以 XML 資訊物件為基礎之事件通報與內容整合系統
An Event Notification and Content Syndication System based on XML Information Object

陳志達　蔡尚榮　杜勇進　陳震洋

南台科大資管系　長榮大學資管系　崑山科大電子系　成功大學電機所

andypony@mail.stut.edu.tw　srtsai@mail.ncku.edu.tw　tu@turtle.ee.ncku.edu.tw

摘要

本文研究如何運用發佈/訂閱之非同步通訊模型及 XML 儲存系統來實現一個以 XML 資訊物件為基礎之事件通報與資訊整合系統。由於一個完整的資訊通常散佈在不同的應用中，以目前在網際網路上利用搜尋引擎蒐集資料的方式，使資訊消費者耗費許多時間在重複過濾大量的資訊。內容整合的目的在於蒐集不同應用的資訊內容，組合成為滿足不同需求的其他應用。系統同時將整合好的 XML 資訊物件透過主動撮合與通報的機制，使資訊消費者能方便且有效的獲得所需的資訊。此外，系統以 XML 資訊物件作為資訊整合的基本單位，使資訊整合者能精確的定義一個完整的 XML 資訊物件。在系統中提供的整合工具能針對不同的需求快速的建立不同的整合應用而不用撰寫程式。研究中透過 X-Bind 的運作產生一個符合整合應用的虛擬資訊集合；系統同時引用 XML 儲存系統來管理訊息，對長期性訊息的維護提供一個良好的解決方案。

關鍵字：XML 資訊物件，內容整合，X-Bind，虛擬資訊集合，XML 儲存系統

Abstract

In this thesis, we apply an asyn- chronous publish/subscribe communication model and the XML technology to build an event notification and content syndication system based on XML technologies. In general, the completed information is distributed in many separated services. Now on Internet, after collecting information by a search engine, information consumers often spend much time on filtering a large number of raw data.

The purpose of content syndication is to collect information from separated services and syndicate the satisfied contents according to user's different demand. Moreover, this system adopts an event brokering and notification service, thus an information consumer can get syndicated information more convenient and efficient. Using the XML Object as a content syndication grain, the information consumer can define a completed XML Object precisely. We also provide a syndicated tool that built integrated service according various application requirements without programming efforts. In this thesis, a Virtual Collection is built through X-Bind operation to matching integrated service and the XML Storage System as the message exchange center to provide effective management for persistent event messages.

Keyword: publish/subscribe communication model, XML Object, content syndication, Virtual Collection, X-Bind, XML Storage System

1. 簡介

伴隨著網路技術的迅速發展，網際網路已經逐漸與人們生活的各個部份緊密相連，每個人可以看到其他人貢獻的資訊，也可以參與資訊的提供。雖然每個人的貢獻有限，但集合起來的資訊量卻是非常龐大的，這就是網路社群的力量與魅力所在，

也因為這個廣大的網際網路社群不斷的擴增，越來越多人利用網際網路分享與蒐集資訊，因此，資訊量也相對的急遽增加；此外，由於網際網路分散的架構讓資訊分散在各地，對於資訊消費者所需的資訊往往無法在一個網頁或文件中找齊，因此常要耗費許多時間在資訊的過濾與取得，如此的操作模式是相當沒有效率地。

所以我們希望提供一個系統，對於來自不同來源的資訊，資訊消費者能精確的定義他所需的資訊，透過系統的資訊組合與主動的通報機制，有效的將符合需求的資訊源源不斷的遞送給資訊消費者。

本論文在於設計一個以 XML 資訊物件為基礎的事件通報與內容整合系統，期望能達成以下的目標：

- 提供一個客製化的 XML[1]資訊整合平台，以提高資訊整合的彈性與效率
  - 不同來源之 XML 資訊之整合
  - 部分 XML 資訊來源的整合機制
  - 自訂整合資訊的輸出格式
  - 設計整合應用的工具
- 整合資訊的撮合與通報 (Event notification service)[5][6]
  - 訊息撮合 ：主動撮合符合關聯與訂閱條件的資訊來源
  - 即時通報 ：將多個不同應用的資訊整合在一起後，資訊消費者可以方便且即時的獲取其整合資訊

2. 系統規劃與設計

2.1 資訊物件整合系統概觀

目前網路上流通著各式各樣不同的資訊格式，我們期望能在如此的環境下將這些異質資料(Heterogeneous Data)來源做進一步的整合，並藉由 publish / subscribe 系統作為資訊流通的交換管道，而其最終的整合結果就是符合 subscriber 想要的一份完整資訊物件。系統選擇以 XML 作為資訊交換的格式，並以 XML 資訊物件(XML Object)作為資訊內容整合的基本單位，圖 2.1 是本事件通報與資訊物件整合系統概念圖。

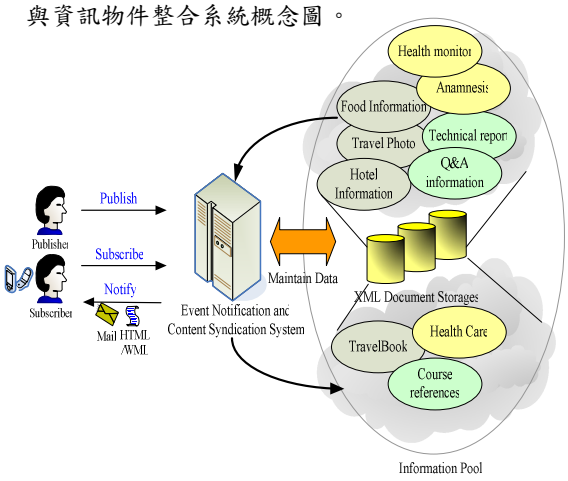

圖 2.1 事件通報與資訊物件整合系統概念圖

在我們系統中其整合資訊的來源就是 XDS[2][3]，而 XDS 中的一個 Collection 搭配本系統提供訊息的撮合及通報之機制，可以視為一個應用服務，例如：旅遊資訊服務、求職求才服務、醫療看護資訊服務等 。整合的對象允許跨越不同應用，整合的結果則代表了一個新應用服務的產生。而這也是我們對資訊內容整合(Content Syndication)所下的定義，透過 Content Syndication 的運作，系統能蒐集不同來源應用的資訊內容，組合成為滿足特定需求的應用。一個 syndicated content 即為一個完整的資訊物件，包含了豐富的資訊格式，包括: 文字、多媒體(照片或影音檔案)，或特定格式的文件(如 MS Word、Power Point 等)，甚至還可以包括另一個完整而且有意義的資訊。

