國立交通大學

電機資訊學院 資訊學程

碩 士 論 文

採用 Web 技術
整合 GIS 與通用性資料庫應用系統之研究
－以海事地理資訊系統為例

A Study on GIS Integrated with a Generic DB Application
by adopting Web Technology
– Using Maritime GIS Application as an Example

研 究 生：黃以德

指導教授：李素瑛　教授

中 華 民 國 九 十 五 年 七 月

採用 Web 技術整合 GIS 與通用性資料庫應用系統之研究
－以海事地理資訊系統為例

A STUDY ON GIS INTEGRATED WITH A GENERIC DB APPLICATION BY ADOPTING WEB TECHNOLOGY - USING MARITIME GIS APPLICATION AS AN EXAMPLE

研 究 生：黃以德　　Student ：Dennis Yi-te Huang

指導教授：李素瑛　　Advisor ：S.Y. Lee

國 立 交 通 大 學

電機學院與資訊學院專班 資訊學程

碩 士 論 文

A Thesis

Submitted to Degree Program of Electrical Engineering and Computer Science

College of Computer Science

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Computer Science

July 2006

Hsinchu, Taiwan, Republic of China

中華民國九十五年七月

採用 Web 技術整合 GIS 與通用性資料庫應用系統之研究
－以海事地理資訊系統為例

學生：黃以德　　指導教授：李素瑛

國立交通大學電機資訊學院 資訊學程（研究所）碩士班

中文摘要

在交通監控中心裡為使管制員充分掌控交通狀況，其管理工作站一般配置兩台螢幕，一台螢幕專供顯示地理資訊圖像外，另搭配一台螢幕顯示其相關的資訊信息。為降低開發與維護成本，資訊服務業者都期望能發展出一套通用性資訊系統，可適用於不同硬體平台、作業系統甚至國家語言等。現今 Web 技術如 Java、瀏覽器的發展宗旨即在實現跨平台的理想，然而因其受限於 HTTP 架構（Request →Process → Reply），與交通監控管理的作業模式並不相符。在監控模式下，是由伺服器端主動通知客戶端事件的發生或改變；這與 Web 模式下由客戶端提出交易請求再獲得結果的方式相異。因此想在目前的 HTTP 架構下，採用 Web 技術來發展通用性的監控管理資訊系統極為困難。

在這篇論文裡，我們提出了五項規範來衡量資訊系統是否已符合通用性要求，這五項規範分別為：硬體獨立性、作業系統獨立性、使用語言獨立性、資料庫工具獨立性及應用系統獨立性。為符合這五項通用性規範，本系統仍採用 Java 技術作為開發平台，並結合 Web 主動推播技術來解決監控管理系統下需能獲取即時資訊的問題；使得工作站上，Web 化監控管理資訊終端能與地理資訊終端相互傳遞訊息。此外，我們探討如何以 RIA (Rich Internet Application) 技術來開發網頁程式，以改善系統反應效率及讀取記錄時網頁需重刷新的缺點，使得操作畫面感覺與傳統桌面程式相同。最後，為了驗證通用性資訊系統，除發展三種不同語文套件外，我們將完成的實作系統在不修改程式碼的情況下，觀察其於不同的硬體、作業系統、資料庫工具及使用者語言間的執行情形，以確定通用性資訊系統的開發可行性與效益，達到降低軟體開發與維護成本的目的。

A Study on GIS Integrated with a Generic DB Application by adopting Web Technology – Using Maritime GIS Application as an Example

Student: Dennis Huang Advisor: S.Y. Lee

Degree Program of Electrical Engineering Computer Science
National Chiao Tung University

ABSTRACT

In a Traffic Service Center, for the purpose of easily monitoring traffic status, operator can supervise the condition through a traffic monitoring station. In general, there are two monitors in a traffic monitoring station. One monitor dedicates to be a GIS client that can display a electronic chart overlapped with moving targets. Another is a DB client to display corresponding text information. To save the cost of development and maintenance, usually the software developers wish to build a generic information system that can run on a variety of environment over different hardwares, OS and user languages. Web technology nowadays such as Java and browser-based application can achieve this goal, but do not fully satisfy the characteristics of a monitoring system. In HTTP protocol, the scenario is request, process and reply. This is different from the model of a monitoring system. For a monitoring system, server will notify clients whenever an event occurs or changes. On the other hand, a HTTP client can have a reply only after its previous request. For this reason, it seems difficult to build a generic information system by adopting Web technology.

In this thesis we propose five criteria that can evaluate whether an information system is generic or not. These criteria are: hardware independence, OS independence, user language independence, database tool independence and application independence. To meet the above criteria, we adopt the Java technology as the development and run-time platform for the implemented system in this thesis. In addition to this, we utilize web push technology to achieve the real-time data acquisition in a monitoring system. Based on this push technology, web DB client can send data to GIS client and receive from it. Meanwhile, we study how to develop the browser-based program by using RIA (Rich Internet Application) technology. With RIA technology, the system response time is improved and there is no need to refresh a whole web page while a user asks for a new data record. As a result, the look and feel inside a browser-based client with RIA is almost the same as a traditional C/S database client. At last, to validate the development possibility of a generic information system, we have tried to run our developed system under a variety of hardwares, OS, database tools and logon with different user language. From the experimental result, we believe this development model is workable to a generic information system and it can dramatically save the cost and improve the software development and maintenance of GIS system.

誌謝

本論文承蒙恩師李素瑛教授長期耐心的指導及教誨。恩師不僅在學術上給予指導，更在待人處世及作研究的態度上亦受益良多；本論文得以順利完成，在此要對恩師至上最無限的感激與謝意。

此外，感謝所有曾教導我的師長、同學和同事，他們增進了我的專業知識並提供我許多論文上的寶貴意見與想法。尤其在我論文寫作過程受到阻擾挫折時，給予我最大的精神安慰與支持。

最後，要特別感謝我的太太及可愛的子女，容忍我在他們的成長中，因求學過程的需要而缺席，並鼓勵我專心完成論文，才有如今的我，僅在此再次獻上我最深的謝意。

目錄

圖目錄

表目錄

1. 緒論

1.1 研究動機

監控管理是許多系統常見的功能，更是交通管理配合地理資訊系統（GIS）下的主要作業項目。海上的船舶交通管理系統（VTMS – Vessel Traffic Management System）管制船舶進入港口、停泊碼頭及離開港口等作業 [8]；空中的飛航管制系統，管制各飛行器於機場之降落、停靠及起飛 [10]；陸上的警車勤務派遣系統，透過 GPS 將警車衛星定位資訊傳入管制中心，使中心指揮人員能受理報案、警車派遣指揮、管制等 [11]。以上監控管理都由操作員於管制員工作站 (Operator Station)進行各項管制作業。該工作站通常配備兩台螢幕，一台螢幕專供顯示 GIS 電子海圖外，另一台螢幕需搭配 GIS 提供各項資料庫訊息。

因成本考量，軟體業者都期望能發展出一套通用性的資訊系統，可適用於不同工作平台、作業系統、國家語言等，如此方能降低軟體開發與維護等成本，提升競爭力以謀求最大利潤。採用 Web 技術如 Java、browser 皆能達成以上目標。然而這些 Web 技術是否適用於上述之監控管理作業呢？

目前 Web 應用系統因受限於 HTTP 架構，其模式皆屬 HTTP Request → Web Server Processing → HTML Page Reply 的作業流程，也就是都由客戶端觸發交易事件為起始。然而，監控管理作業有項重要特性就是資料即時獲取與分析處理，也就是說當遠端感測設備或環境發生變動時，需能即時通知前端操作人員做即時應變處理 (real-time operation)，例如：雷達偵測出兩艘船舶可能發生碰撞之警告。相較於傳統 Web 應用系統，此時監控管理卻是由伺服器端觸發事件並通知前端操作人員(client workstation)告警事件之發生。因此，除非採用 JAVA Applet 技術（配合 RMI, CORBA 等），否則難以在瀏覽器內達成此監控管理功能。

本論文的研究動機，主要在於探討發展一套通用性資訊管理系統時所應採用的 Web 技術，並使其能在各種不同的硬體平台、作業系統、國家語言下運作；同時又能如同傳統 Client / Server 架構之應用系統，可即時監控、有效管理各項設備。

1.2 船舶交通管理系統簡介

港口管制人員可透過船舶交通管理系統針對港區服務範圍（通常以港口為中心 20 海浬內的管轄範圍）內的各式船舶實施持續性的偵測與追蹤，並將結果回傳到監控管理中心，使管制人員可即時瞭解與處理交通狀況，並協助各類船舶在不同的天候條件與時間狀況下安全且有效率的進出港口。簡

言之，建立船舶交通管理系統的目的有三，即資訊即時獲取、助航協助與服務與交通管理服務與監控。[8]

即時資訊獲取需求如下。1、雷達對所有港區內船舶動態作持續性的偵測。2、管制人員可透過 VHF 無線電與海上船舶進行雙向交談。3、可透過 VHF 無線電探向器，協助操作人員找出通話船舶的位置。4、以閉路電視監控港區內特定區域。5、整合來自港口其他單位電腦系統內的船舶、代理商、航行計畫等資訊。6、掌握即時海氣象資訊。

有關助航協助部分，操作員可從船舶交通管理系統取得最新的現況交通情報，以提供需協助的船舶其周圍詳細交通狀況資訊、海氣象資料等。並以 VHF 無線電與船舶通訊協助進出港航行作業。並提供船舶碰撞可能之告警、流錨告警、航道偏離或超速之告警。對於交通管理服務與監控，操作員可達到以下目的。1、監控錨泊區的船舶動態。2、監視航行輔助設施的位置。3、禁區監視-如海底油管區、航道限制等。

由上得知，一套完整的船舶交通管理系統至少應包含：雷達偵測系統、電子海圖顯控系統、VHF 無線電通訊系統、船舶資料庫系統與網路系統。

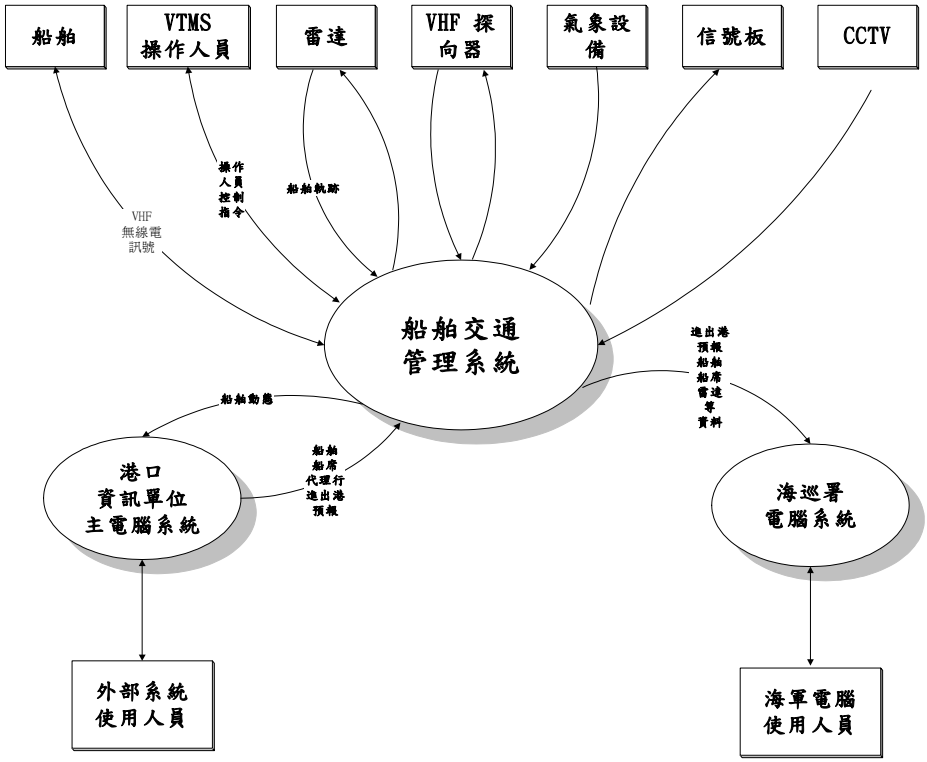

圖 1：船舶交通管理系統設備架構概況

圖 1 表示 VTMS 外部設備與系統間之信號控制及操作方式，詳細操作過程如後。港區內設置數架雷達，持續偵測港口管制區域，並將雷達獲取資料傳回 VTMS 加以處理。操作人員藉由海事專用頻道，以 VHF 無線電與來訪船舶聯絡，並確認其身分及來訪目的。操作人員在與來訪船舶通話時，

可於管制中心設定遠端之 VHF/DF 之目標頻道。待 VHF/DF 定位後，將測得目標物之角度資料傳回 VTMS 加以處理。配合雷達資料，顯示於交通狀況顯示幕上，以協助操作人員確定船舶之位置。

氣象設備應包括風速計、風向計及大氣壓力計等設備，各氣象資料於定時蒐集後，傳回 VTMS 加以分析處理。操作人員可透過此系統顯示交通管制管制措施於信號板上，以告知進出港之船舶目前港區內之狀況。並可於交通管理中心內，遙控遠端 CCTV，並藉由 CCTV 所攝得之影像，以監控港區內船舶之動態，補足雷達所無法涵蓋之區域。

船舶交通管理系統須與現有港口電腦中心資訊系統連線，透過通訊界面，可傳送資料到港口主電腦系統，並可獲得已建立於主電腦的船舶基本資料、代理行基本資料、船舶進出港計畫及船席現況等資料。某些國防系統如海巡署或海軍也可透過通訊介面查詢資料。

1.3 章節概要

在第一章裡，說明何謂船舶交通管理系統，並簡述其功能，來瞭解為何要研究探討通用資訊系統這個主題，以及當前交通監控系統所面臨的困難與挑戰。

在第二章裡，我們探討了當前 WEB 應用系統的發展新趨勢，以及這些趨勢對我們的研究主題所造成的影響。

在第三章裡，提到為了發展通用性資訊系統，我們提出了一套發展規範及評估標準。並規劃了一套通用性資訊系統的發展環境。

在第四章裡，會依據第三章所提出的通用性資訊系統規範，以船舶資料庫資訊系統為實作範例，設計一套船舶用的通用性資訊系統；再針對此項實作成果進行效益分析，並說明此實作系統如何滿足通用性規範上的要求。

在第五章裡，說明了這份論文研究成果，並探討未來的發展方向。

2. WEB 發展趨勢相關主題研究沿革

本論文主要研究動機在探討如何發展出一套通用性的資訊管理系統，並以船舶資料庫應用系統為實作案例。有關通用性資料庫應用系統所應具備之性質請見本論文第 3.2.1 節所述。依照此通用性質規範，應採用 J2EE platform 技術作為通用性資料庫應用系統之開發工具，並整合以下技術主題。

2.1 WEB 應用系統的發展新趨勢

在 1980 年代末期，主從式應用系統(C/S Application)的出現取代了大型主機與終端機的應用架構。一直持續到網際網路的盛行，又使應用系統的開發從主從式架構轉移到瀏覽器式應用系統(B/S Application)。然而傳統以 HTML 為基礎的網頁程式開發都是基於頁面式的伺服器端資料傳遞的模式，為資訊應用系統的使用者衍生了其他問題，他迫使被接受一種線性的、以頁面為基礎的處理模式。事實上，當初發展 HTML 技術的人員一開始從未把它用來取代應用系統開發。

