行政院國家科學委員會專題研究計畫　成果報告

Web 萃取資料之資料管理及資料模式之研究(2/2)

計畫類別：個別型計畫
計畫編號：NSC94-2213-E-009-028-
執行期間：94 年 08 月 01 日至 95 年 07 月 31 日
執行單位：國立交通大學資訊工程學系(所)

計畫主持人：吳毅成

計畫參與人員：蘇瑞元、許俊彬、張修逞

報告類型：完整報告

處理方式：本計畫可公開查詢

中　華　民　國 95 年 9 月 26 日

行政院國家科學委員會補助專題研究計畫成果報告

Web 萃取資料之資料管理及資料模式之研究

計畫類別：■ 個別型計畫　　□ 整合型計畫
計畫編號：NSC 94-2213-E-009 -028 及 NSC 94-2213-E-009 -028
執行期間：　2004 年　08 月　01 日至　2006 年　7 月　31 日
計畫主持人： 吳毅成
共同主持人：
計畫參與人員： 蘇瑞元、許俊彬、張修逞

成果報告類型(依經費核定清單規定繳交)：□精簡報告　■完整報告

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

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立交通大學資訊工程學系

中　華　民　國　九十五　年　九　月　二十　日

中文摘要

隨著全球資訊網(World Wide Web)的快速發展，如何在這些大量資料中萃取出有用的訊息是非常重要的事情，例如：比價系統須萃取出相關電子商務網站中的有用訊息，如產品名稱、價格、購買方式等；其他，如萃取網路上一些新聞、及出版單位之書籍文章目錄等。在我們過去的國科會計畫中，我們設計了一個資料萃取語言叫做 BODED(Browser Oriented Data Extraction Description)語言及其雛形系統來解決這些問題，並已技術轉移至業界。然而如何對所萃取的資料進一步做一般化的管理是接下來的一件非常重要研究課題。萃取出 Web 資料的管理之基本應用有：資料貯存(data storing)、資料查詢(data query)、網站再生工程(web site reengineering) 、網站資料整合(web site integration)等。

這些 Web 資料管理應用的共通需求，是如何將 Web 的資料與資料庫的資料容易地互轉及整合，以便於管理。然而由於 Web 資料較為不規律，解決這些需求並非易事。首要工作是研究明確及有彈性的 Web 資料模式，如此方能簡化這些需求。因此，本計畫將以過去設計出的 BODE 資料萃取系統為基礎，提出一套適用於 BODE 的資料模式，以便於管理所萃取出的資料。主要工作項目如下：(前四項為第一年計畫，而後四項為第二年計畫)

1. 收集並分析過去 Web 資料模式及資料管理的研究。
2. 提出一套適用於 BODE 的資料模式，並對此模式制定資料定義語言。
3. 研究此模式與關聯式資料庫之對應關係。這研究含如何從這資料定義語言自動產生相對應的關聯式資料庫 schema。
4. 研究 BODE 萃取系統與此模式之對應關係。這研究含如何從 BODE 系統所產生的 script，自動產生相對應的資料定義語言。結合前項研究工作，可使 BODE 萃取系統自動萃取網站資料至關聯式資料庫。
5. 研究並設計如何整合不同網站的資料。這須研究及分析如何將不同的資料定義整合於同一個資料定義。
6. 研究並設計適用於此資料定義語言的查詢語言及系統。
7. 研究並設計適用於此資料定義語言的網站再生工程系統。
8. 以現有電子商務網站為實例，展示所研究的系統對萃取資料之管理能力。

Abstract

With the rapid growth of World Wide Web, it becomes very important to extract information from such a huge amount of database. For example, a price comparison system needs to extract information, such as product names, prices, purchase methods, etc, from some related e-commerce sites. Other examples are to extract news from news providers and to collect categories of publishers. In our past projects from NSC, we defined a data extraction language, named BODED (Browser Oriented Data Extraction Description) Language, and designed to solve the problem of data extraction a system for it whose technique has been transferred to industry. However, the way to manage the extracted data for data management is the next very important research topic. The basic applications include: data storing, data query, web site reengineering, and web site integration.

The common requirements for these basic applications are: how to easily translate and integrate data between web pages and databases. However, since web-based data is less regular, the requirements are not easy. The key research is to study a flexible and definitive web data model in order to simplify the requirements. Therefore, this project proposes a data model suitable for BODE for facilitating the management of extracted data, based on the experiences of designing the BODE system. The work items of this project are: (The first four items are for the first-year project, whereas the next four items are for the second-year project.)

1. Collect and analyze the research of web-based data management and data model in the past.
2. Propose a new Web-based data model suitable for BODE and define a data definition language for the model.
3. Study the mapping relation from the data model to relational databases. This research includes how to map the data definition language to database schemas.
4. Study the mapping relation from BODED scripts to the data definition language. Combined with the previous item, we can map BODED scripts directly to database schemas.
5. Study and design the method to integrate information from different web sites. This requires to study and analyze how to integrate different data definitions into one definition.
6. Study and design the query language and the query system suitable for the data definition language.
7. Study and design the web site reengineering system suitable for the data definition language.
8. Demonstrate our system by using it to manage web data of current e-commerce web sites.

Keywords: (data management, data model, data definition model, data extraction, data storing, data query, web site reengineering, web site integration)

一、前言

由於網際網路的普及以及 World Wide Web 的快速發展，有越來越多的資訊，以網頁的方式發布在網際網路上，因此有許多的使用者在網頁上蒐集有用的資訊。全球資訊網（World Wide Web）可說是一個超級大的一個資料庫，內容相當豐富。除了一般生活資訊外，另有大量之專業科技、新聞、商業、金融與人文資料。然而要在 Web 網頁中萃取出有用的資料，並加以彙整與整理，其實並不容易。

過去資料管理(Data Management)的研究主要是在傳統的資料庫上。隨著全球資訊網(World Wide Web)的快速發展，在 Web 上的資料管理也愈形重要。一般而言，資料庫採用較規律且較有結構的關聯模式或物件為基礎的模式，來存取及管理資料。然而，在 Web 上的資料，大多是較為不規律的 semi-structure 格式的文件，如 HTML (HyperText Markup Language) [15]或 XML(Extended Markup Language) [4]。因此，Web 的資料管理是個很難的研究課題。

二、研究目的

在我們過去的國科會計畫中，我們設計了一個資料萃取語言叫做 BODED (Browser Oriented Data Extraction Description)及其雛形系統來解決這些問題。然而如何對所萃取的資料進行儲存，並進一步做一般化的管理也是一件非常重要的事情。萃取出 Web 資料的管理之基本應用有：

- 資料貯存(data storing)：將萃取的資料貯存下來。
- 資訊查詢(information query)：查詢萃取下來的資訊。
- 網站再生工程(Web site reengineering)：可將資料庫內的資料反向顯示成網站。對有些網站年久失修，或使用舊的 Web 伺服器技術，亦可藉由此技術重新修整。
- 資料資料整合(Web site integration)：將不同網站萃取下來的資料整合在同一個資料庫，或甚至整合成一個新的網站。

這些 Web 資料管理應用的共通需求，是如何將 Web 的資料與資料庫的資料容易地互轉及整合，以便於管理。然而由於 Web 資料較為不規律，解決這些需求並非易事。首要工作是研究明確及有彈性的 Web 資料模式，如此方能簡化這些需求。因此，本計畫將以過去設計出的 BODE 資料萃取系統為基礎，提出一套適用於 BODE 的資料模式，以便於管理所萃取出的資料。

三、研究背景：

過去我們進行許多相關的研究，如 WebOQL0、XML Information Set、XQuery and XPath Data Model[16]、Relational Data Model、Object Data Model、OQL、ORDB 等資料庫系統或資料定義語言，並參考 BODE 系統以及語言的特性，改良發展為 DESDL Data Model。在說明如何在網頁上萃取資料以及以有系統的方式儲存這些資料之前，我們先來看一個例子。假設有一個簡化的論文資料庫網站，在這個網站中包含兩層的分類，圖 1 中顯示包含主分類的論文資料庫網站首頁，在主分類網頁中，有超鏈結會連結到含有多個次分類的次分類網頁。其中一個次分類網頁的內容顯示在圖 2 中。次分類網頁中則有超鏈結連結到屬於該次分類的論文列表的網頁。圖 3 則顯示列有好幾篇論文以及其作者名稱、標題與出版商的論文列表網頁。在論文列表網頁的結尾處，有一個 URL 鏈結到下一個論文列表網頁以顯示更多的論文列表。在論文列表網頁的每一篇文章的標題，皆有一個超鏈結連結到論文網頁。圖 4 顯示一個論文網頁，在論文網頁中，有論文的標題、摘要、作者與出版商等資訊。在每一個作者名稱上，皆有一個超鏈結連結到作者資訊的網頁。圖 5 顯示一個位於 wu.html 的作者資訊網頁，在作者網頁中，包含作者的稱謂、電子郵件地址、電話，以及其相關著作。每一篇著作，則又有超鏈結連結到上一層的論文網頁。

```
<TABLE>
  <TR><TD><A href="db.html">Databases</A></TD></TR>
  <TR><TD><A href="al.html">Algorithms</A></TD></TR>
    . . .
</TABLE>
```

圖 1、包含主分類的論文資料庫網站首頁

```
<TABLE>
  <TR><TD><A href="de.html">Data Extraction</A></TD></TR>
  <TR><TD><A href="dm.html">Data Mining</A></TD></TR>
    . . .
  </TR>
</TABLE>
```

圖 2、位於 db.html，包含次分類的網頁

```
<TABLE border=1 width="100%">
  <TR>
     <TD><A href="p1.html">On the Web Data Extraction Model</A></TD>
     <TD>I-C. Wu, J.-Y. Su, L.-B. Chen </TD>
     <TD>SEKE 2005.</TD>
  </TR>
  <TR>
     <TD><A href="p2.html">A Web Data Extraction Description
                          Language and Its Implementation</A></TD>
     <TD>I-C. Wu, J.-Y. Su, L.-B. Chen </TD>
     <TD>COMPSAC 2005</TD>
  </TR>
</TABLE>

<A href="nextpage.html">NEXT</A>
```

圖 3、位於 de.html 中屬於 Data Extraction 次分類的論文列表網頁

```
<TABLE>
   <TR>
      <TD>Title: </TD>
      <TD><B>On the Web Data Extraction Model</B></TD>
   </TR>
   <TR>
      <TD>Abstract: </TD>
      <TD><I>…</I></TD>
   </TR>
   <TR>
      <TD>Authors: </TD>
      <TD> <A href="wu.html">I-C. Wu</A>
           <A href="su.html">J.-Y. Su</A>
           <A href="chen.html">L.-B. Chen</A> </TD>
   </TR>
   <TR>
      <TD>Publisher: </TD>
      <TD>SEKE 2005</TD>
   </TR>
</TABLE>
```

圖 4、位於 p1.html 的論文網頁

```
<P>I-Chen Wu</P>
<TABLE>
  <TR>
    <TD>Title</TD>
    <TD>Associate Professor</TD>
  </TR>
  <TR>
    <TD>Email</TD>
    <TD>icwu@csie.nctu.edu.tw </TD>
  </TR>
  <TR>
    <TD>Tel</TD>
    <TD>035731855</TD>
  </TR>
  <TR>
    <TD>Publisher</TD>
    <TD>
      <TABLE>
        <TR><TD>
          <A href="p1.html">On the Web Data Extraction Model</A>
        </TD></TR>
        <TR><TD>
          <A href="p2.html">A Web Data Extraction Description
                            Language and Its Implementation</A>
        </TD></TR>
        <TR><TD>
          <A href="p3.html">BODE: A Data Extraction Service
                            Description Language</A>
        </TD></TR>
      </TABLE>
    <TD>
  </TR>
</TABLE>
```

圖 5、位於 wu.html 的作者資訊網頁

在上述的例子當中，我們要萃取所有的論文資料，並且將這些資料自動的儲存到資料庫中。為了要萃取整個網站的論文資料，我們必需瀏覽網站中的所有論文列表網頁，然後連結到論文網頁中萃取相關資料。

在之前的研究 0[2][3][8][11][12][13][16][17]中，要萃取網站上的資料並儲存，通常會使用搜尋引擎、撰寫自訂的 wrapper、或是撰寫網頁查詢程式，然後利用特定的系統將網頁中的資料萃取出來。當資料萃取出來後，再利用撰寫好的資料對應程式，將資料對應儲存到資料庫當中。然而，在萃取資料的過程當中，有一些重要的資訊在網頁中的資料被萃取完成，並進入到下一頁進行其他的萃取之後即遺失。比如頁面之間的超鏈結關係。

舉例來說，在圖 2 當中，我們萃取了 Data Extraction 次分類，並萃取了該次分類中的論文列表。當整個網站萃取完成後，該如何知道哪些論文列表中的論文是屬於哪些次分類呢？如果這項資訊未在萃取 Data Extraction 次分類資料與該次分類所鍊結的論文列表網頁資料的期間記錄下來，當這兩頁萃取完成後，該資料即已遺失。因此這一類的資訊，必須要在萃取資料的期間就記錄下來。這樣的記錄通常必須要與網頁資料的萃取系統搭配，在萃取期間紀錄此項資訊。在過去的研究中，Araneus[5]系統與 WebOQL0 即是使用這種模式進行相關訊息的紀錄。

我們使用在之前的計畫中所開發的 BODE 網頁資料萃取系統來萃取網頁上的資料。BODE 系統提供了一個外掛程式(Plug-in)的系統，稱為 BODEDLet。在 BODEDLet 中，我們可以讀取 BODE 系統從網頁上所萃取的資料，並能控制 BODE 系統的資料萃取行為。因此，我們使用 BODEDLet 在萃取網頁資料期間，記錄資料之間的關聯。

四、BODE 系統簡介

BODE 系統是一套 Web 文件之資料自動萃取系統。這套系統主要解決有關網頁文件上的資料自動萃取問題。網頁文件的資料萃取涵蓋兩個部份，一是瀏覽順序，二是資料萃取。

許多網頁並沒有辦法直接以網址來取得。比如許多網站如 104 人力網需要登入才能瀏覽重要資料，有些網站如奇摩站的超連結是經由執行某些 Javascript 程式才會產生。因此在萃取網頁資料前，常常必須瀏覽至所需的網頁後才能做資料萃取。因此網頁間瀏覽的順序相當重要。這個計畫除了解決網頁資料萃取的問題之外，也必須同時解決網頁瀏覽的問題。

為了讓網頁資料萃取更有彈性且更易設計，我們設計了一個以 XML 為基礎的新的描述語言，叫做瀏覽器導向資料萃取描述(Browser Oriented Data Extraction Description)語言，簡稱 BODED，用來描述資料萃取過程中的流覽與資料萃取。

BODED 語言

在 BODED 中，一個 script 程式包函一組網頁的萃取服務，每一個網頁萃取服務可萃取該網站中具有特定格式的網頁，並處理這些萃取出來的資料。例如，將萃取的資料儲存到資料庫中，或利用這些資料來瀏覽下一個網頁。

舉例來說，若要以 BODE 系統萃取圖 1 中的主分類資料，則要以 BODED 語言撰寫一個資料萃取程式。圖 6 顯示一個用來萃取圖 1 中的主分類的 BODED 萃取程式。在這個萃取程式中，會載入論文資料庫網站首頁，並針對每一個主分類萃取主分類的名稱。之後，在顯示該主分類的瀏覽器上，模擬使用者按一下滑鼠左鍵，以瀏覽到下一頁。這種作法是為了解決在有些網頁中，超鍊結是由網頁中的 Javascript 程式來產生的問題。利用模擬使用者按一下滑鼠，可以讓瀏覽器執行該 javascript 並產生正確的下一頁的超鍊結。

BODED script 的最外層的 XML 元素是 BODED 元素(element)，其內包含整個 BODED 萃取程式。這個元素包含兩種不同型態的子元素，即 INIT 與 PAGE。從 INIT 元素當中，BODE 系統得知要在第一個瀏覽器中輸入 url 屬性所指定的超鍊結，以取得網頁。之後載入名稱屬性內容為 MainCat 的 PAGE 元素中的 BODED script 在該瀏覽器上執行。

PAGE 元素使用的 FOREACH 元素是一個迴圈，可以用來指定要對每一個萃取出來的 HTML 元素、屬性或文字，所要執行的動作。比如針對每一個主分類，萃取其名稱，並利用主分類上的超鍊結連結到下一頁。在名為 MainCatPage 的 PAGE 元素中，名為 MainCat 的 FOREACH 元素是一個迴圈結構。BODE 系統在執行該 FOREACH 元素時會產生一個以該 FOREACH 名稱為名的變數。例如在圖 6 中，所產生的變數名稱為 MainCat。在 PAGE 元素所對應的瀏覽器當中，FOREACH 元素先萃取所有的主分類的 HTML 超鍊結元素(由

xpath 屬性指定主分類的 HTML 超鏈結元素所在的位置)。在此假設有 n 個。這時 FOREACH 元素所包含的 BODED script 片斷皆會被執行 n 次。在第 i 次的迴圈中，MainCat 變數內容為第 i 個主分類的 HTML 超鏈結元素。在 FOREACH 當中的每一個元素皆是以 MainCat 變數所指向的元素為根節點，在此根節點上使用 xpath 的相對路徑來萃取資料。例如，在圖 6 的 FOREACH 中，使用 VAR 元素來萃取主分類的名稱到名為 MainCatName 的變數中，並將該變數置於 BODE 系統中的萃取資料集區(extracted data pool)。由於主分類的名稱是在 HTML 超鏈結的文字部分，因此 xpath 屬性中的值為"./text()"。在這個 XPath 當中，"."代表目前的節點，即 HTML 超鏈結元素。在圖 6 的例子中，若是目前在第 i 次的迴圈中，則"."指的是第 i 個主分類的 HTML 超鏈結。而"./text()"則取出目前 HTML 超鏈結的文字部分，也就是第 i 個主分類的名稱。接著利用 EVENT 元素模擬使用者在瀏覽器中的主分類項目上按滑鼠左鍵的 Click 動作。而萃取的規則如果是使用 XPath 語言，則由 xpath 屬性指定。在 BODED 的變數中，可包含被萃取出來的字串資料或指向某個元素的指標。

```
<BODED Name="CSArchive">
  <INIT page="MainCatPage"
        url="http://bode.csie.nctu.edu.tw/index.html" />
  <PAGE Name="MainCatPage">
    <FOREACH Name="MainCat" xpath="//TD/A">
      <VAR name="MainCatName" xpath="./text()" />
      <EVENT name="LinkToSubCat" xpath="." service="SubCat"
             type="OnClick" />
    </FOREACH>
  </PAGE>
  ...
</BODED>
```