圖 2.2 表達了系統中資訊整合的基本運作：

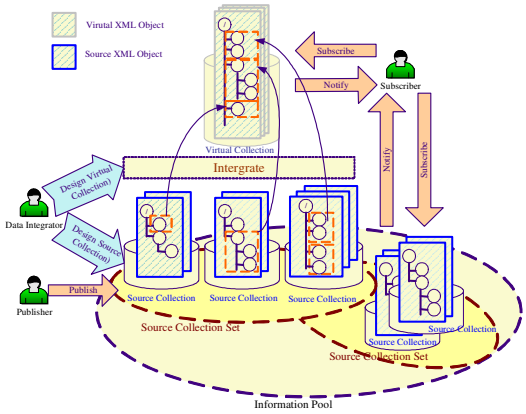

圖 2.2　資訊整合與事件通報的基本運作圖

根據圖 2.2 所示我們首先了解在整合運作中，各個使用者在系統中所扮演的角色如表 2-1。

Actor	Activity
Publisher（資訊提供者）	■ Publish / Seed：發佈新的訊息到既有的應用中
Subscriber(資訊消費者)	針對系統中存在的 Collection 提出以下兩個要求: ■ 訂定訂閱規則（資料篩選條件） ■ 資訊查詢
Data Integrator (資料整合者 /應用設計者)	■ 建立原始資訊集合 (Source collection,SC) ■ 建立整合資訊集合 (Virtual Collection,VC) ■ 管理各個資訊集合內容 :新增、刪除、修改 CO 和 XO
System Administrator	■ 負責 Services 跟 Account 的建立、維護

表 2.1　系統角色

想像 XDS 為一個資訊整合來源的 Information Pool，由資訊提供者(Publisher)不斷提供各種不同應用服務的資訊 ，系統根據資料整合者 (Data Integrator) 依據應用需求所完整定義的一個目標資訊物件樣板(Target XO template)以組合出一個個整合資訊物件[7]，這些整合的資訊物件可以套用到傳統資料庫 view 的概念如圖 2.3：

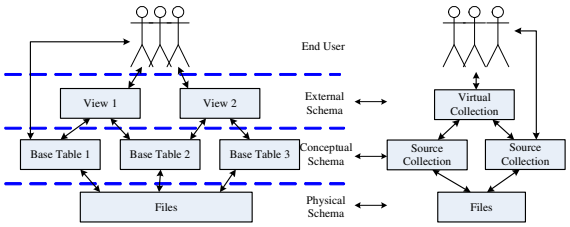

圖 2.3 Database View vs. XML Virtual Collection

View 的概念是動態的、有彈性的；一份 XO 中的一些 elements 或 attributes 可成為一個 view，或多份 XOs 也可成為一個 view，這份重新組合出來虛擬文件(Virtual document[4])就是一個完整的 XO 我們稱為 Virtual XML Object，簡稱 VXO，而這些 VXO 的集合便產生另一個新的整合應用(Virtual Collection，簡稱 VC)，整合應用的資訊來源可能來自數個不同的 collection 我們稱這些 collection 為 Source Collection (簡稱 SC)，而一個整合資訊的所需的 collection 集合則形成一個 Source Collection Set。對於資訊消費者(Subscriber)而言，無論 SC 或 VC 均可從中訂閱取得所需的資訊。

此外，為了達到客制化的資訊需求，資料整合者可能只對來源資訊中的部分內容(可能包含了"元素"或"屬性")有需求而不是整個 XO，這些資料整合者所需求的部分資訊集合我們稱為 Interested part(簡稱 I_part)。

2.2 系統規劃-資訊整合模型

2.2.1 整合資訊樣板

由於 XML 資料模型的特性，欲從兩個以上不同的 collection 中之 XML Objects 組合成一個單一的 collection，其組合的變化是十分複雜的[8]。因此，首先針對以下的 Target Content Syndication mode 提出我們分析所需的定義與假設，並同時搭配實際的應用狀況說明，最後挑選出一個能滿足大部份應用需求的模式作為我們定義整合資訊樣板的基礎。

定義 SCT 代表 Target XO Template 的 Source collection set，表示為:

SCT = { sc_1 , sc_2 , … , sc_n | sc_i € SP,i=1,2,...,n ,n:代表整合應用的 Source collection 個數}

我們以一個 Matched Source XO 為資料組合的基本單位，依據 sc_i 所得到的 Matched Source Set 中的 Source XO 個數，Target XO template 中相對於 sc_i 的 Matched Source Set 之 XO 可能的出現次數(以 $N(mxo)sc_i$ 表示)為" 1 "或" N_i",N_i >1，N_i 等於一個 SC 中 Matched Source Set 之的 XO 個數;i=1,2,...,n ,n:代表整合應用的 SC 個數。

$T(N(mxo)scn)=\{N(mxo)sc_1,N(mxo)sc_2,...,N(mxo)sc_n \mid N(mxo)sc_i=1 \text{ or } N_i, i=1,2,\ldots,n\}$ ，表示 sc_1~sc_n 的 Matched Source XO 在 Target XO Template 的出現次數。

mode1:

運作方式: 將所有 Source Collection 的 Matched Source XO 彼此做 Cross product。

Target XO template 中各個 SC 允許的 Matched Source XO 最大出現次數的集合表示 :

$N(mxo)sc_1=1,N(mxo)sc_2=1,...,N(mxo)sc_n=1$

=> $T(N(mxo)sc_n)=\{1,1,\ldots,1 \mid N(mxo)sc_i=1,i=1,2,\ldots,n\}$

整合結果筆數=$N(mxo)sc_1$* $N(mxo)sc_2$*...* $N(I\_part)sc_n$

mode2:

運作方式: 將所有 Source Collection 的 Matched Source XO 全部 Union 成為一個 XO。

Target XO template 中各個 SC 允許的 Matched Source XO 最大出現次數的集合表示 :

$N(mxo)sc_1=N_1,N(mxo)sc_2=N_2, \ldots ,$
$N(mxo)sc_n=N_3$

=> $T(N(mxo)sc_n)=\{N_1,N_2,\ldots,N_n \mid N(mxo)sc_i=N_i,i=1,2,\ldots,n\}$

整合結果筆數= 1

mode3:

運作方式: 從 SC set 中選擇 p 個 SC，表示為 $\{x_1,x_2,...,x_i\}|x_i,i=1\sim n$ 表示 SC set 中的其中一個 SC，1≤p<n，將這些 SC 的 Matched Source XO 全部 Union 成為一個 XO，再與其他 q 個 SC，表示為$\{y_1,y_2,...y_j\}|y_j,j=1\sim n$ 表示 SC set 中的其中一個 SC，1≤q<n-p 的 Matched Source XOs 做 Cross product。

