第二章 相關文獻探討

本章主要是相關文獻的探討與整理。第一節探討由陳連壎【3】所提出的網路書籍概念及其理論，並簡述已規劃出來的架構及相關系統模組；第二節介紹 SCORM 規範與其相關技術；第三節介紹其他相關技術，其中 LMML【27】是符合 XML 規範之另一種學習領域相關語言，該語言採取與 SCORM 不同的面向，從教材架構模型的角度來對教學單元進行 Metadata 描述；另外也探討了 Web Service 的相關技術以及其重要性，並分析未來 E-Learning 與 Web Service【38】技術結合的可能與重要性。

第一節　網路書籍模式

現今的網際網路上充斥著大量的數位化電子資訊，許多專門學科領域更將他們數位化後的專業領域知識教材透過網頁的方式來呈現與傳播。如果能夠將這些資訊適當地加以整理，站在幫助學習的角度而言，將是一種便利的新管道，由是利用網路資源來編寫書籍或者其他形式教材的趨勢開始逐漸成形。在這樣的前提下，由本實驗室的陳連壎【3】提出了個人化隨取書籍模式及相關理論，其概念指出在結合全球資訊網的優勢之下，未來的書本將具有以下的特點：內容來自全球各地、即時瀏覽、結合多媒體呈現、可依據瀏覽者的需求變更內容以及保持內容的新鮮度。我們將此種書本形式稱為”隨

取書籍”(Book-On-Demand，BOD)【9】，而依此形式所建立的書本為網路書籍(Web Book)【3】【8】。在上述的模式中，教材編寫者可以將所要取材的網頁透過快取機制，輔以自定義的邏輯架構編輯成 Web Book，並透過網路，以快速、不延遲的方式呈現在瀏覽者的面前。使用者可以將編輯好的書籍放置到網路上，與其他使用者分享。

網路書籍的概念略述如下：首先是由網路書的作者(Author)，透過網路書編輯器(Web Book Editor)的協助，整理各網站的資訊形成一本具有書籍章節結構的網路書。在這個網路書籍中提供給瀏覽器端的閱讀者是一種網頁資訊的邏輯觀點(Logical View)，讓閱讀者可以將跨越各個網站的資訊視為一本邏輯式的網路書。閱讀者不需要處理在實體觀點(Physical View)下的各網站的超連結和網址，即能閱讀其網頁資訊。如圖 2-1 所示，作者先利用網路書編輯器擷取各網站的資訊，並以書籍的呈現結構整合成一本依作者邏輯觀點編排的網路書，並登錄發行到書籍伺服器(Book Server)上。書籍伺服器是介於各個資訊網站和使用者端的瀏覽器之間的一種加強功能的代理伺服器(Proxy Server)，而其提供的服務有兩大項目，一為替資訊閱讀者提供邏輯觀點的網路書以取代各網站網頁的邏輯觀點；二為替資訊閱讀者提供網頁動態整理擷取和下載等功能。