因此軟體業者提出 RIA (Rich Internet Application)的技術來做為 WEB 應用系統的使用者介面。RIA 的網頁與一般網頁或是其上所執行的動態程式之進行流程不同，卻和一般單機上所使用的軟體程式進行雷同。簡單地說，RIA 是一種在網際網路上運行的應用程式，有著如同桌面應用程式的行為、功能、快速回應、直覺與體驗，也融合了網際網路應用程式的容易開發的與低成本的特性。

利用 RIA 的技術，可以讓我們在網路上瀏覽各種頁面時，與使用 client 端的程式相同，不會因為 HTML 語言的特性，受限於一個步驟一個步驟 Request 的程式進行流程。在邁入內容的時代之後，如何整合各種多媒體資訊在一個介面中，是未來的趨勢。以數位電視為例，即是利用硬體設備與軟體整合，使本來利用一台電視機只能在同一時間看一個頻道節目的觀看型態，轉換成可以同一時間同時看不同的節目，甚至利用隨選視訊的方式，隨時觀看想要看的節目。相同的道理，在網際網路上的行為，也應該達到這樣的目標。目前在 RIA 的研究上，以 Macromedia 公司的 Flash 產品最具代表。其他的 RIA 應用有 XAML、XUL、Bindows 及 Flex 等。[18]

RIA 中 Rich 的含意有二，分別是資料庫模型的豐富與使用者介面的豐富。

1. 資料庫模型的豐富：

在使用者介面上可以顯示和操作更為複雜的嵌入在客戶端瀏覽器上的資料庫型態。也可以操作客戶端的計算甚至與伺服器端非同步的發送與接收資料。這種模式相對於傳統 HTML 頁面的優點是程式碼在客戶端執行，且與使用者進行互動多而與伺服器互動少。平衡客戶端和伺服器端之間的資料庫模式可讓你創造效率更高和更具有交互作用性的 WEB 應用程式。

2. 使用者介面的豐富：

HTML 只提供使用者一些非常有限的介面控制元件，而 RIA 卻對用戶介面提供了更多靈活且多樣的介面控制元件，並且這些元件可以很容易的與資料庫模型相結合。傳統的 HTML 程式採用線性處理的設計理念，提供使用者一些選擇後再將選擇結果發送到 WEB 伺服器這種單一的模式無法滿足應用程式的靈活交互要求，且頻繁的伺服器請求及前端的頁面更新都使頁面打開緩慢並消耗大量網路頻寬。如果採用 RIA，可以從以前的 WEB 伺服器影響前端的整個頁面，轉移到只有請求的應用部分才做出相對應的變化。

此外值得注意的是傳統網頁程式處理流程的問題。使用者與開發者皆能體驗到傳統網頁應用程式 Request / Response 模式所造成的侷限性，因為網頁是無狀態的，故每當網頁資料或使用者介面需要變更時，必須將整個網頁傳送到伺服器更新，意即頁面必須進行客戶端與伺服器端之間的往返，尤其是當彼此之間若有極大量的資料過度頻繁的往返，將會造成者用者感受到非常漫長的等待。

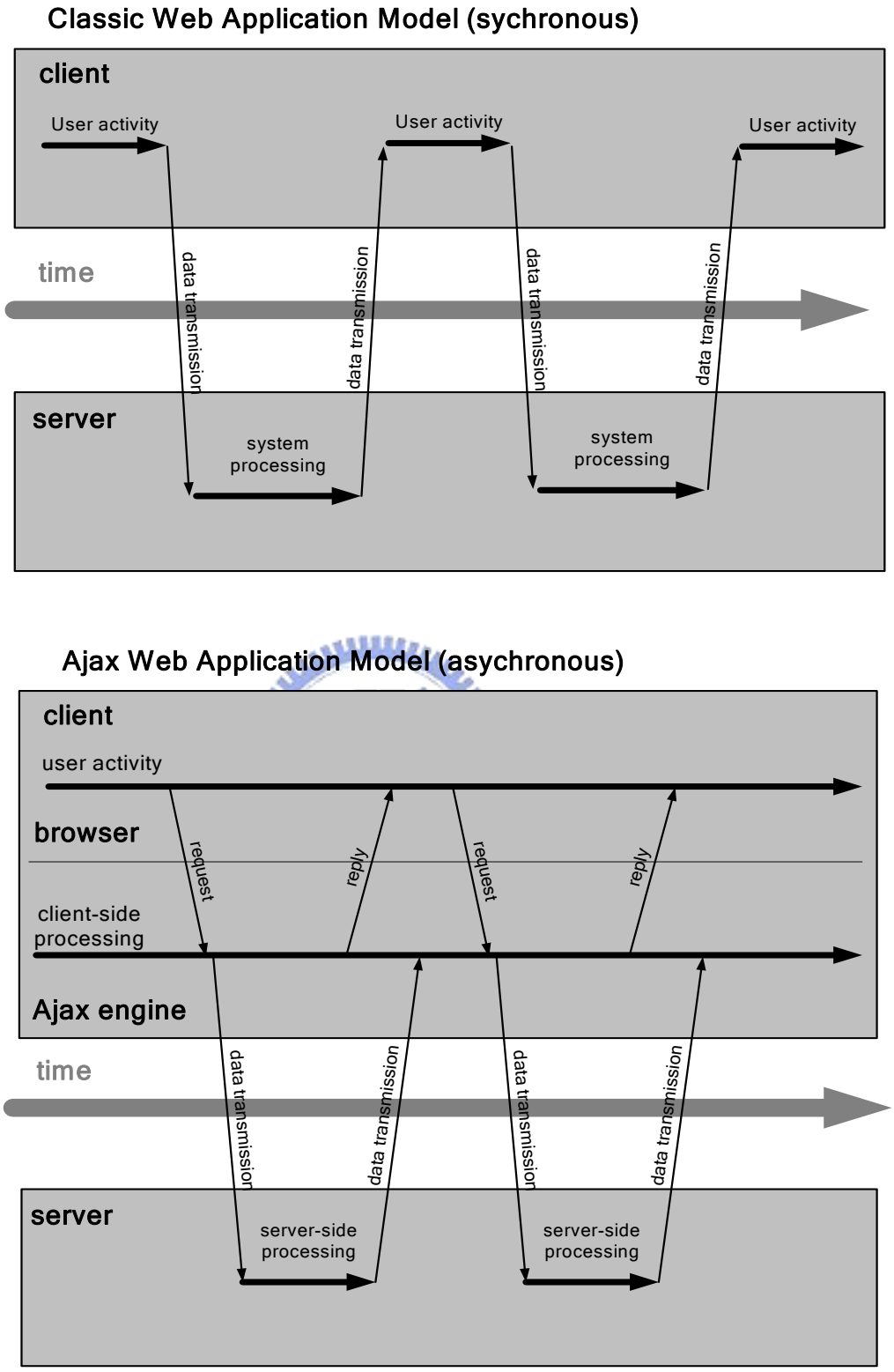

圖 2：傳統網頁應用模式與 Ajax 網頁應用模式處理流程的比較

圖 2 是傳統網頁模式與 Ajax 模式的處理流程比較，其中 AJAX 技術的發展加強了網路頁面的處理流程。AJAX 描述一個跨平台、以客戶端為中心

的網頁應用程式開發途徑，其構想在於建立一個核心概念：使用非同步呼叫，並結合 XML 方式傳輸資料與大量使用客戶端 Script。程式開發人員能夠以客戶端 Scripts 方式透過 XMLHTTP 來進行非同步的 Web 服務呼叫。透過非同步方式進行呼叫，網頁中的資料與使用者介面不需要進行客戶端與伺服器端之間的往返，結果是具有較少的資料傳送與較佳的應用程式效能，非同步呼叫能夠使得網頁應用程式能夠有更佳良好的反應性，因為使用者能夠在呼叫工作仍在伺服器端執行的同時，於客戶端瀏覽器內繼續進行工作。[15]

2.2 微軟新一代視窗作業系統的介面標記語言 - XAML

微軟於 2005 年公布了新一代視窗作業系統代號 “Longhorm”所使用的圖形使用者介面 “Avalon”的 SDK。Avalon 屬於 Longhorn 的一部分，是一個圖形和展示的引擎，主要由新加到 .NET framework 的一個 class 集合而成。

Avalon 定義了一個在 Longhorn 中使用的新標記語言，其代號為 XAML (Extensible Application Markup Languange)，可以使用 XAML 來定義文件、圖像、控制元件的佈局，與使用 HTML 的方式相似。

Avalon 的設計理念，環繞於下列四項主軸：[12]

1. Web 應用程式與視窗應用程式採用相同開發理念
2. 同時整合使用者介面，文件與多媒體內容展現
3. 具備彈性、運用 .NET 技術之應用程式架構 (Application Framework)
4. 善用個人電腦強大圖形運算能力

Avalon 中提供了以 XML 延伸而來的 XAML 標籤 (Tag) 語言，成為定義使用者介面與表達文件內容的主要方式，XAML 可以直接運用於 Web 應用程式與視窗應用程式展現內容與使用者介面，或是類似 ASP.NET 般 Code Behind 方式結合 .NET 環境內的程式語言，如此便可將 XAML 交由美工人員設計，而程式碼部分由開發人員負責，以現今 Web 應用程式分工方式來開發視窗應用程式。

表格 1 為一段簡單的 XAML 代碼，可於螢幕上顯示 Hello World。

```
<TextPanel xmlns="http://schemas.microsoft.com/2003/xaml"
        FontFamily="Comic sans MS"
        FontSize="36pt"
        HorizontalAlignment="Center">
    Hello, world!
</TextPanel>
```

表格 1 : XAML 代碼範例

將 HelloWorld.xaml 檔直接載入到 Microsoft Internet Explorer 的 Longhorn 版本中，然後您將看到類似一個 Web 頁的內容。還可以使用 MSBuild 程式來編譯 HelloWorld.xaml，執行 Hello World 的可執行檔後，您將看到一個類似於 Windows 程式的內容。圖 3 顯示 “hello world”以視窗型態展現及以瀏覽器型態展現的兩種版本。

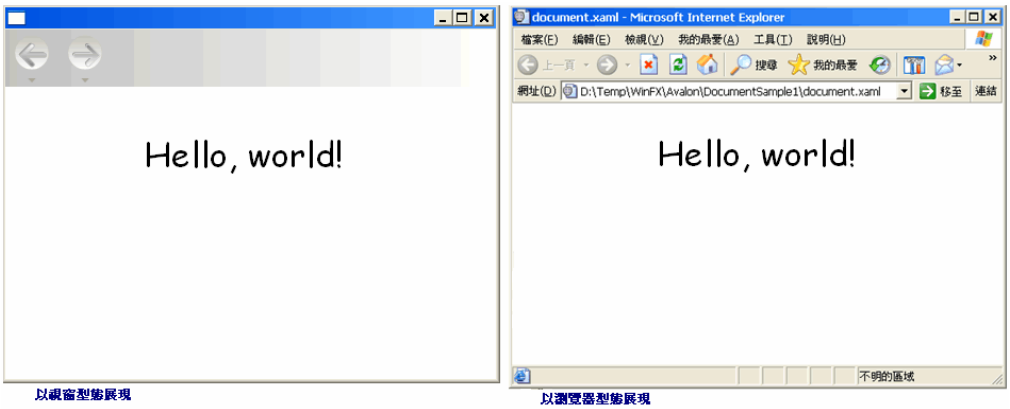

圖 3：XAML 以視窗型態展現及以瀏覽器型態展現的兩種版本

從某種意義上面來說 XAML 和 HTML 頁面非常相似，不過因為基於 XML 擁有比較嚴格的規範，同時因為在 Avalon 下執行，Longhorn 整個作業系統成為其容器，相對於 IE 而言，擁有更加廣闊的空間。

Avalon 和 XAML 的發展象徵著即將改變與過去基於 Windows 的應用程式的開發方式程。在許多方面，設計應用程式的 UI 都將比過去容易得多，而且部署起來也更輕而易舉。很明顯，由於有了用於 UI 定義的 XAML 標記，基於 Longhorn 的應用程式會成為集中 Web 和桌面編程模型的下一步發展方向，並且會合並上述兩種方法的最佳功能。

2.3 MOZILLA WEB 應用程式用戶介面 - XUL

XAML 理念並非微軟所獨創，在 XAML 問世之前，Mozilla 組織訂定 XUL 與 Macromedia 公司產品 Flex 中都有著相同的概念。XUL 代表 XML 使用者介面語言（XML User Interface Language），由 Mozilla 組織推出。流行的 Firefox 流覽器和 Thunderbird 郵件其使用者介面全都是用 XUL 編寫的。利用 XUL，開發人員能構建功能很豐富的應用，可以與 Internet 連接，也可單機執行。

為了讓熟悉 DHTML 的開發人員儘快地學會，XUL 設計為可以為諸如視窗和按鈕等標準介面部件提供跨平臺支援。雖然它本身不是一個標準，但 XUL 所基於的都是標準，如 HTML 4.0、CSS、 DOM, XML 和 ECMAScript 等等。XUL 應用可以在流覽器上運行，也可以安裝在一個客戶電腦上。

XUL 是用來構建豐富動態使用者介面的語言。它是 Mozilla 流覽器的一部分，也是和 Mozilla 相關聯的應用程式，並且可以作為 Gecko 的一部分。

它可適應於所有版本的 Windows、Macintosh，還有 Linux 和其他的 Unix 作業系統。擁有了 XUL 和其他的 Gecko 元件，你就可以創建精密複雜的跨平台應用程式而不用使用特別的工具。

XUL 被設計用來構建 Mozila 應用程式的用戶介面，包括網頁瀏覽器、郵件用戶端和頁面編輯器，都可以透過 XUL 來創建。但是，它同樣可以被用來開發網頁應用程式，比如，當你需要能夠獲得網路資源時，並且需要提供更加豐富的用戶介面。就像 HTML，用 XUL 你可以使用稿本語言來構建一個使用者介面，使用 CSS 層疊樣式表來定義表現層，並且使用 JavaScript 來定義行為。你同樣具有權力通過網路來讀、寫遠端的內容。

但不同於 HTML，XUL 提供更為豐富的使用者介面控制元件，以下是 XUL 最常見的 UI 元件：

- 輸入控制元件，如 text box、check box。
- 工具列提供各類按鈕及內容說明文字。
- 功能表選單包含彈出式子功能表(pop up menus)。
- Tabbed 對話頁。
- 以樹型結構展現資訊關係。
- 鍵盤快捷列。

XUL 可以做到許多 HTML 無法表現的介面效果，例如樹型元件(Tree Element)可以使我們在瀏覽器網頁上做出與 Windows 檔案總管的瀏覽方式。在表格 2 中，描述如何以 XUL Tree 控制元件做出類似 Windwos 檔案總管的瀏覽效果。[4]

```
<tree flex="1">
  <treecols>
      <treecol id="firstname" label="First Name" primary="true" flex="3"/>
      <treecol id="lastname" label="Last Name" flex="7"/>
  </treecols>

  <treechildren>
    <treeitem container="true" open="true">
      <treerow>
        <treecell label="Guys"/>
      </treerow>

      <treechildren>
        <treeitem>
          <treerow>
            <treecell label="Bob"/>
            <treecell label="Carpenter"/>
          </treerow>
        </treeitem>
        <treeitem>
          <treerow>
            <treecell label="Jerry"/>
            <treecell label="Hodge"/>
          </treerow>
        </treeitem>
      </treechildren>
    </treeitem>
  </treechildren>
</tree>
```

