源，所以只能在單一電腦存取的使用者介面與存取控制方式是不可行的，所以 PIA 在設計上必須考慮到資訊存取的簡易性與一般性。

三、結果與討論

3.1 系統需求與分析

依照上述，使用者必須能夠在任何地方皆能自由存取 PIA 的資源，以目前的技術與網際網路環境而言，最適合的使用者介面非 HTTP browser 莫屬，HTTP browser 是目前最成功的顯示與輸入資訊的介面，幾乎沒有作業系統不包含 HTTP browser，就連最近極為熱門的資訊家電(Information Appliance)的嵌入式作業系統通常也包含 HTTP browser 為使用者介面。當使用者端使用 HTTP browser，PIA 必須能夠產生互動的能力，為此，PIA 採用了下列的機制：(1)包含一般的 HTTP Daemon，(2)為了達到與系統硬體平台無關(Platform Independence)的要求，PIA 使用 Java 的技術來開發整個系統，包含 Java servlet 的相關技術[11,12]。

Java 雖然在效能上不及傳統的 C 與 C++，但是其語言在設計之初，即包含網際網路上存取的功能，符合 PIA 主要的需求。且整個系統若全部以 Java 來實作，也就是符合 Sun Microsystems 所提倡的 Pure Java 規範，只要硬體平台提供 JVM (Java Virtual Machine)，整個系統即可直接移植到硬體完全不同的平台。舉例來說系統是在 Windows NT 平台中開發，而在實際測試時卻是在 Linux 作業系統中作最後測試，最後可能在使用嵌入式 ARM 處理器的資訊家電中執行，這完全要拜 Java 的 platform independence 的特性所賜。使用 Java 的 PIA 能夠在未來的各種平台都可以正確的執行甚至包含未來的網路家電，甚至加入 Jini，成為未來家庭中控制 IA 的主控監控裝置，而 PIA 使用的 API 只有網路與檔案兩大類，只要 JVM 提供這兩大類機制，PIA 就可以在該硬體平台上執行。

3. 發展平台與資料存取使用者介面

目前在實作上，PIA 中有關 HTTP Daemon 與 servlet 部分的機制使用 Caucho 公司所開發的 Resin servlet engine [13]來建構。Resin 是一個 open source 的 servlet engine，相當小，全部系統只需要約 950 KB，而 PIA 改寫了 Resin 部分的程式碼以符合 PIA 的需要。因為上述的系統設計架構，PIA 還具備遠端控制的能力，使用者只要通過了系統認證之後，幾乎可以獲得所有對 PIA 的控制能力與所有資料的表現能力。

PIA 系統開發實作主要在 Windows NT 環境配合 JDK 1.3beta、Sun. forte 為主要的開發工具，而測試的平台除了 Windows NT/98 以外，還包含 Redhat Linux 6.1+CLE 0.9 與 blackdown JDK 1.2.2 兩種主要的測試平台。在 client 端部分，因為 PIA 的 HTTP Daemon 輸出的是正常的 HTML code，所以只要有 HTTP browser 就可以成為 PIA 的 client 端，並不需要特殊設定與調整。

3.2 系統模組

PIA 整個系統運作有幾個基礎的物件必須先加以描述：

(1) Task：描述儲存個別單項工作的詳細內容、結果與狀態。

(2) Plug-In Component (以下簡稱 PIC)：PIA 對各項特殊的工作都是由每一個 PIC 所處理執行，PIC 本身就是一個小小的執行單元，在實際實作時是使用 Java 的 Runnable interface 與 MultiThread 來建立起這個機制，PIC 在 PIA 系統裡是一個抽象類別(abstract class)，提供一組 meinber function 與 PIA 進行溝通，而處理所有 Task 的執行單元都是繼承 PIC 這個抽象類別並加以實作真正工作部分。

PIA 整個運作與 HTTP Daemon 十分相似(如上圖)，比較大的不同在於 PIA 會依據使用者所設定時間自動執行各種 Task，使用者也可以隨時觀看相關工作進行的進度。而在 PIA 接受的 HTTP request，可以分為幾類：

(1) 傳統靜態的 HTML request，這部分單純由 HTTP Daemon 直接處理。

(2) 對 PIA 內部的資訊或各個 PIC、Task 結果狀態加以查詢的類別的 request，而這部分的 request 由 HttpRequestHandler 向詢問的 PIA、PIC、Task 直些呼叫其狀態查詢的函式獲得。