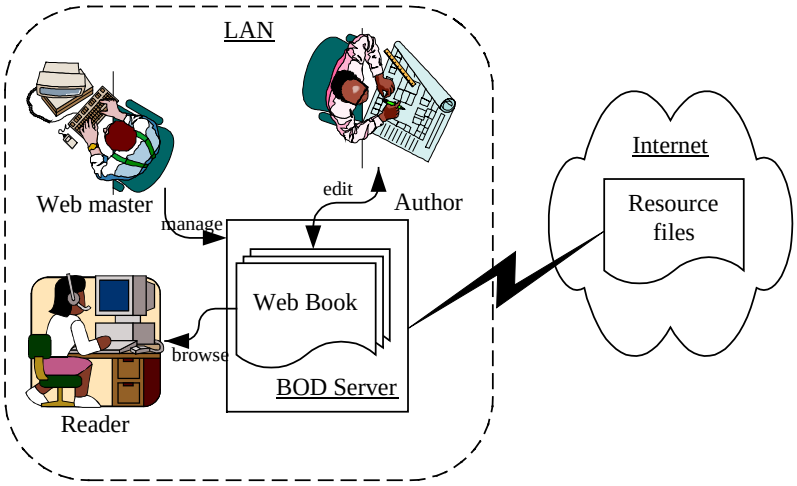

圖 2-1　BOD 模式各部份功能示意圖

Web Book 可應用的範圍相當廣泛，如：可提供企業做為新進員工教育訓練手冊或在職員工的進修教材，其內容除了可涵蓋既有的自編教材外，還可以是取材自網際網路上的資源或透過企業間的交流而取得的相關教材，Web Book 形式的教材可以幫助企業節省花費在教育訓練方面的成本並有效提升企業的人力素質；而在教育方面可以做為教師指定給學生的課外補充教材，甚至可以作為正式教材之用，藉由多媒體的生動活潑來增加學生學習的興趣和成效，另外由於書本的呈現乃是由教師所規劃的邏輯架構為主，故可以針對學生的學習狀況或某些特殊需求隨時進行調整，讓教材更具彈性及適性化(Adaptive)的性質；在另一方面，由於網際網路上的資源包羅萬象，可以提供給各階層的社會大眾作為終身學習的教材之用，為民眾提供另一種自修的管道與媒介。

第二節　SCORM 標準

早期的電腦教學或輔助學習系統由於開發者的知識背景與當時的資訊技術限制，各領域分別發展出來的教材與教學平台都只能符合各自的需求，且加上沒有共同的標準可以依循，不同的組織要採用同樣內容的教材時，通常都要投入大量人力去重新開發，收到的投資效益卻往往遠不如預期。為了降低教材開發時程以及成本，美國國防部於１９９７年推動「先進分散學習計畫」（ADL），ADL的理念是希望透過相關規範的制訂來實踐教材的共享和重用性，且經過物件化後的教材單元可以依照使用者的需求即時的組成(Assemble)，隨時隨地(Anytime, Anywhere)提供學習與輔助(Assistance)，而這些理念正是ADL計畫中”A”所代表的意涵。自１９９９年起，ADL 聯合業界、學界與軍方等單位共同成立ADL Co-Laboratory，成為全球性的組織以及共通標準的制定單位。ADL計畫參照IMS【22】、AICC(Aviation Industry CBT Committee)【10】、IEEE 學習技術標準委員會(Learning Technology Standards Committee，LSTC)【21】及ARIADNE【11】等機構先前所各自建立的相關學習標準，並結合了來自產官學界的眾多廠商與使用者的意見，制訂出SCORM教材規範作為電子化學習的共同技術標準。SCORM 1.2規格書分為三個部分：The SCORM Overview【16】、The SCORM Content Aggregation Model(CAM)【17】和The SCORM Run-Time Environment(RTE)

【18】其架構如圖2-2示，我們將在以下的小節中進一步探討規格書中的相關內容。

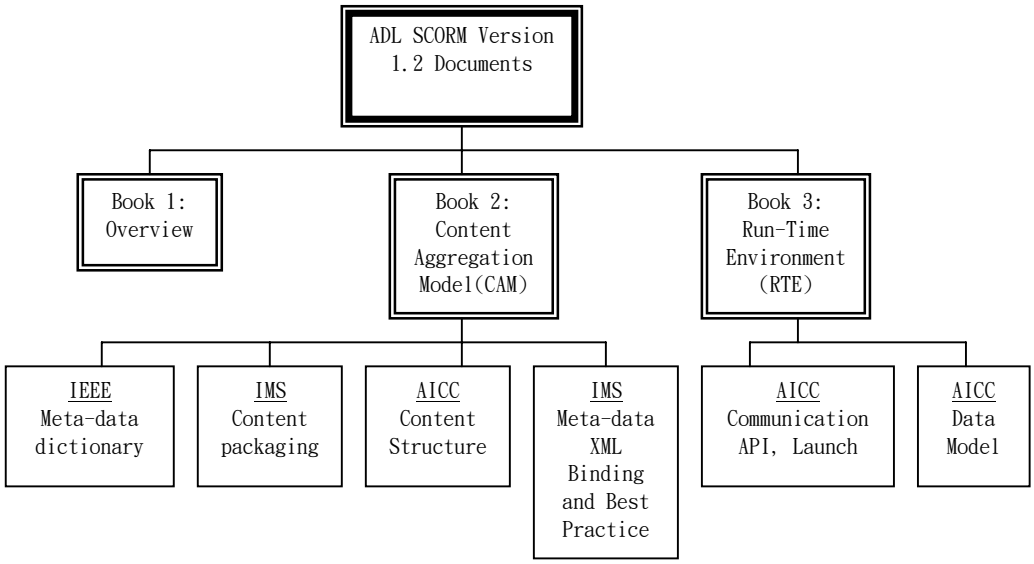

圖 2-2　SCORM 1.2 規格書與相關機構規範關係圖

2.2.1　SCORM 教材內容整合模式(Content Aggregation Model)

SCORM 的課程物件內容整合模式支援元件式教學模組，將教材的編寫過程視為教材元件的重新組合。因應這樣的編輯模式，SCORM CAM【17】定義了下列三種元件模式：

