智慧型服務之網路伺服架構

丁培毅
海洋大學資訊科學系
基隆市北寧路二號
pyting@cyber.cs.ntou.edu.tw

蘇子安
海洋大學資訊科學系
基隆市北寧路二號
b86051@cyber.cs.ntou.edu.tw

摘要

近年來各式辦公室自動化產品、資訊家電產品、以及無線通訊產品進步快速，在區域網路上共享各種服務的設備因而大幅增加。這些設備所共享的網路服務多半需要特殊或是大量的軟硬體資源，有的服務可以透過批次處理的模式來完成，有的則需要透過頻繁交換資料的交談方式來完成，另外使用服務與提供服務的設備可能動態加入或離開，單純的客戶端/伺服端架構面對這樣複雜的要求時，服務品質很容易下降，甚至無法順利完成，因此在本研究中嘗試以監督式的網路伺服架構來輔助傳統的客戶端/伺服端服務模型，使得分散式應用系統中服務的整體效能提高，使得多個需要使用大量頻寬的視訊、音訊智慧型服務可以在區域網路上實現。在這個服務模型中，我們設計了一個服務控管中心，它可以協調眾多的服務要求設備與伺服器間的互動，充分發揮分散式系統的優點，根據各個伺服器的運作特性與服務的本質，適當地分配工作到各個提供服務的伺服器上，除了滿足個別客戶端不同的服務品質要求，也能提高整體網路及伺服器的運作效率。

關鍵詞：文字轉語音系統，智慧型服務系統，分散式系統。

一、前言

隨著區域網路與無線網路技術的發展與成熟，結合網路上遠端資源來運作的服務日新月異，像是線上購物、電子報、網路遊戲、電子字典、資料搜尋、分散式資料庫等等，這些服務在網路資源的運用上可以分為鬆散式與緊密式結合兩大類，其中緊密式的結合通常在伺服端與客戶端之間有密切的交談協定並且傳遞大量的資料，鬆散式的結合則藉由簡單的輸入、輸出資料來定義各個服務。另一方面如何利用分散式系統的概念，整合各個伺服器，提昇整體效率也是非常重要的問題[5]。在本研究中，我們主要考量緊密式結合的智慧型網路服務：例如語音合成、辨認、文件分析、視訊分析、多媒體教學、視訊廣播等等服務，在各種服務並存時，如何能同時應付大量的工作、讓使用者能夠得到滿意的服務，不但完成使用者交付的任務，並且滿足使用者在服務品質上的要求，減少使用者的等待時間，提昇後端服務伺服器的運用效率，本研究中採用分散式系統架構，透過監督式的網路服務控管中心來達成上述目的。

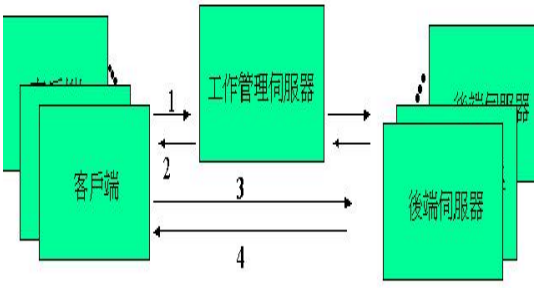
我們將依照系統的實作架構，在第二節中介紹系統整體的運作機制，在第三、四節詳細地介紹系統元件的設計概念與內部的運作，第五節分析本系統與其它系統架構(如 Jini，TSpaces 等)之間的差異，第六節則是結語與未來工作。

二、系統運作基礎機制

(一) 系統元件及運作流程

如圖一所示，本系統在運作時由三類主要成員參與：

1) 客戶端：發出服務請求的應用程式，在範例實作中是一個能夠接收電子郵件並且能夠以中文將郵件內容朗讀出的應用程式。
2) 工作分派與管理伺服器：是本系統的核心元件，其功能包括負責接受各種服務請求、指派後端伺服器工作、管理後端伺服器、分析後端各個伺服器的運作狀況、預測網路上資料流量、調整工作的分派等等。
3) 後端伺服器：提供各種不同的智慧型服務，以滿足客戶端的要求，在本研究中實作一提供中文文字轉語音的智慧型服務。