圖 6、用來萃取圖 1 主分類的 BODED 萃取程式

BODED 使得控制瀏覽順序與資料萃取變的更加容易。簡單的說，BODED 具有下列的特色。

(1) 使用 XPath[17] 當作萃取網頁內部資料的查詢敘述格式。

(2) 能以特定的順序瀏覽網頁。

(3) 可以填寫表單並啟動下一個服務來萃取提交表單後的下一頁。

(4) 支援使用 javascript 所產生的超鏈結。

(5) 支援稱為 BODEDLet 的外掛程式，以處理所萃取的資料。例如，將萃取的資料儲存到資料庫或瀏覽下一頁。

(6) 與目前的瀏覽器的規格一致。

(7) 模擬按一下的動作並啟動下一個服務來萃取下一頁。

此外，BODE 系統包含一個 WYSIWYG 的視覺化的工具，如圖 7，讓使用者能快速的產生能萃取網站資料的 BODED 萃取程式。我們過去的經驗是，透過這視覺化工具的輔助，我們可以在一小時，甚至幾十分鐘內完成 BODED 萃取程式的設計。

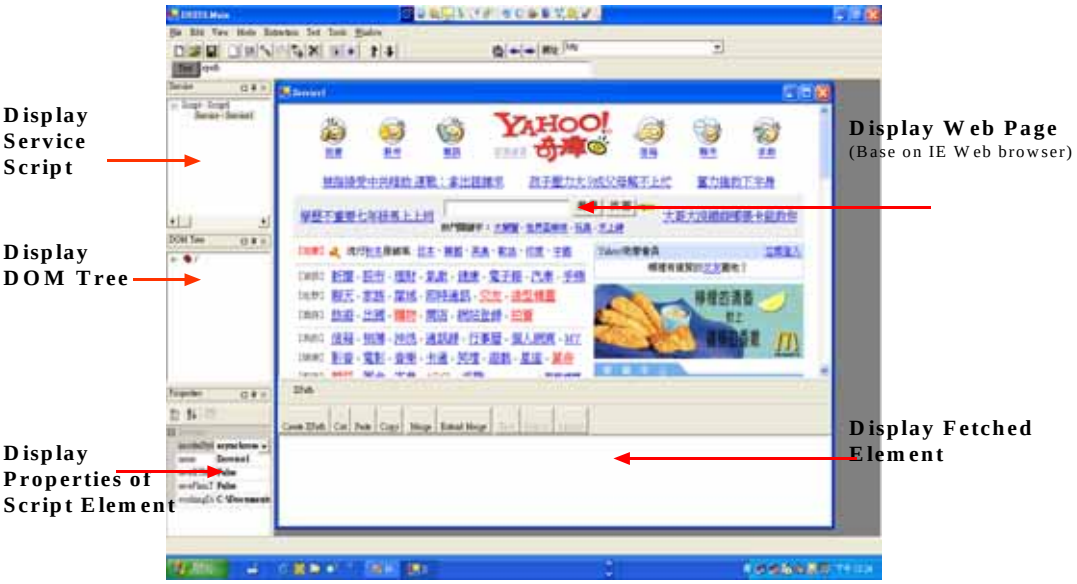


圖 7、BODE 萃取系統的視覺化介面

圖 8 是一個利用 BODE 系統將網頁上的資料萃取出來並儲存到資料庫中的例子。在 BODE 系統中，若是在萃取資料結束後才將資料儲存於資料庫中，則網頁之間連結的關聯資訊已經消失。在圖 1 中，href 屬性為 db.html 的超鏈結與圖 2 的網頁中的內容具有包含的關係，也就是在圖 1 的網頁中，Database 主分類連結到 db.html 的次分類網頁中。其意義為 Database 主分類包含 Data Extraction 以及 Data Mining 次分類。若超鏈結與目的網頁間的關係消失，則無法復原此項關係。因此這項資訊必須在當使用可連結到下一個網頁的元素來建立下一頁的瀏覽器時，儲存這些資訊。BODE 系統提供一套外掛程式介面，可以在瀏覽與萃取網頁的同時，儲存所萃取的資料。同時，這套萃取系統也可以控制 BODE 系統的運作。

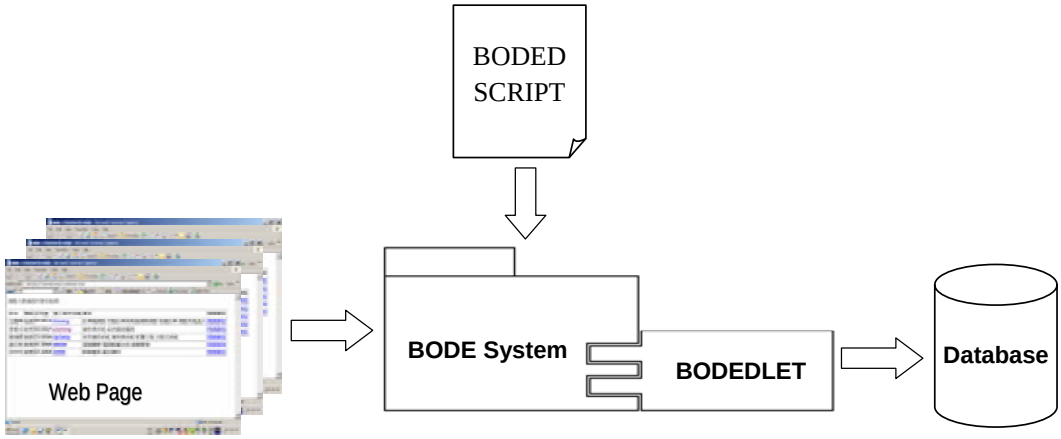


圖 8、使用 BODE 系統萃取資料並儲存到資料庫中

圖 9 是一個利用 BODEDLET 來儲存資料，並瀏覽到下一頁繼續萃取的範例。圖 9 的程式是根據圖 6 的程式，再加入 BODEDLET 元素，以儲存所萃取的資料。BODEDLET 元素中的 code 屬性，指定要載入執行的 BODEDLET 物件。而 archive 屬性則指明該物件位於哪個檔案中。BODEDLET 元素的子元素 PARAM 則描述 BODEDLET 所需要的參數名稱與值。

```
<BODED Name="CSArchive">
  <INIT page="MainCatPage"
        url="http://bode.csie.nctu.edu.tw/index.html" />
  <PAGE Name="MainCatPage">
    <FOREACH Name="MainCat" xpath="//TD/A">
      <VAR name="MainCatName" xpath="./text()" />
      <BODEDLET name="BODEDB" code="BODEDLET.saveDB"
                archive="bodedlet.dll">
        <PARAM name="SavedVAR1" value="MainCatName" />
      </BODEDLET>
      <EVENT name="LinkToSubCat" xpath="." service="SubCat"
             type="OnClick" />
    </FOREACH>
  </PAGE>
  ...
</BODED>
```

圖 9、儲存萃取資料並且控制 BODE 系統瀏覽下一頁的 BODEDLET 範例

五、Data Model

在 Web 資料模式的研究中，我們設計了一個新的資料模式，叫做 BODE 資料模式，並定義描述該模式的語法，用以描述網頁上所萃取的資料。同時，我們也設計一個對應語言。這個對應語言能夠將 BODED script 中指定萃取的資料對應到資料模式中的特定資料欄位。

BODE 資料模式

在 Web 資料模式的研究中，我們定義了 BODE 資料模式(BODE Data Model)以及其描述語言。在 BODE 的資料模式中，視一筆資料紀錄為一個資料物件。一個資料物件中須具備的欄位由資料類別所定義。圖 10 顯示一個描述圖 1 的 BODED 語言所萃取的主分類名稱資料的 BODE 資料模式定義。在圖 10 中定義了名為 MainCat 的類別。在該類別中，有一個欄位叫做 MainCatName，其型態是字串(string)，長度為 20。

描述 BODE 資料模式的語言稱為 Class Definition Language，簡稱 CDL。CDL 是以 XML Schema 架構（Framework）為基礎，是一個用來描述 BODE 資料內容的結構的 XML Language。

目前 CDL 定義了兩個標籤，一個是 CLASS，另一個是 FIELD。CLASS 標籤定義資料物件的類別。其內含 FIELD 標籤。

CLASS 標籤包含兩個屬性，一個是 name，用以指出資料類別的名字。一個是 key，用以指出資料類別是以哪些欄位當作主鍵。在資料物件資料庫中，具有同樣主鍵的資料會被視為是同一筆資料。

FIELD 標籤包含三個屬性，第一個是 name，為該欄位的名稱，第二個是 type，為該欄位的資料型別，第三個是 length，是該欄位所儲存的資料的長度。

```
<CLASS name="MainCat" key="Name" >
   <FIELD  name="MainCatName" type="string" length="20" />
</CLASS>
```

圖 10、主分類項目資料的 CDL 定義

CDL 的主要目的是定義資料類別的結構，依照某一資料類別所建立的資料稱為資料物件(Object)。資料類別內包含一或多個資料欄位，資料類別中的資料欄位必須是基本資料型別，也就是說，在資料類別中，個別的資料欄位中只能存放簡單型別的資料，而不能存放資料物件。

CDL 所定義的資料類別(Class)，其作用就如同資料庫資料的表格(Table)，其內的欄位(Field)則定義一個資料物件所具備的欄位。資料物件(Object)就如同資料庫資料表格的紀錄(Record)。

CDL 定義資料類別時，首先需要先給予資料類別(Class)命名，其次是給予資料類別(Class)內的資料欄位(Field)的名稱以及資料型別。

CDL 的定義了兩種不同的基本資料型別：

(1) 字串 (String)

(2) 數值 (Number)

另外，CDL 可以定義一組資料類別的鍵值(Key)，鍵值是由數個資料類別內的資料成員所組成，相同資料類別但不同資料物件(Object)內的鍵值必然不相同。

Key 的存在，不僅可以區別資料物件，並且可以協助 BODE 系統建立兩個資料物件之間的關聯(Relationship)。

由於 BODE 的資料類別(Class)的作用，就如同資料庫的資料表(Table)，是用來定義內部的組成欄位。資料類別中所儲存的資料物件(Object)，就如同資料庫資料表的紀錄(Record)。基本上，資料類別與資料庫資表是可以直接以 One-to-One 的方式對應的。因此資料類別在定義之後，便可以利用工具將資料類別轉成關聯式資料庫的資料表。其對應方式如下：

(1) 一個資料類別對應到資料庫中的一個資料表。將資料類別名稱對應到資料表時，是以資料類別名稱在加上前置字串"BODE_"的方式命名。

(2) 一個資料類別內，一個基本資料型別的欄位可對應到一個資料表的欄位。

(3) 資料物件與資料物件之間的關聯性，是用一個稱為 Universal Relation Table 的資料表來紀錄。

例如，圖 11 為一個稱為 TEACHER 的資料類別，其對應的關聯式資料庫的 SQL 資料表如圖 12。

```
<CLASS name="TEACHER" key="NAME“ >
   <FIELD  name="NAME"  type="STRING"  length="50" />
   <FIELD name="TITLE"  type="STRING"  length="50" />
   <FIELD name="ROOM"   type="STRING"  length="50" />
   <FIELD name="SPEC"   type="STRING"  length="254" />
</CLASS>
```

圖 11、TEACHER 資料物件的 CDL 定義

```
TABLE BODE_EMPLOYEE
(
  RID LONG,
  DOMAIN     CHAR(32),
  NAME       CHAR(50),
  TITLE      CHAR(50),
  ROOM       CHAR(50),
  SPEC       CHAR(254)
)
```

圖 12、圖 11 中的 TEACHER 資料物件所對應的 SQL 資料表

BODED 資料模式對應語言

為了要讓資料的萃取與儲存能夠自動化的進行，因此我們設計了一個能夠將 BODED 萃取程式中所萃取的資料項目對應到 BODE 資料模式中的資料欄位的資料定義的語言，稱為 BODED Script to Class Mapping Description Language，簡稱 BSCMDL。該語言利用 BODED 語言的標籤結構，將資料對應到物件模式中。圖 13 的對應規則即是將圖 9 中的 BODED 語言對應到圖 10 的類別定義中。在圖 13 中，FOREACH 元素會對應到圖 10 中的 MainCat 資料類別。而 FOREACH 元素中的 VAR 元素則對應到 MainCat 資料類別中的 Name 欄位。因此在 BODE 系統執行當中，便可依據該對應程式將資料自動的儲存於資料庫中的資料表內。

```
<MAP script="CSArchive">
  <PAGE name="MainCatPage" . . . >
    <FOREACH name="MainCat" class="MainCat" . . . >
      <VAR name="MainCatName" field="Name" />
      . . .
    </FOREACH>
  </PAGE>
  . . .
</MAP>
```

圖 13、將 BODED script 中的萃取資料對應到資料模式中的主分類項目資料的 BSCMDL 程式

BSCMDL 資料對應描述語言是以 XML Schema 架構（Framework）為基礎，其定義的標籤如下：

(1) MAP：MAP 標籤內含一個屬性 script，用來指出 Script 的名字。

(2) BLOCK：BLOCK 標籤包含數個重要屬性。name 是用以指出 BODED Script 內的區塊名稱。class 是用以指出相對的資料類別的名稱。bind 是用以指出將從不同的 Event 中所萃取的資料項目加入到某一個資料類別中，成為其中的一個欄位。rclass 是用以指出一個與此一資料類別有關聯的資料類別的名稱，domain 是用以標示及區分不同的資料來源。

(3) VAR：VAR 標籤包含數個重要屬性，name 是用以指出 BODED Script 內的 VAR 標籤的名稱，field 是用以指出相對的資料類別內的欄位名稱。

圖 14 顯示將 BODED script 中兩個不同的 BLOCK 元素以及其所包含的 VAR 元素對應

到 TEACHER 以及 COURSE 資料類別，並建立兩個類別間的關聯的 BSCMDL 程式。TEACHER 是主要資料項目，COURSE 是次要資料項目，rclass 指示 BODE 系統建立 COURSE 與 TEACHER 的關聯：

```
<MAP script="scrpNET">
  <BLOCK name="eachTeacher"  class="TEACHER" domain="NET">
    <VAR name="varname"      field="NAME" />
    <VAR name="varTitle"     field="TITLE" />
    <VAR name="varRoom"      field="ROOM" />
    <VAR name="varSpecialty" field="SPEC" />
  </BLOCK>
  <BLOCK name="eachCourse" class="COURSE" doamin="NET"
          rclass="TEACHER" rname="TEACHER-COURSE">
    <VAR name="varCNO"    field="CNO" />
    <VAR name="varCname"   field="Cname" />
    <VAR name="varCtype"   field="CTYPE" />
    <VAR name="varCscore"  field="CSCORE" />
    <VAR name="varChr"    field="CHR" />
    <VAR name="varTime"    field="CTIME" />
    <VAR name="varCroom"   field="CROOM" />
  </BLOCK>
</MAP>
```

圖 14、將 BODED script 中萃取的教師與課程資料分別對應到資料模式中的 TEACHER 資料類別與 COURSE 資料類別，並建立兩個資料類別間的關聯

圖 15 以書本以及書本明細資料為例，說明 bind 屬性的功能。bind 指示將所萃取的書本明細資料加入 BOOK 資料類別：

```
<MAP script="scrBook">
  <BLOCK name="eachBOOK" class="BOOK">
    <VAR name="VARTITLE"   field="TITLE" />
    <VAR name="VARAUTHOR"  field="AUTHOR" />
    <VAR name="VARINDEX"   field="INDEX" />
    <VAR name="VARPY"      field="YEAR" />
    <VAR name="VARTYPE"    field="TYPE" />
    . . .
  </BLOCK>
  . . .
  <BLOCK name="pageBOOKDETAIL"  bind="BOOK">
    <VAR name="VARISBN"    field="ISBN" />
    <VAR name="VARPRICE"   field="PRICE" />
    . . .
  </BLOCK>
```

```
</MAP>
```

圖 15、將 BODED script 中兩個不同的 BLOCK 所萃取的項目，結合到 BOOK 資料類別

在做 Script 與資料類別對應時，將採取以下原則：

(1) BLOCK 根據其名稱,可指定為 BODED script 中的 PAGE 以及 FOREACH 等元素。藉由執行 BODEDLET，可以將 PAGE 以及 FOREACH 等元素所萃取的資料對應至資料類別。

(2) BLOCK 被啟動的時機，通常是在 BODE 系統結束一個 PAGE 或 FOREACH 等元素的萃取作業的一個循環或是結束之後。

(3) 資料類別與資料類別的結構關聯次序，是由 rclass 屬性所指定的。

由於資料大部份可以從 BODED script 中的 PAGE 或 FOREACH 中的變數而來，為了要提供更簡化的操作，BODE 系統對 BODED Script 稍作修改，並且根據這些修改，提供一個 CDL 及 BSCMDL 產生器，協助使用者產生 CDL 及 BSCMDL。

對於 BODED Script 的更動如下：

(1) 在 PAGE、FOREACH 等標籤內，新增一個屬性 class，用來指示對應的的資料類別名稱。對於未定義屬性 class 的標籤將被 CDL 及 BSCMDL 產生器所忽略。屬性 domain，用來指示對應的的資料類別名稱。屬性 rclass，用來指示關聯此一資料類別的資料類別名稱。屬性 rname，用來設定關聯的名稱。