1. 內容模式(Content Model)。
2. 元資料(Meta-data)。
3. 內容包裹(Content Package)。

以下針對三個部分個別說明：

(1)內容模式

SCORM 將課程物件定義為下列三種層級，分別是 Asset、SCO (Sharable Content Object)和課程整合(Content Aggregation)。Asset 是課程物件最基本的元件型態，可由文字、圖形、聲音、網頁和任何可由瀏覽器開啟的電子檔案格式所組成。SCO 則是 SCORM 執行環境可以操作的最基本單位，由一個或數個 Assets 所組成，具有與執行環境平台溝通的能力。而課程整合則是在上述的兩類物件型態上加上一個邏輯描述架構，使之成為完整的課程，其架構如圖 2-3 所示。

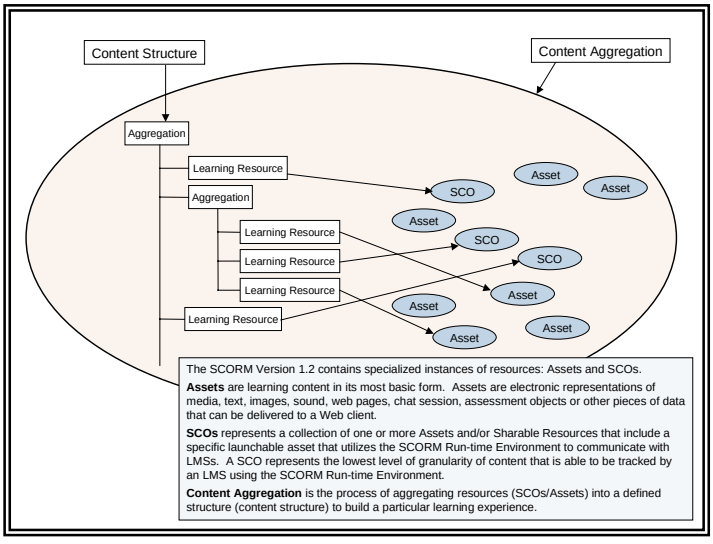

圖 2-3　課程整合架構圖(取自 www.adlnet.org)

(2)元資料

SCORM 的 Meta Data 是參考 IEEE LTSC 和 IMS 兩個組織先前所訂定的元資料 XML 連結(Meta-data XML Binding)規格書而來。規格中將描述資料定義為九種類別(Category)，每一類裡面都包含許多 XML 標籤，共計有 80 個，對於九個類別的內容分述如下：

1. 一般(General)：針對整個教學資料描述其一般性資訊，如標題、使用語言、關鍵字與架構等。
2. 生命週期(Lifecycle)：描述教學資料的的版本、狀態和提供者的相關資料。
3. 描述資料(Meta-metadata)：描述整份 Meta Data 的提供者及其編寫整份 Meta Data 的方法及所參考的 XML Schema【39】等。
4. 技術(Technical)：描述教學資料相關技術方面的訊息，如格式、大小、位置、執行環境需求等。
5. 教育(Educational)：描述教學資源的教育相關訊息，如互動型態、互動程度等等。
6. 版權(Rights)：描述這份教學資源的版權及收費等相關訊息。
7. 關係(Relation)：描述這個教學資源和其他資源間的關係。

8. 註釋(Annotation)：幫助教材引用者更瞭解此份課程物件的意義，類似於大部分知名網路書店的書評機制，內容包含發表評論者、發表評論的時間和其評論內容。
9. 分類(Classification)：補足一般分類的不足，對教學元件可進行更多其他形式的分類。