圖一、系統架構圖

圖一中所標示的步驟 1 至步驟 4 是客戶端要求服務的流程，每項步驟所包含的基本動作如下：

1. 客戶端向工作管理伺服器發出請求，並送出有關請求之服務的相關訊息，包括服務種類，基本參數，(例如在本實作中是要求文字轉語音的服務，以及需要轉換為語音的文件之長度)，以及使用者對於服務品質的基本要求。
2. 工作管理伺服器查詢其內部關於此項服務目前的資訊，考量合格伺服器的工作量與使用者的品質要求後，挑選一適當的後端伺服器，將此伺服器的服務提供資訊回應給要求服務的客戶端應用程式。
3. 客戶端應用程式依據從工作管理伺服器獲得的資訊，連結建議的後端服務伺服器，將工作交予指定的後端伺服器處理，本系統中是將電子郵件的內容送達文字轉語音的伺服器。
4. 後端服務伺服器在收到工作請求後，根據該項服務所定義的協定和客戶端進行連線並開始服務，將結果送回客戶端應用程式，完成工作後並結束服務協定，本系統中後端伺服器將文字內容轉換為對應的語音訊號送回客戶端應用程式播放。

在圖一中有兩個未標示的箭頭，這兩個箭頭代表工作管理伺服器與後端服務伺服器之間，在該服務所定義的服務協定之外的額外訊息溝通，其內容分別為：

a .後端服務伺服器在啟動或關閉時，必須向工作管理伺服器報告其狀態。

b.工作管理伺服器會定時向各個後端服務伺服器查詢工作負載之統計資訊，後端服務伺服器將所要求的資訊回報給工作管理伺服器，以完成統計資訊的登錄。

以上所描述的是以網路通訊為主的幾項主要工作，在各個服務伺服器以及客戶端應用軟體還有很多對應的工作需要一一執行，將在下兩節逐一介紹。

(二)基本通訊格式

本系統中訂立兩種基本的通訊資料封包格式：

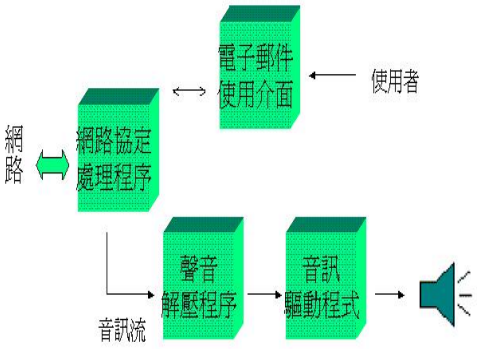
1.命令封包：此封包主要是運載區分各種服務要求之命令，型態為整數，系統內各項服務都會定義這種形式的封包。

2.資料封包：此封包主要為運載服務相關的資料，其內又分為兩段，第一段為資料長度，單位為位元組，型態為整數，第二段為真正的資料，內容隨不同服務而定，基本單位為位元組，位元組的個數即為第一段所記載之資料。

三、客戶端應用程式與後端伺服器

以下我們以一個語音合成的智慧型服務配合電子郵件收發的應用程式為範例來說明整個系統的架構，這樣子的一個應用系統在辦公場合、會議場所中非常容易出現：老闆帶著隨身的 PDA，透過無線區域網路的連結收發電子郵件，並且利用區域網路上所提供的語音合成服務來將文件轉換為聲音，充分地運用區域網路中提供的資源；場景稍微轉換一下，也可以由瀏覽器中將網站上定義好的資料或是新聞性資訊轉換為語音訊號輸出。

(一)客戶端應用程式



圖二、客戶端應用程式架構圖

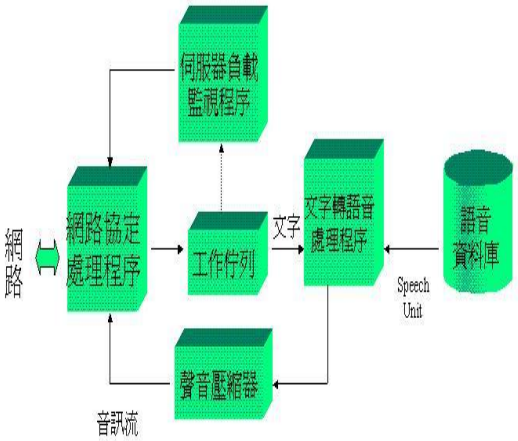
如同前文所提到的，客戶端應用程式是服務運作流程的起點，它最主要的工作就是根據使用者的操作來發出服務的請求，如圖二所示，客戶端是一個收發電子郵件的應用程式，經由 POP3 協定收下電子郵件之後，在區域網路上尋找工作管理伺服器，在該伺服器上要求所需要的語音合成服務，工作管理伺服器經過整體的考量後將適合的語音合成伺服器回應給客戶端應用程式，然後應用程式依循上節所述的程序直接向語音合成伺服器要求服務。