Target XO template 中各個 SC 允許的 Matched Source XO 最大出現次數的集合表示 :

$N(mxo)sc_1=1 \text{ or } N_1,N(mxo)sc_2=1 \text{ or } N_2, \ldots ,N(mxo)sc_n=1 \text{ or } N_n$

=> $T(N(mxo)scn)$:至少有 1 個 SC 的 $N(mxo)sc_i=N_i$ 最多有 n-1 個 SC 的 $N(mxo)sc_i=N_i$

整合結果筆數: = 1* $N(mxo)scy_1$* $N(mxo)scy_2$...* $N(mxo)scy_j$

mode4:

運作方式: 選擇一個 SC 作為 Primary SC，根據此 PSC 中的各個 Matched Source XO 為基礎，將其他 Secondary SC 中相關的 Matched Source XO 與其 Primary Source XO 做 Union。

Target XO template 中各個 SC 允許的 Matched Source XO 最大出現次數的集合表示 :

$N(mxo)sc_1=1 \text{ or } N_1,N(mxo)sc_2=1 \text{ or } N_2, \ldots ,N(mxo)sc_n=1 \text{ or } N_n$

=> $T(N(mxo)sc_n)$:有 1 個 SC 的 $N(mxo)sc_i=1$ n-1 個 SC 的 N(mxo)sci=Ni

整合結果筆數= N(mxo)psc

綜合以上的分析，進一步思考根據資訊消費者蒐集資訊的經驗與需求都會傾向以某個應用的資訊集合做為所需應用的資訊主體，再搭配其他次要應用的資訊內容來延伸補足資訊主體的不足。因此，我們選擇以 mode4 做為整合資訊樣板的基礎。例如:“醫療看護資訊”應用，以“個人病歷”資訊內容為主，整合此病人相關的檢查報告與即時看護

資訊。考題應用中，出題老師想以現存的一個"考題 1"的應用，包含有是非、選擇、填充三個部份，作為考試的題庫出題，但在某次考試中可能需要加入另一個"考題 2"應用中的"配合題"部份，以產生另一個考題的應用。此"考題 1"就是 PSC，"考題 2"為 SSC。在以上所舉的整合應用中，其主要資訊無法完全滿足資訊消費者的需求，必須藉由其他相關資訊一併參考，以使得整合應用服務的資訊內容更為完整。以下定義 Primary Source Collection 與 Secondary Source Collection：

Primary Source Collection：為整合應用中主要需求的資訊內容，根據 PSC 的資訊內容為基礎蒐集所有相關的資訊，以作為延伸和擴充主體資訊內容的不足。一個整合應用中只能包含一個 PSC。由於整合應用(VC)是以 PSC 的資訊內容為主，所以最終 VC 中的 VXO 個數就等於 PSC 中的 XO 個數。**Secondary Source Collection：**為資訊消費者在整合應用中次要參考的資訊，作為擴充和補足 PSC 資訊不足的資訊來源。一個整合應用中允許一個以上的 SSC。

一個整合應用的資料來源除了 PSC 與 SSCs 之外，進一步思考若資訊消費者所需的資訊在既存的應用服務中不符合需求，或甚至尚未成形，因此，需要規劃一個部份讓資料整合者在建立新的整合應用時，可以完全重新定義所需新增的客制化(Custom part)資訊內容，此部分的資訊內容為等待將來有符合的 SC 之資訊 ready 後，再整合進來，或是提供讓 publisher 自行發佈此部分 的資訊內容。圖 2.4 表達了一個完整的 Target XO Template 所包含的部份。

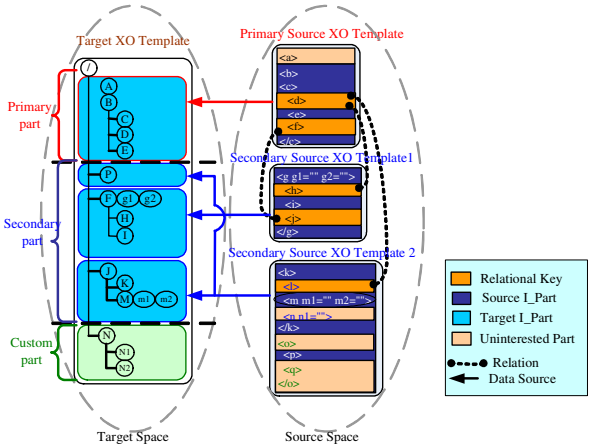

圖 2.4 整合資訊樣板

2.2.2 資訊間的關聯性

確定了整合資訊樣板規劃後，首先我們需爲整合應用蒐集所有相關的 XOs，以待整合。整合資訊的資料來源為 XML Storage 中多個不同的 XML 資訊物件，這些不同應用來源的 XML 資訊物件彼此間可能隱藏著某種程度的關連性，以下將探討如何將隸屬於不同應用中的 XML 資訊物件搭建起彼此間的相互參照，以同時向多個不同的資訊來源中搜集整合資訊所需的相關資訊。

何謂 X-Relation

為了從不同的 Source collection 中搜集所有與整合應用相關的資訊，Source collection 間必須能搭建起資訊間的相互參照，以便讓系統能有跡可尋找出所有符合的相關資訊。以下爲 X-Relation 做一定義:

定義:在不同 Source collection 間，彼此間各自擁有一個文件節點的資料內容隱含有相同的資訊意義，我們爲他們之間搭建起一個資訊間的相互參照稱之。而所對應的文件節點稱為 Binding key。

Binding Key 實際上是建立在某些資料欄位(element 或 attribute)中，如圖 2.5 中 Hotel 的<Place>元素與 TravelBook 的<Location>元素均是代表"地點"的意義；而所謂的 X-Relation 就是為不同 Collection 之 XML Object 中的 Binding Key 搭建一

個資料的對應關係，使不同 collection 中之 XML Object 其文件節點(Document node)的資料內容隱含有相同意義的節點彼此做一個參照的連結。在一份 XML Object 中 Binding Key Set 記錄了所有與其他 XML Object 有關聯的 Binding Key。

透過 X-Relation 的定義，我們了解 X-Relation 的主要目的就是爲不同資訊集合間做 data binding 的動作。就邏輯上的觀點，是將多個不同的 Source Collection 視為同一個 collection，以便在做資訊整合時為系統提供一道線索，從多個 Source Collection 中搜集所有符合整合應用需求的 XML Objects。進一步思考 collection 間可能存在一個以上的 X-Relation，藉由 X-Relation 間的不同搭配以滿足不同的應用需求，因此，透過 Relation rule 便能完整描述 collection 間相互參照的資訊。