(3)內容整合

SCORM 將教材以壓縮檔的格式進行打包的動作，將該課程所需的元件包裹成一個檔案，以利於不同的教學系統間可以交換共享該份教材。教材包裹(Package)內容除所需元件的實體檔案及對應的 Meta Data 以外，另外附上一清單檔(Manifest)來描述包裹內部檔案的相關訊息，並以 Meta Data 描述該份包裹的相關訊息。Manifest 的架構具有很高的彈性，可由發展者定義內部元件的組織狀態，並允許巢狀架構，也可包含子清單。該壓縮檔稱為包裹交換檔(Package Interchange File，PIF)，圖 2-4 為 PIF 檔的架構概念示意圖。

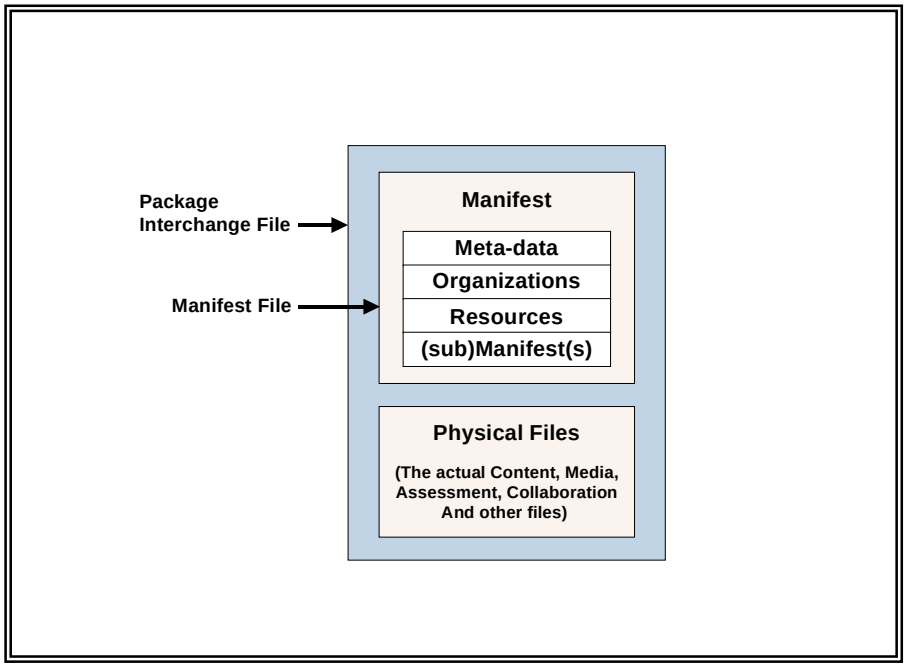

圖 2-4　PIF 檔案架構概念圖(取自 www.adlnet.org)

程式碼 2-1 是依照 SCORM 1.2 Content Package 規範所編寫的一個範例檔案，我們可以從裡面清楚看出 SCORM 將資源的邏輯架構與實體分開描述的作法。

```
<?xml version="1.0" encoding="utf-8" ?>
- <manifest identifier="MANIFEST01" version="1.1"
        xmlns="http://www.imsproject.org/xsd/imscp_rootv1p1p2"
        xmlns:imsmd="http://www.imsglobal.org/xsd/imsmd_v1p2"
        xmlns:adlcp="http://www.adlnet.org/xsd/adlcp_rootv1p2"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.imsproject.org/xsd/imscp_rootv1p1p2 imscp_rootv1p1p2.xsd
        http://www.imsglobal.org/xsd/imsmd_rootv1p2p1 imsmd_rootv1p2p1.xsd
        http://www.adlnet.org/xsd/adlcp_rootv1p2 adlcp_rootv1p2.xsd">
    <metedata />
  - <organizations>
      - <organization identifier="MANIFEST01_ORG1" structure="hierarchical">
            <title>ch1</title>
          - <item identifier="MANIFEST01_ITEM1"
                identifierref="MANIFEST01_RESOURCE1" isvisible="true">
              <title>Chapter 1</title>
            </item>
        </organization>
    </organizations>
  - <resources>
      - <resource identifier="MANIFEST01_RESOURCE1" type="webcontent"
            adlcp:scormtype="sco" href="S01_power.htm">
          - <metadata>
              <schema>ADL SCORM</schema>
              <schemaversion>1.2</schemaversion>
              <adlcp:location>./S01.xml</adlcp:location>
            </metadata>
```