(3) 對於個別 Task 所產生的結果加以存取的 request，這部分的 request 主要的做法類似傳統 HTTP rewritten 的方式來作個別 Task 最後結果的存取。

(4) 對 PIA 或特定 PIC 要求進行特殊命令或動作，這部分也是由 HttpRequestHandler 這個 servlet 來對 PIA 與個別 PIC 直接進行動作。

PIA 系統主要包含了下列幾個重要的模組來負責各部分機制的運作：

(1) HTTP Daemon

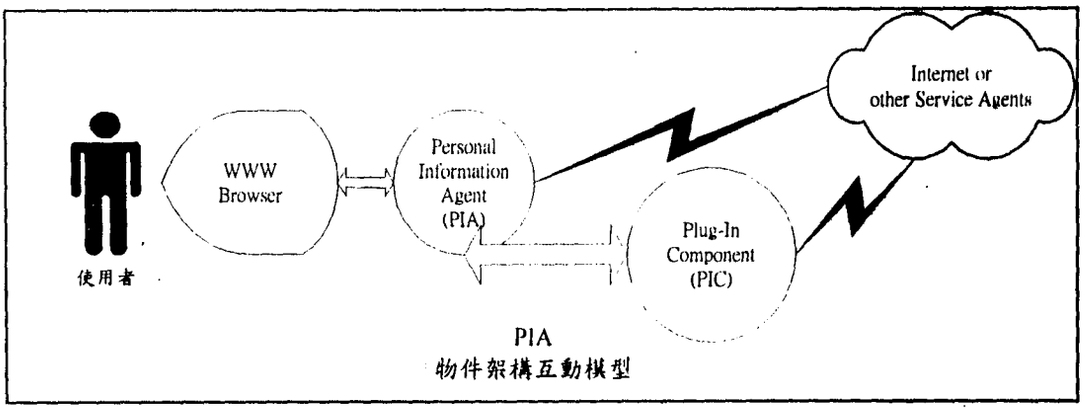
負責建立接受 client 的 HTTP request 的機制，處理部份靜態的回應與系統訊息，如果 client 的 request 屬於 servlet，則將 request 交由負責的 servlet 來回應。

(2) HttpRequestHandler

是 PIA 最重要的一個 servlet，是整個系統的 dispatcher 與檢查各種 request 的完整性與相關參數是否完備，並負責阻隔不合法存取與處理密碼認證等。

(3) URLHandler

提供 PIA 最基本的網際網路存取能力，包括簡易的 HTTP 資源抓取，FTP 檔案抓取，POP3 監視等最基本的服務項目。



PIA
物件架構互動模型

(4) TaskEditor
是一個 servlet 專門用於導引使用者更容易對 Task 作新增、修改與刪除的工作，由 client 的 HTTP request 並不能直接呼叫到這個物件，也是由 HttpRequestHandler 將 request 傳道此物件再行處理。

(5) TaskTrigger
根據使用者所設定的工作敘述，及預設的時間對 ServEngine 發出服務請求，這是 PIA 在資訊處理自動化的核心。TaskTrigger 對於 PIA 在執行時期來說是比較特殊的部分，因為 TaskTrigger 是單獨執行的且擁有自己的執行緒。使用者對於一些經常性的工作與需要自動化部分可由 TaskEditor 來描述欲建立的工作程序與執行時間。

(6) ServEngine
儲存所有與 PIC 與 Task 的資訊，所有對於 PIC 與 Task 的操作查詢也是由此物件負責處理。接受由 HttpRequestHandler 與 TaskTrigger 所提出的需求，並依據需求向系統提出完成需求所需要的資源並呼叫 URL Handler 完成該需求或呼叫相關的 PIC。

(7) Monitor
提供所上述物件紀錄所有動作的進行與紀錄錯誤的機制，對於系統動作與除錯有相當的幫助。

(8) 其他 PICs
目前已完成數個 PIC 的可以用於長時間監視其他電腦，或是其相關的服務是否有任何異常與測試系統運作的測試 PIC 等。

3.2.1 運作流程

PIA 可以算是設殊化且簡單化的 HTTP server，在靜態的資料的基本運作流程上與 HTTP server 相同，比較大的相異處在於利用 servlet 與 PIA 的物件如 ServEngine、TaskTrigger、URLHandler 等 PIA 系統管理與資料存取物件進行溝通與資料的存取，對於使用者所會提出的幾類 request 會透過下列流程加以回應:

(1) 靜態網頁的 request 由 HTTP Daemon 直接回應。
(2) 對於 PIA 內部物件資訊如 ServEngine、TaskTrigger、與系統資訊的查詢由 HttpRequestHandler 直接向該物提出查詢的要求並直接將結果回應給使用者。
(3) 對於 Task 的新增部分 HttpRequestHandler 必須先查詢 PIA 有多少 PIC 以註冊在系統中，並詢問使用者要建立何種 Task 並交由哪一個 PIC 來負責此 Task，然後由個別 PIC 提出個別 PIC 運作所需的各項參數並交由使用者加以填寫，而後再將所需的參數傳入 PIC 直接執行或傳入 TaskTrigger 加以紀錄，並依據所定義的時間或間隔加以產生 Task。
(4) Task 的刪除與修改就包含兩部分，分別是正在 PIA 系統執行的 Task 與紀錄在 TaskTrigger 中所的 Task，兩者的分別由 HttpRequestHandler 與 TaskEditor 兩個 Java servlet 加以負責這部分管理工作。

3.2.2 任務管理與自動化

Task 在 PIA 系統中是最基本的資料結構，系統最初起始後，系統並不會有任何的 Task 在執行的狀態，而 TaskTrigger 利用 Java 的 serialization 機制會紀錄儲存前一次 PIA 被關機前排定在 PIA 內的 Task，所以當 PIA 重新開始時並不需要作特殊的初始化動作。而使用者可以建立一個新的 Task 並強制 PIA 立即加以執行，或設定 Task 排入 TaskTrigger，而 TaskTrigger 會依據使用者設定的時間或執行間隔定期執行。在 PIA 中有許多物件都會直接參考操作 Task 物件，如 PIC 本身獲取該 Task 所指定的工作內容、ServEngine 進行 Task 的新增、刪除等，而 Task 物件在定義上使用 Java 的 Properties 物件來建立彈性紀錄 Task 內容的機制，所以雖然工作的內容與所要的參數都是由不同的 PIC 來執行的 Task，都可以正確的紀錄要完成工作內容。而 TaskTrigger 每隔 3 分鐘檢查一次內部的 Task 表單是否有符合條件的 Task 需要被啟動，如果符合就將 Task 交由 ServEngine 排如 PIA 系統中，轉交給 PIC 開始執行。

3.2.3. 遠端操作與控制

PIA 因為在設計使用場合與使用者介面的考量下，在設計架構上採用了使用 HTTP Daemon 的來作為 PIA 處理輸入、輸出的，也造成了 PIA 在遠端控制上的能力，這部分幾乎不需要再加以特殊設計，而 PIA 在執行電腦端也提供了簡易的 on-line monitor 提供使用者在 PIA 執行電腦觀察 PIA 執行的狀態。

3.2.4. 安全性

因為 PIA 在架構使用 HTTP daemon 的架構，難免就會在出現安全上的顧慮，在安全性的考量

下，PIA 同時使用密碼認證與 Java Servlet Session 的兩種機制來保護使用者的資訊不會直接暴露被直接存取的危險，首先使用者必須先通過系統的使用者 id 與密碼的認證，通過之後系統將會記錄通過認證的該次 Session key 加以追蹤使用者的使用狀態，並以此控制使用者在整個操作上的流程與系統反應，當使用者結束該次連線以後必須重新再認證。這部分也是 PIA 修改部分 Caucho Resin 提供的 source code 的主要原因，因為 Resin 所提供的 HTTP Daemon 功能與安全權限設定上十分有限，所以在安全性考量下有些需要改進的地方，舉例來說就移除了 HTTP Daemon 的 Directory listing 功能以增加其安全性。