實作上當多人在同一區域網路上要求服務時，語音資料量可能相當龐大，後端伺服器在運算量允許的狀況下可以提供資料壓縮的選項，因此語音資料傳送回來的時後可能已經過適當的壓縮，必須先經過解壓縮，才能順利透過客戶端平台上聲音播放的功能來播放出合成的語音。

(二)後端服務伺服器

後端服務伺服器在系統中佔有重要的地位，客戶端的要求需要靠它們來完成，在實作時我們假設每個後端伺服器一次只能處理一件工作，但是若有多件以上的工作分配到同一台伺服器，則採取先來先服務(FCFS)的機制，其餘的工作就先暫存起來，如此在處理工作分派的機制時可以較為簡化，至於一次可以處理多

個工作的伺服器基本上也可以用多個伺服器的模型來取代。

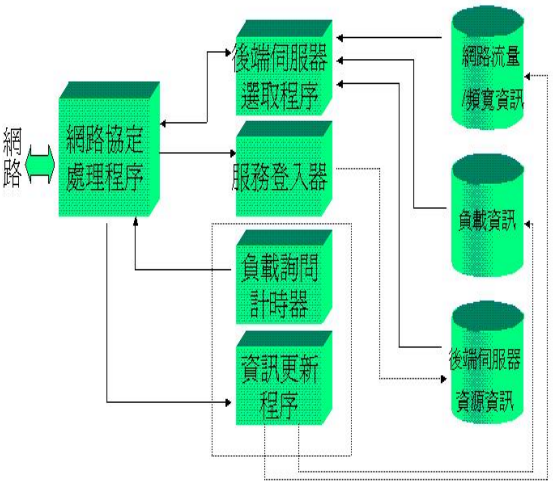


圖三、後端服務伺服器架構圖

如圖三所示，一個語音合成服務伺服器由五個元件加一個資料庫組成，網路協定處理程序依照訂立的協定，傳輸或接送資料，伺服器負載監視程序負責當工作管理伺服器發出詢問負載資訊時，計算並回報當時伺服器的負載狀況，工作佇列則如同前述，因為系統假設後端伺服器一次只能處理一件工作，所以其餘的工作先暫存起來，聲音壓縮器則負責將音訊資料作適當的壓縮，以促進網路頻寬的有效運用。聲音壓縮器是一個可以調整的模組，如果服務伺服器的工作量不大、網路的交通擁擠、同時客戶端有足夠的運算能力的話，就可以做複雜一些的壓縮，反之如果上面三點之一沒有達到的話，就不需要太複雜的資料壓縮，這些都是可以動態由工作管理伺服器來調整的，目標是使各個使用者可以滿意，同時使系統整體的運用效率提昇。

四、工作管理伺服器

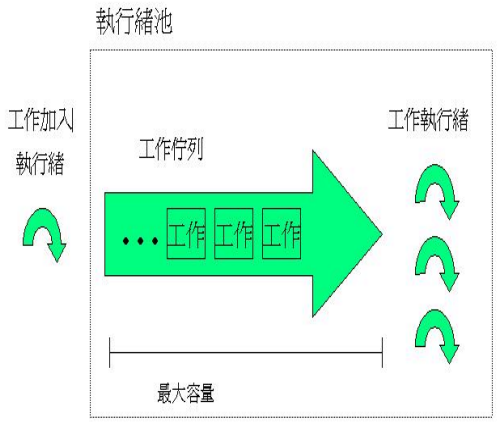
工作管理伺服器在整個系統中是扮演管理、調節的角色，從後端服務伺服器的登入、登出，客戶端要求服務，取得後端服務伺服器的負載資料都必須透過它來協調，在各項工作當中，工作管理伺服器最主要的任務是平衡後端服務伺服器們的負載以及維持網路上資料傳送量不要超過臨界點，使各個服務伺服器以及區域 (無線) 網路能發揮最大的效能，假設工作管理伺服器不存在，或是運作不正常，使用者自行挑選後端服務伺服器的話，就有很大的機會造成一台伺服器有做不完的工作，而另一台卻無事可做的情形，或是造成網路頻寬不足以負荷所有的要求的狀況。工作管理伺服器可以依據使用者的要求(例如：等候服務時間最短)，服務的特性 (視訊/音訊/文字)，要求服務的數量(視訊長度，品質)來選擇適當的後端服務伺服器，平衡各後端服務伺服器間的負載。