程式碼 2-1　imsmanifest.xml 檔案範例

```
            <file href="S01_power.htm" />
            <file href="pics/01_power.GIF" />
            <file href="pics/ball00.GIF" />
            <dependency identifierref="MANIFEST01_RESOURCE2" />
            <dependency identifierref="MANIFEST01_RESOURCE3" />
            <dependency identifierref="MANIFEST01_RESOURCE4" />
        </resource>
      - <!-- Assets -->
      - <resource identifier="MANIFEST01_RESOURCE2" type="webcontent"
              adlcp:scormtype="asset" href="pics/01_power.GIF">
          - <metadata>
              <schema>ADL SCORM</schema>
              <schemaversion>1.2</schemaversion>
              <adlcp:location>pics/01_power.xml</adlcp:location>
            </metadata>
            <file href="pics/01_power.GIF" />
        </resource>
      - <resource identifier="MANIFEST01_RESOURCE3" type="webcontent"
              adlcp:scormtype="asset" href="pics/ball00.GIF">
          - <metadata>
              <schema>ADL SCORM</schema>
              <schemaversion>1.2</schemaversion>
              <adlcp:location>pics/ball00.xml</adlcp:location>
            </metadata>
            <file href="pics/ball00.GIF" />
        </resource>
      - <resource identifier="MANIFEST01_RESOURCE4" type="webcontent">
            <file href="scripts/APIWrapper.js" />
            <file href="scripts/AUFunctions.js" />
        </resource>
    </resources>
</manifest>
```

程式碼 2-1 imsmanifest.xml 檔案範例(續)

2.2.2 SCORM 執行環境

傳統的網路教學中，不同的教學系統通常採用各自定義的資料格式和資料處理方式，造成不同平台間教材交換時的許多問題，SCORM 針對此點，特別定義了執行環境的規範，內容包含三個部分：啟動機制(Launch)、應用程式介面(Application Program Interface，API)和資料模組(Data Model)【18】。Launch 是規定一個共通的方式讓學習系統可以啟動教材，教材透過 API 與系統進行訊息溝通，而系統也可以從這些訊息中控制教材的進行狀態以及追蹤學習者的動作，規範中將 API 訂為八個函式呼叫，並規定所有的教學管理系統(LMS)平台在實作時，不論所使用的技術或採用的程式語

言為何，都必須將所提供的 API 呼叫依照規範所訂定的名稱來命名，各函式的功能和代表含意也必須與 SCORM 所規定的相同，並應提供可用來呼叫這些函式的 API Adaptor。以下對八個函式的命名與代表意義作簡略的說明：

- 控制 SCO 狀態的有兩個函式：
  1. LMSInitialize(“ ”)：通知 LMS 一個 SCO 被啟動。
  2. LMSFinish(“ ”)：通知 LMS 被啟動的 SCO 已經完成，亦即學習者已經完成該 SCO 的學習活動。
- 學習狀態中，用來處理錯誤的有三個函式：
  1. LMSGetLastError(“”)：SCO 透過 API 與 LMS 溝通的過程中如果發生錯誤，系統會進入錯誤狀態，透過本函式可以取得相關的錯誤代碼，有助 LMS 得知前一個發生錯誤的 API 函式呼叫，取得錯誤類型的代碼。錯誤代碼由三個位元的數字構成共分四種類別，各以 1~4 為起始數字，1xx 代表一般錯誤，2xx 代表語法錯誤(Syntax Error)，3xx 代表 LMS 造成的錯誤，4xx 為 Data Model 錯誤，如果沒有發生錯誤時呼叫本函式則傳回的數值為 0。
  2. LMSGetErrorString(“errornumber”)：將錯誤代碼傳入此函式，則可以得知該錯誤代碼所代表的錯誤類型和意義。例如：

LMSGetErrorString(“403”)傳回的字串為 Element is read only，表示引起錯誤的元素屬性為唯讀。

3. LMSGetDiagnostic(parameter)：將錯誤代碼傳入此函數後的回傳字串除了代碼本身所代表的意思之外，還有教材提供者(Vendor)所提供的關於此錯誤類型的額外敘述。

- 控制資料傳輸的有三個函式：

1. LMSGetValue(data model element)：將 Data Model 的元素(Element)或群組(Group)名稱傳入本函式中，傳回與 SCO 的基本資料相關的字串，目前可傳入本函式的值共有四種：
   i. datamodel.group.element:傳回目前 Data Model 中該元素欄位所存入的值。
   ii. datamodel._version：傳回目前 LMS 所支援的 Data Model 版本。
   iii. datamodel.element._count：傳回目前 Data Model 陣列(Array)中的元素個數。
   iv. datamodel.element._children：傳回目前元素或者類別中可被 LMS 支援的所有元素個數。