(2) VAR 標籤內，新增一個屬性 field，用來指示對應的的資料欄位名稱。isKey 則表示是否為鍵值。type 指定型態。length 則指定欄位長度。若 field 屬性不存在，CDL 及 BSCMDL 產生器將產生一個與 VAR 相同名稱的欄位，若 type 屬性不存在，則預設為 String。若 length 屬性不存在，則預設長度為 32 的 String。isKey 若為 true，則該欄位為鍵值欄位。若 isKey 不存在，則該欄位為非鍵值欄位。

例如圖 16 的 BODED Script，經由 CDL 及 BSCMDL 產生器，可以得到圖 17 的 CDL 程式，以及圖 18 的 BSCMDL 程式。

```
<BODED name="scrpNET" InitialService="pageTeacher"
       url=".../index.html">
  <PAGE name="pageTeacher">
    <FOREACH name="forTeacher" xpath=".../TR[position()>=1]"
             class="TEACHER">
      <VAR name="varName" xpath="/TD[0]" />
      <VAR name="varTitle" xpath="/TD[1]" />
      <VAR name="varRoom" xpath="/TD[2]" />
      <VAR name="varSpecialty" xpath="/TD[3]" />
      <BODEDLET name="SaveDB" code="BODEDLET"
                archive="BODEDLETDB.dll">
        <PARAM name="domain" value="NET" />
      </BODEDLET>
    </FOREACH>
  </PAGE>
</BODED>
```

圖 16、萃取教師資料的 BODED script

```
<CLASS name="TEACHER">
  <FIELD name="varName"      type="STRING"  length="32" />
  <FIELD name="varTitle"     type="STRING"  length="32" />
  <FIELD name="varRoom"      type="STRING"  length="32" />
  <FIELD name="varSpecialty" type="STRING"  length="32" />
</CLASS>
```

圖 17、圖 16 中的 BODED script 經由 CDL 產生器產生的 CDL 程式

```
<MAP name="scrpNET"
  <BLOCK name="forTeacher" class="TEACHER" domain="NET">
    <VAR name="varName"    field=" varName" />
    <VAR name="varTitle"     field=" varTitle" />
    <VAR name="varRoom"    field="varRoom" />
    <VAR name="varSpecialty"   field="varSpecialty" />
  </BLOCK >
</MAP>
```

圖 18、圖 16 中的 BODED script 經由 BSCMDL 產生器產生的 BSCMDL 程式

由 CDL 及 BSCMDL 產生器，產生 CDL 及 BSCMDL 的原則如下：

(1) BODED Script 的 PAGE、FOREACH 等標籤，將會對應到 CDL 所定義的資料類別。若此類標籤有一個 class 屬性，則表示要產生資料類別定義。否則，表示不會產生資料類別定義。

(2) BODED Script 的 VAR 標籤將會對應到 CDL 所定義的資料欄位。對應方式限定為一對一。若 BODED Script 的 VAR 標籤不含 field 屬性，則表示對應的資料欄位名稱與 VAR 名稱相同。若 BODED Script 的 VAR 標籤有一個 field 屬性，則表示使用給定的資料欄位名稱及資料型別。

(3) 如果 BODED Script 的 VAR 標籤沒有指定對應的資料型別，系統將此一欄位預設型別為字串。

(4) 如果 BODED Script 的 VAR 標籤沒有指定對應的資料長度，系統將此一欄位預設長度 32。

(5) 一個 BODED Script 的 VAR 標籤對應的資料欄位，隸屬於包含此一 VAR 的 BODED Script 的 PAGE、FOREACH 等標籤對應的資料類別之內。而且不得超越其範圍。

BODE 的資料模式處理系統

由於資料的對應與儲存必須在 BODE 系統執行當中進行。因此在 BODE 系統上，是以一個 BODEDLET 外掛程式來實作資料模式處理系統，如圖 19。這個資料模式實作(Data Model Implementation)在執行時讀取類別定義與對應規則，將萃取出來的資料對應到物件資料庫中。並建立資料之間的關聯。

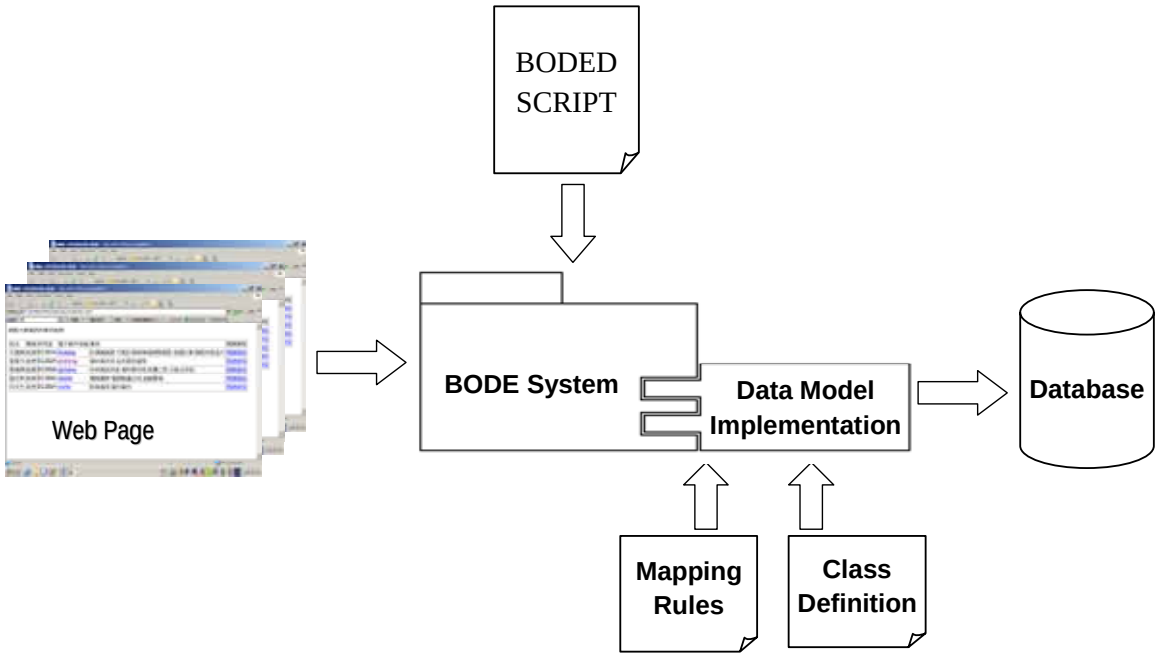


圖 19、在 BODE 系統中的資料模式實作

資料關聯

在網頁當中的資料與資料之間，有些會具有特定的關係，這樣的關係我們稱為關聯。以資料庫的角度來看，關聯大致有四種，即一對一、一對多、多對一以及多對多等關聯，如圖 20。在網頁當中的資料，也存在這四種關係。例如圖 4 的論文網頁當中，論文的標題(Title)、摘要(Abstract)與出版商(Publisher)具有一對一的關係。也就是一篇論文當中，會有一個標題、一份摘要以及由一個出版商出版。這些資料都是屬於一篇論文中的屬性，且對於一篇論文當中，具有唯一性。因此我們可以建立一個論文的類別，如圖 21 中的 Paper 類別，並將這些彼此之間是一對一的資料，作為論文類別中的欄位。在 BODED 萃取程式中，要萃取的資料是由 VAR 標籤來描述該資料位於何處。由於 BODED 萃取程式的 XML 結構中，僅有變數類型的標籤才能指定要萃取的資料。因此，這些標籤會被對應到類別中的欄位。為了使資料定義更加簡單，在 BODE 資料模式中規定 CLASS 當中不能包含其他 CLASS。因此 CLASS 沒有辦法表達多層的關係。而可以包含 VAR 標籤的其他標籤，例如 PAGE 或是 FOREACH 標籤，由於包含了欄位，因此會被對應到資料類別。

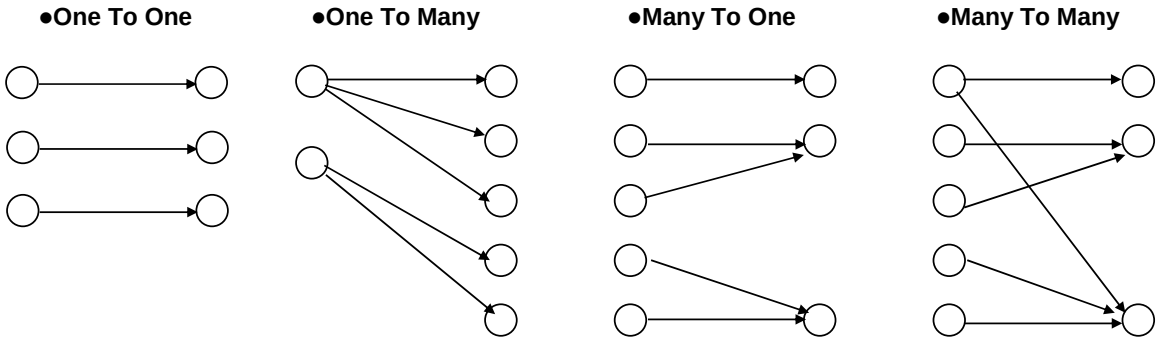


圖 20、資料間的一對一、一對多、多對一與多對多關聯

```
<CLASS name="Author">
  <FIELD name="AuthorName" type="string" length="20" />
</CLASS>
```

```
<CLASS name="Paper" key="Title" >
  <FIELD name="Title" type="string" length="50" />
  <FIELD name="Abstract" type="string" length="300" />
  <FIELD name="Publisher" type=" string" length="100" />
</CLASS>
```

圖 21、圖 4 論文網頁的類別定義

```
<PAGE name="Paper" class="Paper" . . .>
  <VAR name="Title" field="Title" />
  <VAR name="Abstract" field="Abstract" />
  <VAR name="Publisher" field="Publisher" />
  <FOREACH name="Authors" class="Author"
           parentrelation="Paper">
    <VAR name="AuthorName" field="AuthorName" />
  </FOREACH>
</PAGE>
```

圖 22、將圖 23BODED SCRIPT 對應到圖 21 資料模式中的類別定義的對應程式

```
. . .
<PAGE name="Paper">
  <VAR name="Title" xpath="//TR[0]/TD[1]/B[0]" />
  <VAR name="Abstract" xpath="//TR[1]/TD[1]/I[0]" />
  <VAR name="Publisher" xpath="//TR[3]/TD[1]" />
  <FOREACH name="Authors" xpath="//TR[2]/TD[1]/A">
    <VAR name="AuthorName" xpath="." />
    <EVENT name="LinkToAuthor" xpath="." page="Author"
           type="OnClick" />
  </FOREACH>
</PAGE>
. . .
```

圖 23、用來萃取圖 4 論文網頁的 BODED SCRIPT 片段

舉例來說，圖 22 是一個能將圖 23 的 BODED 萃取程式中所萃取的欄位對應到圖 21 所定義的類別中的對應規則。由於圖 23 中的 VAR 元素會被對應到 CLASS 定義中的 FIELD，而圖 23 的 PAGE 包含對應到 Paper 類別中的欄位的 VAR 元素，因此 PAGE 被對應到 Paper 類別。在 BODE 資料模式的對應規則中，規定 BODED 萃取程式中的 VAR 元素的上一層元素必須被對應到一個類別。

然而在圖 4 當中，除了論文的標題、摘要與出版商之外，還有作者的名字。在作者的名字上，有超鏈結連結到作者網頁中。由於一篇論文可能不只有一個作者，因此一篇論文與其作者之間可能具有一對一或是一對多的關聯。圖 23 是一個用來萃取圖 4 的網頁的 BODED 萃取程式片段。在這個萃取程式中，除了萃取論文的標題、摘要與出版商之外，並使用 FOREACH 元素依序萃取作者的名稱與超鏈結。並利用連結到作者網頁的超鏈結，建立一個新的瀏覽器，並連結到作者網頁當中，以名稱為 Author 的 PAGE 服務繼續萃取作者的資料。

當資料之間具有一對多、多對一或多對多的關係時，就沒有辦法以目前單純的 CLASS 定義來定義這樣的關係。因此 BODE 的資料類別在設計時，必需使系統能夠處理及紀錄各種關聯。而且，因為即使是 BODE 在萃取資料的同時，也很難得知兩個資料類別之間的關聯性屬於那一種，因此 BODE 資料類別在處理 Relationship 時，必須使用一般性通用性的做法。為了保持簡單的資料定義格式，並且同時保留可以彈性的描述各種物件之間的關係，BODE 使用一個 Universal Relation Table 去紀錄關聯資訊，無論是何種關聯資訊，皆可由 Universal Relation Table 紀錄。

為了要將資料物件(Object)與資料物件的關聯性轉換成紀錄(Record)與紀錄之間的關係，BODE 系統可以使用以下方法：

(1) 給予每一個 Object 一個 RID。

(2) 將資料物件與資料物件的關聯紀錄儲存於 Universal Relation Table 中。

Universal Relation Table 的結構如圖 24：

```
TABLE BODE_RELATION
(
  CID LONG,
  EID LONG,
  PNAME  CHAR(50),
  CNAME  CHAR(50),
  RNAME  CHAR(50),
  DOMAIN  CHAR(50)
)
```

圖 24、Universal Relation Table 結構

其中，PNAME 和 CNAME 是用來紀錄資料類別名稱，PNAME(Container Class Name)紀錄父資料類別的名稱。CNAME 紀錄子資料類別的名稱。PID 是一個 PNAME 所紀錄的資料類別的一個 Record 的 ID，CID 是一個 CNAME 所紀錄的資料類別的一個 Record 的 ID，DOMAIN 是用以標示及區分不同的關聯。

例如，圖 25 表格的第一列資料，其意義是：一個叫做 Teacher 的資料類別的一筆記錄與一個叫做 Course 的資料類別的一筆記錄存在關聯，此一關聯的被指定一個叫做 CSINFO 的 DOMAIN。TEACHER 為此一關聯的父資料類別，COURSE 為此一關聯的子資料類別。

PID	CID	PNAME	CNAME	RNAME	DOMAIN
670296712	670297531	TEACHER	COURSE	TEACHER-COURSE	CSINFO
670296112	670296922	PaperList	Paper	PaperList-Paper	ARCHIVE
670296922	670297201	Paper	Author	Paper-Author	ARCHIVE
...	...	...	...		...

圖 25、描述老師/課程以及論文列表/論文以及論文/作者的關聯資料表片段

例如圖 26 的教師網頁與課程資訊之間存在有一 Hyper Link，此一 Hyper Link 具有關聯意義"教師以及此教師所開課程"的關聯。經由 Universal Relation Table，我們得以保留來自網頁上的關聯資訊，儲存到 Universal Relation Table 中。

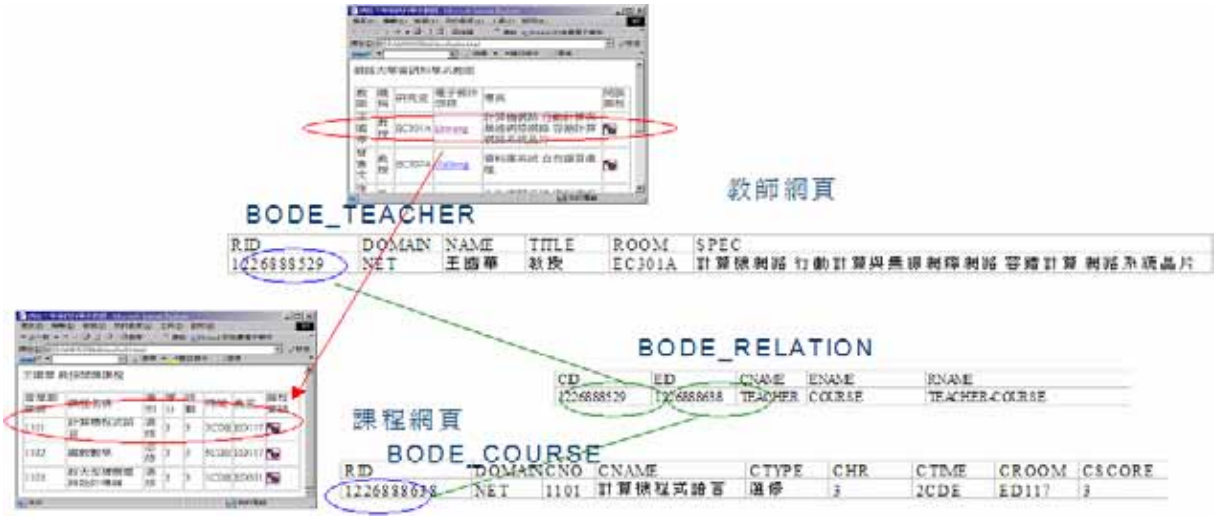


圖 26、資料關聯示意圖

當使用者需要應用到資料庫的資料時，藉由儲存在 Universal Relation Table 的關聯資訊，可以利用由上而下的方式，追蹤並建立如圖 27 的資料關聯樹。

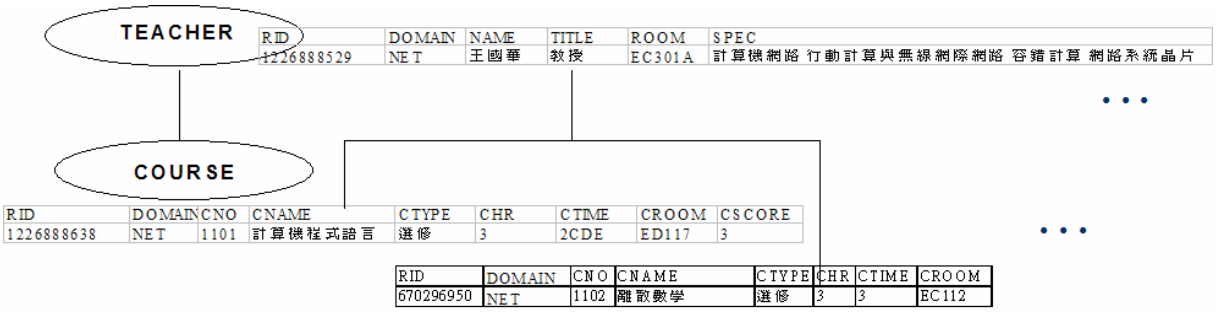


圖 27、資料關聯樹示意圖

使用此一 Universal Relation Table 具有以下的優點：

(1) 可以紀錄各種關連性，如一對一、一對多、多對一或多對多。由於在 Web 資料萃取的應用上，在某些網站的某些資料之間可能具有一對一的關係，但是在其他網站上，同樣的資料之間可能具有一對多或多對一等關係。