表格 2：以 XUL tree element 做出類似 Windows 檔案總管的瀏覽效果

將以上文件放入 FIREFOX 瀏覽器後，顯示結果如圖 4。有了這些介面元件，可使我們將桌面應用程式移植到 WEB 應用系統，且不用犧牲只有在桌面應用程式才能運用的介面操作。

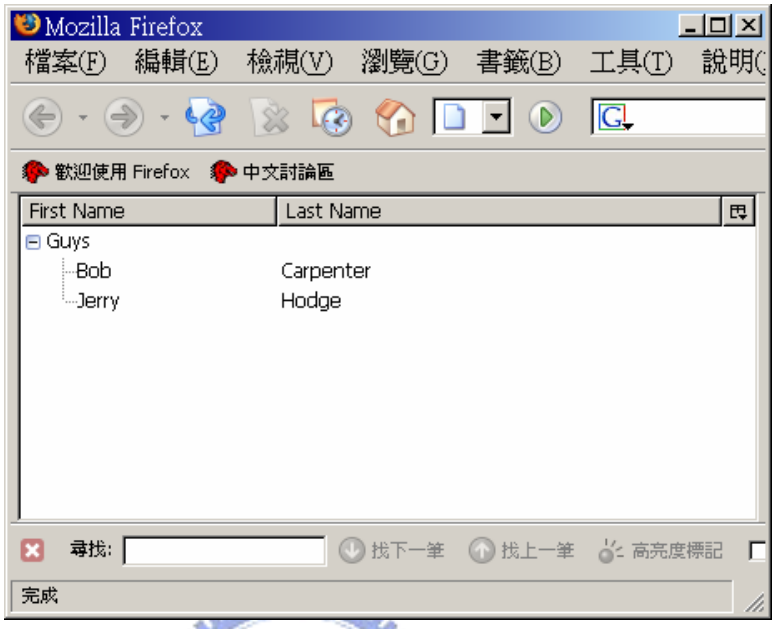

圖 4：XUL tree element 的執行結果

3. 通用性資訊系統研究內容與方法

為探討通用性資訊系統的發展規範，本論文以船舶資料庫應用系統為研究對象，瞭解其 WEB 化所可能面臨的問題。包括客製化的需要、需開發不同語言版本的資訊系統以及能與地理資訊應用程式相互傳遞訊息等。藉此我們歸納出發展通用性資訊系統所應具備的五項特質，並以此規範做為評估系統滿足通用性的程度。再訂定系統的開發環境及所採用的伺服器呼叫客戶端的技術。

3.1 船舶資料庫應用系統 WEB 化所面臨的問題

交通監控中心內設有許多監管人員。為達監管目的，監管人員使用的工作站通常由一台資料庫終端機(DB Client)配屬一台地理資訊終端機(GIS Client)所組成，其配置如圖 5 所示。地理資訊終端機（GIS Client）可提供海圖地理資訊，包含海岸線輪廓、水深、航道、禁航區、錨泊區等地理資訊（如圖 6）。除此類靜態地理資訊外，系統透過雷達、AIS 等設備，獲取航行中的船舶目標位置後，並將所有船舶位置動態標示於海圖上。監控人員可透過本終端機瞭解航行於海面上的船隻動態情形，並予以適當的導航服務。資料庫終端機（DB Client）可提供各類文字資料，包含船舶資料、航商資料、船舶預報、船舶動態訊息及氣象水文等（如圖 7）。

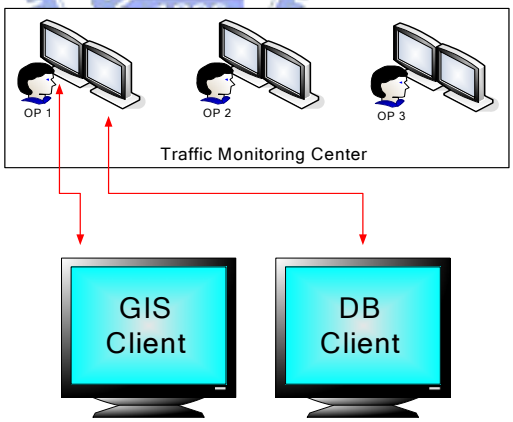

圖 5：交通監控管理中心工作站的配置

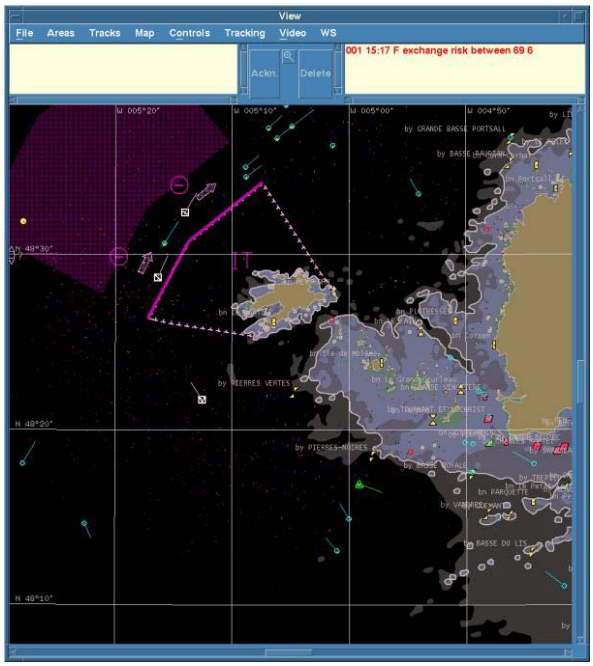

圖 6：地理資訊終端機畫面

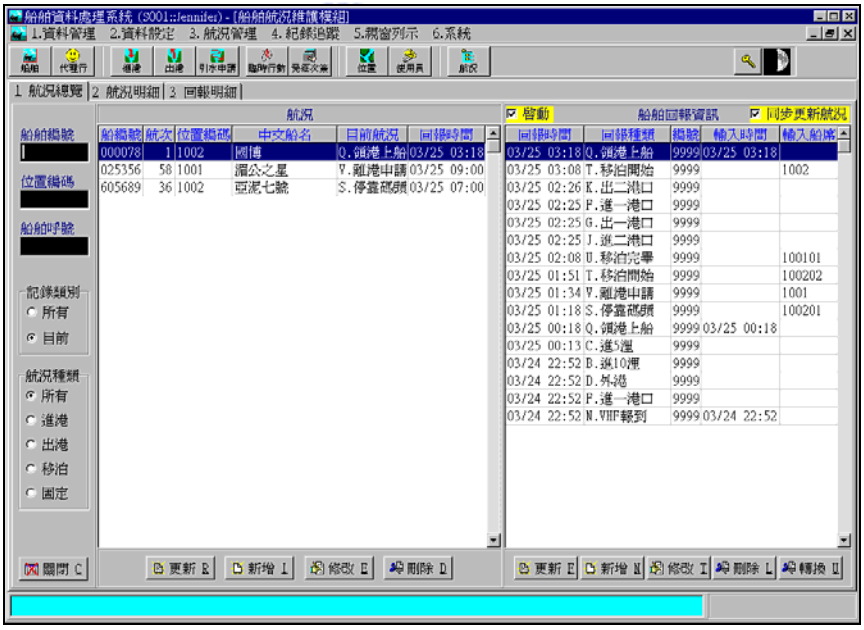

圖 7：資料庫終端機畫面

綜合以上要求，監控人員為能達成交通管制作業，其工作台位如圖 8 所示，並包含下述次系統。

1. 海圖地理資訊次系統 (GIS Client)
2. 船舶資料庫次系統 (DB Client)
3. 閉路電視次系統 (CCTV)
4. 無線線通訊次系統 (Radio)

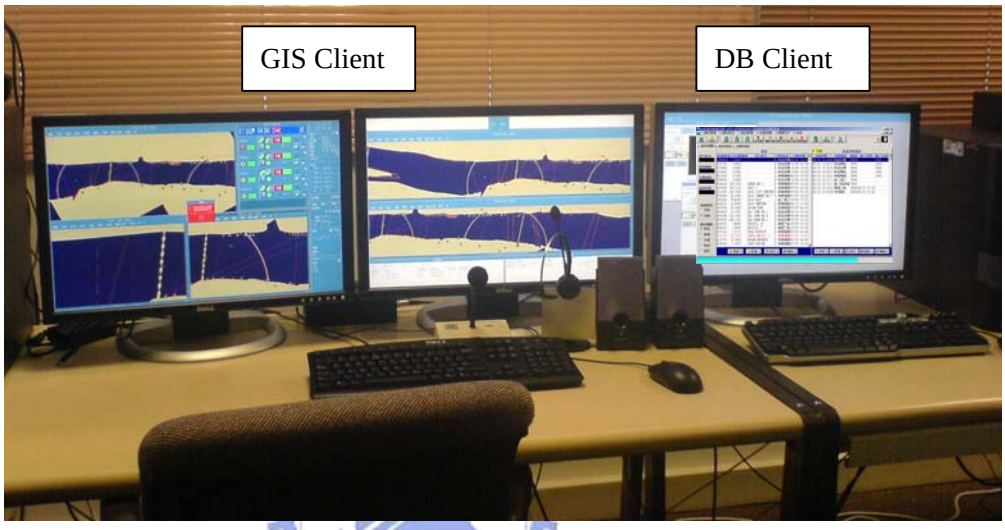

圖 8：船舶交通管理中心管制台位

地理資訊畫面與資料庫畫面需分於兩台終端機原因：

- 監控人員能全覽海上交通動態

  為使監控人員能全面掌控海域交通狀況，地理資訊畫面能即時提供所有位於港區內之船舶動態，並判斷船舶是否處於危險之情況。這些情況包括：可能與其他船舶發生碰撞、駛入禁航區、流錨等。因此，該畫面上並不適宜再擺置其他資料庫資訊，以免妨礙監控人員之作業。

- 資料庫次系統為客製化系統（Customerized System）

  雷達系統屬於高科技國防工業，目前擁有此發展技術的公司多來自歐美日等國，並受於該國政府的國防機密管制，禁止相關開發技術跨國移轉。是故雷達系統多為英文操作介面。

  但船舶交通管理作業多需配合各國港口的船舶資料庫資訊（如船舶資料、船舶計畫等）以達成監控作業。而這些資訊多以當地文字儲存資料庫內（如台灣採繁體中文、大陸採簡體中文、加拿大採法文等），因此資料庫畫面往往無法與原廠之地理資訊程式互相整合於同台終端機。且因各國法令不同，致使交通管制作業各有所異，開發廠商必須

與各港口詳談其管制作業方式，以發展出一套適合該港口的資料庫系統。

有鑑於船舶交通管理的資料庫次系統需進行客製化，並與原廠提供之其他次系統整合。以下是軟體開發人員所面臨的問題：

1. 如何發展出一套通用性的資料庫應用系統，以期能符合不同國家、不同港口的作業需要。該資料庫應用系統又應具備那些性質才能符合所謂通用性的要求？
2. 因開發廠商的客戶對象遍及世界，廠商開發完成一套資料庫應用系統後，針對不同的國家港口，能否僅需提供附加的語言資源檔(Language Resource Bundle)，就能有一套屬於該國語文介面的的資料庫應用系統？
3. 國際港口的船舶來自世界各地，該船舶基本資訊如船名等都有屬與該國的文字名稱。此資料庫應用系統是否能存放與處理不同的語文內容？
4. 因廠商維護人員所使用的語言可能與客戶不同。當客戶使用系統發生問題時，因其使用當地語言介面，維護人員要如何來瞭解客戶所遭遇之問題？
5. 各國客戶經濟發展情況不同，某些國家可能較為富裕，有能力購置較為先進的電腦設備，某些國家可能較為貧窮，僅能購置較便宜的電腦設備。廠商所開發之資料庫應用系統是否能在不同的硬體平台與軟體平台上執行呢？
6. 因目前地理資訊應用程式都採用第三代程式語言開發（如 C 語言）。如果軟體人員採用 WEB 技術來開發資料庫應用系統，其所面臨的另一項問題是：WEB 系統如何與傳統地理資訊應用程式相整合？同操作台位上的終端畫面彼此訊息應如何即時傳遞？
7. 監控軟體具有群組協同作業、即時監控兩項特性。當甲監控員對某船舶所設定的告警狀態，如何能透過 WEB 資料庫介面即時同步告知共同值班的其他監控人員？

本文主要在於探討如何發展出一套通用性的資訊管理系統，以解決上述問題。詳細說明如後。

3.2 方法說明

3.2.1 通用性資料庫應用系統的五項特質

針對軟體開發，本論文在此提出所謂通用性資料庫應用系統，應以能符合下述五項獨立性質(Independency)為目標：

1. 硬體獨立性 (Hardware Independency)
2. 作業系統獨立性 (OS Independency)
3. 使用者語言獨立性 (User Language Independency)
4. 資料庫管理工具獨立性 (DB Tool Independency)
5. 應用系統獨立性 (Application System Independency)

詳述各獨立性質如下：

1. 硬體獨立性:
   - 應用軟體不可直接控制硬體，需透過作業系統存取與管理不同種類的硬體設備。
   - 硬體獨立性可讓用戶在選用硬體時有更大的彈性。如經費較低時，應用軟體可搭配便宜的個人電腦 (Intel-based PC) 交給客戶。如經費較高時或應用系統屬重要關鍵系統(Mission Critical System)時，能採用較高等級的電腦來大幅提升應用系統的執行效率與可靠性。
2. 作業系統獨立性:
   - 應用軟體應可跨平台(cross-platform)且在不同作業系統下皆能執行(OS portability)。
   - 符合作業系統獨立性的應用軟體可在不修改、重編譯程式的狀況下，仍可在微軟視窗平台（如 Windows XP, Windows 2003 等）或 UNIX 平台（Linux, Solaris 等）上運作。
   - 作業系統獨立性好處在於可視用戶經費或需要，選取免費作業系統（如 Linux）或昂貴但管理功能強大作業系統（如 Windows 2003）。
3. 使用者語言獨立性:
   - 資料庫應用系統應為多國語言版本。配合客戶所在國家可採用該國語言版本執行，並能顯示與儲存該國文字資料與訊息。
   - 資料庫應用系統採用 I18N (internationalization) 及 L10N (localization) 模式技術開發與建置。[5]
     - 開發時應採用 I18N 模式(I18N Model) 開發。
     - 交貨時(delivery stage)，以 L10N 模式配置適用於客戶之資訊系統。
   - 為便於軟體業者維護作業，當系統安裝於客戶端時，除選用的該國語言版本外(nataive language)，另應配置一套預設語言版本（default language）做為維護人員與使用人員間的溝通模式，此預設語言通常為英語版。
   - 實施 I18N 模式時應符合：[5]

- 可顯示客戶所屬之國家語文 (native language)。
- 可輸入客戶所屬之國家語文。
- 可採用客戶所屬之國家語文顯示各類應用系統訊息。
- 有關日期、貨幣等顯示格式，應能符合客戶所屬之國家習慣。

■ 語言資源檔(Resource Bundles)：應用程式原始碼與訊息介面等文字內容應分開。針對不同國家搭配一套該國語文適用的語言資源檔。如需交付新語文版時，可依據現有語言資源檔重新翻譯製作一份。

■ 應用程式支援 Unicode 並採用 UTF-8 字元編碼。

4. 資料庫管理工具獨立性:

■ 與後端資料庫連結時應採用 JDBC 或 ODBC，如此後端資料庫管理系統就可使用各類型的資料庫管理工具。

■ 採用之資料庫管理工具應能符合 ANSI/ISO SQL92 標準並支援 Unicode。

■ 開發應用系統時，應注意所有程式碼都需 100%符合 ISO/ANSI SQL, ODBC 及 JDBC 等標準。如此才能保證所開發的資料庫應用系統能配合多類型的資料庫管理工具。（如： PostgreSQL, Oracle, Sybase, Informix 等）