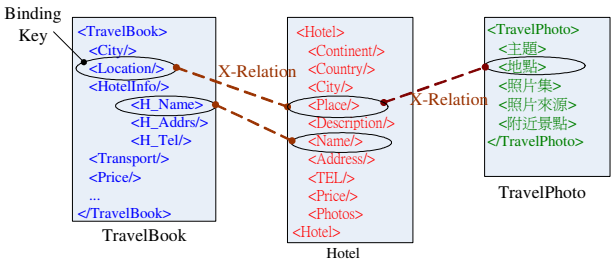

圖 2.5　資訊間的 Binding

關聯規則-Relation Rule

Relation rule 實質上的意義在於描述 Source Collection 間的整合條件，系統透過此資訊，爲整合應用蒐集所有 SSC 中與 PSC 相關的 XML 資訊物件。以下為 Relation rule 做一定義:

定義:由一個以上的 X-Relation 所組成，藉由 X-Relation 間的運算(and/or)，完整描述不同 Source Collection 間的關聯性。

詳細歸納出 Relation rule 的目的有以下兩點:

- Data binding :透過 X-Relation 將不同 Source Collection 間的資訊鏈結在一起視為同一個資訊集合，以整合不同應用的資訊來源。
- Data filtering :依據整合應用需求，爲整合資訊來源做初步的資訊過濾。

- Relation rule 的型態

根據不同的應用需求，資訊間的 relation rule 可區分為以下兩種型態：

- 單一關聯規則-Single Relation rule

在欲建立關聯的兩個不同應用的 XML Object 之間只存在一個關聯，因此，不會有 X-Relation 間的組合問題。

- 多重關聯規則-Multiple Relation rule

在欲建立關聯的兩個不同應用的 XML Object 之間存在兩個以上的關聯，而關聯與關聯之間需以"&"(AND)、"|"(OR)、"!"(NOT)做串接以搭配出符合應用需求的關聯規則。如圖 2.6，在課程參考資料整合應用中，以"課程資訊"為 PSC"論文期刊"為 SSC，透過"課程資訊"的 <CourseName> 元素與 <Adviser> 元素分別和"論文期刊"的 <Field> 與 <Author> 建立 X-Relation，並將這兩個 X-Relation 以"&"組合，以精確的過濾出符合 PSC 中相關的 XML 資訊物件。

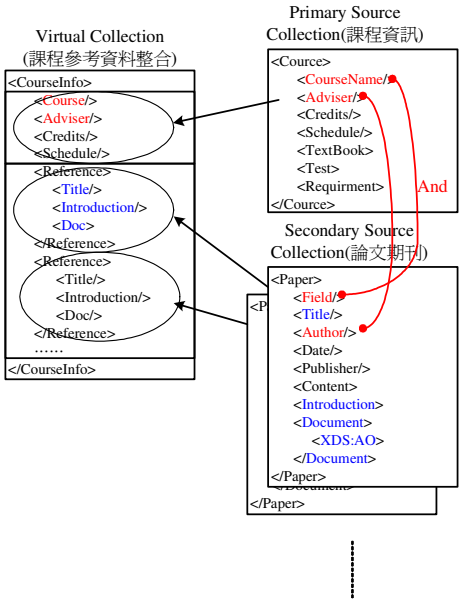

圖 2.6　Multiple relation rule

2.2.3 內容整合運作-Content Syndication Process

本小節將討論整個資訊整合運作的演算法，將前幾節所討論的內容結合在一起，利用 X-Bind 的運作產生上述所需的 VC。此外，爲了達到客制化的自訂輸出格式，我們提供了一些基本的 XML 轉換處理以達到目標。

歸納之前討論的幾個重點，最終的目的就是產生一個符合資料整合者所定義的 VC。透過 X-Bind 使整個運作能順利的完成，以下爲 X-Bind 作一定義。

定義:根據 source collection 間的關聯規則，從一個 primary source collection 與 N 個 secondary source collection 中取得符合的 Source XO 之 I_part，組合成一個單一的 Collection。一個整合好的 VXO 是以 primary matched source XO 的資訊為主並嵌入所有在 secondary matched source XOs 中相關的資訊內容。

由於 XML 具有 well-defined 的良好文件結構，因此，在整個資訊整合的處理過程，我們可以很精確的萃取出部份的文件片段內容，並且將他們重新組裝成符合新應用所需的文件結構。我們將整個演算法切割為四個細步階段。

系統是建立在 publish/subscribe 的通訊模式上，因此，系統會爲所有整合應用週期性的整合新發佈到 XDS 的 Source XO，以確保整合應用中的 VXO 都是最新的。當系統設定的比對期間(matching interval)結束，則系統觸發 X-Bind 演算法，開始執行 Stage1。

以下針對各個 Stage 做說明:

Stage1：Get matched source set

根據 CSO 中的 Relationa rule 與 second chance filter rule 分別從資料整合者所選定的各個 Source collection(PSC 與 SSCs)中取得符合的 XO 集合(Matched Source Set)。

Stage2：Extract source I_part

根據 CSO 中的 I_part Mapping 紀錄的來源資訊，從 Stage1 所取得的 Matched Source XO 中抽取出 Source I_part。

Stage3：Transform source I_part

根據 CSO 中 I_part Mapping 所定義的 Target XO template 透過系統的 XML-transforming operation 將 Source I_part 轉換為對應的 Target I_part。

Stage4：Create Virtual XML Object，此階段為實際產生出一個個組合好的 VXO，集合成 VCO，包括四個子程序

4-1 產生所定義的 Custom part : 系統根據資訊整合者重新定義的部份產生出客制化的部分資訊。

4-2 根據 CSO 中的 I_part Mapping 記錄了 Target I_part 彼此間的順序，組合 Stage3 中所有相關的 Target I_part 成一個完整的 VXO。

4-3 實際加入 XO 中所包含的 Attachment file。

4-4 加入所需的 AP metadata 以包裝成符合 XEBS 的 Event 格式。

4-5 將整合好的一份完整 VXO 加入 VC。

2.3 系統設計

本小節將討論系統中針對 X-Bind 運作所需紀錄的資訊規劃描述整合應用的格式，並對於整合好的 XML 資訊物件分析其不同操作需求的必要性，並探討是否實際儲存整合資訊物件。

資訊物件化的概念讓資料整合者能完整的定義他所需的整合應用(也就是 XML 資訊物件)，一旦定義好後，系統就能從不同的資訊集合蒐集並整合成他想要的 XML 資訊物件集合(VC)[8]。因此，我們需要設計一個能完整描述整合應用的格式，讓系統能動態的根據不同的整合應用需求，產生各種不同的客製化應用。我們稱這描述整合應用的格式為 Content Syndication Object (簡稱 CSO)，系統中