(2) 同樣的資料之間，可能存在不只一種關係。關聯資料表可以紀錄某一項資料物件所具有的所有可能的關係。經由此一關聯資料表，可以得知某兩個物件之間具有那些關聯性。

由於在萃取圖 4 中資料時，由於一篇論文可能擁有多位作者，因此這個關係是一對多的關係，圖 21 顯示論文相關資訊的類別定義。在圖 21 中定義了 Paper 以及 Author 類別，在 BODE 資料模式中如何描述 Paper 與 Author 類別之間的關係呢？

在 BODE 資料模式中物件之間的關係是定義在對應規則中。以圖 21 的類別來說，Paper 與 Author 類別之間的關係由圖 22 中的對應規則來指定。在圖 22 當中，名為 Paper 的 PAGE 元素其 class 屬性值為 Paper，表示要對應到 Paper 類別。而名為 Authors 的 FOREACH 其 class 屬性值為 Author，表示對應到 Author 類別。在圖 22 的例子裡，我們在名為 Authors 的 FOREACH 上以 parentrelation 指定 Author 類別與 Paper 類別具有關聯。如此在資料萃取進行中，當建立 Paper 物件後，繼續建立 Author 物件時，BODE 資料模式處理系統就會將這兩個物件的關聯建立在關聯資料表中。對於跨網頁的資料物件之

間的關聯也是以 parentrelation 屬性指定。

跨網頁間的關聯則是由 BODE 資料模式處理程式，也就是圖 9 中的 BODEDLET 來處理。BODE 資料模式處理程式會紀錄目前的資料物件的 ID，之後執行一個 EVENT 元素以建立一個新的瀏覽器來瀏覽下一頁。並且將新的網頁服務，即圖 9 中 EVENT 元素所指定的 SubCatPage 網頁服務連結到新的瀏覽器中執行。同時在 SubCatPage 網頁服務中，建立一個內定的變數，將目前資料物件的 ID 設到該變數中。如此新的網頁服務即可得知前一個網頁服務目前的資料物件是哪一個。且依據此資訊建立網頁間的關聯。

若是資料之間具有多對一的關係。比如不同的論文可能作者是同一人，因此這些由某一個作者所撰寫的論文的網頁中，都有作者的超鏈結連結到作者網頁，這時若沒有特別的機制，會使得同一個作者的資料在資料庫中同時出現好幾次。而違反了一致性的原則。要解決這樣的問題，就必須在類別的定義上加入 key 屬性，指定該類別的物件是由哪些欄位來決定唯一值。圖 21 中的 Paper 類別即指定類別的標題(Title)欄位為 key。因此當萃取資料時，發現具有同樣 key 值的資料被萃取時，則不再產生新的物件，直接將資料庫中已存在的物件與上層的資料物件作關聯。如此可確保多筆論文資料物件中的同一個作者皆關聯到同一個作者的物件。當 BODE 資料模式處理系統遇到具有相同 key 值的資料時，就會停止繼續往下萃取，在建立好物件的關聯後，直接返回上一層繼續萃取。

一但資料與其間的關聯資訊由 BODE 系統儲存到關聯式資料庫中後，即可以關聯式資料庫的 SQL 查詢來查詢資料。

使用標準 SQL 查詢 CDL 定義的資料類別的資料

BODE 使用關聯式資料庫儲存 BODE 所萃取的網路資料，因此仍然可以直接應用 SQL 存取存 BODE 所萃取的網路資料。但是，由於使用 Universal Relation Table 建立類別資料之間的關聯，所以當使用者需要存取經由 BODE 系統萃取所得並且儲存於資料庫中的資料時，仍然可以透過 Universal Relation Table 發覺資料間的關聯。Universal Relation Table 在 BODE 系統資料庫中的資料表名稱為 BODE_RELATION。

以下將以 TEACHER 及 COURSE 兩個資料類別作為例子將說明如何存取兩個資料表的資料。

假設使用者要查詢所有教師與課程資料，由於 BODE_TEACHER 為父資料類別，BODE_COURSE 為子資料類別，SQL 的建立方式如下：以 Universal Relation Table 的 PID 關聯 BODE_TEACHER 資料表，以 Universal Relation Table 的 CID 關聯 COURSE 資料表。即可找到所有有關聯的 TEACHER 與 COURSE 資料。

```
SELECT T.*, C.*
FROM   BODE_TACHER T, BODE_RELATION R, BODE_COURSE C
WHERE  T.RID = R.PID
  AND  C.RID = R.CID
```

圖 28、BODE 資料模式資料的查詢

網站資料的整合

為了能夠將從各個不同網站抓取，具有不同資料類別的資料，對應到一個共同的資料類別中，需要有一套工具能夠將不同的資料類別欄位定義，對應到共同的資料欄位定義。這個過程我們稱為資料格式轉換。資料格式轉換是資料管理中，非常重要的一件工作。例

如，有一位消費者，想要在購買商品之前，先從不同的銷售網站，取得產品銷售資訊。進行規格及售價上的比較。為了方便進行比較的工作，此一消費者需要將萃取自不同網頁的資料，匯整到同一組資料類別，並且建立新的關聯，以方便進行比價作業。

當要進行資料轉換時，如果使用單純的 SQL 描述資料的轉換，並且建立新的 Relationship 其實並不容易。在這個研究中，我們提出一套語言，稱為資料類別對應到資料類別的描述語言(Class to Class Mapping Description Language)，簡稱 CCMDL 來進行資料轉換。

CCMDL 資料轉換語言也是以 XML Schema 架構（Framework）為基礎，是一個用來描述如何將 BODE 資料模式中的資料類別轉換成另一個資料類別的 XML 語言。此一轉換語言除了可以轉換資料之外，還可以建立新的 Relation。圖 29 為 CCMDL 的一個範例。

CCMDL 資料轉換語言定義的標籤如下：

(1) TRANSLATIONS：TRANSLATIONS 內含有一至數個資料轉換規則作為成員。

(2) TRANSLATION ： TRANSLATION 是用以指出一組相關聯的資料轉換。TRANSLATION 包含一個重要屬性 mode，目前 TRANSACTION 分成兩種 model，分別是 General、Aggregation。

(3) SOURCE，描述資料來源，一組 TRANSLATION 擁有一組資料來源描述，其中包含來源 CLASS、RELATION 以及 WHERE 三種元素。

(4) CLASS，描述來源資料類別，並且可以指定來源資料的別名。

(5) RELATIONSHIP，描述來源資料之間的關聯，並且可以分別指定擁有特定 DOMAIN 的資料及特定 DOMAIN 的關聯。RELATION 有三項參數，第一個資料類別及其 DOMAIN，第二個資料類別及其 DOMAIN，最後一個是兩個資料類別間 RELATION 的 RNAME。資料類別的 DOMAIN 及最後一個參數可以省略。

(6) WHERE，描述來源資料的限制條件，例如

(7) CLASS，CLASS 是用來描述資料轉換時的目的地資料類別，CLASS 標籤包含數個重要屬性，name 是用來指出目的資料類別名稱。doamin 是用來指示資料來源。rclass 指出將與那一個資料表的資料建立關聯。rname 是用來指定關聯的的名稱。mode 是用指示重複資料的處理方式，包含 OVERWRITE 及 PREEMPT 兩種模式。

(8) FIELD，FIELD 是用來描述資料轉換時的目的地資料類別的組成欄位及資料來源，FIELD 標籤包含數個重要屬性，name 是用以指出欄位的名稱，value 是用以指出資料來源或是依運算式。

```
<TRANSLATIONS>
 <TRANSLATION model="Aggregation">
  <SOURCE>
   <CLASS>TEACHER AS TEACHER</CLASS>
   <CLASS>COURSE AS COURSE</CLASS>
   <RELATIONSHIP>TEACHER[NET],COURSE[NET],TEACHER-COURSE</RELATIONSHIP>
   <WHERE>CTYPE='必修'</WHERE>
  </SOURCE>
  <CLASS name="ALLCOURSES" wmode="PREEMPT">
     <FIELD field="CTITLE" value="COURSE.TITLE"/>
  </CLASS>
  <CLASS name="ALLTEACHERS" mode="PREEMPT" rclass="ALLCOURSE" rname="ALLCOURSE-TEACHER">
```

```
      <FIELD field="NAME"  value="TEACHER.NAME"/>
      <FIELD field="TITLE" value="TEACHER.TITLE"/>
      <FIELD field="ROOM"  value="TEACHER.ROOM"/>
   </CLASS>
  </TRANSLATION >
</TRANSLATIONS>
```

圖 29、使用 Aggregation 的 CCMDL 程式

CCMDL 的模式

CCMDL 提供了以下兩種資料轉換的模式：

- General 模式 – 預設的資料轉換模式。使用"GENERAL"模式時，系統不會考慮 DOMAIN。任何一個 KEY 的資料仍然只有一組。若是遇到相同 KEY 的資料時，則以" OVERWRITE "或是"PREEMPT"模式處理。
- Aggregation 模式 – 使用"AGGREGATION"模式時，系統會先考慮 DOMAIN。在相同 DOMAIN 下的資料,任何一個 KEY 的資料仍然只有一組。若是遇到相同 KEY 的資料時，則以" OVERWRITE "或是"PREEMPT"模式處理。

首先，我們以兩個來自不同網頁的資料為例，進行資料格式及結構的轉換，第一組網頁資料是"教師關聯教師所開的課程"，第二組網頁資料是"學校課程關聯授課講師"；我們將要從中得到"課程名稱關聯教師"。此一例子我們將假設教師的姓名不會重複，且分別以課程名稱及教師的姓名為鍵值。

(1) 使用"GENRAL"模式時，任何相同課程名稱將視為單一課程資料。

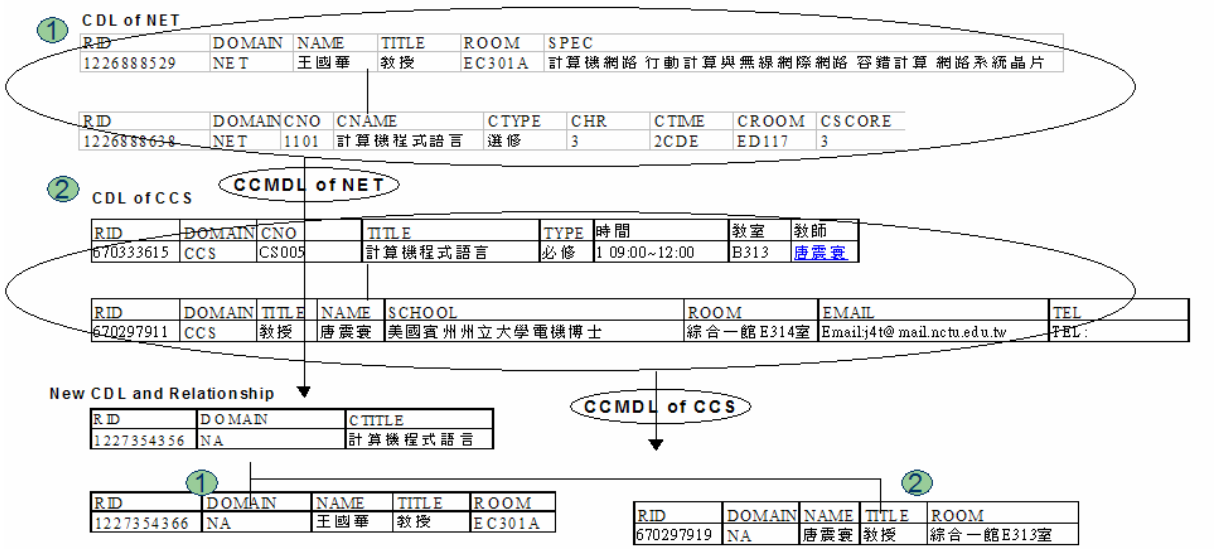


圖 30、CCMDL 資料轉換(GENRAL 模式)示意圖

(2) 使用"AGGREGATION"模式時，不同來源相同課程名稱的課程資料將不被視為單一課程資料。也就是說，鍵值作用會被 domain 所限定。

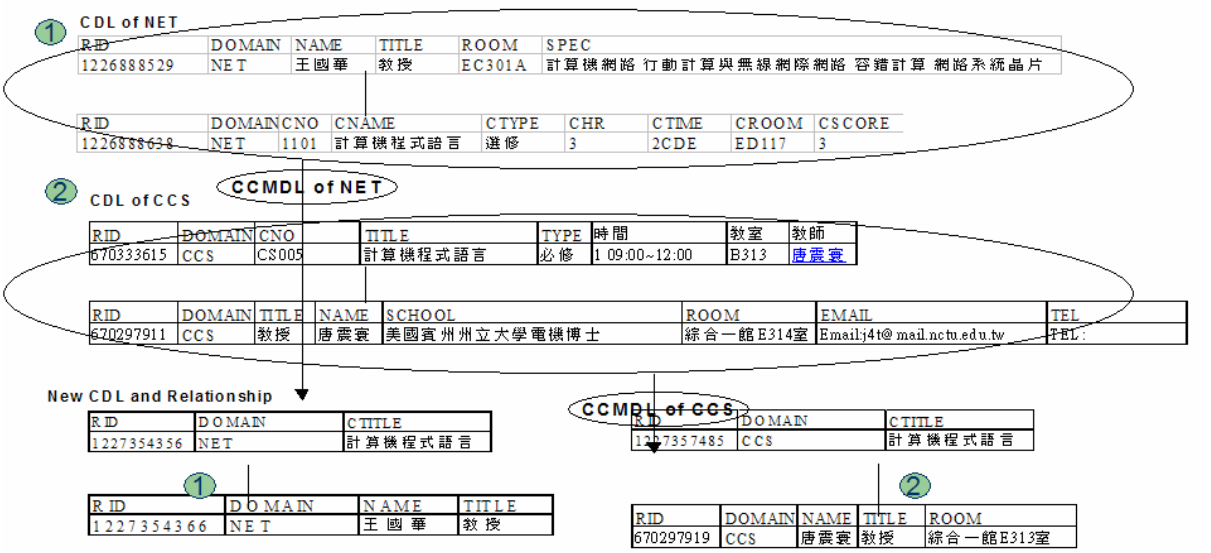


圖 31、CCMDL 資料轉換(AGGREGATION 模式)示意圖

接著，以圖書資訊為例，一般而言，ISBN 可以用來做為圖書資料的 KEY，以下將說明如何以不同的資料轉換的模式，彙整來自不同網頁的圖書資料。圖 32 與圖 33 的書籍資料分別萃取自兩家不同的出版商，其 Domain 不同代表是不同的資料來源。

RID	DOMAIN	ISBN	TITLE	PRICE
12200001	儒林	0-13-062221-4	C# HOW TO PROGRAM	1200

圖 32、儒林書局網站萃取資料

RID	DOMAIN	ISBN	TITLE	PRICE
12202220	天瓏	0-13-062221-4	C# HOW TO PROGRAM	1300

圖 33、天瓏書局網站萃取資料

(1) 使用 GENRAL 模式時，系統不會考慮圖書資料的資料來源。一組 ISBN 僅會有一筆圖書資料。

RID	DOMAIN	ISBN	TITLE	PRICE
12300546	儒林	0-13-062221-4	C# HOW TO PROGRAM	1200

圖 34、使用圖 32 以及圖 33 的資料，在 GENERAL 模式下進行資料彙整的結果

(2) 使用 Aggregation 模式時，系統會先考慮圖書資料的資料來源。不同資料來源的資料，將會並存。相同資料來源的資料，僅會有一筆資料。

RID	DOMAIN	ISBN	TITLE	PRICE
12300546	儒林	0-13-062221-4	C# HOW TO PROGRAM	1200
12302256	天瓏	0-13-062221-4	C# HOW TO PROGRAM	1300

圖 35、使用圖 32 以及圖 33 的資料，在 AGGREGATION 模式下進行資料彙整的結果

假設有一位消費者，欲購買一部數位相機，在購買前，先從不同的銷售網站，取得產品銷售資訊。接著，將來自不同來源的資料彙整到相同的資料表中，以方便規格及售價上的比較。

(1) 使用"GENERAL"模式，並且以產品資訊 PRODUCT 及 PRODUCER 為 KEY，進行資料匯整。使用此一方式，系統會忽略 DOMAIN，將使用相同 KEY 值的資料，視為重複的資料。因此一個資料類別的任何一個 KEY 值只允許一筆資料(以下例

