一個基於 Web Service 的洪氾預警系統

A Flood Forecasting System Based on Web Services

吳佩珊，張雅惠
國立臺灣海洋大學資訊工程系
Pei-Shan-Wu，Ya-Hui-Chang
Department of Computer Science and Engineering
National Taiwan Ocean University
{19957019, yahui}@ntou.edu.tw

摘　要

由於洪氾預警系統的建立，時常需要藉由各式各樣的地理水文資料進行傳遞交換，再經過異質平台的演算模組進行通訊合作，因此模組彼此之間的溝通橋梁成為重要的研究議題。在本論文中，我們採用 SOA（Service-Oriented Architecture）的概念，設計一個完善的系統架構，並基於 Web Service 的技術實作有效的資訊整合平台，讓各演算模組迅速地取得所需資料，且讓異質平台的演算模組直接進行通訊。我們並且將各模組演算的結果利用 Google Map API 的技術，搭配圖表和地圖適當地呈現於網頁上，達到快速預警的目的。

一、 緒論

台灣地區雨量豐沛，年平均降雨量高達2500公釐，為世界平均值的2.5倍。常年因颱風與豪雨事件所帶來的降雨，而釀成嚴重的淹水災害，造成人民生命及經濟上的重大損失。近年來，由於全球氣候變遷的加劇影響，使得降雨狀況更加趨於極端而超過預期，導致災害範圍和程度相較於以往更加慘重。有鑑於淹水發生頻繁，突顯出工程手段並不能完全杜絕水患，所以我們希望建立一個洪氾預警系統，藉由即時的偵測資料有效模擬推估未來的水情預報，並利用合適的淹水警戒值研判預警區域，以掌握黃金時間疏散避災減少損失。

一個完善的洪氾預警系統，需要結合數個具有不同功能的水利模組，如「降雨逕流演算模式」、「河道斷面演算模式」、「二維淹水演算模式」和「海岸越波演算模式」等。由於各模組會各自針對特定目標，利用即時的偵測資料進行接連的演算模擬過程，模式之間資料的傳遞與程式的協調成為系統是否能順利運作的關鍵因素。所以本篇論文的主要研究重點，即是建立一個有效的資料交換與作業整合平台，讓在不同作業系統下，使用不同程式語言所開發的應用程式，也能直接進行通訊和迅速地傳遞交換資料。我們同時會將模組的演算結果自動匯入資料庫，最後利用圖表和地圖等工具做適當的網頁畫面呈現，可以清楚地顯示洪氾預警區域，以協助相關單位即時做出正確的決策支援，減少人員傷亡及財務的損失。

為了有效地達到在Windows作業環境和Linux作業環境的資料交換，我們設計了數類Web Service。Web Service應用程式提供一組基於HTTP協定的通用服務，用戶端和伺服端可直接藉由HTTP進行互動，且其資料交換的協定是以XML（Extensible Markup Language）為基礎的技術。由於其具有良好的擴充性，所以基於Web Service的應用近來受到很大的關注。我們將在後續的章節中，說明我們所設計的Web Service具有何種功用，且如何將其實作出來。

接下來簡介本篇論文的架構如下：在第二

節，我們會介紹四個水利模組的主要功能及其相關性。在第三節將會敘述我們所提出的洪氾預警系統架構，其如何滿足各水利模組資料交換的需求。在第四節則會進一步說明我們如何設計合適的Web Service，以建構出完整的系統程式流程。而整個系統實作介面將會於第五節做成果展示。最後在第六節討論相關研究，以及在第七節提出本篇論文的結論。

二、 水利模組介紹

本篇論文建立的洪氾預警系統包含數個水利模組，各模組對應的演算模式與之間所傳遞的資料，如圖 1 所示。由於水利演算模式的細節研究非屬本篇論文著重的焦點，所以在本節中，我們僅簡略介紹各水利模組的功能以及與其它模組的相關性，以使讀者了解各模組間的資料流向與複雜的相互作用。