一個整合應用對應到一個 CSO。由於 XML 可自訂標籤，便於程式自動處理，且無論使用任何語言實作，只要能產生符合規格的 XML 文件就可以輕鬆的在不同應用程式間交換資料，所以我們系統利用 XML 格式來描述 CSO，以下將針對 CSO 的設計細節作說明：

針對一個整合應用需要描述的三個主要資訊分別為:來源資訊集合、關聯規則與部分來源資訊(Source I_part)和目標資訊物件樣板(Target I_part)的對應，分別對應到以下 CSO 中的三個部份，如圖 2.7:

1. **Source Collection Set**

   -描述此整合應用對應的資料來源集合 (Source Collection)。

2. **Relation Rule**

   -描述 Source Collection 間的關聯規則。

3. **I_part Mapping**

   -目標資訊物件樣板的 文件結構，描述整合應用的輸出格式。

   -目標與來源 I_part 的對應。

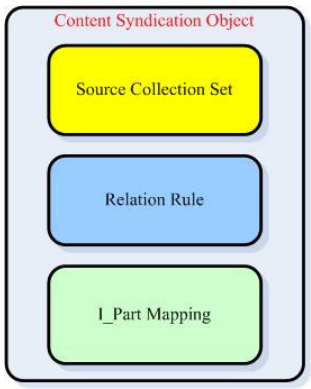

圖 2.7　Content Syndication Object

●Source Collection Set 的部份：

作用：此部分資訊主要用來蒐集所有與整合應用相關的來源資訊物件　，各個 Source collection 中所定義的訂閱條件會搭配關聯規則向 Source collection 做搜尋後，並根據過濾規則 以取得符合的來源資訊集合　(Matched Source Set)。所包含的資訊有：

- 整合應用的 metadata：例如:整合應用名稱、collection type、資料整合者與整合應用的建立日期。
- 來源資訊集合。
- 各來源資訊集合的訂閱條件。

使用時機：系統在資訊整合運作啟動時首先根據此部分的資訊與 Relation Rule 的資訊，以完成取得 Matched Source Set 的工作。

●Relation Rule 的部份：

作用：此部分主要描述不同 XML 資訊物件間的關聯規則。系統根據此關聯規則到各個 Source collection 中蒐集所有相關的 XML 資訊物件等待做下一步的整合　，若 Source Collection Set 的部分有訂閱條件之資訊，則需再進一步的過濾以取得最後的　Matched Source Set。所包含的資訊有：

- 各個 Source collection 間 XO 的關聯規則
  -關聯對應的元素
  -關聯間的運算(AND 或 OR)
  -關聯的比對形式(~或=)
  -關聯元素的 pattern
- PSC 針對 Relational key pattern 的訂閱條件

使用時機：系統在資訊整合運作啟動時首先根據此部分的資訊與 Source Collection Set 的資訊，以完成取得 Matched Source Set 的工作。

●I_part Mapping 的部份：

作用:此部份主要描述目標資訊物件樣板的文件結

構，並針對所選定的 PSC 與 SSC 從中挑選出有興趣的部分資訊集合(I_part)，最後將所定義的目標資訊物件中的文件節點與來源 I_part 中做一對應。所包含的資訊有：

- ◆ 目標資訊樣板結構
- ◆ 目標與來源 I_part 的對應

使用時機：

當系統取得整合應用的 Matched Source Set 後，須從每個 Matched Source XO 中抽取出資料整合者有興趣的部份時；接著將每個 Source I_part 轉換成目標資訊物件樣板的形式。最後將各個 Target I_part 根據目標資訊物件樣板的文件結構產生一份組合好的 XO。當資料整合者定義好整合應用後，系統在啟動資訊整合運作之前必須在 XDS 中事先建立好一個 VC，而由於再 XDS 中建立一個 Collection 需要準備對應的 Schema，因此，系統需要利用此部分資訊組合出此整合應用的 Schema。

3.系統實作

圖 3.1 是系統整體的元件架構圖。

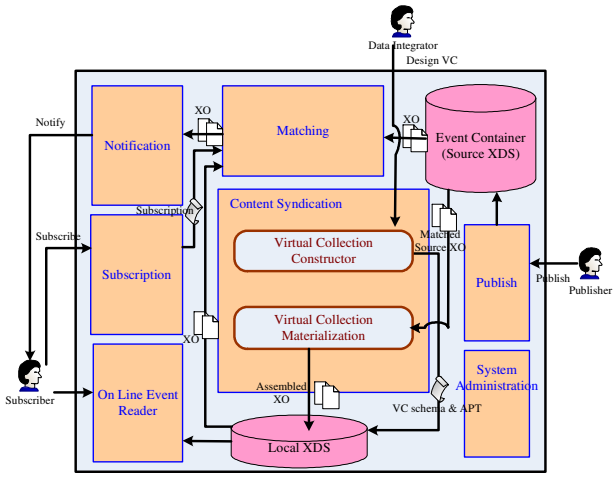

圖 3.1　系統元件架構圖

配合圖 3.1 我們先簡單描述一下系統中主要的運作：

1. Publisher 透過 Event Editor 產生 XML Event，並把它儲存在 Event Container，等候資訊整合與 Matching Daemon 的撮合[9]。
2. Data Integrator 透過 Virtual Collection Constructor 設計並建立所需的整合應用。
3. 資訊整合元件中的 Virtual Collection Materialization 會定時爲每個整合應用根據 Data Integrator 所定義的 Content Syndication Object 整合出符合的 XO，並儲存在 Local XDS 中以供分享。
4. Subscriber 透過 Subscription Form 的幫助，他可以針對儲存在 Source XDS 中的應用(Source Collection)或存在 Local XDS 的整合應用(Virtual Collection) 下達 subscription rule，並產生 subscription 記錄在系統的 subscription database 中。
5. Matching Daemon 根據 User 的 Subscriptions，定時為 User 在 Event Container 與 Local XDS 中尋找符合條件的 event。
6. 如果 Matching Daemon 找到符合 User Subscription 的 event，就把 event 交給 Event Presentation 配上適當的 Style Sheet。然後再透過 Delivery Gateway，送到 Subscriber 的手上。

系統中的兩個 Storage server 之用途:

- ●Source XDS:儲存在網路上由 publisher 主動發佈的訊息，蒐集來自四面八方的 XML 資訊物件，作為資訊整合的來源資訊。
- ●Local XDS:系統本身用來儲存、維護與管理整合應用的儲存空間。思考資訊整合的資料來源可能來自不同的 Source XDS，因此我們將整合應用統一存放在 local XDS 中。