圖四、工作管理伺服器架構圖

參考圖四，工作管理伺服器由五個元件及三個資料庫組成，其中網路協定處理程序依照訂立的服務運作協定，傳輸或接送資料。後端服務伺服器選取程序從三個資料庫中取得相關資訊，包括伺服器基本能力(例如中央處理器的工作頻率)，有關工作的相關資訊(本實作中包括電子郵件的大小，單位時間之工作完成量)，還有網路以及負載的狀況，將這些資訊加以計算，所得結果與使用者的要求經過加權後，再算出最後的結果，依據此結果選擇出適當的後端服務伺服器[4, 6]。服務登入器負責處理後端服務伺服器登記其服務及登入登出的事務。負載詢問計時器依照事先定好的週期，對各後端服務伺服器發出回報負載資訊的要求。資訊更新程序負責將各後端伺服器回報的負載以及其他資訊存入資料庫中。後端伺服器資源資訊資料庫存放各種服務相關的靜態資訊，以便伺服器選取程序能夠依照適當的規則來選取服務伺服器。負載資訊資料庫則存放各個登錄的伺服器的動態資料，網路流量/頻寬資訊資料庫存放著網路使用狀況的動態資訊。

在實作上，因為客戶端應用程式所發出的服務要求以及後端服務伺服器的登入和回報負載資訊，並不是一個個循序進來，而是隨機產生的，有可能在一短時間內爆出大量的工作，且這些工作需要網路上的溝通，可能會發生阻塞的情況，為了能同時進行多項工作，增加效率並縮短反應時間，可以用多執行緒來實作這個工作管理伺服器。但若有太多的執行緒產生，工作管理伺服器的資源可能不夠使用，所以必須限制執行緒的數量。實作上工作管理伺服器採用執行緒池 (Thread Pool) 的方法來實作，它不但可以控制執行緒的數量，更可重複利用執行緒，省去創造及刪除執行緒時的額外負荷。



圖五、執行緒池架構圖

執行緒池的工作模型其實就是生產者與消費者模型，生產者產生工作，將工作放入佇列中，消費者從佇列中取出工作進行處理，並且要考慮到工作佇列是客滿還是空的情況，以控制生產者及消費者的運作。

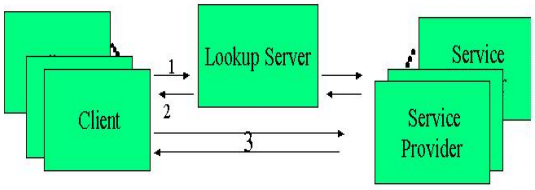
五、與其它系統架構的比較

(一)與 Jini 的差異

我們在設計與實作這個系統時，仔細評估過透過 Sun 的 Jini 架構來實作的可行性[1, 2, 3]，Jini 基本上強調能將網路應用軟體的開發由程序導向程式設計的層次提升到物件導向程式設計的層次，Jini 替所有的服務建立基本的物件服務並且在 Java 原有的 RMI 架構下將網路通訊協定完全包在高階的物件架構下，程式設計師可以輕易地越過各服務提供者與網路的藩籬，利用他人提供的服務，來建立自己的應用程式。

設計服務伺服器時可以專注於服務本身的功能而不需要煩惱網路之功能限制。

如圖六所示，Jini 系統架構可以簡單地描述如下：基本上由三部份組成，分別是 lookup server、service provider 及 client，而這三種元件利用三個主要的程序進行溝通，這三個程序分別是 discovery、join 及 lookup。



圖六、Jini 系統架構圖

圖六中所標示 1~3 的流程是 client 與 lookup server 溝通的流程，其所完成的工作如下：

1. Clinet 利用 discovery 程序找到 lookup server。
2. Client 利用 lookup 程序尋找它所要的服務。
3. Client 與 service provider 進行溝通。

另外兩條線則是代表 lookup server 與 service provider 之間的溝通：service provider 利用 discovery 程序找到 lookup server 後，再利用 join 程序登入到 lookup server 中。

我們所實作的系統架構與 Jini 的靜態結構相似，但兩者間在設計目標及系統運作流程上有一些差異存在，Jini 設計的目的是提供一個方便的架構及模組讓程式設計師能很方便地建立能夠與整個基於 Jini 的服務系統溝通的應用軟體，而我們實作的系統則是強調透過管理後端服務伺服器的運作，分派適當的服務使得負載平衡，以便滿足使用者的要求，提昇區域網路上服務運作的效能，系統運作上有三點主要的差異：