■ 資料庫管理工具獨立性的優點是可配合客戶的預算或需要，交付不同之資料庫管理工具。也許是免費的 PostgreSQL 自由軟體，或是昂貴但高效率的 Orcale 商用軟體。

5. 應用系統獨立性:

■ 所開發的應用系統如可能與其他系統整合使用時，應使其保有運行獨立性質。

■ 舉例說明，船舶資料庫應用系統屬於船舶交通管理主系統下的次系統並與 GIS 次系統相整合。如考慮其運行獨立性可採下列三種方式販售給客戶：

- 應用系統可在不整合 GIS 的情況下，單獨販售給客戶作為船舶計畫與船舶報告的工具。
- 船舶資料庫應用系統也可搭配 GIS 系統一起販售給客戶。
- 如客戶已有舊 GIS 系統時，也可在瞭解此 GIS 介面規格後，與既有 GIS 系統整合使用。

3.2.2 通用性資料庫應用系統的開發環境

基於前節的性質要求與條件限制，本實作系統以三層式(3-tier)應用程式架構為基礎，採 J2EE platform 技術平台作為通用性資料庫應用系統之開發環

境。其架構圖如圖 9 所示。在作業系統環境方面，伺服器與客戶端電腦都採用 MS Windows XP。此外，另安裝 Red hat Enterprise Linux WS，觀察開發環境(IDE)與執行環境(Run-time Environment)跨平台功能是否滿足本系統通用性的要求。

客戶層以 Firefox 瀏覽器做為執行平台，可執行以 XUL 為基礎的 RIA 頁面程式。瀏覽器使用 HTTP 通訊協定連結到中間應用層(Middle Tier)，WEB 伺服器端採用 Apache Jakarta Tomcat 以作為 JSP 容器，以便管理動態性 HTML 內容 (dynamic-content HTML)。Servlet 可透過 JDBC 連結到後端層(Back-end Tier)的資料庫。本系統採用 PostgreSQL 做為主要的資料庫管理工具，同時在另一台電腦安裝 SYBASE 以觀察本系統連結不同後端資料庫種類時是否滿足獨立性的要求。此外 Servelet 也可透過 TCP/IP 協定連接到傳統桌面應用程式。本系統開發環境使用工具 (IDE – Integrated Development Environment) 採用 Netbean 4.0。

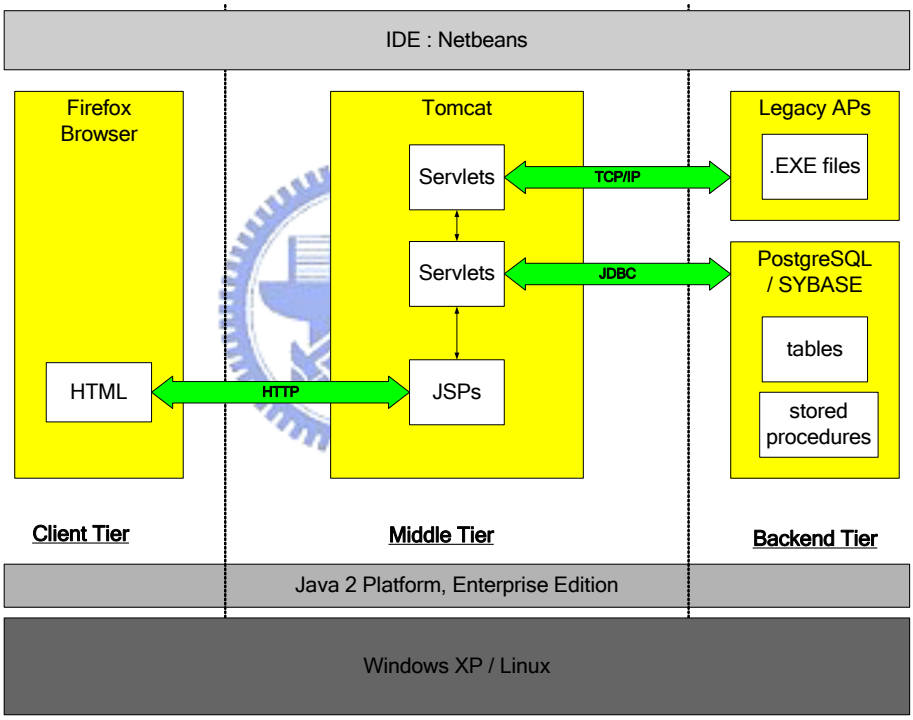

圖 9：J2EE 平台資料通訊協定架構圖

3.2.3 伺服器端主動呼叫客戶端瀏覽器所採用之技術

當地理資訊客戶端(GIS Client)需發送事件到資料庫客戶端(DB Client)時，卻受限於傳統桌面程式無法直接發送事件到 WEB 客戶端瀏覽器。在目前 WEB 應用系統下，地理資訊客戶端程式可透過 TCP/IP 協定與 WEB 伺服器端程式（如 Servlet）相互傳送訊息，但 WEB 伺服器端該如何將該訊息轉送到相對應的資料庫客戶端瀏覽器內呢？

目前伺服器端主動通知客戶端瀏覽器（Server-side notification to browser-client）可採用的技術有：Java applets 與 RMI (Remote Method Invocation)、CORBA 或配置一些其他公司開發的 TCP/IP messaging 元件，皆可達成目的。但這些技術都有開發困難、維護繁雜、瀏覽器相容性及防火牆管制等問題。

在此本系統採用 PUSHLET 機制解決上述問題。PUSHLET 機制主要採用 Java Servlet 開發而成，配合前端瀏覽器 JavaScript 及 HTTP Streaming 技術，WEB 伺服器可透過 PUSHLET Servlet 主動發出訊息到前端資料庫終端瀏覽器內。

PUSHLET 的作業原理與在瀏覽器內觀看多媒體影片的運作模式相類似，都是採用 HTTP 串流(HTTP Streaming)技巧。換言之，一般 HTTP 連結方式是瀏覽器取得 HTML 文件後隨即關閉連結；但就 HTTP Streaming 而言，將 HTTP 保持連結並隨時等待新的串流資訊從伺服器端送達客戶端瀏覽器內。也就是利用這種技巧，WEB 系統如採用 PUSHLET 機制，WEB 伺服器可發送事件到特定的客戶端瀏覽器。[14]

主要運作流程如下：

1. 客戶端瀏覽器向 WEB 伺服器訂閱(subscribe)想要收到的事件(Event)類別。並使用 HTTP 串流技巧與伺服器建立一個永久的 HTTP 連結。
2. WEB 伺服器保存所有客戶端瀏覽器相關資訊，如 IP 地址及訂閱事件等。
3. 當 WEB 伺服器端有新事件產生時，WEB 伺服器內的 Java Servlet 將查閱所有曾訂閱該事件的客戶端，並逐一送出事件到訂閱的客戶端瀏覽器。
4. 訂閱事件將以 JavaScript 的型態送達客戶端，瀏覽器是透過“執行”送達的 JavaScript 事件，搭配 DOM(Document Ojbect Model)更新 HTML 頁面。

客戶端訂閱事件流程(Event Subscription Scenario)

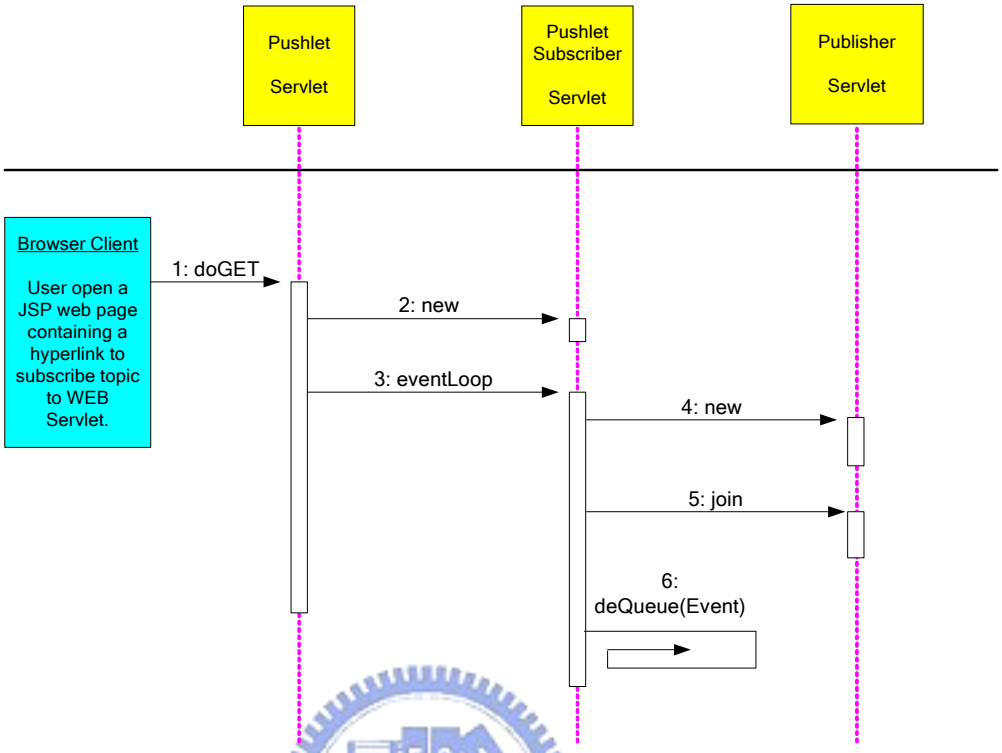

圖 10：訂閱事件循序圖

圖 10 說明當客戶端瀏覽器首次連結到 WEB 伺服器端時的訂閱事件流程。其流程依序如下：[14]

1. 使用者開啟網頁時，網頁內含有超連結(Hyper Link)以 doGET 方法呼叫 Pushlet Servlet，向 WEB 伺服器訂閱註冊。
2. 因為 Pushlet 可能同時收到許多不同客戶端的註冊請求，每收到一項請求時，它將個別建立一個新的 PushletSubscriber Servlet 來處理不同的客戶端。
3. PushletSubscriber 將持續執行直到 eventLoop 結束為止。
4. 在 eventLoop 程序內，PushletSubscriber 將要求 Publisher Servlet 建立一項事件類別，並加入(join)該事件類別。
5. 最後在 eventLoop 程序內，設計一迴路反覆執行 deQueue 程序。當 Queue 內無任何事件時，PushletSubscriber 將進入休眠狀態(Sleep state)，待有新事件產生時，再恢復執行。此新事件產生後的流程如圖 11所示。

伺服器端發佈事件流程(Publish Events Scenario)

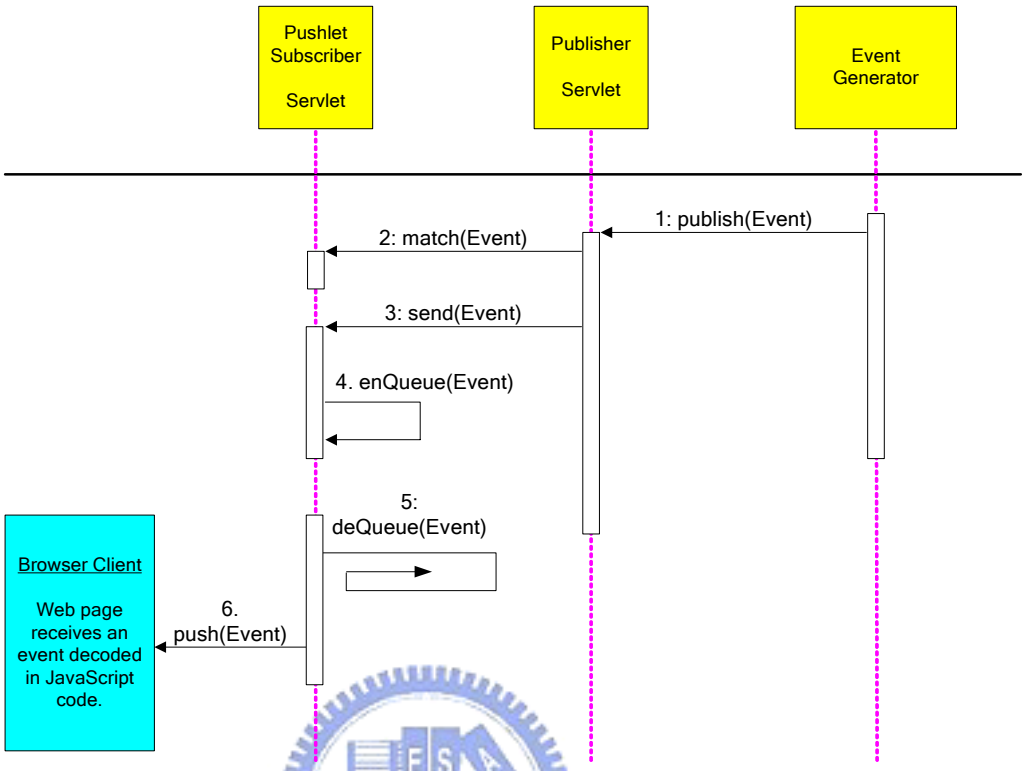

圖 11：發佈事件循序圖

圖 11 說明當有新事件發生時的發佈流程，依序如下：[14]

1. 產生事件的程式(Event Generator)呼叫 Publisher Servlet 的 publish 程序，發佈此項新事件。
2. Publisher 逐一檢查所有 PushletSubscriber Servlet，如符合其訂閱主題(Subscribed Topic)時，將呼叫 send(Event) 送出該事件。該新事件將放入 PushletSubscriber 的佇列中(enQueue)。
3. 之前 deQueue() 進入休眠狀態的程序將被喚醒，並將該事件以 JavaScript 的型態透過 HTTP Streaming 的方式送到客戶端瀏覽器內。
4. 客戶端瀏覽器收到 JavaScript 後，以執行該程式段的方式更新網頁內容。

4. 通用性船舶資料庫應用系統實作

於訂定前章所述系統通用性的五項規範後，並以伺服器主動呼叫客戶端瀏覽器的技術為基礎，我們設計出一套船舶資訊系統與地理資訊系統間能相互傳遞訊息的軟體架構。並藉由四種可能的介面訊息傳遞方式，來驗證所設計的架構是否已符合需要。在此，我們也規劃實驗環境的設備需求，並介紹實作系統的功能特色，然後再逐一評估其滿足通用性的程度，同時在與其他系統比較下，說明本系統確實更能符合通用性資訊系統的規範。