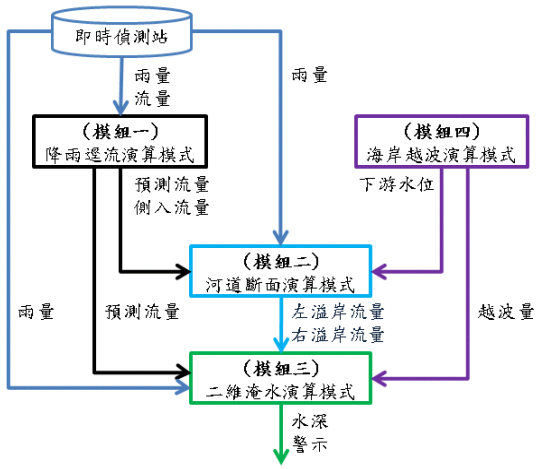

圖 1：水利模組關係圖

如圖 1 所示，首先，整個系統流程的起始點為模組一，該模組將執行「降雨逕流演算模式」。此模組會配合地理資訊系統的應用，將研究流域劃分為上游次集水區、支流次集水區，以及殘流域等多個演算區塊，以考慮空間降雨的非均勻特性，而後進行降雨逕流演算，並結合氣象單位偵測站的即時降雨預報資料，發展即時逕流校正方式，以提高預測的精度與效能。其所模擬預測的資料為未來一至三小時的上游入流量、河道支流入流量和殘流域側入流量，該資料將分別提供模組二和模組三所需使用。

系統的另一個起始點為模組四，該模組則會執行「海岸越波演算模式」。由於颱洪期間的沿海水域時常同時出現暴潮與巨浪，而河川大量的洪水抵達出海口位置，若受到潮汐漲落及暴潮影響，將導致河川的迴水效應。如此一來，海岸區域不僅會受到原本河川洪水引致的淹水災害，亦會受到高水位和巨大波浪聯合引致的越波流量，而加重海岸區域淹水的嚴重程度。因此，此模組的目的在於結合暴潮水位的模擬分析，演算河海口暴潮水位以及海岸保護結構物越波流量的即時資料，進而依此演算結果模擬沿海低窪區域的海水溢淹情形。此模組輸出的資料，將做為模組二和模組三執行演算時的下游邊界條件。

其次，模組二接收到模組一和模組四的資料後，將接續執行「河道斷面演算模式」。此模組的目的在於考慮不確定土石量影響，以建立集水區河道斷面的洪水模擬演算。在執行演算之前，此模組必須先依據河川斷面測量的地形資料，繪製出各河川的斷面圖。當颱風來臨時，此模組會即時考量上游陡坡河川側坡崩塌、土石流由支流河谷進入主流河川，以及土石隨洪水進入下游平原河川段等不同河川淤積型態，進而預測河川斷面的水位高度、左溢岸和右溢岸的流量，以提供模組三所需的資料，並依此演算結果研判河川將於何段溢堤。

最後，模組三集結之前三個模組所提供的演算結果，以執行「二維淹水演算模式」。此模組利用多核心伺服器進行平行運算數值模擬，並配合空間與時間的離散處理方式，以發展高效能的二維淹水演算，縮短傳統二維淹水演算計算耗時的缺失，進而模擬演算河川下游低窪地區的水深高度，以做為即時淹水預警區域的研判依據。

由上述討論可知，各模組之間存在著複雜的資料傳遞關係。同時，由於模組三和模組四的程式會在 Linux 系統上執行，而模組一和模組二的程式將在 Windows 環境下執行，所以更加深資料傳遞的困難。以下兩節，我們將提出對應的

系統架構設計，以及如何利用 Web Service 的技術來實作此系統。

三、 系統整體架構

如同之前所討論，我們希望建立一個有效的資料交換與作業整合平台，讓各水利模組快速地取得與傳遞資料，並協調各水利演算模式之間的執行作業，使其不僅能夠進行跨平台的通訊合作，亦能夠達成資源共享的功能。在本節中，我們將詳述所提出的系統架構。