子使用 OVERWRITE 模式處理重複資料) 。此一例子顯示使用"GENERAL"模式不利於規格及售價上的比較。

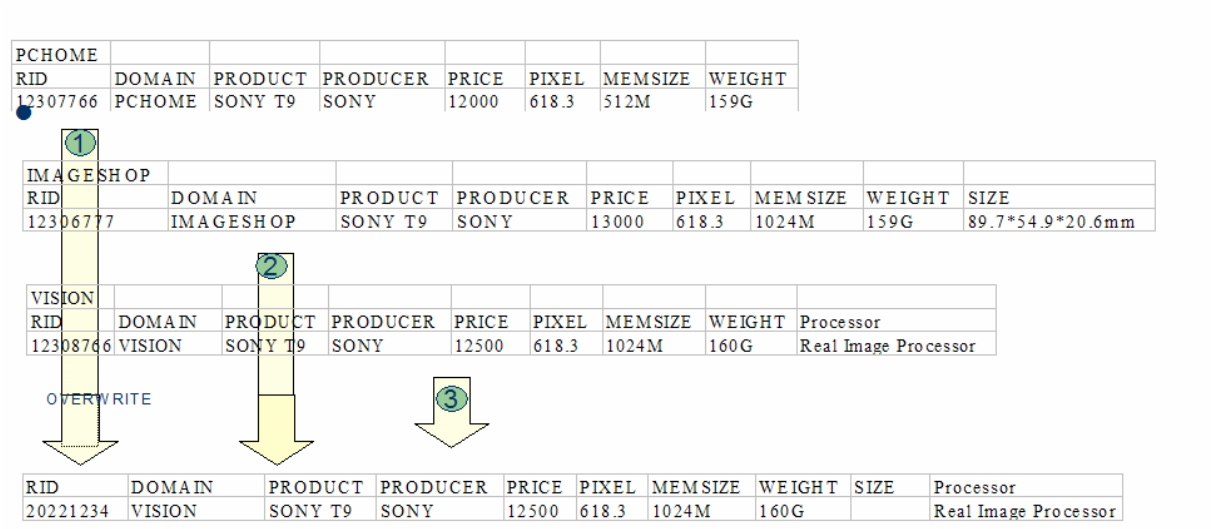


圖 36、CCMDL 資料轉換(GENRAL 模式)

(2) 使用"Aggregation"模式，並且以產品資訊 PRODUCT 及 PRODUCER 為 KEY，進行資料匯整。使用此一方式，系統會將使用相同 KEY 值，但是不同 DOMAIN 的資料，視為不同的資料。相同 KEY 值，但是相同 DOMAIN 的資料，視為重複資料的資料。(以下例子使用 OVERWRITE 模式處理重複資料)

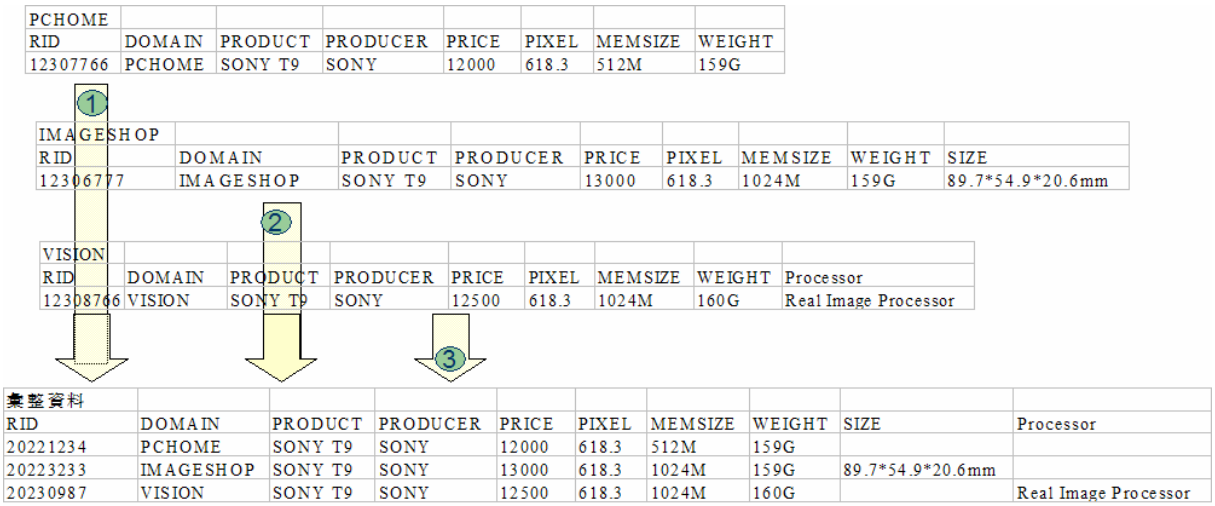


圖 37　　CCMDL 資料轉換(AGGREGATION 模式)

資料重複資料的處理

當所抓取的兩筆資料，其 KEY 值相同，則代表其中一筆資料重複。

當我們要將不同來源的資料進行彙整時，往往會遇到重複的資料，對於如何處理重複性的資料，取決於應用本身。為了要滿足各類需求，BODE 提供了以下兩種重複性的資料的處理方式：

(1) 取代資料庫現有資料 (Overwrite) - 使用" OVERWRITE "模式時，若資料已經存在，系統會以新的資料及舊的資料 ID，更新資料庫中的資料。否則，系統會產生新的資料 ID 儲存資料。此時， DOMAIN 僅有紀錄資料來源的功能。

(2) 保留資料庫現有資料 (Preempt) - 使用"PREEMPT"模式，假若資料已經存在，系

統會保留資料庫中原本存在的資料。否則，系統會產生新的資料 ID 儲存資料。此時， DOMAIN 僅有紀錄資料來源的功能。

例如，我們在網站資料更動前後，分別萃取教師的基本資料：

(1) 假使我們使用"OVERWRITE"，若是此一教師的資料已經存在資料庫時，系統將會保留原有的資料庫中的資料 ID，但是資料庫中的其他非鍵值資料欄位將會被更新。

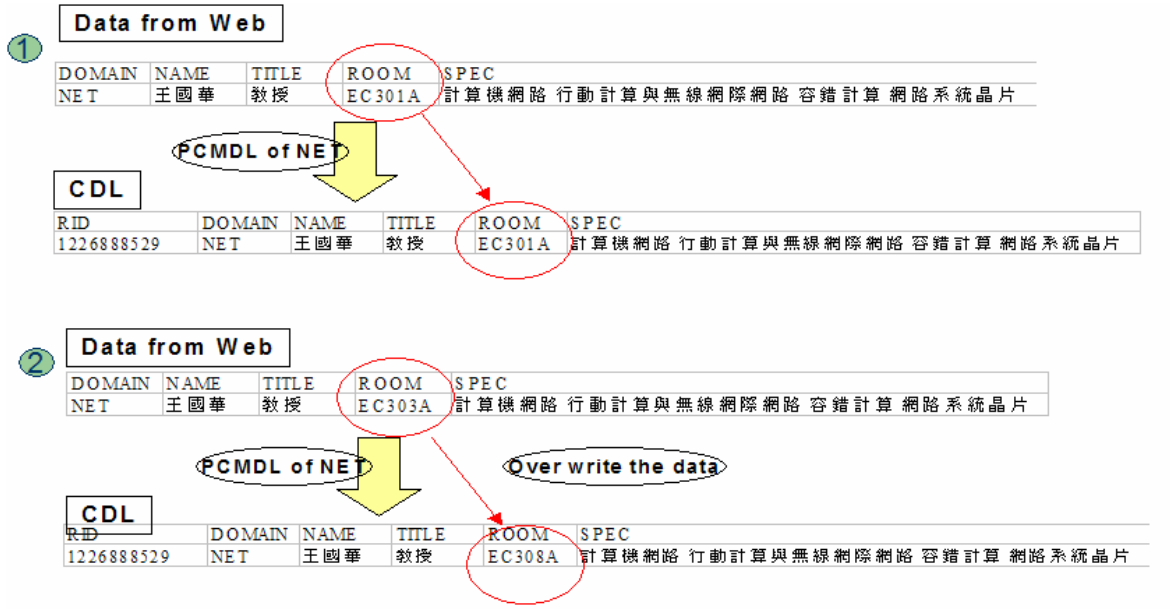


圖 38、處理重複資料的示意圖(OVERWRITE 模式)

(2) 假使我們使用"PREEMPT"，若是此一教師的的資料已經存在資料庫時，系統將會不會對資料庫作任何資料的更易的動作。

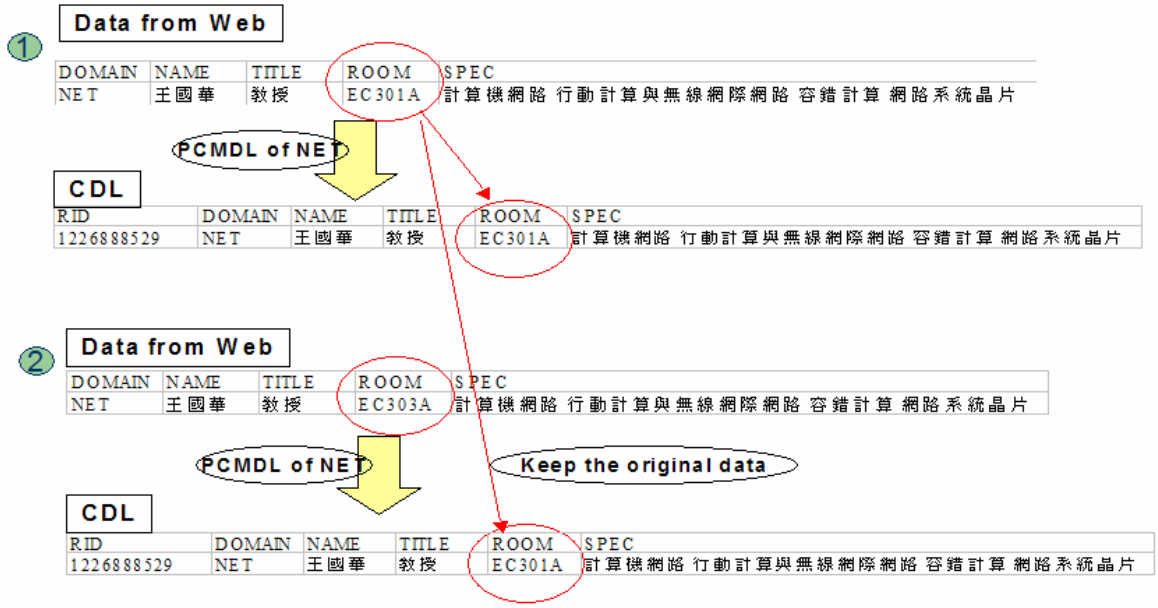


圖 39　　處理重複資料的示意圖(PREEMPT 模式)

例如，我們要彙整來自不同圖書館網頁資料，並且以"中央圖書館"的圖書資料作為最高優先權的資料，我們可以用以下方式：

(1) 使用"PREEMPT"模式，先匯入"中央圖書館"的圖書資料，接著再匯入其他圖書館的圖書資料。此時，ISBN 為 KEY，DOMAIN 僅僅作為區別資料來源的功能。 在進行轉換時，資料庫出現相同 KEY 的 資料時，系統將會保留資料庫中的原有資

料。否則，系統會產生新的資料 ID，並且將資料存入資料庫中。

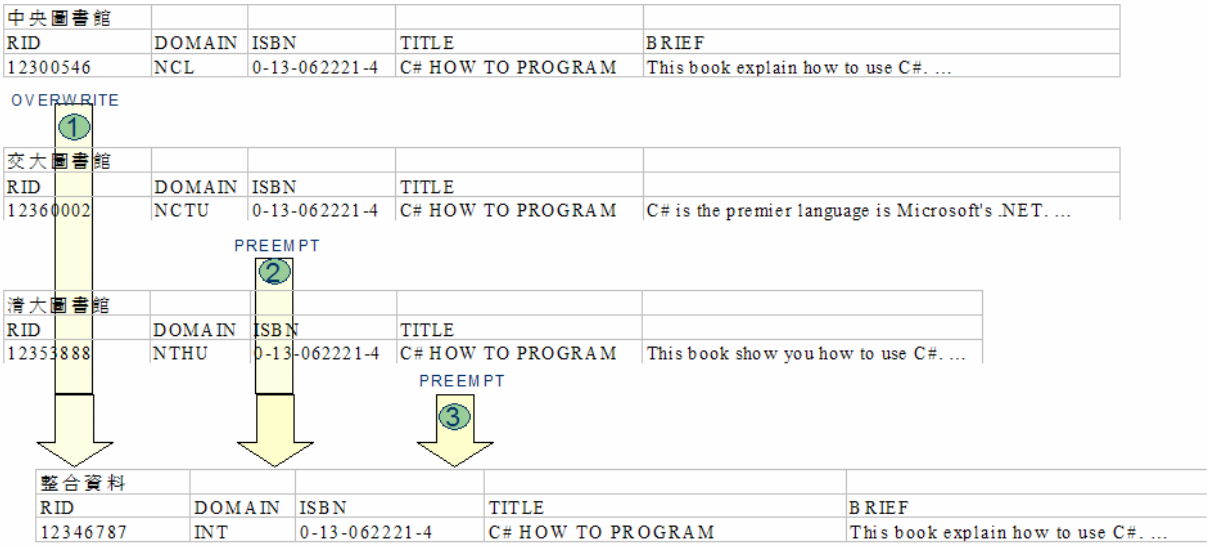


圖 40、處理重複資料的示意圖(OVERWRITE 模式)

(2) 或者是，使用"OVERWRITE"模式，先匯入其他圖書館的圖書資料，接著再"中央圖書館"的圖書資料。此時，ISBN 為 KEY，DOMAIN 僅僅作為區別資料來源的功能。在進行轉換時，資料庫出現相同 KEY 的資料時，系統先取得原有資料 ID，刪除原有的資料，將新取得的資料以舊有的資料 ID 存入資料庫中。否則，系統會取得新的資料 ID，並且將資料存入資料庫中。

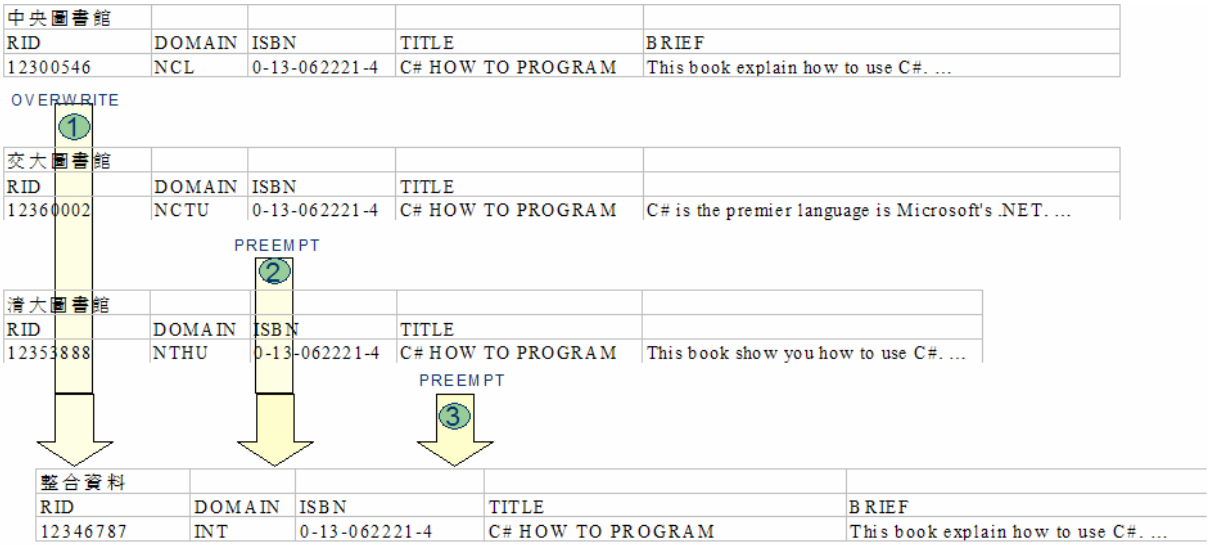
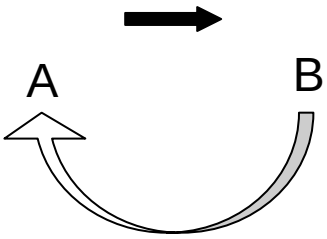


圖 41、處理重複資料的示意圖(PREEMPT 模式)

CCMDL 資料轉換語言的限制

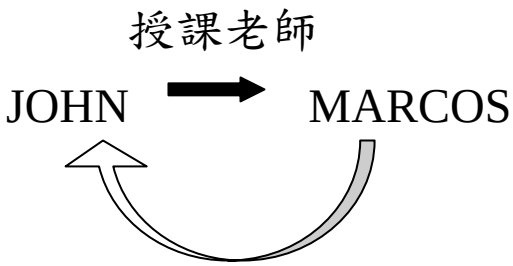
CCMDL 主要適用於具有以下特性的關聯。首先是，資料類別之間的關聯，必需具有方向性。而且，其衍生關聯必需要具有遞移性。例如圖 42，若存在資料 A 關聯資料 B，則存在反向關聯(資料 B 關聯資料 A)，而且具有意義，但是兩者關聯的意義並不一定相同。



反向關聯

圖 42、資料 A 與資料 B 的關聯與反向關聯

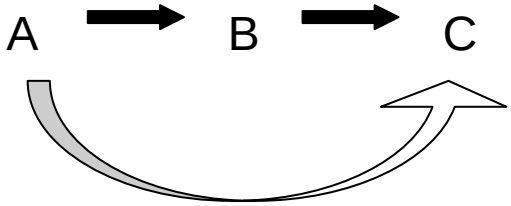
舉例來說圖 43 存在 JOHN 是 MARCOS 的老師，其反向關聯是 MARCOS 是 JOHN 的學生。師生關係具有方向性。



學生

圖 43、JOHN 與 MARCO 的師生關聯

圖 44 是一個衍生關聯的示意圖。若存在資料 A 關聯資料 B，且資料 B 關聯資料 C，其衍生關聯(資料 A 關聯資料 C)具有意義，則稱此一特性為關聯的遞移性。



衍生關聯

圖 44、資料 A、B、C 的衍生關聯

在圖 45 中存在 Johnson 是 John 的兒子，且 Johnson L.是 Johnson 的兒子，則 Johnson L.必定是 John 的子孫。由父子關係，可以衍生出來親族關係。