4.1 系統設計

4.1.1 軟體架構設計

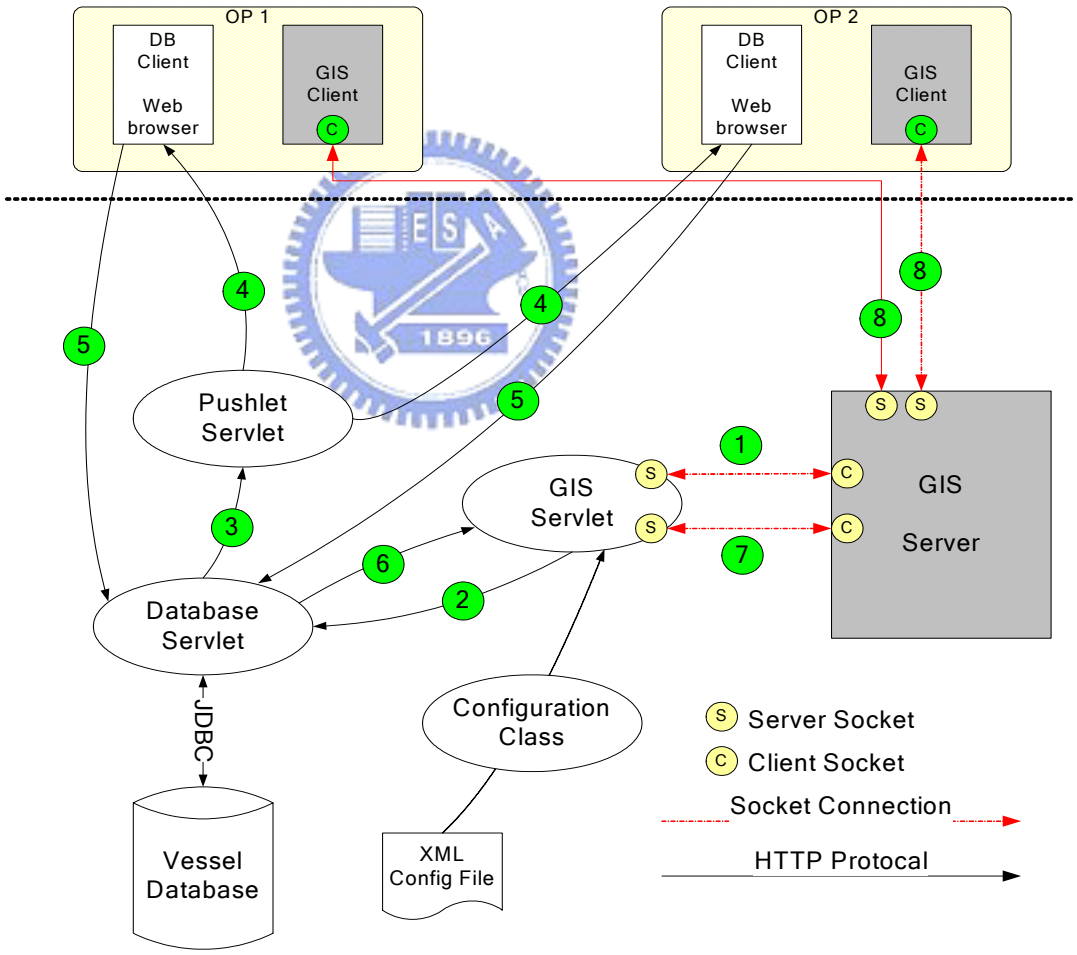

圖 12：GIS / DB 軟體介面訊息架構圖

圖 12 說明 GIS 與 DB 之間軟體介面訊息的傳遞方式。GIS 與 DB 之間的訊息介面主要是透過 GIS Servlet 來協調、傳遞。GIS Servlet 程式是採用 Java Servelt 標準所撰寫。本章內容主要在描寫 GIS Servlet 如何與 DB Server 及 GIS Server 協調溝通，以協助 DB 及 GIS 之間的通訊。簡述上圖功能如下：

- 灰色方塊內屬於 GIS 軟體系統範圍，係指外部開發軟體系統或現有商業 GIS 等軟體。
  - GIS Client 包括一套港區電子海圖(Electronic Sea Map)並配合最新之船舶動態交通顯示狀況。
  - GIS Server 提供 Socket Server port connection 等待 GIS Client 連線。因管制中心有多位管制員同時工作，每台管制員工作台都將連線到 GIS Server，
  - 當 GIS Server 啟動時，需自動透過 Socket connection 連線到 GIS Servlet。此時 GIS Server 扮演 client 角色。
- GIS Servlet 的 server socket connection。
  - GIS Servlet 與 GIS Server 之間，由兩條專用永久的 socket 所連結。(dedicated socket connections)
  - GIS Servlet 於啟動時，將產生兩個 server socket，並等待 GIS Server 連結。
  - 圖 12 內標籤 1 的連接線路，是 GIS Servlet 接收來自 GIS Server 的介面訊息命令(Interface Message Command)。當 GIS Servlet 收到命令後，將透過原連線傳送回應訊息(response message)給 GIS Server。
  - 圖 12 內標籤 7 的連接線路，是 GIS Servlet 傳送來自 DB Server 命令到 GIS Server，GIS Servlet 並需於原連接線路上，等待 GIS Server 回應訊息。
- GIS Servlet 於初次啟動時讀取設定配置檔(Configuration file)。所有管制中心內，各台工作台兩套系統的 IP 對應關係，都將定義於此檔中。詳見 4.2.1節內有關設定配置檔的敘述說明。
- 連線型態以虛實線段分類，其中 Socket Connections 都以虛線表示，包括標籤 1, 7 及 8 等連接線路；HTTP Connections 都以實線表示，包括標籤 2, 3, 4, 5 及 6 等連接線路。
- GIS Servlet 存取資料庫的方式採 HTTP request 到 Database Servelt 讀取或寫入各類資料。
- DB Client → GIS Client 之互動顯示(Cross Highlighting)：如操作員想知道在 DB Client 內某艘船舶位於 GIS Client 何處時，可在 DB Client 上選取某艘船舶，並在 GIS Client 中標示出所選取之船舶。參照上圖，其訊息流程為：DB Client 送出一個 HTTP 請求到 GIS Servlet，GIS Servlet 再透過 socket connections 送出訊息命令到 GIS Server，GIS

Server 根據指定之 IP，透過 socket connection 要求特定之 GIS Client 標示該船舶。

- GIS Client → DB Client 之互動顯示：同理，操作員可在 GIS Client 選取某艘船舶，並要求 DB Client 跳出視窗顯示該船舶明細資料。參照上圖，其訊息流成為：GIS Client 透過 socket 傳送訊息命令到 GIS Server，GIS Server 將再透過另個 socket 傳送訊息命令到 GIS Servlet，然後 Servlet 將傳送一個 HTTP 請求到 DB Servlet，然後 DB Servlet 將透過 Pushlet 將一個事件推到對應之前端 DB Client 上。

4.1.2 介面訊息傳遞方式

從圖 12 內我們得知，GIS Client 與 DB Client 間的訊息傳遞是透過其所屬的 GIS Server 與 DB Server 之間的通訊、協調來完成。其介面訊息依傳遞方向可分為兩種型態，分別是從 GIS 傳送訊息或要求到 DB，以及從 DB 傳送到 GIS。

從 GIS 到 DB 的訊息傳送包含以下兩種型態：

- GIS 到 DB 互動顯示 (Cross-highlighting)

  操作員希望藉由 GIS Client 上的船舶移動軌跡，連結到 DB Client 並顯示該船的資料庫內詳細資訊。

- GIS 事件廣播或儲存 (GIS event broadcast or save into database)

  GIS Server 發出 GIS 事件資料到 DB Server。如 GIS 內某艘船舶通過自動報告線(report line)或船舶碰撞警告等。

從 DB 到 GIS 的訊息傳送包含以下兩種型態：

- DB 到 GIS 互動顯示 (Cross-highlighting)

  在 GIS Client 上，操作員很難在海圖內的數百艘船舶中迅速找到某艘船舶。因此，操作員可在 DB Client 的船舶名單中，藉由所指定的船舶連結到 GIS Client 並標示該船舶的移動軌跡在海圖中的位置。

- 資料提供 (Data Supply)

  在某些應用情況下，DB Server 需定期主動提供最新資訊到 GIS。

  GIS Server 發出資料庫存取需求到 DB Server。如 GIS 需要讀取船舶資料，或是寫入船舶事件狀態到資料庫等。

各類訊息傳遞方式如以下各節說明。

4.1.2.1 GIS 到 DB 互動顯示

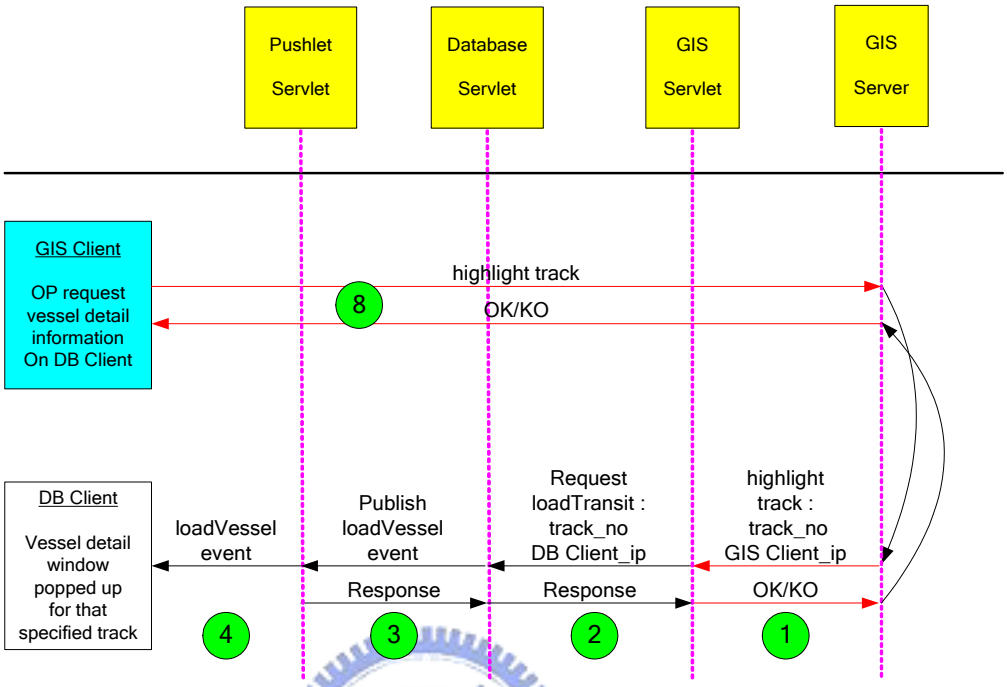

圖 13：互動顯示 - 從 GIS 發出命令在 DB 上顯示船舶明細資料

圖 13 描述了從 GIS Client 發出命令到 DB Client 的循序圖 (Sequence Diagram)。箭號線段上的標號（上圖中有 (1), (2), (3), (4), (5) ）代表了 «圖 12：GIS / DB 軟體介面訊息架構圖 » 裡的流程線段編號。其處理程序如下：

- 操作員在 GIS Client 以滑鼠雙擊選取的船舶軌跡。
- GIS Client 以 TCP/IP Socket 方式送出 highlight track 訊息到 GIS Server。
- GIS Servere 將 Client 的 ‘IP 地址’ 及所要求互動顯示的 ’船舶編號’ 再以 TCP/IP Socket 方式轉給 GIS Servlet。其中因 GIS Servlet 已是屬於 DB Server 的程式，流程至此等於訊息已從 GIS 轉達 DB Server。
- GIS Servlet 依據傳來的 GIS Client IP 地址轉換成預先定義在配置檔內的 DB Client IP， 並以 HTTP 方式對 Database Servlet 送出 loadTransit(tra ck_no, DB_Client_IP)命令。
- Database Servlet 再以 HTTP 方式對 Pushlet Servlet 提出發佈 (publish) “loadTransit” 事件的請求。
- 最後 Pushlet Servlet 會透過內部機制（詳見圖 11：發佈事件循序圖）將 “loadTransit” 事件送達指定的 DB Client。

- DB Client 收到 “loadTransit”事件後，會以 JavaScript 的方式迫使瀏覽器跳出一個船舶資料明細視窗畫面。

4.1.2.2 DB 到 GIS 互動顯示

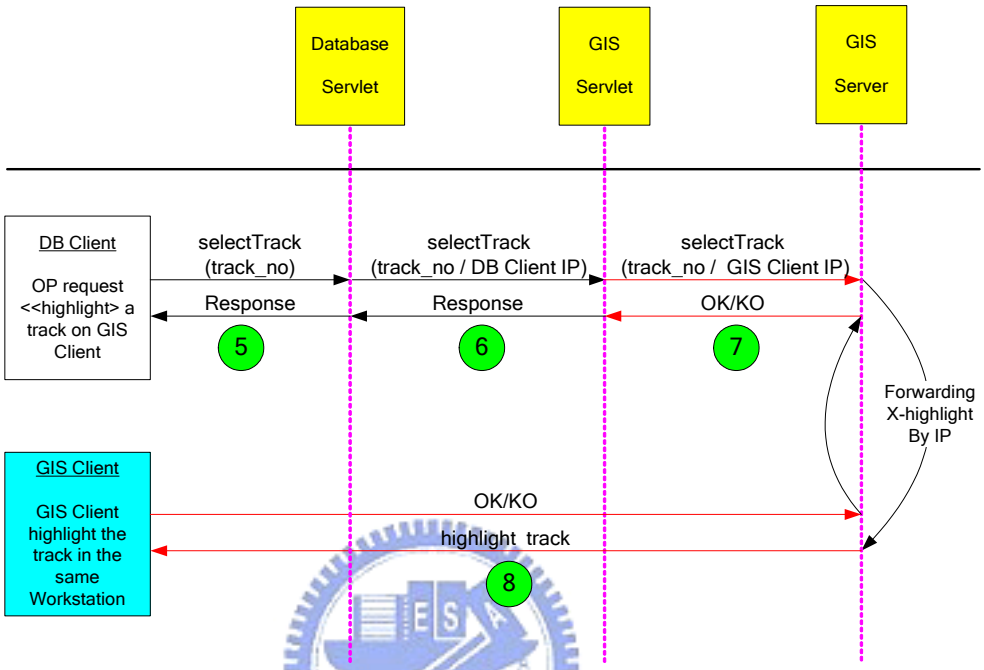

圖 14：互動顯示 – 從 DB 發出命令在 GIS 上標示船舶軌跡

圖 14 描述從 DB Client 發出命令到 GIS Client 有關互動顯示的循序圖 。其處理程序如下：

- 操作員在 DB Client 的船舶名單內，以滑鼠雙擊選取的船舶。
- DB Client 以 HTTP 請求(Request)方式，以船舶編號(track No)為參數送出 “selectTrack(track_no)” 請求命令到 Database Servlet。
- Database Servlet 得知請求 DB Client IP 地址，並再以 HTTP 方式，送出 “selectTrack(track_no, DB Client IP)” 到 GIS Servlet。
- GIS Servlet 依據傳來的 DB Client IP 地址轉換成預先定義在配置檔內的 GIS Client IP，並以 TCP/IP 方式送出 selectTrack(track_no, GIS Client IP) 到 GIS Server。
- GIS Server 依據 GIS Client IP 並以 TCP/IP 方式送出 highlightTrack(track_no)到指定的 GIS Client。
- GIS Client 收到後，依據 track_no 標示海圖內的指定船舶。