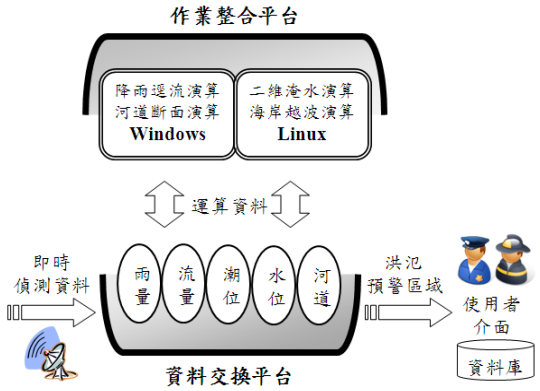

圖 2：系統架構圖

我們所規劃的資訊整合平台，其架構大致如圖 2 所示。首先，我們接收來自外部即時偵測站蒐集的資料，例如雨量、流量等。由於這些資料會被不同的水利模組加以利用，而其中部分水利模組又必須等待其他水利模組的演算結果繼續做運算，所以這些即時取得和運算產生的資料皆必須被存放至資料交換平台中。同時，它們也會自動被匯入關聯式資料庫中，以便建立使用者介面，提供相關人員查詢分析之用。至於使用者介面的設計，我們利用 Google Map API 的技術將與時間相關的資料以統計圖表的方式呈現，而與空間相關的資料則搭配地圖呈現給使用者。

其次，我們依照各水利模組分別在不同作業系統下執行的需求，建立一個整合的作業環境，以協調程式之間的執行運作，使其以正確的順序執行，如圖 1 所示。針對此點，我們採用事件驅動的處理方式，等偵測到所需資料均被演算輸出後，再呼叫下一個程式執行，並同時將演算結果自動匯入關聯性資料庫，以便於日後資料的查詢與維護。其次，由於部分水利模組於不同的作業環境下執行，需要進行跨平台的資料傳遞與通訊合作，因此我們採用 Web Service 的技術協助解決。

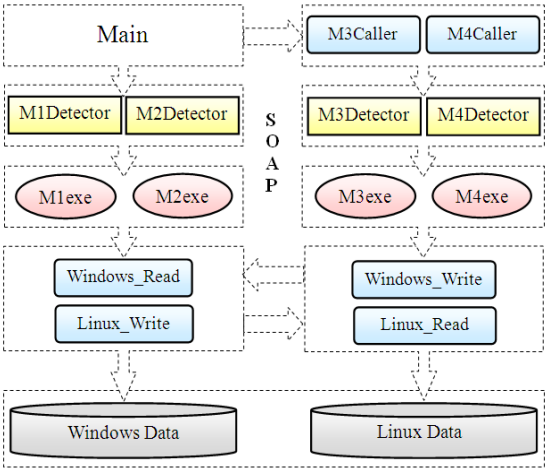

圖 3：程式流程圖

在本系統中，我們所撰寫的自動偵測程式和 Web Service，其完整的系統程式流程如圖 3 所示。首先，程式流程的起始點為一個主控台程式，亦即「Main」，用以掌控以下所有的程式物件，其中直角矩形表示各水利模組執行前的資料偵測程式，橢圓形表示水利模組的應用程式，共有 M1 至 M4 四個水利模組，圓角矩形則表示 Web Service。

特別針對 Web Service 的部分，依其不同功用又可劃分為兩類。第一類為跨平台的通訊合作，如「M3Caller」可讓在 Windows 環境下的「Main」程式，跨平台呼叫 Linux 系統下的「M3Detector」執行。另一類為跨平台的資料傳遞，其需要藉由 Read 和 Write 兩個 Web Service 結合來達成，如「Linux_Read」是在 Linux 系統下，進行讀取水利模組演算後的結果，而「Linux_Write」則是在 Windows 環境下，透過資料交換協定 SOAP，接獲「Linux_Read」回傳的讀取資料，並將其寫入資料交換平台中。這樣一來，位於 Linux 系統裡的資料即可順利傳送至 Windows 環境裡。從圖 1 我們知道各水利模組均有傳遞資料的需求，因此四個水利模組均會有一組對應的 Read 和 Write