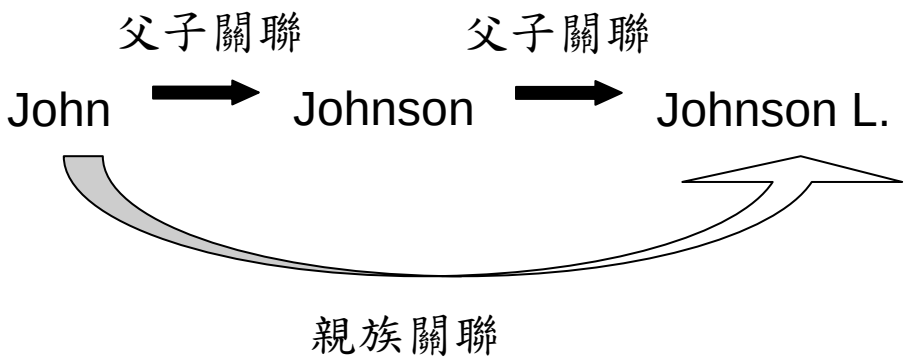


圖 45、John、Johnson 以及 Johnson L. 的衍生親族關聯

然而，衍生關聯並不一定具備實質的意義。例如圖 46 中 Johnson 是 John 的兒子，且 Tom 是 Johnson 的同事，但是，經由父子關係及同事關係，並不會產生具有實質意義的關連。

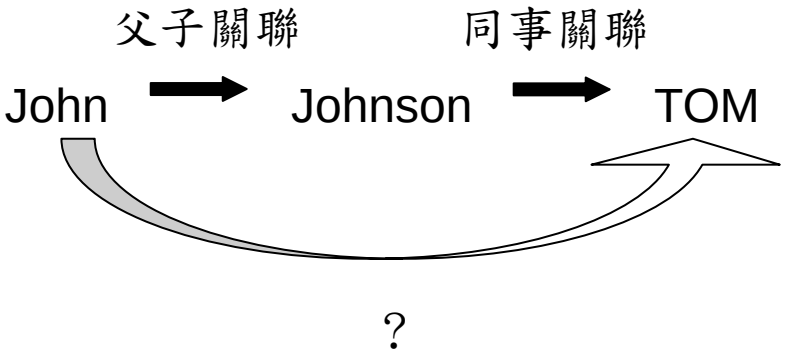


圖 46、John、Johnson 以及 TOM 的衍生關聯

關連是否具有遞移性，無法從關聯的組合方式直接推論，必需審視衍生關聯是否具有意義。並非所有的衍生關聯都具有遞移性。假設，衍生關聯不具有遞移性，就有可能會產生錯誤或是不合理的資料。

本研究對於上述的衍生關聯是否具有遞移性，進行研究，並提出數種具有遞移性關聯的特性：

(1) Composition

例如圖 47 中，汽車是由各種模組裝置組合而成；也可以將汽車視為由許多更小的零件所構成。

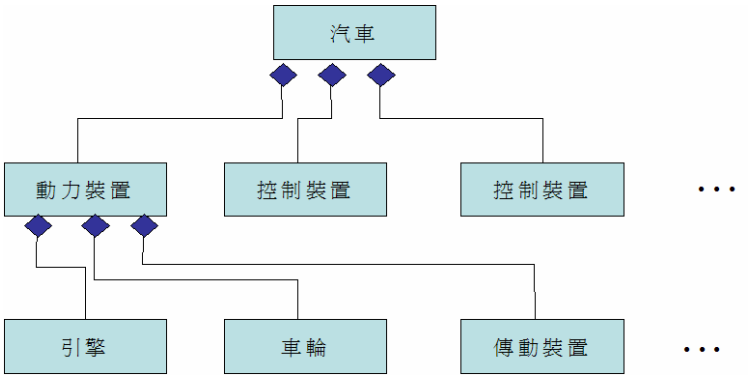


圖 47、汽車與零件的 Composition 衍生關聯

(2) Inheritance

例如圖 48 中，汽車繼承車的特性且混合動力油汽車繼承汽車的特性，混合動力汽車必定繼承車的特性。

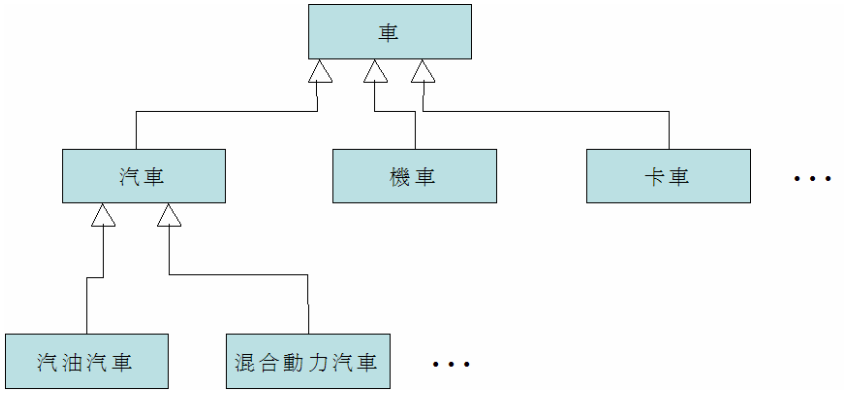


圖 48、車子種類間的 Inheritance 衍生關聯

(3) Generalization/Specification

例如圖 49 中，某一銷售網站網頁資料分類方式，是以產品種類由大而小進行切割。大的產品種類中包含許多較小的產品種類。例如電子產品包含電視類的產品、電話類的產品等。而電視類產品又包含各種不同廠牌的產品。

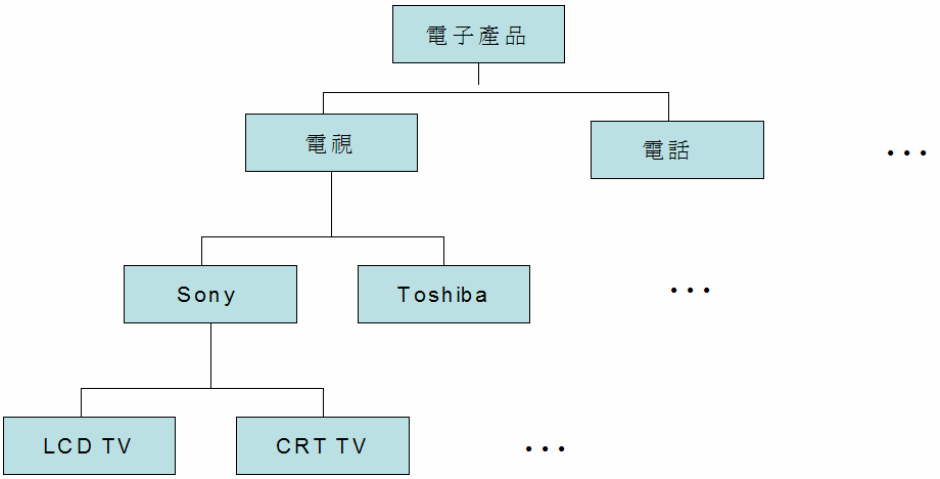


圖 49、產品種類間的 Generalization/Specification 衍生關聯

網站資料匯整

萃取不同網站的資料，是 Web 資料管理的一件非常重要工作。例如，消費者想要在購買商品之前，先從不同的銷售網站，取得產品銷售資訊。進行規格及售價上的比較。為了方便進行比較的工作，往往需要整合這些萃取資料到同一組資料表。在 BODE 系統中，可使用兩種方式來完成：

(1) 在萃取網站網頁資料的時候，將來自不同網站的萃取資料，直接存入同一組資料類別。並且以不同的 DOMAIN 來區分來自不同網站的網頁內容。此一方式較適用於結構相似的網站。

例如，圖 50、圖 51、圖 52 三個不同網站，因為三個不同網站的結構及資料內容大致相符；如果想要整合來自 A、B、C 三個不同網站的產品資訊，可以採用第一種方式。

首先是聯合三個不同網站的資料項目，定義共用的資料類別，圖 53 為其示意圖。接著是針對不同的網頁，撰寫個別的 BODED script 及描述個別的資料對應方式，並且指定個別

的 DOMAIN，區別不同的資料來源。

最後是執行個別的 BODED script，萃取不同網站的產品資訊儲存到資料庫中。圖 54 為使用此方式整合不同網站資料的示意圖。

```
…
<TABLE>
   <TR>
      <TD>PRODUCT</TD>
      <TD>EENCODE</TD>
      <TD>PRICE</TD>
      <TD>COUNTRY</TD>
   </TR>
   <TR>
      <TD>NOTEBOOK001</TD>
      <TD>710626981315</TD>
      <TD>NT40000</TD>
      <TD>TAIWAN</TD>
   </TR>
   …
</TABLE>
…
```

圖 50、Web_A.html

```
…
<TABLE>
   <TR>
      <TD>PRODUCT</TD>
      <TD>EENCODE</TD>
      <TD>PRICE</TD>
      <TD>WEIGHT</TD>
   </TR>
   <TR>
      <TD>NOTEBOOK001</TD>
      <TD>710626981315</TD>
      <TD>NT40000</TD>
      <TD>3000g</TD>
   </TR>
   …
</TABLE>
…
```

圖 51、Web_B.html

```
…
<TABLE>
   <TR>
      <TD>PRODUCT</TD>
      <TD>EENCODE</TD>
      <TD>PRICE</TD>
      <TD>SIZE</TD>
   </TR>
   <TR>
      <TD>NOTEBOOK001</TD>
      <TD>710626981315</TD>
      <TD>NT40000</TD>
      <TD>15</TD>
   </TR>
```

```
    ...
</TABLE>
...
```

圖 52、Web_C.html

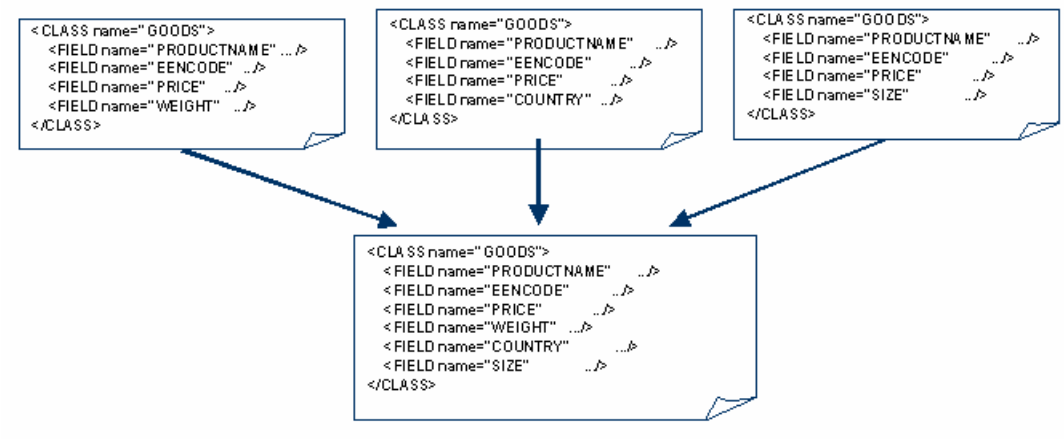


圖 53、聯合來自不同的網站的資料欄位型成資料類別

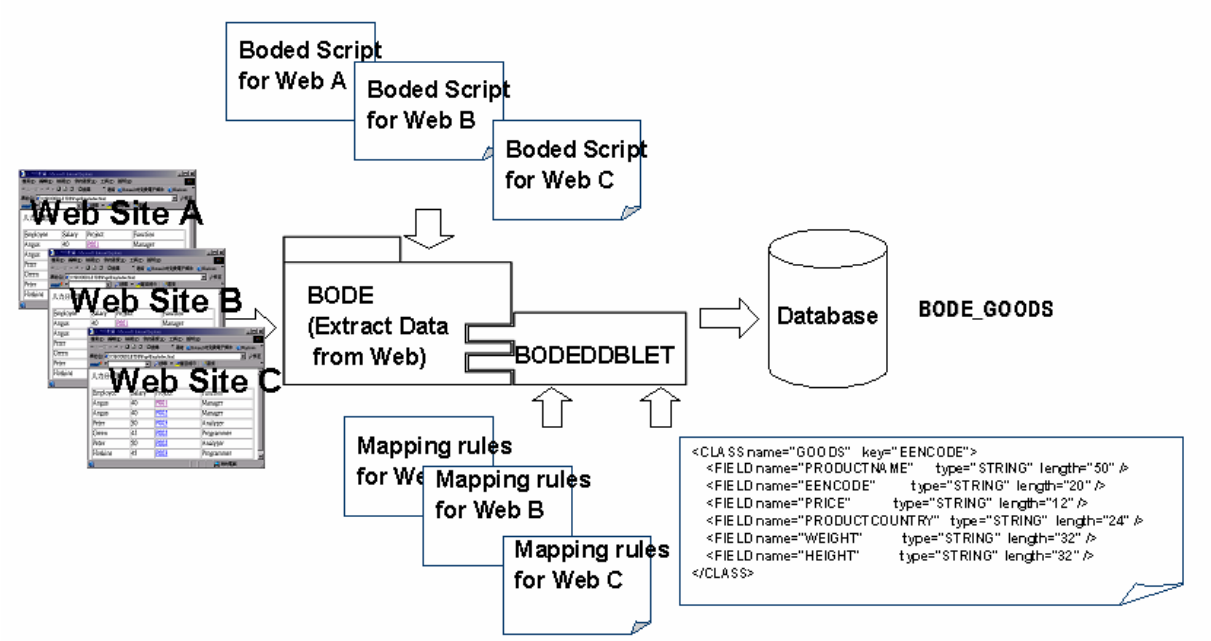


圖 54、整合不同網站資料的示意圖

(2) 先將網站網頁資料萃取出來，並且將來自不同網站的萃取資料，存入個別的資料表中。接者再將萃取的網站資料匯整到共同資料表。也就是說，我們可以用 CCMDL 來描述資料整合作業，然後交一般化的資料整合系統來完成。

在撰寫 CCMDL 之前，我們要進行網頁資料的結構分析。找出網頁資料的結構特性，判斷是否適合用 BODE 資料轉換語言進行資料匯整。

假設，有一 BODE 的使用者，要進行筆記型電腦的比價，並且以名人購物網(MREN)，順發以及燦坤等銷售網站，作為商品資訊的來源。首先，要先針對網頁資料的結構進行分析。

以上述網站為例，名人購物網(MREN)的階層結構如圖 55。順發購物網(SUNFAR)的階

層結構如圖 56。燦坤購物網(TKEC)的階層結構如圖 57。

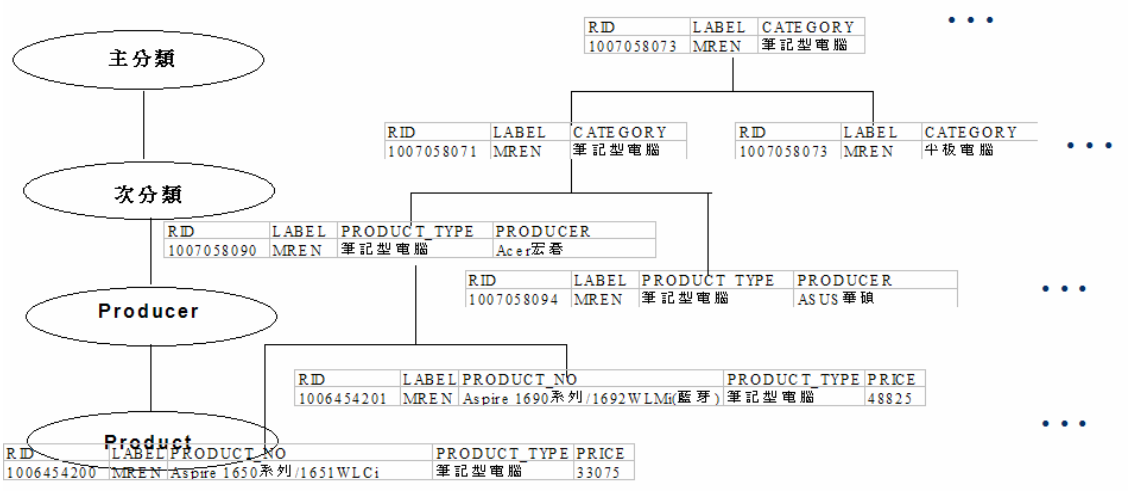


圖 55、名人購物網(MREN)的階層結構

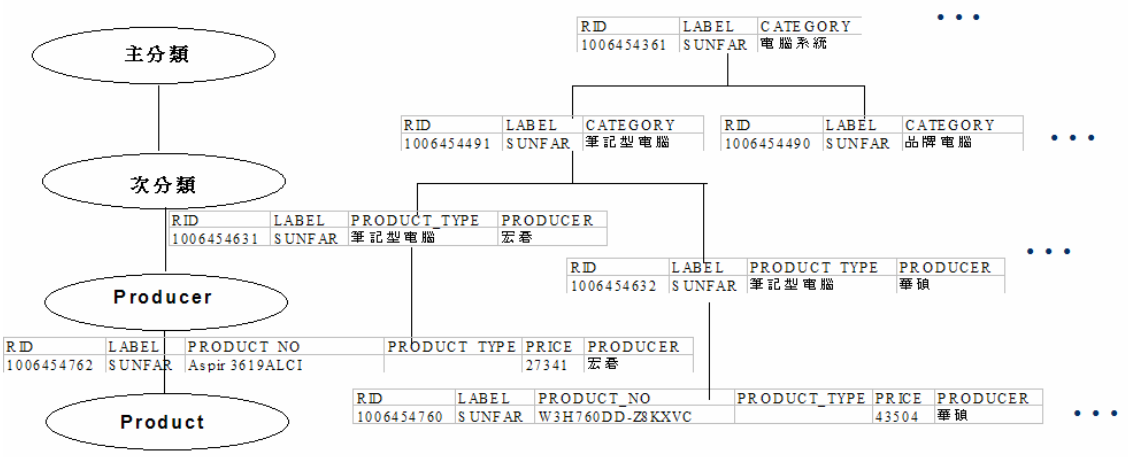


圖 56、順發購物網(SUNFAR)的階層結構

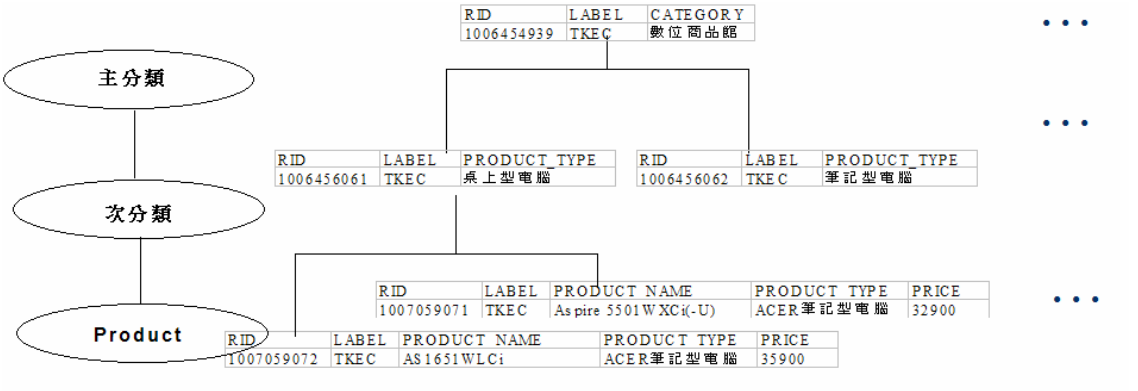


圖 57、燦坤購物網(TKEC)的階層結構

根據以上的網站的結構分析，可以發現以上三個網站的網頁結構都具備 Specification 的特性，資料的衍生關聯具有遞移性，也就是名人購物網(MREN)，順發(SUNFAR)以及燦坤(TKEC)等銷售網站的網頁資料，適合用 BODE 資料轉換語言進行資料重組。