2. LMSSetValue(data model element, value)：將第二引數的值傳入所指定的 Data Model 元素中，例如將學生姓名存入 cmi.core.student_name 元素中。
3. LMSCommit("")：API Adaptor 在處理數值時有時候會將其先放在暫存記憶(Cache)中，執行本函式後會強制將數值全部寫進資料庫中。

有了上述八個函式加上 SCO 藉以和 LMS 溝通用的 API Adaptor 後，還必須用來儲存各種資料的統一資料庫格式，Data Model 規範便是訂出統一的資料格式，使 LMS 可以更詳盡的紀錄學習者的相關資料和學習狀態與歷程等，且其記錄的型態也不會因為系統的不同而有所差異。SCORM 的 Data Model 規範是取自 AICC 的 CMI Data Model 規範，將用來記錄相關訊息的變數命名及規格作統一的規範，以幫助 LMS 可以更方便的追蹤學習者的相關資料、學習進度以及學習歷程等許多相關訊息。SCORM Run Time Environment 的巨觀架構如圖 2-5 所示。

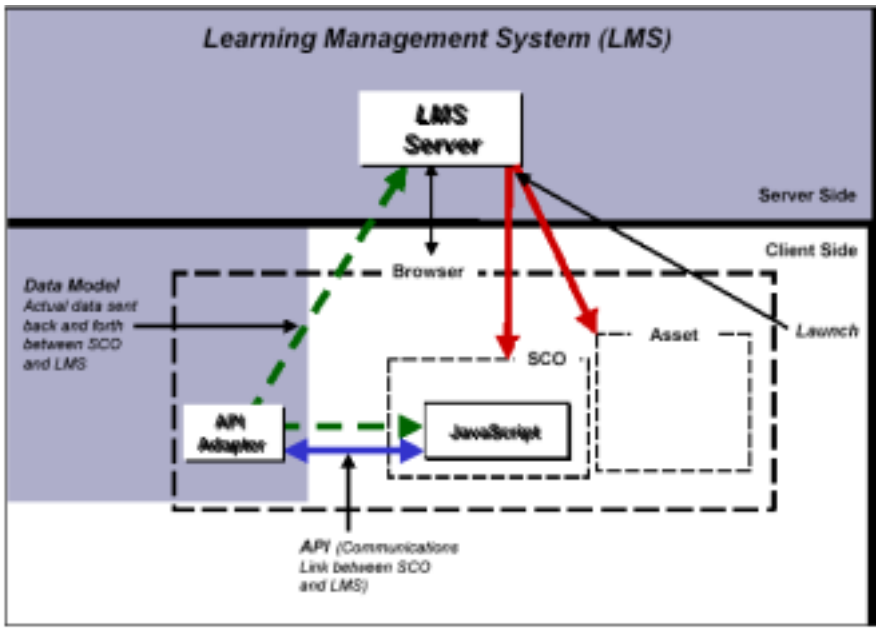

圖 2-5　SCORM RTE 架構圖(取自 www.adlnet.org)

第三節　其他相關技術

本節共分兩個部分，第一部份討論以 XML 為基礎的其他教育相關語言，主要是以 LMML【27】為例，並就其與 SCORM 不同之處深入探討；第二部分討論 Web Service 技術目前的概況，並評估未來 E-Learning 與 Web Service 相關技術互相結合的可能性與重要性。

2.3.1　Learning Material Markup Language(LMML)

今日，各式的教材常藉由 Html, Pdf, Word 文件或者 Powerpoint 投影片等電子格式加以儲存並呈現透過網路瀏覽的動作呈現在讀者面前。這樣的呈現方式不但必須依賴特定的閱讀軟體(Viewer/Reader)才能讀取，且文件本身