Content Syndication 軟體元件為資訊整合的主要元件，負責建立資訊整合應用並具體化整合資訊。我們將在之後的章節詳細說明它所包含的兩個子元件 Virtual Collection Constructor 與 Virtual Collection Materialization 的詳細實作內容。

3.1 Virtual Collection Constructor

在執行整合運作之前，系統必須先搭建起整合應用的運作環境，包括：

- 提供給資料整合者設計整合應用的工具。
- 根據設計的整合應用建立所需的系統資源。

Virtual Collection Constructor 軟體元件針對以上兩項工作分派由兩個子元件來執行：

- Content Syndication Editor
- Virtual Collection Generator

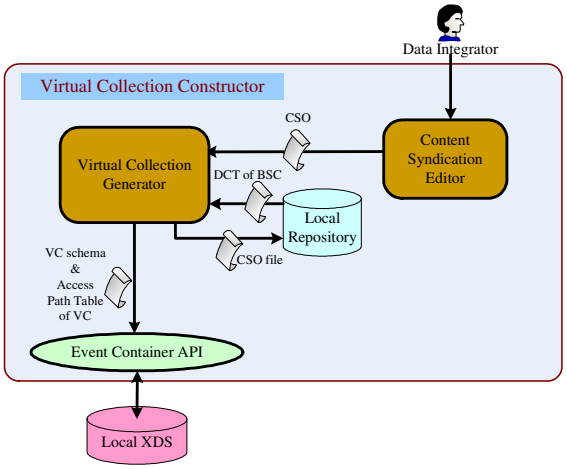

圖 3.2　Virtual Collection Constructor 的子元件互動

Virtual Collection Constructor 的子元件互動(如圖 3.2)

1. 資料整合者透過 Content Syndication Editor 設計整合應用，並產生對應的 CSO，再將 CSO 之 JDOM Tree 傳遞給 VC Generator。
2. 藉由 VC Generator 根據 CSO 資訊，組合出對應此整合應用所需的 XML Schema，以實際在 Local XDS 中建立一個儲存 VXO 的 Collection，並在系統中建立、儲存對應此整合應用所需的資源。

3.2 Virtual Collection Materialization

此軟體元件為整合運作的核心部份並以 Daemon 的形式存在系統中，因此，它會週期性的被觸發執行；其主要功能就是實作出.2.2.3 小節中所設計的 X-Bind 演算法。

圖 5-6 簡單描述子元件彼此間的互動：

Virtual Collection Materialization 子元件間的互動

1. Content Syndication Object Manager 將 CSO file 轉換成 JDOM Tree[10][11] 並儲存成特定的資料結構，供系統方便參考使用。
2. Content Syndicating Engine 根據 CSO 資訊從 Source XDS 蒐集所有符合的 Matched Source Set 等待進一步的整合。再依據 CSO 資訊與 Matched Source Set 分別抽取出 Source I_part 後根據 Target XO template 轉換成對應的 Target I_part，接著以各個 Primary source XO 為基礎，組合所有相關的 Target I_part 成符合需求的 VXO，再包裝成 event 後儲存在 Local XDS。

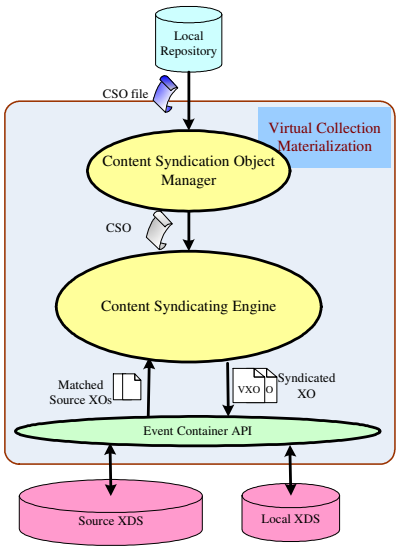

圖 5-6、Virtual Collection Materialization 子元件間的互動

4. 系統應用與展示

現代人生活品質的逐漸提升 ，旅遊的風氣也漸漸盛行，坊間也隨之也推出琳瑯滿目的旅遊參考書籍，讓讀者在想要前往某地旅遊或參觀時，能夠依照書上的內容與指引，得知旅遊景點的資訊，如旅館、交通、景點介紹等等。基於上述的需求，資訊整合者針對旅遊的應用，可以利用我們的系統，整合出滿足特定需求的旅遊資訊。想像一個整合應用的劇本，某些旅遊景點會應季節的改變而影響旅遊的人潮，旅行社期望就當時的時間點針對目前最熱門的幾個景點整合出符合民眾需求的旅遊手冊。

假設在目前的 Collections 中已經存在一個 Travel Book 的應用，但其內容並不夠完整無法滿足資訊消費者的需求，因此，旅行社想要透過其他兩個 Source Collection，“Hotel”與”Travel Photo” 的資訊內容來補足欠缺的部份，並同時從這三個 SC 中分別篩選出想要的部份資訊內容，以取得一個完整的客制化旅遊資訊，此外，旅遊手冊同時整合景點美食的資訊，雖然在既有的應用中不存在，但可事先定義出此部分的資訊需求，待將來再做進一步的整合。就之前章節的討論，我們了解在這個整合應用中，Travel Book 為 PSC，Hotel 與 TravelPhoto 為 SSC，而新增的美食資訊為 Custom part。在下一節中將利用上述的應用劇本展示本系統實作的成果。

首先假設系統中已存再整何應用所需的三個 SC，分別為 Travel Book、Hotel 與 Travel Photo。整個設計整合應用的步驟如下：

Step1 : Select Source Collections

Step2 : Define X-Relation

Step3 : Define VC Schema

Step4:Set Data Filter rule

當整合應用儲存至系統後 ，將會週期性的為各個做整合的運作，以隨時都能掌握最新的資訊內容。我們以資訊消費者的角色登入系統 ，並利用 On-line event reader 觀看整合後的結果。如圖整合出來的結果即是針對 ”阿里山”與”合歡山”這兩個

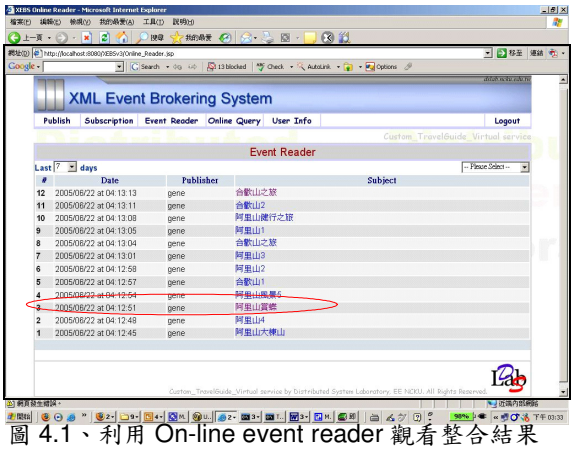

圖 4.1、利用 On-line event reader 觀看整合結果

觀看第 3 筆的 event 內容如圖 4.2、4.3、4.4 與 4.5，分別代表 Target XO Template 中的 primary part、secondary part 1、secondary part2 與 custom part。