依據網站的結構以及資料內容，分別定義以上三個網站使用的資料類別。名人購物網(MREN)的資料類別定義如圖 58。順發(SUNFQAR) 銷售網站的資料類別定義如圖 59。燦坤(TKEC)銷售網站的資料類別定義如圖 60：

```
<CLASS name="MREN_PRODUCT_MAINCAT"   key="CATEGORY" >
  <FIELD name="CATEGORY" type="STRING" length="32" />
</CLASS>
<CLASS name="MREN_PRODUCT_2NDCAT"    key="CATEGORY" >
  <FIELD name="CATEGORY" type="STRING" length="32" />
</CLASS>
<CLASS name="MREN_PRODUCT_3THCAT"    key="PRODUCT_TYPE, PRODUCER">
  <FIELD name="PRODUCT_TYPE" type="STRING" length="32" />
  <FIELD name="PRODUCER"     type="STRING" length="64" />
</CLASS>
<CLASS name="MREN_PRODUCT"   key ="PRODUCT_SN" >
  <FIELD name="PRODUCT_SN"   type="STRING"  length="64" />
  <FIELD name="PRODUCT_TYPE" type="STRING"  length="32" />
  <FIELD name="PRICE"        type="NUM,BER" length="7" />
</CLASS>
```

圖 58、名人(MREN)購物網的資料類別定義

```
<CLASS name="SUNFAR_PRODUCT_MAINCAT" key="TITLE">
  <FIELD name="CATEGORY" type="STRING" length="32" />
</CLASS>
<CLASS name="SUNFAR_PRODUCT_2NDCAT"   key="TITLE">
  <FIELD name="CATEGORY" type="STRING" length="32" />
</CLASS>
<CLASS name="SUNFAR_PRODUCT_3THCAT" key="TITLE">
  <FIELD name="PRODUCT_TYPE" type="STRING" length="32" />
  <FIELD name="PRODUCER"     type="STRING" length="64" />
</CLASS>
<CLASS name="SUNFAR_PRODUCT" key="PRODUCT_NO">
  <FIELD name="PRODUCT_TYPE"  type="STRING" length="32" />
  <FIELD name="PRODUCT_NO"    type="STRING"  length="64" />
  <FIELD name="PRICE"         type="NUMBER"  length="7" />
</CLASS>
```

圖 59、順發(SUNFQAR) 銷售網站的資料類別定義

```
<CLASS name="TKEC_PRODUCT_MAINCAT" key="CATEGORY">
  <FIELD name="CATEGORY" type="STRING" length="32" />
</CLASS>
<CLASS name="TKEC_PRODUCT_SUBCAT" key="PRODUCT_TYPE">
  <FIELD name="PRODUCT_TYPE" type="STRING" length="32" />
</CLASS>
<CLASS name="TKEC_PRODUCT">
  <FIELD name="PRODUCT_NAME" type="STRING" length="64" />
  <FIELD name="PRODUCT_TYPE" type="STRING" length="32" />
  <FIELD name="PRICE"        type="NUM,BER" length="7" />
</CLASS>
```

圖 60、燦坤(TKEC) 銷售網站的資料類別定義

經由 BODE Scrip 萃取名人購物網(MREN)，順發(SUNFQAR)以及燦坤(TKEC)等三個銷售網站的資料之後，BODE 系統便會將萃取所得的資料存入資料庫中。

但是，為了要進行比價，使用者必需匯整資料。於是，定義了一個匯整資料的類別。其定義如圖 61。且其架構如圖 62。

```
<CLASS name="PRODUCT_TYPE" key="PRODUCER,TYPE">
  <FIELD name="PRODUCER" type="STRING" length="64" />
  <FIELD name="TYPE"     type="STRING" length="64" />
</CLASS>
<CLASS name="PRODUCT" key="SN">
  <FIELD name="SN"    type="STRING"  length="64" />
  <FIELD name="PRICE" type="NUM,BER" length="7" />
</CLASS>
```

圖 61、名人、順發與燦坤網站資料匯整的資料類別定義

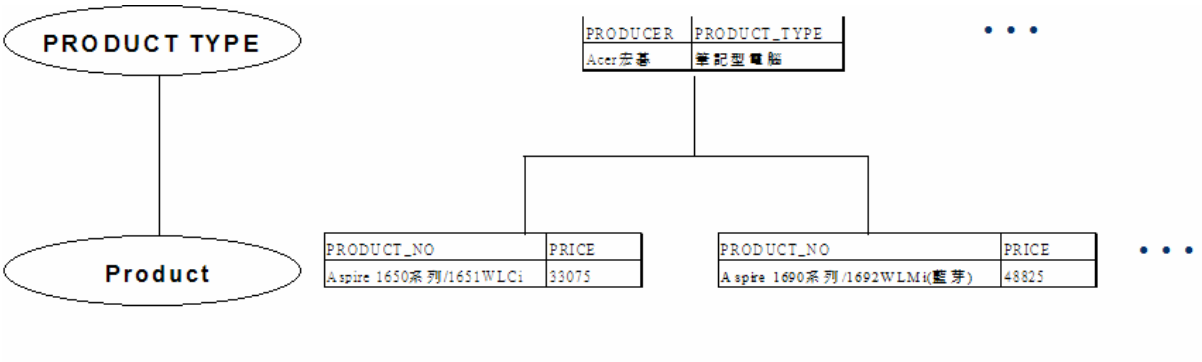


圖 62、名人、順發與燦坤網站資料匯整的資料類別架構

接著是，利用資料轉換語言，將名人購物網(MREN)，順發(SUNFAR)以及燦坤(TKEC)等銷售網站的資料，匯入匯整資料的類別。其示意圖如圖 63。

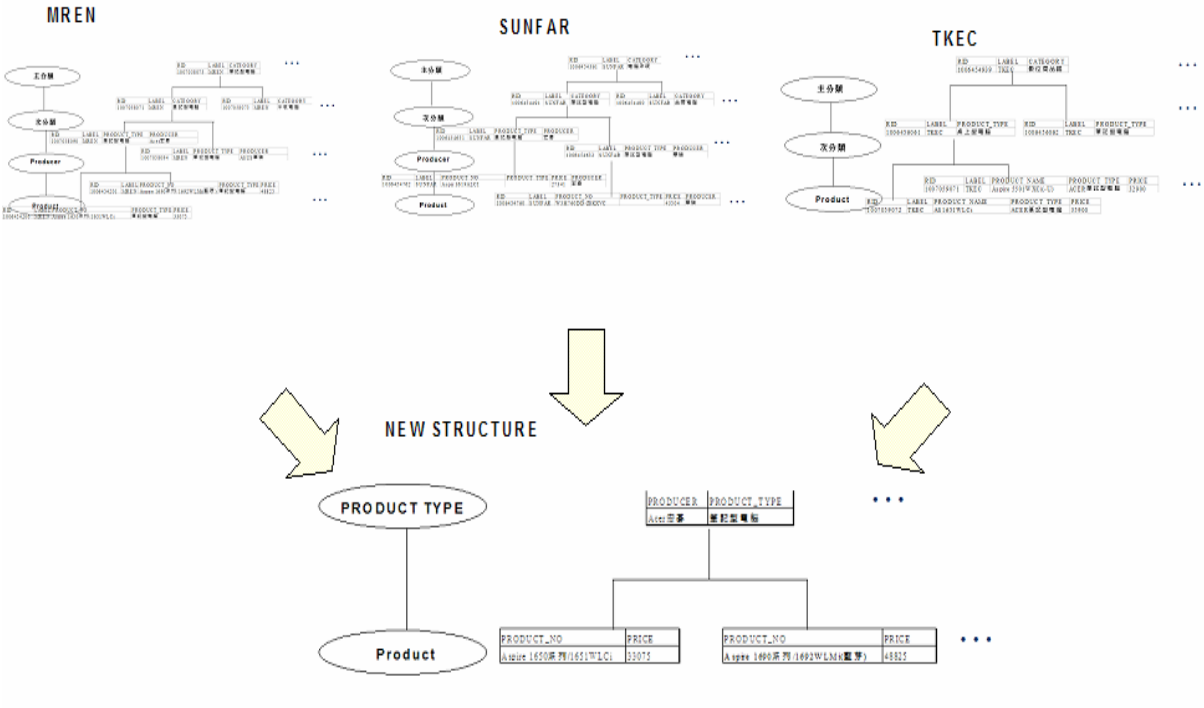


圖 63、將名人、順發與燦坤網站資料匯整到匯整資料類別示意圖

為了要讓來自不同來源的資料能夠並存，資料轉換語言將以 AGGREGATION 模式轉換資料。名人(MREN)購物網的資料轉換程式如圖 64。順發(SUNFAR)銷售網站的資料轉換程式如圖 65。燦坤(TKEC) 銷售網站的資料轉換程式如圖 66。

```
<TRANSLATIONS>
  <TRANSLATION mode="AGGREGATION">
    <SOURCE>
      <CLASS>MREN_PRODUCT_MAINCAT</CLASS>
      <CLASS>MREN_PRODUCT_2NDCAT</CLASS>
      <CLASS>MREN_PRODUCT_3THCAT</CLASS>
      <CLASS>MREN_PRODUCT_PRODUCT</CLASS>
      <RELATION>MREN_PRODUCT_MAINCAT,MREN_PRODUCT_2NDCAT</RELATION>
      <RELATION> MREN_PRODUCT_2NDCAT, MREN_PRODUCT_3THCAT </RELATION>
      <RELATION> MREN_PRODUCT_3THCAT, MREN_PRODUCT_PRODUCT </RELATION>
    </SOURCE>
    <CLASS name="PRODUCT_TYPE" mode="PREEMPT">
       <FIELD field="PRODUCER" value="MREN_PRODUCT_3THCAT.PRODUCER" />
       <FIELD field="TYPE" value="MREN_PRODUCT.PRODUCT_TYPE" />
    </CLASS>
    <CLASS name="PRODUCT" mode="PREEMPT">
        <FIELD field="SN" value="MREN_PRODUCT.PRODUCT_SN" />
        <FIELD field="PRICE" value="MREN_PRODUCT.PRICE" />
    </CLASS>
  </TRANSLATION>
</TRANSLATIONS>
```

圖 64、名人(MREN)購物網的資料轉換程式

```
<TRANSLATIONS>
  <TRANSLATION mode="AGGREGATION">
    <SOURCE>
      <CLASS>SUNFAR_PRODUCT_MAINCAT</CLASS>
      <CLASS>SUNFAR_PRODUCT_2NDCAT</CLASS>
      <CLASS>SUNFAR_PRODUCT_3THCAT</CLASS>
      <CLASS>SUNFAR_PRODUCT_PRODUCT</CLASS>
      <RELATIONSHIP>SUNFAR_PRODUCT_MAINCAT,SUNFAR_PRODUCT_2NDCAT</RELATIONSHIP>
      <RELATIONSHIP>SUNFAR_PRODUCT_2NDCAT,SUNFAR_PRODUCT_3THCAT</RELATIONSHIP>
      <RELATIONSHIP>SUNFAR_PRODUCT_3THCAT,SUNFAR_PRODUCT_PRODUCT</RELATIONSHIP>
    </SOURCE>
    <CLASS name=" PRODUCT_TYPE" mode="PREEMPT">
      <FIELD field=" PRODUCER" value="SUNFAR_PRODUCT_3THCAT.PRODUCER" />
      <FIELD field=" TYPE" value="SUNFAR_PRODUCT.PRODUCT_TYPE" />
    </CLASS>
    <CLASS name=" PRODUCT" mode="PREEMPT">
      <FIELD field="SN" value="SUNFAR_PRODUCT.PRODUCT_NO" />
      <FIELD field=" PRICE" value="SUNFAR_PRODUCT.PRICE" />
    </CLASS>
```

```
  </TRANSLATION>
</TRANSLATIONS>
```

圖 65、順發(SUNFAR)銷售網站的資料轉換程式

```
<TRANSLATIONS>
  <TRANSLATION mode="AGGREGATION">
    <SOURCE>
       <CLASS>TKEC_PRODUCT_MAINCAT</CLASS>
       <CLASS>TKEC_PRODUCT_SUBCAT</CLASS>
        <CLASS>TKEC_PRODUCT</CLASS>
        <RELATIONSHIP>TKEC_PRODUCT_MAINCAT, TKEC_PRODUCT_SUBCAT</RELATIONSHIP>
        <RELATIONSHIP>TKEC_PRODUCT_SUBCAT, TKEC_PRODUCT</RELATIONSHIP>
    </SOURCE>
    <CLASS name="PRODUCT_TYPE" mode="PREEMPT">
      <FIELD field=" PRODUCER" value="alpha(TKEC_PRODUCT.PRODUCT_TYPE)" />
      <FIELD field=" TYPE"   value="TKEC_PRODUCT_SUBCAT.PRODUCT_TYPE" />
    </CLASS>
    <CLASS name="PRODUCT" mode="PREEMPT">
      <FIELD field="SN"      value="TKEC_PRODUCT.PRODUCT_NAME" />
      <FIELD field=" PRICE"  value="TKEC_PRODUCT.PRICE" />
    </CLASS>
  </TRANSLATION>
</TRANSLATIONS>
```

圖 66、燦坤(TKEC)銷售網站的資料轉換程式

資料匯整及重組時，必須先將來源資料展開，然後依據轉換描述將資料拆開，並建立關聯，下圖是以名人購物網(MREN)的一筆筆記型電腦為例，說明其處理過程。

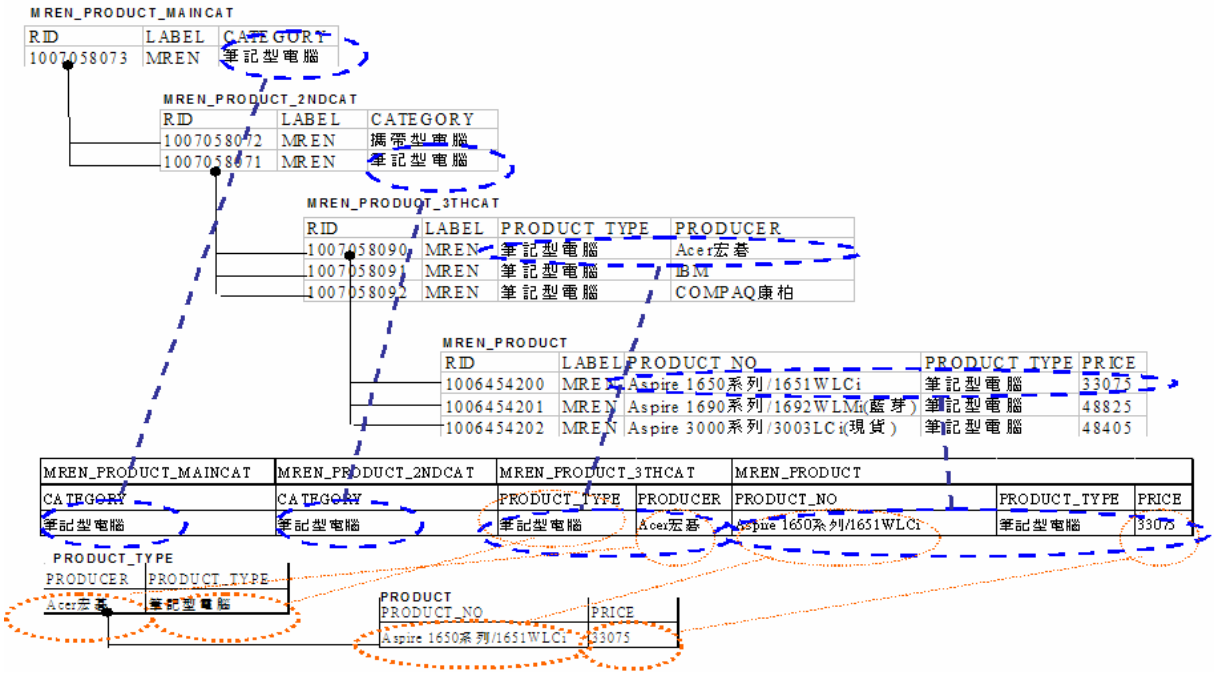


圖 67、資料匯整及重組程序示意圖

使用 AGGRAGATION 模式，將名人購物網(MREN)，順發(SUNFQAR)以及燦坤(TKEC)等三個銷售網站的資料轉換及重組後，來自三個網站的相同鍵值的資料會個別存在資料庫中。最後是，由人工或其他的系統進行比價的作業。