1. 在 Jini 系統中 lookup server 跟其它 server 只有在登入時有過互動，但在智慧型服務網路伺服架構中，不但在後端服務伺服器登入時有溝通，工作管理伺服器也會定期與各後端伺服器聯絡，在服務伺服器發生問題時工作管理伺服器也會在一定的時間內得到訊息，作出適當的處理。
2. 在 Jini 系統中 lookup server 只存放 service provider 提供哪些服務，而在智慧型服務網路伺服架構中，不但存有後端伺服器提供哪些服務，更存有後端伺服器的負載狀況及網路運作狀況。
3. 在 Jini 系統中 client 只有告訴 lookup server 它需要何種服務，而在智慧型服務網路伺服架構中，客戶端可以提出特別的要求，並且告訴工作管理伺服器足夠的訊息，讓工作管理伺服器除了能夠挑選出讓使用者滿意的後端服務伺服器之外，也可以同時對其它的客戶端繼續保持最佳的服務狀態。

由上述內容可以知道，Jini 強調對於程式設計上的便利性，而智慧型服務網路伺服架構則是強調對於系統完善管理並且滿足所有客戶要求的功能。

(二)與 TSpaces 的差異

TSpaces 是 IBM 開發的一個分散式服務環境[7]，IBM 希望創造一個平台，讓現存的所有服務及應用程式不管是在哪種作業系統下運作都能互相溝通。以技術方面來講，可以把 TSpaces 當作一個可以藉由網路進行存取，並具有資料庫能力的儲存器，其他的應用程式可

以藉由其所定義的 tuple 來與 TSpaces 溝通。在實作方面，TSpaces 以 Java 作為開發工具，運用 Java 使得 TSpaces 更容易跨平台，而一般的應用程式只要透過 Java 程式，將資訊轉換成 tuple 的型式，就可以與 TSpaces 伺服器進行溝通。

TSpaces 與本研究中的系統架構相類似，TSpaces 伺服器類似於我們的工作管理伺服器或是 Jini 的 lookup 伺服器，其它部分也有類似的對應，但如同本系統與 Jini 的差異，TSpaces 設計的目的是希望創造出一個平台，讓現存的應用程式能夠這個平台互相溝通，而我們的系統是強調藉由工作管理伺服器，進行監督管理及分配工作的任務，達到增加整體系統效能的目的。

另外在實務上許多 IA 產品的平台因為成本與技術的考量以致於都不是 Java 的平台，我們的系統架構單純，可以移植到任一個特殊平台上來運作。

六、結語

在本研究中智慧型服務網路伺服架構利用分散式系統的架構，將數個後端服務伺服器整合在一起，並配合工作管理伺服器，進行工作分配以及管理各後端服務伺服器的運作。工作管理伺服器負責接受客戶端要求，為其選擇適當的後端伺服器，接下來則由後端伺服器完成工作，所以工作管理伺服器並不會被複雜的資料處理(例如實作中的文字轉語音等)工作給拖垮，可以應付大量的工作要求並能維持服務品質，所以如同第一節所說，我們的系統可以處理需要大量傳輸資料或複雜計算的多媒體服務。未來將逐步加強系統 QoS 的部分，並且加入認證以及安全性方面的機制，使工作管理伺服器的功能可以更健全，同時也可以在同一個網路區段中設計多個工作管理伺服器，以解決容錯問題。

七、參考文獻

[1] Bill Venners, “Object，the Network，and Jini”. http://www.artima.com/jini/jiniology/intro.html

[2] Bill Venners, “Separating UI and Functionality”. http://www.artima.com/jini/jiniology/Separation.html

[3] Bill Venners , “The Jini Vision”., http://www.artima.com/jini/jiniology/vision.html

[4] Eager, D.L, Lazowska, E.D., and Zahorjan, J.,” Adaptive Load Sharing in Home -geneous Distributed Systems”, IEEE Trans. On Software Engineering, vol. SE-12, pp.662-675, May 1986.

[5] Joseph A. Giampapa, Octavio H. Juarez-Espinosa and Katia P. ”Configuration management for multi-agent systems”; Sycara; Proceedings of the fifth international conference on Autonomous agents, 2001, Pages 230 – 231.

[6] *Rajkumar, R., Lee, C., Lehoczky, J., Siewiorek, D.* “A resource allocation model for QoS management”,Real-Time Systems Symposium, 1997. Proceedings., The 18th IEEE , 1997 Page(s): 298 – 307.

[7] “TSpaces”, http://www.almaden.ibm.com/cs/TSpaces/index.html