圖 4.2　syndicated XO 之 metadata 與 primary part:From TravelBook

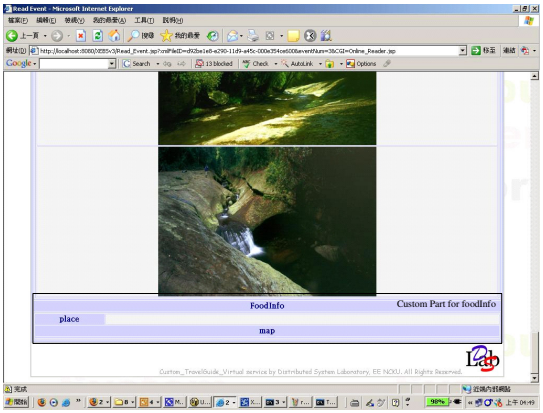

圖 4.5　syndicated XO 之 custom part for FoodInfo

圖 4.3　syndicated XO 之 secondary part 1:From Hotel

圖 4.4　syndicated XO 之 secondary part 2:From TravelPhoto

5. 相關研究

5.1 XQuery(XML Query)

XQuery[12]是一種為可延伸標記語言 (XML) 所設計、類似於結構性查詢語言 (SQL) 的程式語言。XQuery 是由 W3C XML Query 工作小組所制定的 XML 查詢語言。到目前為止仍處於 Working Draft 階段，還未正式定案，XQuery 是源自一種 XML 查詢語言叫 Quilt，Quilt 引用多種其它語言，包含 XPath 1.0、XQL、XML-QL、SQL 及 OQL，Quilt 的許多語法皆繼承自 XQL 以及 XML-QL。XQuery 1.0 版本包含 XPath 2.0。

各種 XML 查詢語言之間的關係如圖 5.1 所示

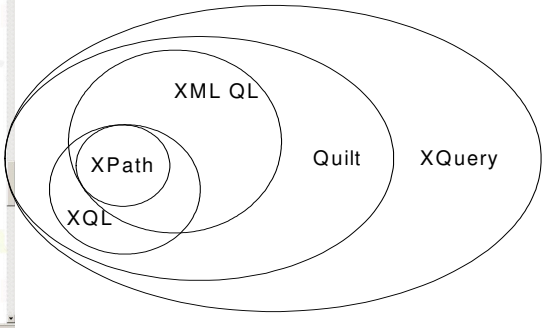

圖 5.1　各種 XML 查詢語言之關係

XQuery 是 XPath[13]的延伸，與 XPath 不同的地方是，XQuery 提供一系列的資料類型和控制

結構的編程環境。XQuery 主要可從 XML 文件與內含 XML 的資料庫中，查詢或擷取資訊。就好像 SQL 一直是查詢關聯式結構資料庫的標準語言。

XQuery 可分析 XML Storage，讓使用者可以輕易且充分的檢索、分析大量又不同來源的相關資訊，以 XQuery 為基礎發展資訊整合 (Information Integration)，為未來不可獲缺的一環。

XQuery 幾個重要特性整理如下：

1.具有路徑的觀念，可以指定 XML 文件中的任何元素或屬性
2.具有完整的條件限制語法
3.支援完整流程控制語法
4.具有自訂輸出格式的功能
5.可自訂資料型態
6.可定義函式

XQuery 語法表示(FLWOR) 是由 For、Let、Where、Order by 及 Return 子句等結構而成，並以特定的順序組合而成。一個 FLWOR 表示法繫結了一個或多個變數，再使用這些變數建構結果。

XQuery 所規劃的功能與本系統所提供的 tools 作一些分析：

1. 對於 XQuery 整合的 XML 文件，必須清楚地一一列舉 XML 文件檔名；而本系統是以一個資訊集合體為一個整合對象，對 XQuery 而言，不適用本系統所欲面對的應用。
2. XML Storage 要對 XQuery 有支援；而本系統工具在後端 XML Storage 不需支援即可對資訊作操作。
3. XQuery 有自訂輸出標籤的功能，為了滿足客製化的整合資訊輸出，本系統也提供此功能。

5.2 RSS

RSS 是由 Netscape 在 1999 年所開發，它代表三種可能的縮寫：RDF Site Summary、Rich Site Summary 或 Really Simple Syndication；RSS 是一種以 XML 為基礎的描述格式，主要用來整合(syndicate)各網站超連結的清單，並藉由附屬的一些 metadata 資訊讓資訊消費者能決定是否要進一步瀏覽完整的詳細內容 [14][15]。對於網站開發者(Web developer)而言，特別像是以發佈新聞為主的網站(news-like site)或是個人網頁空間(Weblog)，能利用它來描述與整合網站內容以建立出屬於自己的 RSS feed(也就是一個符合 RSS 規格的檔案)。當然他不只是用在新聞發佈上，任何 list-oriented information，只要能將資訊內容以列項 (itemize)的方式表達就能透過 RSS 將他們整合起來後提供給資訊消費者；像是 wiki 中哪些網頁是"最近改變"的、CVS 的 changelog 或甚至是一本書的改版或校訂的歷史，而 RSS 也不一定只能應用在 Web page 上，像是 Instant messaging 服務就適合利用它來傳送 headline，如同手機上的簡訊服務。

5.3 XEBS

XEBS(XML Event Brokering System)[16][17] 系統是實驗室的研究成果，主要設計以 Publish/Subscribe 訊息交換模型建立訊息撮合及通報系統，希望帶給使用者更有效率的資訊取得方式。XEBS 系統最大的特色是利用 XML 格式標準、自我表述等特性，建立以 XML Object 為訊息載體的 Content-based 通訊模型，並提供準確的訊息過濾機制。

在 XEBS 中提供 general-purpose 的訊息撮合以及通報系統，系統能夠針對不同的應用需求，在不用撰寫程式的情況下能快速的建立不同的應用服務。以資訊提供者的觀點，可以透過方便的介面輕易的發佈訊息。資訊消費者能夠準確的訂閱想要的訊息，並透過 HTTP 或 E-mail 的方式，取得系統通報的訊息。XEBS 系統中並使用 XML Object 的概念，提供自給自足、完整的資訊個體，由於 XML 具有明確格式、自我表述的特性，能讓使用者作精確的訂閱。