兩個 Web Service，而有關 Web Service 如何實作將在下一節中詳加說明。

四、 Web Service 設計

在本節中，我們藉由範例程式深入說明 Web Service 的設計，以下所提及的範例程式均是採用 C#程式語言撰寫。在 Windows 環境下，我們使用 Microsoft Visual Studio 2008 軟體，而在 Linux 系統下，我們則是使用 MonoDevelop 2.4 軟體，透過這兩個軟體可以協助我們讓分別在 Windows 環境下和 Linux 系統下，各自所架設的網站伺服器，均能執行 ASP.Net 的 Web 應用程式，以順利進行 Web Service 應用程式的共享。

首先，我們設計的第一類型 Web Service，提供跨平台的通訊合作，以便跨平台呼叫應用程式執行。我們以「M3Caller」為例，其程式碼如圖 4 所示，於第 05 行我們利用 System.Diagnostics 裡的 Process 物件呼叫「M3Detector」偵測程式執行。由於我們將該 Web Service 放置於 Linux 網站伺服器（http://140.121.196.3）內特定的資料夾，所以在 Windows 環境下的「Main」程式，可透過「Web 參考」的方式，輸入該 Web Service 所處的網址，亦即（http://140.121.196.3/Flood/Program/M3Caller/Service1.asmx），即可該 Web Service 參考進來當作一般物件使用。

其次，第二類型 Web Service 提供跨平台的資料傳遞，該動作可視為一組 Web Service 的結合，其中一個為讀取資料的 Web Service，如 Linux 系統下的「Linux_Read」，另外會有一個對應的 Web Service，目的是執行寫入資料，如 Windows 環境下的「Linux_Write」。假設我們要將「M3exe」在 Linux 系統下演算後的結果傳送到 Windows 環境下，其「Linux_Read」的範例程式如圖 5 所示。其中第 02 到 05 行先定義我們所傳遞參數的物件，並將其命名為「readData」，裡面可以包含多個字串參數，以對應多個傳遞資料檔案的需求。而第 06 到 24 行為該 Web Service 所提供的服務，其包含一個「Read」的主函式，該函式會協助於 Linux 系統下執行讀取檔案的動作。一開始先於第 09 行宣告一個型態為「readData」的參數物件，並將其命名為「RData」。接著於第 10 行我們利用 System.IO 裡的 StreamReader 物件宣告一個檔案讀取變數，將其對應到特定的讀取檔案位置，此為「/var/www/Flood/TwoD_M3/M3Output1.txt」。而第 12 到 14 行則藉由 StreamReader 物件裡 Peek 函式所回傳的整數是否為-1，以判斷檔案內容是否讀取結束，若尚未結束則將目前讀取到的資料以串接的方式儲存於「RData」裡對應的字串參數中，若讀取結束則在第 15 行將檔案關閉。至於第 16 到 21 行以此類推，差異僅在於讀取不同的檔案到對應的字串參數中。等所有檔案均讀取結束後，最後在第 22 行將整個「RData」參數物件回傳。

至於另一個對應的「Linux_Write」，其範例程式如圖 6 所示。其中第 02 到 15 行為該 Web Service 所提供的服務，此處包含一個「Write」的主函式，該函式會協助於 Windows 環境下執行寫入檔案的動作。首先，我們將先前產生的「Linux_Read」透過「Web 參考」的方式引用進應用程式中，並於第 05 行宣告一個該 Web Service 的物件，此處將其命名為「Data」。接著我們利用 System.IO 裡的 StreamWriter 物件宣告一個檔案寫入變數，對應到特定的寫入檔案位置，此為「C:\Inetpub\wwwroot\Flood\TwoD_M3\M3Output1.txt」。而第 08 行即可藉由「Data」物件讀取 Linux 系統下的檔案，並利用 StreamWriter 物件裡 WriteLine 函式將接收到的字串寫入對應的檔案中，最後再將檔案關閉。至於第 10 到第 13 行的作用如上，差異僅在於抓取「RData」裡不同的回傳字串寫入對應的檔案中。

```
01 namespace M3Caller{
02   public class Service1 : System.Web.Services.WebService{
03     [WebMethod]
04     public void M3Caller(){
05       Process.Start(@"/var/www/Flood/Program /M3Detector");
06     }
07   }
08 }
```