六、實作

由於在 BODED 萃取程式當中，在標籤結構中，有特定的標籤會包含其他標籤。而這些被包含的標籤若是變數 VAR，則是要被萃取的資料。同一層的變數通常具有一對一的對應關係。因此包含這些 VAR 變數的標籤，可對應到類別。而被包含的 VAR 變數中的內容由於具有一對一對應關係，因此可以對應到類別中的欄位。所以類別的定義可以由 BODED 萃取程式自動的產生。由於圖 6 中的 BODED 萃取程式具有兩個主要的結構一個是 PAGE，另外一個是 FOREACH。因此當使用圖 6 的 BODED 萃取程式自動產生類別時，會產生兩個類別。一個是由 PAGE 結構所產生的類別，名稱為 MainCatPage。另一個是由 FOREACH 結構產生，名為 MainCat 的類別。由於其內沒有任何變數，因此該變數並沒有任何欄位。不過在 BODE 資料模式處理系統執行時，會產生 MainCatPage 類別到 MainCat 類別的一對多關係，並儲存於關聯資料表當中。

```
<CLASS name="MainCatPage">
   <FIELD name="MainCat" type="class" classname="MainCat" /> ??
</CLASS>

<CLASS name="MainCat" key="MainCatName" >
   <FIELD name="MainCatName" type="string" length="50" />
</CLASS>
```

圖 68、由圖 6 中的 BODED 萃取程式自動產生的類別定義

圖 69 顯示由圖 6 中的 BODED 萃取程式自動產生的對應規則。對應規則自動產生的原理與自動產生類別一樣。由於 BODED 萃取程式中的每一個包含變數的標籤，會被自動產生成一個類別，因此這些結構也會自動的對應到該類別中。這些標籤中所包含的變數標籤，也就是 VAR，由於在自動產生類別時被自動對應到類別中的欄位。自動產生的對應規則會將這些變數對應到類別中的欄位。

```
. . .
<PAGE name="MainCatPage" class="MainCatPage" >
   <FOREACH name="MainCat" class="MainCat"
            parentrelation="MainCatPage">
     <VAR name="MainCatName" field="Name" />
   </FOREACH>
</PAGE>
. . .
```

圖 69、由圖 6 中的 BODED 萃取程式自動產生的對應規則

圖 70 是一個萃取整個論文資料庫網站的 BODED 萃取程式。而圖 71 則是由圖 70 的 BODED 萃取程式中自動產生的類別定義。由於程式無法事先得知所要萃取的資料長度為何？因此在圖 71 的 CLASS 中的 FIELD，其 length 屬性值為 variable，亦即可變長度。

```
<BODED name="CSArchive">
  <INIT page="MainCat"
        url="http://bode.adm.mcu.edu.tw/index.html" />
  <PAGE name="MainCat">
    . . .  <!-- 詳見圖 6 -->
  </PAGE>
  <PAGE name="SubCat">
    <FOREACH name="ForeachSubCat" xpath="//TD/A">
      <VAR name="SubCatName" xpath="." />
      <EVENT name="LinkToPaperList" xpath="." page="PaperList"
             type="OnClick" />
    </FOREACH>
  </PAGE>
  <PAGE name="PaperList">
    <VAR name="PaperItem" xpath="//TR" />
    <FOREACH name="ForeachPaperItem" from="PaperItem">
      <VAR name="Title" xpath="./TD[0]" />
      <VAR name="Authors" xpath="./TD[1]" />
      <VAR name="Publisher" xpath="./TD[2]" />
      <EVENT name="LinkToPaper" xpath="./TD/A" page="Paper"
             type="OnClick" />
    </FOREACH>
  </PAGE>
  <PAGE name="Paper">
    . . .  <!-- 詳見圖 23-->
  </PAGE>
  <PAGE name="Author">
    <VAR name="Name" xpath="//P[0]" />
    <VAR name="Title" xpath="//TR[0]/TD[1]" />
    <VAR name="Email" xpath="//TR[1]/TD[1]" />
    <VAR name="Tel" xpath="//TR[2]/TD[1]" />
    <FOREACH name="Publishs" xpath="//TR[3]/TD[1]//TR">
      <VAR name="Publish" xpath="./TD[0]/A[0]/text()" />
    </FOREACH>
  </PAGE>
</BODED>
```

圖 70、可萃取圖 1 到圖 5 的整個論文資料庫網站的 BODED 萃取程式

```
<CLASS name="MainCatPage">
</CLASS>
```

```
<CLASS name="MainCat">
  <FIELD name="MainCatName" type="string" length="variable" />
</CLASS>

<CLASS name="SubCatPage">
</CLASS>

<CLASS name="SubCat">
  <FIELD name="SubCatName" type="string" length="variable" />
</CLASS>

<CLASS name="PaperListPage">
</CLASS>

<CLASS name="PaperItem">
  <FIELD name="Title" type="string" length="variable" />
  <FIELD name="Authors" type="string" length="variable" />
  <FIELD name="Publisher" type="string" length="variable" />
</CLASS>

<CLASS name="Paper">
  <FIELD name="Title" type="string" length="variable" />
  <FIELD name="Abstract" type="string" length="variable" />
  <FIELD name="Publisher" type=" string" length="variable" />
</CLASS>

<CLASS name="Authors">
  <FIELD name="AuthorName" type="string" length="variable" />
</CLASS>

<CLASS name="Author">
  <FIELD name="Name" type="string" length="variable" />
  <FIELD name="Title" type="string" length="variable" />
  <FIELD name="Email" type="string" length="variable" />
  <FIELD name="Tel" type="string" length="variable" />
</CLASS>

<CLASS name="Publishs">
  <FIELD name="Publish" type="string" length="variable" />
</CLASS>
```

圖 71、由圖 70 的 BODED 萃取程式自動產生的類別定義

```
<MAP script="CSArchive">
  <PAGE name="MainCatPage" class="MainCatPage" >
    . . .  <!-- 詳見圖 69 -->
  </PAGE>

  <PAGE name="SubCatPage" class="SubCatPage"
        parentrelation="MainCat" >
    <FOREACH name="SubCat" class="SubCat"
             parentrelation="SubCatPage">
      <VAR name="SubCatName" field="Name" />
    </FOREACH>
  </PAGE>

  <PAGE name="PaperListPage" class="PaperListPage"
        parentrelation="SubCat" >
    <VAR name="Papers" xpath="//TR" />  //??
    <FOREACH name="PaperItem" class="PaperItem"
             parentrelation="PaperListPage">
      <VAR name="Title" field="Title" />
      <VAR name="Authors" field="Authors" />
      <VAR name="Publisher" field="Publisher" />
    </FOREACH>
  </PAGE>

  <PAGE name="Paper" class="Paper" parentrelation="PaperItem">
    <VAR name="Title" field="Title" />
    <VAR name="Abstract" field="Abstract" />
    <VAR name="Publisher" field="Publisher" />
    <FOREACH name="Authors" class="Authors"
             parentrelation="Paper">
      <VAR name="AuthorName" field="AuthorName" />
    </FOREACH>
  </PAGE>

  <PAGE name="Author" parentrelation="Paper"
        parentrelation="Authors">
    <VAR name="Name" field="Name" />
    <VAR name="Title" field="Title" />
    <VAR name="Email" field="Email" />
    <VAR name="Tel" field="Tel" />
    <FOREACH name="Publishs" parentrelation="Author">
      <VAR name="Publish" field="Publish" />
    </FOREACH>
```

```
  </PAGE>
</MAP>
```

圖 72、由圖 70 的 BODED 萃取程式自動產生的對應規則

最佳化

由於在實作上，經由自動產生的類別，如圖 71 中的 MainCatPage、SubCatPage 以及 PaperListPage 等並不具有任何的欄位，因此其存在上的意義只是為了儲存網頁與其中資料的關聯，當這個關聯不重要時，反而會造成層次過多。因此我們可以在對應規則上，使用 noclass 屬性來避免 BODE 資料模式處理程式產生真正的物件。圖 73 即是一個使用 noclass 屬性來避免 BODE 資料模式處理程式產生 MainCatPage 類別的資料物件。在使用時只要將 noclass 屬性設為 true。BODE 資料模式處理程式僅記錄這一個物件與前一個網頁中的物件的關聯。並將這個關係繼承到 FOREACH 元素所對應的 MainCat 類別當中。

```
<PAGE name="MainCatPage" class="MainCatPage" noclass="true">
  <FOREACH name="ForeachMainCat" class="MainCat"
           parentrelation="MainCatPage">
    <VAR name="MainCatName" field="Name" />
  </FOREACH>
</PAGE>
```

圖 73、利用 noclass 來避免產生 MainCatPage 類別

此外，在圖 72 中，對於一個作者，會自動產生兩個物件，一個是 Authors，另外一個是 Author，如圖 71 所顯示。這兩個物件都是指同一個作者。第一個物件的 Authors 內只包含作者的名稱。第二個物件是 Author，則包含作者的完整資訊。而 Author 的 parent 是 Authors。由於具有 parent-child 關係的兩個作者物件是指同一個作者，這兩個作者物件的屬性之間是屬於一對一的關係。因此可以將這兩個類別合併成一個類別。在圖 74 中，將名為 Author 的 PAGE 利用 upbinding 屬性合併到 Author 類別當中。作法是將 upbind 屬性值設定成父層的類別，也就是 Author。如此不但可降低產生的類別層次，也可以確保這兩組資料的一對一的關係。

```
<PAGE name="Paper" class="Paper" parentrelation="PaperItem">
   <VAR name="Title" field="Title" />
   <VAR name="Abstract" field="Abstract" />
   <VAR name="Publisher" field="Publisher" />
   <FOREACH name="Author" class="Author"
            parentrelation="Paper">
      <VAR name="AuthorName" field="Name" />
   </FOREACH>
</PAGE>

<PAGE name="Author" parentrelation="Paper" upbinding="Author">
   <VAR name="Title" field="Title" />
   <VAR name="Email" field="Email" />
   <VAR name="Tel" field="Tel" />
   <FOREACH name="Publishs" parentrelation="Author">
      <VAR name="Publish" field="Publish" />
   </FOREACH>
</PAGE>
```

圖 74、利用 upbinding 將兩個物件合併

七、成果與討論

在 BODE 資料模式的研究中，我們設計了一個 BODE 資料模式，並且定義了 BODE 物件定義語言 Class Definition Language，簡稱 CDL。並描述 BODE 資料模式中的資料定義如何轉換為關聯式資料庫的資料表。此外，在 BODE 資料模式中，利用關聯資料表 (Relation Table)，建立物件之間的各種關聯。因此 BODE 資料模式除了具有簡單的資料定義格式，且同時可以更彈性的方式描述各種物件之間的關係。我們也實作 BODE 資料模式處理系統。

在 BODE 物件定義語言上，利用 CLASS 與 FIELD 標籤建立資料物件的類別。由於 BODED script 中，即包含許多萃取資料的描述。因此，我們設計一個可以根據 BODED script 中有關描述要萃取哪些資料的描述，將萃取的資料欄位轉換成資料類別定義的對應語言，稱為 BODED Script to Class Mapping Description Language，簡稱 BSCMDL。根據這個語言，BODE 資料模式處理系統可以輸入一個 BODE script，然後輸出一個資料物件定義程式。並且根據此資料物件定義程式，可以直接在關聯式資料庫中建立相對應資料表。而沒有 BSCMDL 程式的情況下，亦可根據預設值直接從 BODED script 轉換出資料類別定義，並建立相關的關聯式資料庫。省去設計資料類別的時間。

對於來自不同網頁的資料匯整，在此研究中，我們研究資料之間的關聯性，以及在轉換過程中，這些關聯可能發生的變化，並據此提出資料類別的轉換語言，稱為 CCMDL。根據 CCMDL，BODE 資料模式處理系統可以知道如何將一個資料類別中的資料，轉換為另一個資料類別中的資料。

此外，由於自動產生的類別會產生許多僅具備關聯關係而沒有實質欄位的類別，因此，我們也設計了一些最佳化的方法，簡化類別的層次。

八、參考文獻

[1] Gustavo Arocena and Alberto Mendelzon. “WebOQL: Restructuring Documents, Databases, and the Web”, In *Proceedings of ICDE*, 1998, Orlando, Florida.

[2] R. Baumgartner, S. Flesca, G. Gottlob. “Visual Web Information Extraction with Lixto”, In Proceedings of the 27th International Conference on Very Large Data Bases (VLDB'01), 2001.

[3] Alin Deutsch, Mary Fernandez, Daniela Florescu, Alon Levy, and Dan Suciu. “XML-QL: A Query Language for XML”, In *Proceedings 8th International World Wide Web Conference (WWW8)*, 1999. Computer Networks 31(1116) : 1155-1169.

[4] M.Fernandez et al.,“Declarative Specification of Web Sites with STRUDEL,” VLDB Journal,vol.9,no.1,2000,pp.38-55.

[5] G. Mecca, P. Merialdo, P. Atzeni. “ARANEUS in the Era of XML”, *IEEE Data Engineering Bullettin, Special Issue on XML*, September, 1999

[6] Steve Holzner. “XML Complete”, McGraw-Hill, 1998.

[7] David Konopnicki , Oded Shmueli. “Information gathering in the World-Wide Web: the W3QL query language and the W3QS system”, *ACM Transactions on Database Systems (TODS)*, Volume 23 Issue 4, Dec. 1998.

[8] C.A.Knoblock et al.,“Accurately and Reliably Extracting Data from the Web:A Machine Learn-ing Approach,” IEEE Data Eng.Bull.,vol.23,no.4,2000,pp.33-41.

[9] Phillip Merrick, Charles Allen. “Web Interface Definition Language”, W3C NOTE, Sep. 1997. http://www.w3.org/TR/NOTE-widl.

[10] Microsoft Corporation. “WebBrowser Control”, Programming and Reusing the Browser, MSDN Library, 2002. http://msdn.microsoft.com/library/default.asp?url=/workshop/browser/webbrowser/browser_control_node_entry.asp.

[11] Jonathan Robie, Joe Lapp, David Schach. “XQL：XML Query Language”, Workshop on XML Query Languages, Dec. 1998.
http://www.w3.org/TandS/QL/QL98/pp/xql.html.

[12] W3C Consortium, “XML Query”, Apr. 2000.
http://www.w3c.org/XML/Query.

[13] W3C Consortium. “XQuery 1.0: An XML Query Language”, W3C Working Draft, 16 Aug. 2002.
http://www.w3.org/TR/xquery/.

[14] W3C Consortium. “Extensible Markup Language (XML) 1.0 (Second Edition)”, W3C Recommendation, Oct. 2000.
http://www.w3.org/TR/2000/REC-xml-20001006.

[15] W3C Consortium, “HTML 4.01 Specification” W3C Recommendation, Dec. 1999.
http://www.w3.org/TR/html4/.

[16] W3C Consortium. “XQuery 1.0 and XPath 2.0 Data Model”, W3C Working Draft, Aug. 2002.
http://www.w3.org/TR/query-datamodel/.

[17] W3C Consortium. “XML Path Language (XPath) 2.0”, W3C Working Draft, Aug. 2002.
http://www.w3.org/TR/xpath20/.

[18] I-Chen Wu, J.-Y. Su, and L.-B. Chen. “On the Web Data Extraction Model”, submitted to the Proceedings 17th International Conference on Software Engineering and Knowledge Engineering (SEKE2005), 2005.

[19] I-Chen Wu, J.-Y. Su, and L.-B. Chen. “A Web Data Extraction Description Language and Its Implementation”, The 29th Annual International Computer Software and Application Conference (COMPSAC 2005), Edinburgh, Scotland, July, 2005.

[20] I-Chen Wu, J.-Y. Su, and L.-B. Chen. “The User Guide of the BODEDlet Plug-in System”, internal document, 2004.

九、計劃成果自評

在 BODE 資料模式當中，我們使用類別的宣告來定義一對一的資料，可以簡化對資料的認知與描述。由於網頁上的資料之間的關聯通常並不固定，某些同樣的資料在一個網站上是一對一，但是在另外一個網站上則是一對多或是多對一。因此我們使用關聯資料表(Relation Table)來記錄資料物件之間的關聯。而關聯資料表是在 BODE 資料模式處理系統在萃取資料的過程當中動態建立的。因此在網站上實際資料間的關聯會真實的紀錄在關聯資料表中。關聯資料表可以表示一對一、一對多、多對一與多對多的關聯。同時也可以表示不同種類的關係。

本計劃對網頁資料的自動萃取所需的資料表示與儲存問題，提出一套較有彈性的解決方法，利用兩個以 XML 為基礎的物件定義語言以及資料模式對應語言來描述資料以及 BODED 萃取程式中所要萃取的資料與資料模式的對應規則。並提供由 BODED 萃取程式自動產生資料類別定義與對應規則。由於 BODE 系統具有視覺化操作介面，且資料模式與對應規則可自動由 BODED 萃取程式產生，這使得我們在操作，控制瀏覽順序、資料萃取以及資料儲存與使用上變的更加容易。本系統具有下列的特色。

(1) 使用與瀏覽器一致的視覺介面。
(2) 能以特定的順序瀏覽網頁，並紀錄網頁之間的資料關聯。
(3) 利用外掛程式來處理萃取的資料，並將資料自動的儲存到資料庫中，具有容易擴充與更新的特性。
(4) BODE 資料模式具有簡單的資料定義格式
(5) 可以彈性的方式描述各種物件之間的關係，且這些關係可隨不同的應用以及不同的網站類型隨時的加入。
(6) 可由 BODED 萃取程式自動產生資料模式中的類別定義。
(7) 可由 BODED 萃取程式自動產生對應規則，將要被萃取的資料對應到資料庫類別定義中。

此外，我們實作了 BODE 資料庫模式處理系統，可以自動的處理與儲存網頁上所萃取的資料。

我們除了完成了上述工作外，我們更將計畫中所研究發展的成果，發表論文到 SEKE 2005 [18] 及 COMPSAC 2005 [19]（接受機率：72/278~25.9%），兩個不錯的國際會議。這顯示此 BODE 系統的研究已經被國際相關領域肯定。目前，我們更進一步地整理這些論文，投稿到 ACM Transactions on Internet Technology 這個知名期刊。