5.4 XML Document Storage(XDS)

XML Document Storage[2][3]是我們實驗室所

發展出來的一套 XML 檔案儲存系統，XDS 與一般的檔案系統一樣保障被儲存資料的恆久性 (Persistency)。其發展的目的是為了提供一般 XML 資料共享應用系統一個儲存、搜尋與管理 XML 資料的管道。不同於其他 Native XML Database，如 Xindice 與 eXist。XDS 以 XML 及物件的概念，模型化網際網路資訊，提供資訊存取、查詢的管理服務。

XDS 提供了底下以 XML 特性為主的存取功能：

- **提供一份 XML 文件為單位的輸入/輸出介面**
  XDS 所提供的輸入/輸出介面不同於 SQL 語法而相當的單純，輸入介面可以直接置入一份 XML 文件的內文，然後取得該文件的 ID；而輸出的介面則是指定目標 XML 文件的 ID，然後可以直接將一份 XML 文件的內文取出。
- **提供以 XML Schema 標準為主的資料模型**
  XDS 是以 XML Schema 標準[18]來當成區分資料的依據，在 XDS 中必須以一個 XML Schema 來建置一個 collection，而每一符合特定 collection 中之 XML Schema 規範的文件會被 XDS 解讀成是一個 XML Object，並會被分配到對應的 collection 裡頭。
- **提供 XPath 的 query 介面**
  XDS 所提供之 XPath 的 query 方式邏輯簡單且 query 的內容非常容易來解讀。

6. 結論

網際網路中雜亂龐大且分散的資訊，讓資訊消費者往往需花費許多時間在重複的資料蒐集與過濾，為了解決上述的問題，利用 publish / subscribe 系統的模型，將事件利用撮合的機制，自動地推播到使用者端。而網路上繁雜不一致的資訊格式，也限制了資訊間的流通與降低重用的價值，因此，以 XML 作為訊息交換的格式與標準，對於資訊消費者而言更能精確取得完整的資訊外同時也達到客製化的展現。對於資訊整合者，更能清楚且方便從不同的應用來源定義出所需的資訊物件，除了大大縮短整合應用的時間外，同時也提升了資訊的重用性。

本論文研究提出一套可以整合不同應用來源中的部分資訊之系統，系統會自動的幫助使用者蒐集與篩選相關的資料內容，並且提供適度的整合彈性以滿足客製化的整合應用需求。本系統具有以下特點:

- **整合跨越不同應用的資訊**
  藉由資訊間的關聯定義，讓不同應用間搭建了彼此間的參照，對於整合好的資訊物件集合，資訊消費者只需提出查詢便能得到一份來自不同應用的完整資訊物件。
- **提供更方便且有效率的 XML 物件整合工具**
  - 降低開發新應用的時間: 使用者利用系統提供的整合工具設計建立整合應用，為使用者節省了許多繁雜重複的動作。
  - 客製化的整合資訊: 使用者能透過系統挑選來源資訊物件中有興趣的部份，並定義想要的客制化輸出資訊樣板。
- **以 XML 物件為資訊整合的單位**
  - 提供以 XML 物件方式定義資訊需求
  - 搭配 Storage Server 管理 XML 物件
  - 更小顆粒度的資訊共享與重用
- **結合 Publish/Subscribe 系統與 Storage Server**
  - 主動的訊息撮合與通報:藉由 publish / subscribe 的通訊模式，改變了原本資訊消費者以流覽為主的被動模式，使得資訊的取得更方便、有效率。
  - 減少訊息通報的頻率: 我們將訊息儲存在 Storage Server 中，因此，即時的資訊可以使用 Online event reader 的 Notification，以減少通知資訊消費者者的頻率。

提高資訊重用性: 傳統 publish / subscribe 通訊模

型，訊息一旦 publish 並經過 event broker 遞送給符合的訂閱者後，此訊息的生命週期便將結束，若往後有其他相同的訂閱需求時，則之前符合的訊息將無法重用。

7. 參考文獻

[1] World Wide Web Consortium. **Extensible Markup Language(XML)**. http://www.w3.org/XML/

[2] 王志弘，蔡尚榮,An XML Storage System Supporting Binary Contents , Master thesis, Dept. of EE,NCKU, June 2004.

[3] 李嘉銘，蔡尚榮, An XML-based Information Server – The Storage System , Master thesis, Dept. of EE,NCKU, June 2001.

[4] T. R. Gruber. Model-based virtual document generator.Technical report, Stanford Knowledge System Laboratory,1995.

[5] A. Carzaniga. Architecture for an Event Notification Service Scalable to Wide-area Network. Ph. D. thesis, Politechnico di Milano, Milano, Italy.

[6] G. Coulouris, J. Dollimore and T. Kindberg, Distributed Systems Concepts and Design, Addison-Wesley Publication, 2001

[7] Miro Lehtonen. Document Assembly with XML Structured Source Data. In Proceedings of XML Finland 2001, pages 52-60, November 2001.

[8] Kristin Tufte and David Maier, Aggregation and Accumulation of XML Data, IEEE Data Engineering Bulletin 24(2) : 34-39 (2001)

[9] Miro.Lehtonen,Renaud.Petit,Oskari.Heinonen, Greger.Linden.A Dynamic User Interface for Document Assembly.DocEng'02,November 8-9,2002,McLean,Virginia,USA.p134-p141.

[10] Hunter、Brett McLaughlin, "JDOM", http://www.jdom.org .

[11] Jason Hunter and Brett McLaughlin. Easy Java/XML integration with JDOM. JavaWorld. http://www.javaworld.com/javaworld/jw-05-2000/jw-0518-jdom.html.May 2000.

[12] World Wide Web Consortium. XQuery 1.0: An XML Query Language. W3C Working Draft 20 December 2001. http://www.w3.org/XML/Query .

[13] World Wide Web Consortium, XML Path Language. http://www.w3.org/TR/xpath.html.

[14] Ben Hammersley.Content Syndication with RSS.O'Reilly,March 2003.

[15] Mark Pilgrim.What is RSS ? http://www.xml.com/pub/a/2002/12/18/dive-into-xml.html.December 18, 2002.

[16] 陸志恆、蔡尚榮, Building a Content-Based Event Brokering Service with an XML Storage System, Master thesis, Dept. Of EE, NCKU, July 2003.

[17] 謝政育，蔡尚榮, An Event-Based Service Embedding Lively Contents in XML Objects , Master thesis, Dept. of EE,NCKU, June 2004.

[18] World Wide Web Consortium.XML schema , http://www.w3.org/XML/Schema.