圖 4：M3Caller 範例程式

```
namespace Linux_Read{
 public class readData{
  public string M3Out1;
  public string M3Out2;
 }
 public class Service1 : System.Web.Services.WebService{
  [WebMethod]
  public readData Read(){
   readData RData = new readData();
   StreamReader file_M3Out1 = new StreamReader
                    (@"/var/www/Flood/TwoD_M3/M3Output1.txt");
   while(file_M3Out1.Peek() != -1){
    RData.M3Out1 = RData.M3Out1 + "\r\n" + file_M3Out1.ReadLine();
   }
   file_M3Out1.Close();
   StreamReader file_M3Out2 = new StreamReader
                    (@"/var/www/Flood/TwoD_M3/M3Output2.txt");
   while(file_M3Out2.Peek() != -1){
    RData.M3Out2 = RData.M3Out2 + "\r\n" + file_M3Out2.ReadLine();
   }
   file_M3Out2.Close();
   return RData;
  }
 }
}
```

圖 5：Linux_Read 範例程式

```
namespace Linux_Write{
 public class Service1 : System.Web.Services.WebService{
  [WebMethod]
  public void Write(){
   Linux_Read.Service1 Data = new Linux_Write.Linux_Read.Service1();
   StreamWriter file_M3Out1 = new StreamWriter
          (@"C:\Inetpub\wwwroot\Flood\TwoD_M3\M3Output1.txt");
   file_M3Out1.WriteLine(Data.Read().M3Out1);
   file_M3Out1.Close();
   StreamWriter file_M3Out2 = new StreamWriter
          (@"C:\Inetpub\wwwroot\Flood\TwoD_P3\M3Output2.txt");
   file_M3Out2.WriteLine(Data.Read().M3Out2);
   file_M3Out2.Close();
  }
 }
}
```

圖 6：Linux_Write 範例程式

五、 系統實作介面

本論文建立的系統是以台灣南部的林邊溪為案例，進行洪氾預警的模擬演算。在使用者介面的設計，我們利用 Google Map API 的技術將與時間相關的資料以統計圖表的方式呈現，而與空間相關的資料則搭配地圖呈現。以下我們顯示部分介面所呈現的結果。

針對統計圖表部分，我們以河川流量折線圖為例，如圖 7 所示。我們將模擬結果與林邊溪下游的新埤流量偵測站所測得的資料進行成效比對，實線表示實際偵測的資料，而虛線表示我們的模擬結果。由圖 7 我們可以觀察出，隨著橫軸時間的遞移，縱軸的河川流量值有逐漸增強的趨勢，而我們的模擬結果亦達到不錯的成效。另外，針對地圖部分，我們以二維淹水預警區域為例，如圖 8 所示。在圖 1 中的水利模組三會模擬演算出河川下游低窪地區的水深高度，我們以此做為即時淹水預警區域的研判依據，將水深高度達一定警戒標準的二度分帶座標點，於地圖上以紅色圓點標記，因此使用者可以清楚地從介面得知哪個區域需要緊急防災。

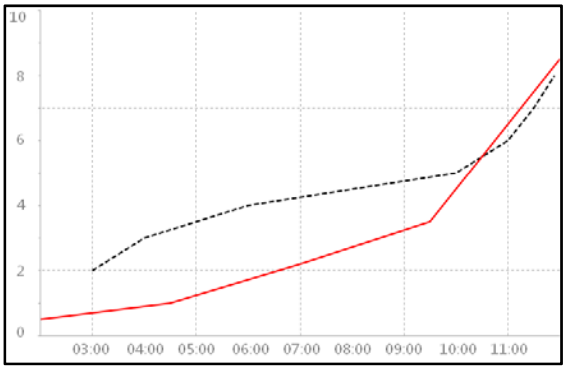

圖 7：河川流量折線圖

圖 8：二維淹水預警區域

六、 相關研究

本論文的相關研究可概略分為「洪水預警系統」和「SOA」兩大部分，以下分別討論之。首先，洪水預警系統由於經常需要透過各式各樣地理水文資料的交換，進行合作計算出結果，然而獨立的電腦程式因缺乏資料的通訊能力而產生問題，所以有研究者提出了建構於全球資訊網的即時洪水預警系統[1]。而論文[5]提出的 Web-based Flood Forecasting System(WFFS)，進一步利用多層次架構的優點，設計三層分別負責不同工作，其中 presentation tier 包含使用者介面和使用者互動，application tier 負責執行水文模組，data tier 用來儲存整個系統所需的資料。由於工作經過良好的分割，即使修改其中一層，也不會使得另一層需大幅修改。