4.1.2.3 GIS Server 發出事件警告到 DB Server

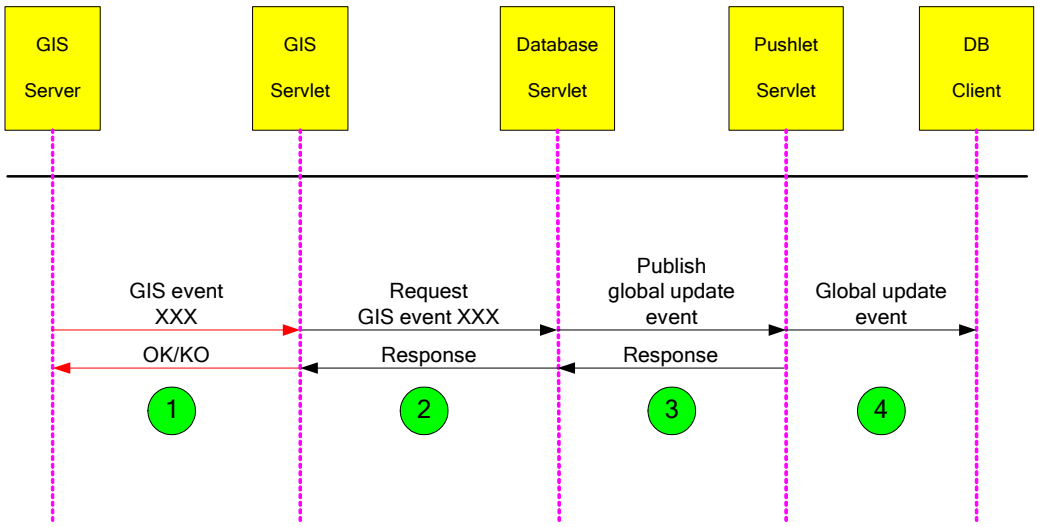

圖 15：GIS 發出事件警告到 DB

圖 15 描述從 GIS 發出警告事件到 DB 的循序圖。其中 XXX 代表警告事件，可能包括 GIS 監控下的船舶通過自動報告線或船舶碰撞警告等。其處理程序如下：

- GIS Server 偵測到某項警告事件後，以 TCP/IP 方式傳送到 GIS Servlet。
- GIS Servlet 以 HTTP 方式將此事件送到 Database Servlet。
- Database Servlet 再以 HTTP 方式對 Pushlet Servlet 提出發佈 (publish) “global update” 事件的請求。同時也將此事件儲存於資料庫內。
- Pushlet Servlet 會透過內部機制將 “globalUpdate” 事件送達訂閱此事件 DB Client。
- DB Client 收到事件後，會以 JavaScript 的方式採取必要的更新動作，如文字訊息更新、顯示告警視窗或於列表視窗 (Listbox) 內增加此事件等。

4.1.2.4 DB Server 送出請求到 GIS Server

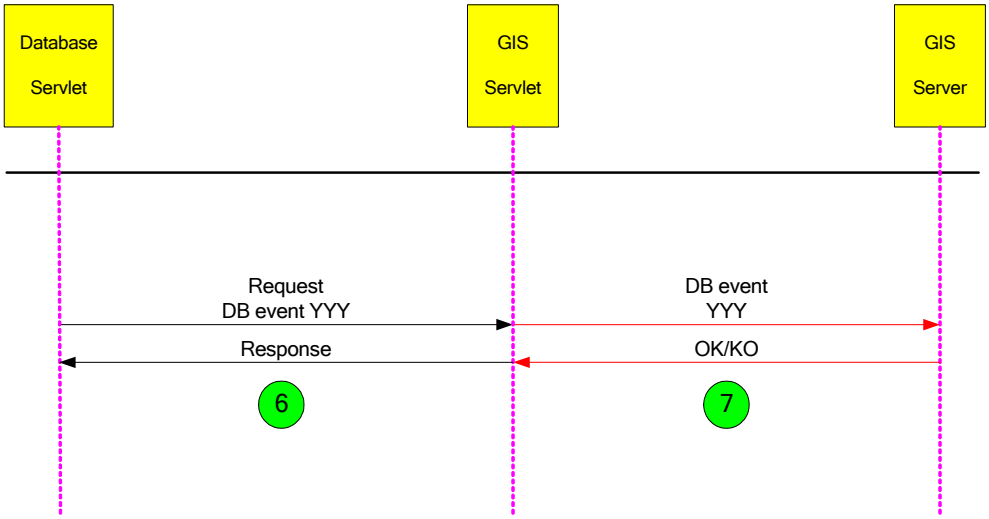

圖 16：DB 送出更新請求到 GIS

圖 16 描述從 DB Server 發出資料更新請求到 GIS 的循序圖。其中 YYY 代表更新請求，可包括更新 GIS 船舶軌跡內容（如中文船名更新、載貨資料更新等），或是發出特殊監控船舶請求等。特殊監控是指由資料庫設定並需於 GIS 上加強監控管制之船舶。其處理程序如下：

- DB Server 收到資料更新請求後，以 HTTP 方式將此資料更新傳到 GIS Servlet。
- GIS Servlet 再以 TCP/IP 方式傳送此更新資料到 GIS Server。
- GIS Server 可視功能需求，通知 GIS Client 處理此更新。

4.2 操作介面功能展示

4.2.1 實驗環境準備與設定

圖 17 描述實作本系統時所建置的實驗環境，該圖說明為模擬交通監控中心的網路環境，除一台伺服器外，另配置兩台工作站，每台工作站搭配一具 GIS Client 及一具 DB Client，共計五台電腦。

伺服器軟體配置如下：[1,2,6]

- OS: Red Hat Enterprise Linux WS 3, Windows XP Professional.
- DBMS: PostgreSQL v8.0.1, SYBASE ASE 11.9.2
- Web AP Server: Jakarta Tomcat v5.0.28
- Java Runtime : JRE 1.5.0-01
- Web Application : PIMX.war (Web Application Archive)

- GIS Server 模擬程式。

二具 DB Client 的軟體配置如下：

- OS: Windows XP Professional, Red Hat Enterprise Linux WS 3
- Browser: Firefox v1.0.6

二具 GIS Client 的軟體配置如下：

- OS: Windows XP Professional
- GIS.exe: GIS Client 模擬程式。

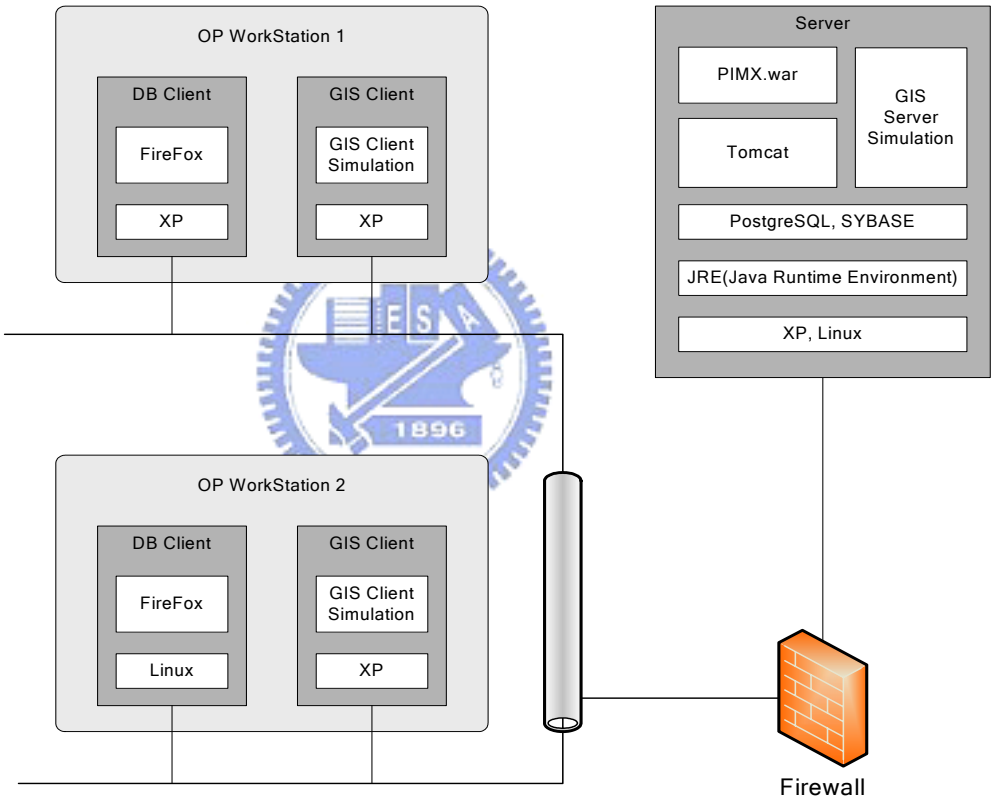

圖 17：實驗環境網路系統架構及軟體配置

由圖 17 內軟體配置可知，本系統採用輕量化用戶端程式架構(Thin Client)。採用此主從架構的設計方式，可使用戶端電腦負荷減輕，進而可使用較廉價電腦來工作。更大的優點是系統組態工作(System Configuration)集中在伺服器端執行，大幅降低系統管理人員的工作。

設定組態檔(Configuration File)

當伺服器端 GIS Servlet 被啟動時（透過 startDaemons Socket 訊息命令），將透過 config.class 讀取 GIS/DB paired IP 環境配置資訊。

環境配置檔採用 XML 格式，範例如表格 3 配置檔所示。取得組態設定資訊後，系統將可得知各工作站的終端電腦配對關係。

```
<?xml version="1.0" encoding="UTF-8"?>

<config>
   <SytarServlet>
      <IP_relation>
         <paired_IP_count>
               <value>3</value>
         </paired_IP_count>
         <relation_list>
               <IP_paired_1>
                     <gis>10.10.135.201</gis>
                     <db>10.10.135.101 </db>
               </IP_paired_1>
               <IP_paired_2>
                     <gis>10.10.135.202 </gis>
                     <db>10.10.135.102 </db>
               </IP_paired_2>
               <IP_paired_3>
                     <gis>10.10.135.203 </gis>
                     <db>10.10.135.103 </db>
               </IP_paired_3>
         </relation_list>
      </IP_relation>
   </SytarServlet>
</config>
```

表格 3：船舶資料庫應用程式環境配置檔

製作語言資源檔(Language Resource Bundle)

為測試資料庫應用系統於不同語言版本下的介面功能，本系統設計了三套語言資源檔，分別為：正體中文、簡體中文以及英文。開發時依照 I18N 的開發指南及其發展套件撰寫螢幕畫面程式。例如：

VesselForm.jsp（船舶畫面）的程式片段：

```
<textbox id    ="vessel.id"    hidden="true"  />
<label    value="<i18n:message key="basic_vessel_shipID"/>" />
```

本系統實作時建立三個語言資源檔，其定義方法如表格 4。當系統登入時，依照所選擇的語文種類，以 i18n 訊息鍵(message key)查詢對應的語言資源檔並顯示其標籤。表格 4 以 “basic_vessel_shipID” i18n 訊息鍵做為使用者畫面所應顯示的文字種類。

- 正體中文語言資源檔名：i18n-sypim_zh_TW.properties

  檔案內容片段：

```
# Vessel Basic File
basic_vessel_shipID=代碼:
basic_vessel_nameEnglish=英文船名:
basic_vessel_callSign=呼號:
```

- 簡體中文語言資源檔名：i18n-sypim_zh_CN.properties

  檔案內容片段：

```
# Vessel Basic File
basic_vessel_shipID=代码:
basic_vessel_nameEnglish=英文船名:
basic_vessel_callSign=呼号:
```

- 英文語言資源檔名：i18n-sypim_en_US.properties

  檔案內容片段：

```
# Vessel Basic File
basic_vessel_shipID=Ship ID:
basic_vessel_nameEnglish=English Name:
basic_vessel_callSign=Callsign:
```

表格 4：船舶資料庫應用系統所定義的三種不同語文資源檔

開發人員可依據不同國家將資源檔文字內容翻譯成該國語文，便可輕易做出不同語文版本的資料庫應用系統，如法文版、韓文版、日文版等。

4.2.2 與傳統桌面程式操作介面相同的客戶端網頁程式

資料庫終端操作介面如採用一般網頁程式以 HTML 為開發基礎，其操作介面將顯得呆板、反應時間緩慢，並受限於 HTML 特性需一個步驟一個步驟的 page request 來進行流程，這些因素都使得習慣於操作傳統軟體程式的使用者排斥使用網頁程式來取代其原有工作。

相較而言，本系統採用 RIA (Rich Internet Applicatoin)技術開發網頁程式，DB Client 仍是以瀏覽器為基礎的精簡型客戶端，但畫面操作元件豐富，且其操作流程與傳統桌面程式完全相同。

使用者登入主畫面後，其功能選項如下圖所示。

圖 18：DB Client 的功能菜單選項主畫面

圖 18 為 DB Client 的功能菜單畫面，在功能菜單裡，畫面的顯示及操作方式都與傳統桌面程式完全相同。在不需更新頁面的情況下，使用人員可以滑鼠瀏覽資訊系統的所有功能。在點選一階功能選項（如基本資料）後，系統也以不更換頁面的方式直接在該選項下，顯示二階功能選單。當使用者選取“船舶基本資料”選項後，出現圖 19 畫面。使用者可按下 [查詢] 鍵設定欲查詢的船舶條件。本系統會像傳統桌面程式般彈出一個查詢條件視窗供使用者輸入查詢條件。

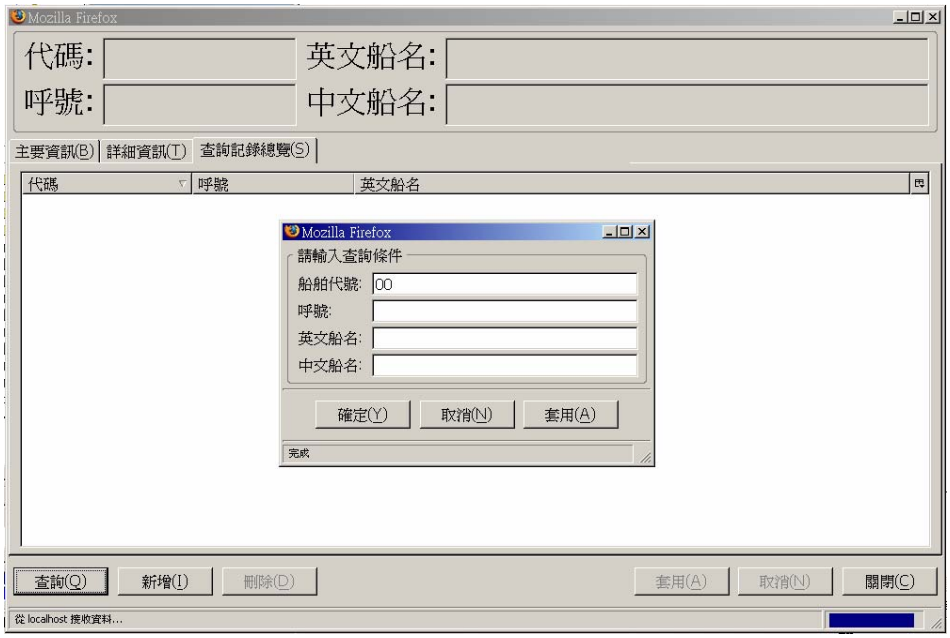

圖 19：船舶資料畫面

使用者輸入查詢條件後按下[確定]鍵，就把條件送到後端 WEB 伺服器內進行資料查詢處理，完成後本系統的 WEB 伺服器只把查詢的結果資料送到前端瀏覽器內，而非將整個頁面傳送到前端。瀏覽器收到後僅需更新欄位內容，不需將畫面全部重新顯示（結果如圖 20 所示）。因此，對使用者而言，將感覺系統反應時間變快，同時操作也比一般網頁程式順暢。