總是在某種程度上出現缺乏結構性的缺失。如此一來，知識或教材管理系統所需用到的描述資訊(Meta)將會容易發生遺失或只能對個別教材在某些細部單元設定特定的訊息的現象。對於教材的需求一般而言可分為兩方面來討論：一方面，學習者需要在個人感興趣的部分上有適合個人的特定呈現方式及搜尋行為；另一方面，教材編寫者則偏好能夠容易找到現存的線上資料加以重用，以及將手頭現有的教材重新整合以滿足不同學習族群的需求。而上述的教材格式在這些概念層次及操縱模型的向度上面臨了很大的挑戰。針對這些需求，Christian 等學者【12】提出一個 Meta-modeling 的方法來架構出符合上述需求的系統，並且依循 W3C 的 XML 規範制訂出一個稱為 LMML 的語言，該語言主要是為了加強在教材內涵及網路查詢層面的知識之自我描述性，而針對不同學習領域的個別需求也可以彈性地加入適當的標籤已符合該領域的特性，程式碼 2-2 為一個簡單的 LMML 文件範例。

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet href="toc.xsl" type="text/xsl"?>
<lmml version="-//DE.UNI-PASSAU.DAISY//DTD LMML-CS 1.1//EN">
<section title="Learning Material Markup Language 1.1" author="Christian Süß und Stefan Winter">
<section type="course" uri="../intro.xml" label="secallg" title="Allgemeines"/>
<section type="course" title="MediaObjects" uri=../mediaobject.xml" label="secmediaobject">
<section type="course" title="ContentObjects" uri="../contentmodul.xml" label="seccontentmodul"/>
</section>
</lmml>
```

程式碼 2-2　LMML 範例程式

不同特性的教材往往有其各自不同的需求，在教材的內容以及編寫架構上都不盡相同，而依據不同概念模型編寫出來的教材在與其他學習單元間的關連及用語模式等也都各異其趣，不過我們仍然可以大致歸納出，儘管各教

材概念都以不同的格式呈現在讀者面前，但其基本架構通常都是以一個特定的學習單元作為根節點(Root Node)，然後以一定的架構或順序呈現出來，各觀念通常使用超連結的方式來操作與彼此之間的連結關係，而索引、詞彙或術語則可以獨立於整體結構之外獨立處理。對於整份教材的概念層次模型或者是個別學習單元本身則可以透過 Metadata 來加以描述，以符合不同操作情境下的使用者的需求。

有鑑於上述的特色以及需求，LMML 將描述教材用的 Metadata 分為兩個部分，針對抽象概念層次的模型教學單元實體的模組抽離開來，概念模型層次方面主要是描述各模組間的連結關係以及彼此之間的關係，而實體層次則是分門別類將教學單元區分為文章段落、表格、列表、媒體以及其個別的動機、定義、範例、習題和測驗題等部分。LMML 描述模型的架構可以表示成如圖 2-6 的形式。

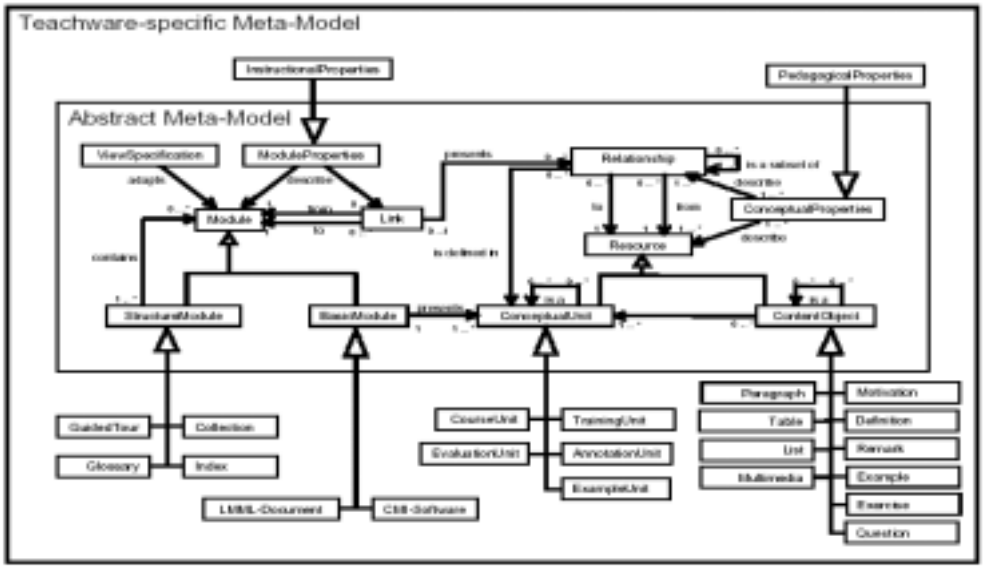

圖 2-6　LMML 描述架構狀態圖(取自 LMML)

LMML 語言為教材的實體檔案編排提供了一套嚴謹的定義與規範方法，不但可以使得教材本身的結構更加嚴謹，也可以增加教材本身的自我描述性，而在抽象層次上，也提供了一套可以通用於各類教材間的架構模型，在教材的管理維護與重用性的考量上，提供了有別於 SCORM 的另一種選擇。

2.3.2 Web Service 及其相關技術