3.2.5. Plug-In 機制設計概念

使用者對於某一方面可能會有需要特殊的服務或計算，例如股市資料的分析取得、分散式計算等等特殊話的需求，對此 PIA 必須提供簡易擴充功能的介面與機制，服務提供者只要依據相關 PIC 格式撰寫 PIA 的相關的功能模組就可以為 PIA 所呼叫操作。除獲得資料以外，並可利用 CORBA 等分散式計算的機制獲得額外的計算能力與完成特殊需求。另外資訊服務專業與高品質也可以利用 Object Web 的運作模式下來提供：例如網際網路提供相當大量的資訊但是其中也混雜許多使用者不需要的資訊，利用 PIA 與特殊 software agents 的合作，集合一群有相同特質的使用者形成一個虛擬社群(Virtual Community)，利用使用者共同參與討論與資訊評分機制來增加資訊的品質。實際現在的軟體有提供上述部分的功能如網頁的分類搜尋(IQ 搜尋家)、資料的離線抓取與自動化(Teleport)等，但是 PIA 在這方面著重的是如何整合這些資訊幫助使用者與建立一個服務連結介面，讓不同的服務可以在相同的介面下運作，而且可以任意的增加與刪減這些服務，或加以排程協助使用者完成工作。遠程來看當電子商務漸漸成熟，PIA 彈性的 plug-in 介面就可已變成電子商務的交易與查詢介面。

PIA 載入 PIC 到系統除了利用利用 Java 在電腦主機端的 class loader 機制外，也可以直接透網路如 HTTP 或 FTP 動態載入 PIC 進到 PIA 來執行，如此一來將提供更加有彈性的管理機制，服務提供端 update 更新版本的 PIC 就不需要使用者在額外進行更新的工作。

經由 plug-in 的方式增加軟體的附加價值與彈性在實際軟體與實作介面上可以說是相當普遍的，例如 Netscape 的 WWW 瀏覽器中的 plug-in 與微軟的 COM(Common Object Model)規格。如在 Netscape 的 WWW 瀏覽器上利用接收檔案的副檔名呼叫相對應的 plug-in 程式，可以播放串流音訊(Stream Audio)；而在微軟的 COM 可以讓程式設計師利用相同的 query interface function 獲得操作物件的指標，進而獲得物件的資料與操作物件成員函示的能力。PIA 在功能擴充上的作法也是引用相同的概念，以建立一個可以擴充功能的作業平台。

四、計畫成果自評

本計劃實作了 PIA 的雛形，完成了預期 90%的進度，其成果發表如[2]。目前在進行的是加入傳呼的能力，讓 PIA 中之 PIC 都可將個人所關心的訊息隨時傳呼，對使用者而言是非常省時省力、有效率的做法。另外，PIA 對外的通訊機制，也在構建中，希望能達到群組通訊的能力。未來，也可將 PIA 嵌入於控制設備中作為主控的軟體。綜觀 PIA，深具發展潛力。

五、參考文獻

1. 鍾興臺、余國棟，全球分散式物件網路上的個人資訊助理，2000 網際網路與分散式研討會，May 12，2000。
2. 余國棟，全球分散式物件網路上的個人資訊助理，碩士論文，淡江大學，2000。
3. 郭信義，網際網路上的個人資訊助理，碩士論文，淡江大學，1998。
4. Hsing-Tai Chung, A Study on Intelligent Web Agents, *NSC project, 87-2313-E-032-014.*
5. Scott M. Lewandowski, Framework for Component-Based Client/Server Computing, *ACM Computing Surveys*, Vol.30, No.1, March 1998.
6. Imran Khan and Howard C. Card, Adaptive Information Agents Using Competitive Learning, *Journal of Network and Computer Application*, 21, p69-89, 1998.
7. David M. Chess, Security Issue in Mobile Code Systems, *HICL IBM Watson Research Center*, 1998.
8. Danny B. Lange, Mitsuru Oshima, Programming and Deploying Java Mobile Agents with Aglets, *Addison-Wesley*, 1998.
9. Jacques Ferber, Multi-Agent Systems, *Addison-Wesley*, 1999.
10. Richard Murch, Tony Johnson, Intelligent Software Agents, *Prentice Hall*, 1999.
11. Danny Ayer, Hans Bergsten etc., Professional Java Server Programming, *Wrox Press*, 1999.
12. Jason Hunter with William Crawford, Java Servlet Programming, *O'reilly*, 1999.
13. Caucho Resin Servlet Engine, http://www.caucho.com