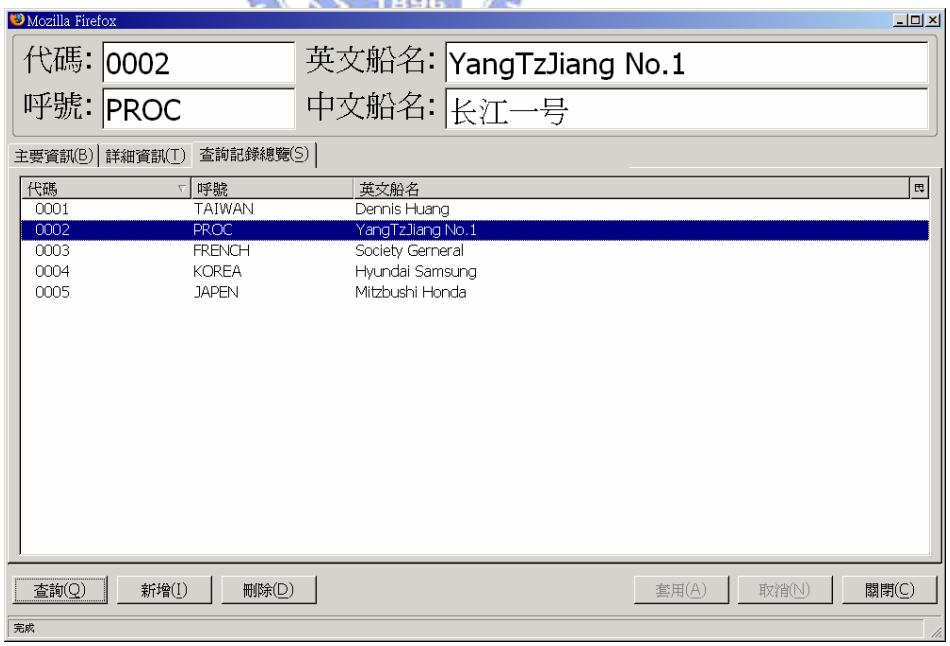

圖 20：顯示符合查詢條件的船舶畫面

圖 20、圖 21視窗畫面全採網頁程式，但操作感覺與傳統軟體程式完全相同，其反應時間與執行效率也與傳統 Client/Server 架構相當。

4.2.3 多國語文介面

於開發本系統時，已將與文字顯示相關的視窗元件如：選單文字、畫面標籤、按鍵標籤及訊息文字等獨立為一套 “語文資源檔” (Languange Resource Bundle)。對於開發人員，僅要專注於發展一套程式；如欲移植到不同國家所使用的語文時，只要將該語文資源檔翻譯成該國語文即可完成一套專屬於該國語文的資訊系統。

在本系統內，我們完成了三套語文資源檔，分別為正體中文資源檔、簡體中文資源檔與英文資源檔。圖 21 為系統開始時，出現以下畫面供使用者選取所要採用的語文介面。

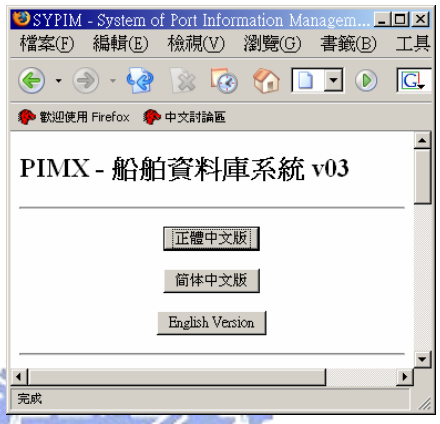

圖 21：使用者選取要採用的語文介面

正體中文介面可參閱前節 圖 19。當使用者選取簡體中文版時，隨後出現的文字訊息包括畫面選項、文字標籤、訊息提示等，都將以簡體中文顯示，如圖 22 所示。同理，如使用者選取英文版時，隨後的文字顯示也都將採用英文，如圖 23 所示。

圖 22：簡體中文版的船舶資料畫面

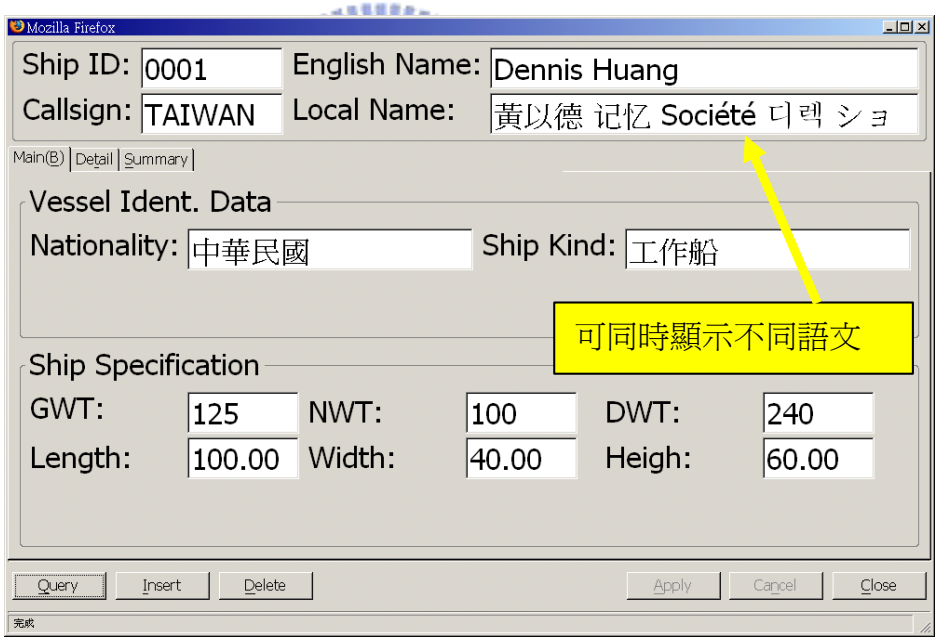

圖 23：英文版的船舶資料畫面

本系統除能顯示不同語文視窗與訊息外，亦能儲存與處理不同語言文字於資料庫內。為達成此功能，本系統一律採用 UTF-8 文字編碼方式來編寫網頁程式、製作螢幕畫面及做為資料庫的儲存設定。圖 23 中我們可以看到在 Local Name 欄位裡，可同時存放並顯示正體中文、簡體中文、法文、韓文及日文等文字。

4.2.4 客戶端瀏覽器畫面資料同步更新

所謂“客戶端瀏覽器畫面資料同步更新”是指當工作站使用者修改某欄位值後，WEB 伺服器內的 PUSHLET 機制用推播技術將修改的資料送到前方其他工作站，若其他工作站的使用者正好再瀏覽該筆記錄，則瀏覽器畫面資料將自動更新。為減少網路資料傳送流量，本系統僅對每次使用者有修改的欄位資料進行傳送作業，如此可大幅提昇效率。

在監控管理中心通常同時有三位以上監控人員值班，值班人員彼此間常需相互協調共同完成監控管理作業，因此於其工作站上，需能在監控人員不必按下更新鍵該客戶端瀏覽畫面就能自動即時顯示最新的資料狀態，否則可能在參考舊資訊的情形下做出錯誤的判斷導致嚴重的交通事故。

為符合監控管理的需要，本系統可在客戶端瀏覽器內達成畫面資料自動與其他工作站同步更新的功能。在圖 24 中，第一工作站監控人員負責與海上船舶進行 VHF 通訊，在船舶通訊員告知該船預定進港的時間後，監控人員予以輸入時間並按下“套用”鍵。資料庫客戶端將修改後的進港時間送到 Web Server 並寫入資料庫中。Web Server 在獲知啟德一號船舶的進港時間已修正後，透過 PUSHLET 機制把修正後的進港時間分別送到第二工作站與第三工作站的資料庫終端，其他工作站監控人員就可在無人工更新方式下獲取最即時的通報資訊。

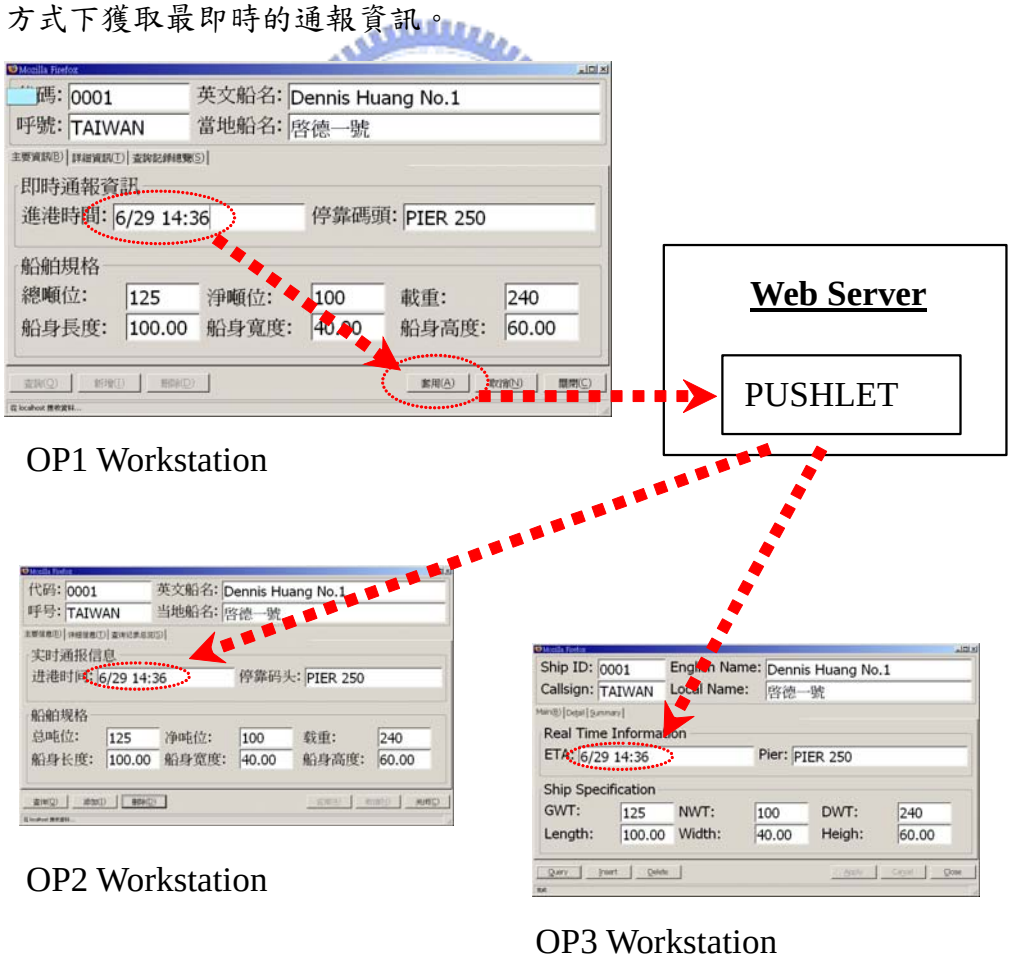

圖 24：修改進港時間欄位後，其他客戶端瀏覽器畫面資料自動同步更新

除此之外，自動同步更新可用在監控管理週邊設備的即時資訊蒐集與動態顯示。監控管理常見的週邊設備包括氣象設備、VHF 通訊設備、雷達設備等等，這些設備即時資料與狀態都是監控人員於進行管制作業時所需要的重要資訊。

4.2.5 網頁與客戶端桌面程式間的即時訊息交換

所謂即時訊息交換是指使用者可直接在網頁畫面上發送訊息到特定的應用程式並加以處理；反之亦然，也可以在傳統桌面程式發送訊息到特定的網頁畫面，並在網頁畫面接收訊息後，能做出對應的處理動作。如4.1.1節所述，我們提出一套 GIS 與 DB 終端的介面連接架構。在此架構下，以瀏覽器為基礎的資料庫終端頁面程式可與其對應的 GIS 客戶端應用程式相互傳遞訊息，進而處理、顯示。

在監控管理中心裡，管制人員於工作站上進行監控作業時，常需查看 GIS 客戶終端內某項動態船舶的相關的資料庫明細。為減少管制人員鍵盤輸入工作，如能點選 GIS 的動態船舶圖示並在 DB 終端畫面上立即顯示，將可大幅提升工作效率。此外，如船舶聯繫管制人員時，管制人員應於資料庫終端輸入相關資訊，並可在資料庫終端點選後，於 GIS 終端機內，將對應的動態船舶目標圖示將以閃爍的方式顯示，以方便操作人員瞭解該船舶在海圖內動態位置。

圖 25 說明當操作人員欲瀏覽 Ship 0003 的資料庫明細內容時，可於 GIS 終端程式以滑鼠雙擊 Ship0003 動態船舶圖示，GIS 終端程式發出請求到 GIS 伺服器，再透過 WEB 伺服器的推播技術選定對應的資料庫終端電腦，並促使該資料庫終端自動查詢並顯示 Ship0003 的資料庫明細內容。

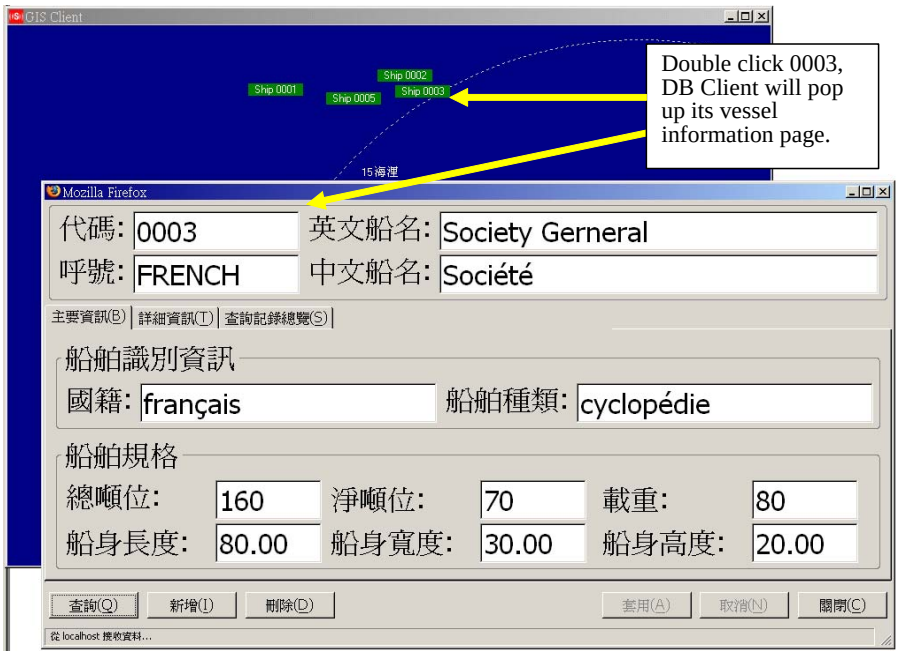

圖 25：於 GIS Client 雙擊 Ship 0003 後，DB Client 自動彈出該船資料明細

反之，在圖 26中說明當操作人員欲從資料庫終端的船舶總覽畫面裡得知其位於 GIS 終端電腦的動態圖示位置，可用滑鼠雙擊 0002 船舶，完成後對應的 GIS 終端電腦內該動態船舶圖示將以閃紅的方式顯示，以便操作人員立刻知道該船位於海圖何處。

圖 26：於 DB Client 雙擊 Ship0002 後，GIS Client 閃紅該船舶圖示

透過以上的操作展示，我們使得網頁程式與傳統桌面程式間能相互傳遞訊息，並使我們相信網頁程式已能完全取代以往只能由應用程式所達到的功能。

4.3 系統評估

本系統經過系統分析、設計、建置與實作後，完成一套實驗應用系統通用性的程式。為了評估系統實作成效，將與其他國家所使用的船舶資料庫系統作一比較。以下是其他國家港口使用中的船舶資料庫應用系統之簡述。[8,9]