XML 是一種具有彈性而且可以由使用者自行定義標籤的網路標記語言，截至目前為止，已經有許多領域向 W3C 提出依照其各自領域需求而制訂出符合 XML 規範的語言，目前在電子商務的領域中最炙手可熱的標準，首推 Web Service 規範以及其相關技術。許多主要的技術提供者，如 IBM【20】、微軟【28】以及昇陽【33】等知名的資訊廠商都各自別出對 Web Service 的定義，簡單歸納其定義，我們可以說「Web Service 是一種可透過標準 XML 訊息存取的網路操作介面【19】，負責將自網路上接收到的 XML 資料傳入現存的軟體系統或元件中，並將處理後的結果以 XML 訊息回傳給呼叫此介面的用戶端。Web Service 具有元件的概念，可以用數個服務來組成一個新的服務，對外只要提供給用戶端標準的呼叫介面就可，實作時所採用的硬體平台或者本身的程式邏輯則可以充分隱藏，此一特點與物件導向程式設計的觀念相仿。」

在眾多 XML 語言中，與 Web Service 相關的技術與服務有 SOAP (Simple Object Access Protocol)【32】，UDDI(Universal Description, Discovery and Integration)【37】和 WSDL(Web Service Description Language)【40】等。SOAP 將訊息以標準的 XML 語法加以包裹，透過標準的超文件傳輸協定(HTTP)協定，在不同的系統間進行訊息傳遞的工作。而 UDDI 則是線上的公開註冊機制，服務提供者登錄後的服務可以透過 UDDI 讓其他使用者進行查找以及整合等行為，我們大致可以將 UDDI 類比為現實世界中的工商名錄或相關類似的服務。WSDL 則是描述該 Web Service 本身的細節，包含該服務本身的功能說明及其所提供給使用者叫用的介面等。

然而目前 UDDI 和 WSDL 本身所能夠提供的功能以及其所能表現的語意(Semantic)仍然是很有限的，透過關鍵字查找服務後加以整合並不能夠完全滿足用戶的需求，故而在語意描述的方面，W3C 另外制訂了資源描述架構規範(Resource Description Framework，RDF)【31】來滿足服務提供者的需求，其他機構如美國國防部先進研究計畫處(US Defense Advanced Research Projects Agency，DARPA)也以 RDF 為基礎針對語意描述方面訂出了 DAML 語言【14】。使用前述的技術並加入語意描述語言的 Web Service 我們稱之為 Semantic Web Service。

就 SCORM 的技術層面來討論，我們認為未來其應該加入 Web Service 的技術或理念，方能使得 SCORM 的應用層面更加廣泛。舉例來說，目前雖

然號稱符合 SCORM 規範的教材元件可以達成分享的目標，但卻遲遲不見相關技術的出現或者更進一步的規範，如果可以透過 Web Service 的機制，在教材系統上加入相關技術，使教材的提供成為該公司的 Web Service，並在類似 UDDI 的機制上註冊與發佈，使用者在搜尋教材時的困難度應該會比目前還要低，而以多個 Web Service 重組成一新的服務的作法，也將會使得教材的重組和重用更加容易，且 Web Service 為了因應電子商務的需求，在收費機制方面的制訂與隱私的保全方面都要比目前 SCORM 現有的規範來得完善，所以採用 SCORM 為共通標準外如果能夠結合 Web Service，相信會為產業帶來更多的效益，也更能真正落實 E-Learning 產業化的目標。我們可以預期未來結合 Web Service 及語意描述的技術，並將提供教材視為對外提供的服務，針對使用者的個別需求，從教材元件庫中動態組合出合適的教材，將能促使 E-Learning 與產業緊密結合，徹底實現 E-learning 跨領域、跨時空學習的理念。