另一方面，荷蘭 Delft Hydralics 發展一套洪水早期預警系統，稱作 FEWS（Flood Early

Warning System）[4]，該系統採用 XML 作為資料的表示方式，能解決各單位測量站資料格式不一的問題，所以臺灣水利署與其合作開發出適用於台灣水文環境的 FEWS-Taiwan 系統。論文[7]進一步利用 FEWS-Taiwan 作為資料的接收與運算平台，並將其資料庫結合 Google Map API 的技術[2]，所以能透過網頁在 Web 上做即時的線上查詢與警戒測站的標示。

其次，我們討論 SOA（Service-Oriented Architecture）的相關研究。Web Service 應用程式提供了一組基於 HTTP 協定的通用服務，用戶端透過 HTTP 傳送函數呼叫給伺服端，而伺服端以 HTTP 將結果回傳給用戶端，所以可以同時支援多個位於不同作業平台的用戶端使用。為了讓用戶端和伺服端能夠依據相同的訊息規格傳遞資訊，W3C 定義了一組標準的資料交換協定，即「SOAP」（Simple Object Access Protocol）[6]，該協定主要是以 XML 為基礎的技術，擁有良好的擴充性，其傳送的資料就是 XML 元素的資料。

由於 SOA 的概念能提供網路服務單位建構具有彈性、可重複使用的整合性介面，加速達到網路服務提升的目標，所以近期廣泛地被應用。論文[3]即是依此概念，利用 Web Service 技術進行複雜的演算程式，並建立一個整合性介面，提供特定的地理水文專家進行演算的參數調整。不同於該篇論文，我們主要是藉由 Web Service 技術協助整個洪氾預警系統的建立，並將演算結果以最具親和力的畫面呈現給一般大眾。

七、 結論

本篇論文研究如何以 Web Service 為基礎，實作資訊整合平台，以協助開發出完善的洪氾預警系統。近年來，由於 Web Service 的相關研究成為熱門的議題，許多針對跨平台分享資源的解決方式亦備受探討，除了本論文所使用的 SOAP 技術，另外還有 REST（Representional State Transfer）等技術。未來我們期望也能運用該技術實作此系統，進一步比較 SOAP 與 REST 兩者的優劣，以有效提升資訊整合平台的執行效率。

誌謝 感謝鄭惟臣同學與蔣宜晏同學協助實作此系統。

參考文獻

[1] W. Al-Sabhan, M. Mulligan, and G.A. Blackburn, “A real-time hydrological model for flood prediction using GIS and the WWW”, *Computers, Environment and Urban Systems*, vol. 27, no. 1, pp. 9-32, 2003.

[2] Google Map API, http://code.google.com/intl/zh-TW/apis/maps

[3] Carlos Granell, Laura Dıaz, and Michael Gould, “Service-oriented applications for environmental models: Reusable geospatial services”, *Environmental Modelling & Software*, vol. 25, no. 2, pp. 182-198, 2010.

[4] Delft Hydraulics, “Delft-FEWS, an open shell flood forecasting system”, http://www.wldelft.nl/soft/fews/int/index.html

[5] Xiang-Yang Li, Kwok-wing Chau, Chuntian Cheng, and Y. S. Li, “A Web-based flood forecasting system for Shuangpai region”, *Advances in Engineering Software*, vol. 37, no. 3, pp. 146-158, 2006.

[6] Simple Object Access Protocol（SOAP）, http://www.w3.org/TR/#tr_SOAP

[7] 黃博炫, 謝龍生, 朱子偉, 傅金城, “洪水災害早期預警系統建置之研究”, 第18屆水利工程研討會論文集, 2009。