- 本系統 – 通用性船舶資料庫應用系統

  開發環境採用 Netbeans, J2EE。後端 DBMS 採用 PostgreSQL，前端軟體使用 Firefox 瀏覽器。

- 高雄港 – 高雄港船舶資料庫應用系統

  開發環境採用 Delphi，後端 DBMS 採用 SYBASE、前端軟體為編譯後的 Delphi 桌面程式。

- 挪威 – Navtek 港口資訊管理系統

  開發環境採用 Visual Basic + MS-C，後端 DBMS 採用 MS-SQL，前端軟體為編譯後的 VB 桌面程式。

- 中國青島 – 青島海事局 VTS 船舶交通管理輔助信息系統

  開發環境採用 ASP，後端 DBMS 為 MS-Access，前端軟體採用微軟 IE 瀏覽器。

在本文 3.2.1節 通用性資料庫應用系統的五項特質裡，我們提出了五項規範來評估資訊系統是否滿足“通用性”的要求。茲就這五項規範逐一評估本系統與其他港口系統的差異。

在硬體獨立性的比較中（如表格 5），本系統因採用 J2EE 環境及一律採用 Open Source Software（如 Netbeans, Tomcat, PostgreSQL 等）所以從 PC、Macintosh 到工作站或大型電腦都可執行。如此，本系統可因應不同客戶等級要求，在客戶預算充裕且希望硬體速度快、穩定時，則可建議客戶採用昂貴的大型電腦。反之，若預算較少時，可採便宜普及的個人電腦做為執行平台。

高雄港系統因採用 DELPHI 開發，其桌面程式僅能在 Windows 上執行，故無法於中型工作站或大型電腦上執行。而後端 SERVER 部分因採用 SYBASE 資料庫其系統軟體本身及支援各型電腦。挪威系統因採用 VB、MS-SQL 等微軟產品，所以不論是客戶端或伺服器端的程式都無法在非 Intel-based 的個人電腦上執行。青島系統的執行問題與挪威系統相同。

其次是作業系統獨立性的比較（如表格 5），我們發現本系統除的伺服器端軟體除無法在 Windwos98 上執行外，其餘各類作業系統都可執行。挪威系統因採用微軟產品 MS-SQL、VB，所以無法在 UNIX 的環境下開發與執行。青島系統則採用更低階的 DBMS: MS-ACCESS，所以還能在已瀕臨淘汰的 Windwos 98 上開發與執行。

比較項目	本系統		高雄港		挪威		青島	
	Client	Server	Client	Server	Client	Server	Client	Server
硬體獨立性 (Hardware Independency)								
PC: Intel-based	✓	✓	✓	✓	✓	✓	✓	✓
Macintosh	✓	✓		✓				
工作站: Sun Ultra 80 wks	✓	✓		✓				
大型電腦: SGI Origin 2000	✓	✓		✓				
作業系統獨立性 (OS Independency)								
Windows 98	✓						✓	✓
Windows XP	✓	✓	✓	✓	✓	✓	✓	✓
Windows Server 2003	✓	✓	✓	✓	✓	✓	✓	✓
Solaris	✓	✓		✓				
Linux	✓	✓						
Mac OS X	✓	✓						

表格 5 : 硬體獨立性、作業系統獨立性比較表

船舶交通管理系統的地理資訊終端軟體多以 UNIX 來開發，跨作業系統執行能力的另一項好處是，資料庫應用程式可以直接在地理資訊終端上用瀏覽器執行，高雄港、挪威及青島系統因使用 MS-Windows 作業環境無法做到。

使用語言獨立性的比較上（如表格 6），本系統提供三種語文資源檔；並於需要時可很容易的編寫其他國家所使用的語文檔，如法文、韓文等。編寫語文檔對於程式本身並無任何影響。相對而言，高雄港僅能提供正體中文版，挪威僅有英文版，而青島則只有簡體中文版。因此當開發這些軟體的業者，如果要發展其他語文版本時，將付出極高的代價。

此外，本系統因為同時提供英文版，可以做為原廠維護人員與客戶之間的溝通交流，對於原廠人員在維護客戶程式時，也便利許多。至於資料庫而言，因本系統全採用 UTF 編碼方式儲存文字，所以可以接受萬國語文於同一記錄內。而其他系統則僅能存放單一的語言文字。

至於資料庫工具獨立性的比較（如表格 6），本系統開發時，所有程式碼都 100%符合 ANSI/ISO SQL92、ODBC 及 JDBC 標準，因此後端不論採用哪種資料庫，只要能符合 SQL92 標準，本系統就能夠透過 JDBC 予以連接。這樣的好處是可配合客戶預算，選取適當的資料庫。至於高雄港、挪威及青島系統因程式開發時已使用其選用資料庫的專有特性，如要改採其他資料庫，則需花費極大的成本來改善。

最後是應用系統獨立性的比較（如表格 6），除高雄港系統因為設計時已與地理資訊系統緊密結合而無法單獨使用外，其餘三系統皆可在不搭配 GIS 的情況下單獨使用。目前四個系統都必須配合專屬的 GIS 及港口客戶來使用。

本文“5.2節、未來研究方向探討”裡，提及三項資訊系統客製化的概念，如能將此三項概念納入本系統之設計內，則將可不限定僅能與原廠 GIS 相整合；亦可在和客戶討論需求分析後，以專案客製準備的方式，在不更改原有程式或增加少許新需求程式的狀況下，完成一套專屬該客戶的資料庫應用系統。如客戶已有自己的 GIS 系統，也可以在瞭解該 GIS 的介面規格知識下，與既有之 GIS 系統整合使用。

比較項目	本系統	高雄港	挪威	青島
使用語言獨立性 (User Language Independency)				
正體中文	✓	✓		
簡體中文	✓			✓
English	✓		✓	
資料庫工具獨立性 (DB Tool Independency)				
MS Access	✓			✓
PostgreSQL	✓			
MS-SQL	✓		✓	
SYBASE	✓	✓		
應用系統獨立性 (Application Independency)				
可單獨使用	✓		✓	✓
不限定與原廠 GIS 整合				
不限定客戶				

表格 6：使用語言獨立性、資料庫工具獨立性、應用系統獨立性的比較

在開發平台的比較裡（表格 7），本系統採用 Netbeans / J2EE 做為開發平台，除 Windows98 無支援 J2EE 平台外，其餘 OS 皆可使用。其餘三系統皆採用微軟系統做為開發環境，無法在其他 OS 上進行開發。其中青島系統因採用 ASP，所以也可在 Windows 98 上進行設計。

特別注意的是本系統可在不同語文版 OS 的平台上開發，例如可在簡體版上開發，再把產生的執行檔(.WAR)移到任何 OS 包括繁體版上執行。這樣的好處是一份專案可以同時發包給許多國家的軟體業者來共同完成，完成後並不需要在其他語文版 OS 上重建執行檔(rebuild run-time code)。

在執行環境的比較裡（表格 7），本系統採用 RIA 技術開發前端頁面程式，所以互動操作屬於無接縫式(Seamless Inteactive)，其操作感覺與 DELPHI、VB 等所產生的桌面程式相同。相對而言，青島使用 ASP 網頁程式設計，所開發出來的應用系統架構為 HTTP 為基礎，在選取功能或瀏覽資料時，都會將整個頁面重新整理，造成使用者操作上的不順暢。

本系統的另一項特色是當使用者更新資料時，其他的使用者上的 DB 終端畫面可自動更新。相對而言，其他三個系統都必須以人工方式提出更新需求才可看到最新的資料。還有一項特色，本系統雖屬 WEB 應用程式，但

前端瀏覽器仍可與 GIS 客戶端互動顯示與傳遞資訊；青島系統雖也屬 WEB 應用程式，但是卻無法在瀏覽器內與 GIS 客戶端有任何系統間的互動、傳輸。

至於防火牆限制方面，本系統以瀏覽器執行，只要防火牆開放 HTTP port 服務，則使用人員可在防火牆外以瀏覽器連線到本系統的伺服器來執行，且不影響與 GIS 客戶端的互動顯示。高雄港與挪威系統採用 Client/Server 作業模式，如果防火牆不開放 DBMS service port，則客戶端將無法穿透防火牆連線到伺服器端。

最後一項比較，本系統因包括顯示、處理及儲存全都採用 UTF-8 內碼，因此可在支援 Unicode 的 OS 上直接執行。高雄港的程式採用 Big5 內碼，青島系統的程式採用 GB 內碼，因此在不支援該內碼的 OS 下執行時將無法正常顯示中文。挪威系統因為只有英文版，故不予比較。

比較項目	本系統	高雄港	挪威	青島
開發平台 (Development Platform)				
Windows 98				✓
Windows XP	✓	✓	✓	✓
Windows Server 2003	✓	✓	✓	✓
Solaris	✓			
Linux	✓			
Mac OS	✓			
可在不同語文版 OS 平台上開發	✓			
執行環境 (Run-time Environment)				
無接縫互動操作 (Seamless Interactive)	✓	✓	✓	
客戶終端資料自動同步更新	✓			
GIS 客戶端互動顯示	✓	✓	✓	
可以瀏覽器執行	✓			✓
穿透防火牆限制	✓			✓
可在不同語文版 OS 執行	✓		n/a	✓

表格 7：開發平台、執行環境的比較

5. 結論與未來發展

5.1 結論

國際化是資訊服務業者開拓市場並往高階市場邁進的必經之路，軟體產業在國內市場太小，為擴大市場規模其機會應在國際市場。所以軟體公司只有大型化、國際化才有競爭力與生存空間。因市場擴大造成客戶對象繁雜，軟體系統在國際化後所面臨的挑戰就是如何解決研發成本與維護成本擴張的問題。為降低成本，軟體業者都期望能發展出一套通用性資訊系統，可適用於不同硬體平台、作業系統甚至國家語言等。

我們以本論文所提及的五項通用性規範為基礎，實作一套通用性船舶料庫應用系統，並將完成的實作系統在不修改任何程式的情況下，觀察其在不同的硬體、作業系統、資料庫工具及使用者語言間的執行情形。在這些不同環境下，本系統都能有一致的優良執行表現。

在這套實作系統裡，我們採用 RIA 技術來做為 WEB 應用系統的使用者介面開發。使得本系統在網際網路上運行時，有著如同傳統桌面應用程式的行為、功能、快速回應、直覺與體驗，也融合了網際網路應用程式的容易開發的與低成本的特性。更重要的是能使那些習慣於操作傳統軟體程式的使用者不再排斥使用 WEB 應用程式。

除了改善 WEB 應用系統的操作介面外，本文也探討客戶端瀏覽器內的網頁程式如何與傳統桌面程式相互傳遞訊息，以達互動作業的功能。本實作系統捨棄一般常用的 Java Applets/RMI 或 CORBA 等複雜的技術，採用網頁推播技術(Push Technology)來解決此問題。在此推播機制下，簡化了系統開發及維護，並提升了軟體的相容性也解除防火牆管制等問題。使得本系統更能符合通用性資料庫的規範宗旨。

經由本系統實作時的開發經驗與完成結果分析，已能確定通用性資訊系統的發展可能與效益，達到軟體業者降低研發成本及簡化維護作業的目的。

5.2 未來研究方向探討

未來可在本實作系統所完成的基礎上，納入以下的考量，以期進一步強化資訊系統的通用性。以下是未來發展可改進及建議的部分：

- 以“服務代理程序”為基礎的模組設計(Service agent-based Modular Design)

  因客戶的不同需求，為便於資訊系統客製化(customization)，其過程如能採用類似物件導向的觀念，將資訊系統分割成許多的服務代理程序。每個服務代理程序都可被設定為功能啟動或關閉。這些服務代理程序的啟動狀態可定義於應用系統配置檔裡。

- 系統功能與行為以外部參數檔來設定(Parameters Provision by External File)

  應儘可能將系統的行為定義於外部參數檔內，例如讀取氣象設備的週期、提供語音警示、業務流程等。於系統啟動時，將讀取外部參數檔以決定資訊系統的功能與行為。

- 使用者介面可因人而異 (Defferent Presentation Views from Users)

  當使用者登入後，除權限控管外，更可根據系統的事先定義，對於同一程式畫面以不同的呈現方式出現，包括顯示風格、欄位是否顯示、資訊顯示方式等。

- 客戶端資料庫引擎 (Client Database Engine)

  目前系統對於多筆資料的傳送方式，採取後端 WEB 伺服器以 RDF (Record Definition File)的格式把多筆記錄傳送到前端瀏覽器的 XUL Tree Element 內。如果前端使用者查詢條件過寬，將造成後端伺服器一次傳送大量資料記錄到前端而佔用網路頻寬。

  改善方法可考慮在前端的 JavaScript 內設計一組中間的資料庫引擎，負責與後端的 Servlet 聯繫，做為中間媒介控制每次傳送的資料筆數。

- JAVA 的報表產生工具 (Report tool in Java)

  本系統並未實作報表，未來可研討如何以 JAVA 的報表產生工具來做為發展通用性資訊系統的一部分。

參考文獻

[1] Inderjeet Singh, Beth Stearns and Mark Johnson, “Designing Enterprise Applications with the J2EE Platform, Second Edition.”, 2003.

[2] 鄭吉峰, “Java Server Page 觀念與應用實務”, 學貫出版社, 2002.

[3] 王佑中, “Web 動態技術入門”, 第三波出版社, 2001.

[4] Mozila Foundation, “Introduction to the XML user Interface Language (XUL)” http://www.xulplanet.com

[5] Tomohiro KUBOTA, “Introduction to i18n” http://www.debian.org/doc/manuals/intro-i18n/

[6] SQL Support in Mozilla http://www.mozilla.org/projects/sql/index.html

[7] JavaScript Gude http://wp.netscape.com/eng/mozilla/3.0/handbook/javascript/

[8] 高雄港務局全球資訊網 http://www.khb.gov.tw/

[9] 青島科技港全球資訊網 http://www.stport.net

[10] 交通部民用航空局飛航管理系統計畫 http://www.anws.gov.tw/CNSATM/ATM_1_chinese.htm

[11] 黃宗福, “智慧型 e 化勤務派遣系統”, 2003.

[12] 奚江華, “微軟 ASP.NET 2.0 的 AJAX 利劍 ～ Atlas Framework”, 2005.

[13] Markus Kuhn, “UTF-8 and Unicode FAQ for Unix/Linux”, 1999.

[14] Van Den Broecke, “Pushlets - Whitepaper”, pages: 5-8, 2002.

[15] Jess james Garrett, “Ajax: A New Approach to Web Applications”, pages: 3-6, 2005.

[16] Kevin Mullet, “The Essence of Effective Rich Internet Applications”, page 11-15, 2003.

[17] Christine Perfetti and Jared M. Spool, “Macromedia Flash : A New Hope for Web Applications”, page 6-8, 2005.

[18] 吳信輝, “網頁技術的新趨勢—RIA（Rich Internet Application）”, page 2-5, 2004.

[19] Patrick Niemeyer & Joshua Peck, “Exploring JAVA”, page 120-165, 1997.

[20] Bert Jenninga, Han-Jin Lee, Thomas P. Marian and Valeriy Latypov, “Report on The 10th International Symposium on Vessel Traffic Services”, page 11-15 , 2004.

[21] Thomas P. Marian, “The Role of VTS in Network Centric Commerce”, page 3-6, 2004.

[22] David N. Lloyd, “Integrating VTS with Port Services”, page 10-15, 2004.

[23] Harry Fuecks, “Web based XUL Filemanager”, 2004.

[24] Jim Inscore and Nicholas Kassem, “Designing Enterprise Applications with the J2EE Platform”, 2002.