國立交通大學

資訊管理研究所

碩 士 論 文

以目錄服務為基礎達成企業知識入口網站間的知識交換之機制

A Directory Service Based Mechanism for Knowledge Sharing Between Corporate Knowledge Portals

研 究 生： 吳擁宙

指導教授： 黃景彰 博士

中華民國九十二年六月

以目錄服務為基礎達成企業知識入口網站間的知識交換之機制

學生：吳擁宙　　指導教授：黃景彰 博士

國立交通大學資訊管理研究所碩士班

中文摘要

資訊技術的發達使一般企業可以容易地進行知識的編寫及保存。而隨著企業規模的擴大，企業的知識文獻便有可能分散儲存在各個不同的儲存地點。在此種分散式的環境下，企業應提供一個適當有效率的方式，使欲利用文獻的使用者能夠方便、正確、快速地取得該文獻。

本研究提出知識藍圖系統，使企業的知識入口網站間能方便地進行知識交換。本研究以目錄伺服器做為企業內知識文獻之元資料（metadata）的儲存者。知識藍圖系統運用了目錄服務可以一個單一的邏輯架構呈現分散在不同地區、位置的資訊之優點，並結合企業知識入口網站之 web 介面，提供予企業內各使用者一個整合的環境，可進行位在不同處之知識文獻之分享、查詢、瀏覽等工作，達到知識交換的目的。

關鍵字：LDAP，目錄服務，企業知識入口網站，元資料

A Directory Service Based Mechanism for Knowledge Sharing Between Corporate Knowledge Portals

Student：Yung-Chou Wu　　Advisor：Jing-Jang Hwang

Institute of Information Management
National Chiao Tung University

ABSTRACT

The advancement of information technology enables enterprises to codify and store organizational knowledge without difficulties nowadays. Enterprises grow in time with the founding of new branches, departments, or project groups, resulting in the proliferation of knowledge documents that are inadvertently hoarded in diverse physical locations. Under this distributed environment, an effective and efficient means needs to be developed for users to access right knowledge documents accurately.

Accordingly, the research proposes a directory-based knowledge blueprint system that grants knowledge sharing between corporate information portals. The present study uses an LDAP-compliant directory server as the repository of the metadata of the knowledge documents in the enterprise. Leveraging directory service's feature of displaying distributed information in one logic view, the system provides users with an integrated portal environment where users can share, search, and browse dispersed knowledge documents.

Keywords: LDAP, Directory Service, Enterprise Knowledge Portal, Metadata

誌謝

首先要感謝我的指導老師黃景彰教授，在兩年的學習中，對於我的研究給予相當大的彈性，並適時提供鼓勵，讓我受益良多。同時，也要感謝四位口試委員：陳安斌教授、廖耕億教授、陳昱仁教授和劉興華總經理在口試過程中提供我許多寶貴的意見。

另外也要感謝資訊安全實驗室的成員及學長、姐們，星吏、定翰、曉玲、婉淇，和方仁威、林樹國兩位學長無論是在課業、研究或是其他待人處事上，均曾給過我不少的幫助及啟發。

JBORG 成員的家甫、小竺跟冠宇是資管界的奇蹟。有了你們，研究變得更有趣了。在你們身上我見到什麼是天才、搞怪、認真打拚…各種奇異特質的集合體。你們是我研究所生活的不定時炸彈，呃…不，該說是靈感、創意及樂趣的來源。感謝你們不管在研究苦悶，或不苦悶的時候（就是無時無刻啦），與我一同研究 CS、War3，讓我飽嘗勝利滋味。

感謝可愛的女友鳳儀，經常在我研究繁忙之時來吵…呃…噓寒問暖，讓我永遠保持研究的動力及進取的心。

最後，要感謝家人的支持。沒有家人，是不會有我跟這篇論文的。

感謝所有在有意或無意間曾幫助我的人。

目錄

圖目錄

表目錄

第一章 緒論

1.1 研究背景與動機

科技的發展日新月異，以往書信仍是人與人、或是企業間傳遞訊息的重要媒介。自網際網路問世乃至興盛以來，書信已逐漸退為次要的信息媒介。網站、電子郵件成為民眾、企業發佈、接收資訊的主流媒體。不僅止於資訊傳遞的革新，網際網路的普及化所帶來的是全面性的影響，如：個體的人際關係(網路社群)、娛樂行為(線上遊戲)、企業的商業模式、交易流程、學術的資料查詢、甚至是國家的戰爭等等。這些變化可說是空前的。舉例來說，對現代的企業而言，撰寫、保存、傳遞、簽核繁複的公文、知識文件對員工是一大負擔。企業若能妥善利用網路進行各項作業，能夠讓效率提升，對組織營運績效將能夠有所幫助。

就現今而言，企業與企業間的購併頻繁；企業與企業間的合作也是再平常不過。再者企業內部門的增減、人員流動等的速度也比以往來得迅速，這些由網路所帶來的改變是無可避免的。無法跟上十倍速前進的時代的企業，將會遭到無情的淘汰。在這種時空環境之下，許多問題因應而生。例如：當文件分散儲存在企業間或企業各部門間之內，該如何設計一有效之共享、傳遞方式，以讓需要某份文件的人能在簡單的查詢後便可迅速、方便地取得正確的資料。針對該問題，目前的解決方式大約有以下數種：FTP、電子郵件、新聞群組、作業系統內建的檔案分享系統等等。以 FTP 方式解決文件傳遞及共享的優點是速度快，而缺點在於其對於一檔案所能顯示的描述資訊極為有限，使用者需要由檔名猜測該檔案是否為自己所需的檔案。此種方式於企業內文獻檔案數量尚少之時影響不大，然一旦檔案數量增加至一定的幅度，或是檔案的版本太多，則難以評斷。最後只能將檔案全數下載回本機，再一一開啟確定來解決這個問題。電子郵件的應用時機有限，主要在收方及送方皆有收送檔案的需求、且已經互相通知時，在其他例如要搜尋知識文獻等等場合較難派上用場。新聞群組可以快速地發佈知識，然而若

要由群組中搜尋某一特定知識之內容則較不容易。至於一般作業系統內建之分享系統，如微軟 Windows 內的網路芳鄰。優點在於使用方便。然並不是所有企業都百分之百使用 Windows 系統，若加上其它如 Unix、Linux、Netware 平台的系統，則會增加許多整合上的困難。

企業資訊入口網站（Enterprise Information Portal, EIP）的觀念延伸自網際網路的入口網站。它的目的在以單一的資訊窗口，讓企業可以協調決策支援、協同合作、知識管理以及企業應用整合等目標，以提升整體的營運績效[1]。企業知識入口網站在企業中扮演著所有員工儲存、搜尋、分享、使用知識的單一入口。藉由知識入口網站，員工可方便存取企業內部所有的資源及文獻，藉之以解決自己成長以及工作之需求。如同前文所述，在企業與企業間交易更頻繁、往來更密切的今日，各個企業自身的知識入口網站若能互相分享資訊，對於具有合作關係的企業而言，會有無比的助益，也提供了企業在目前變化複雜的環境下更高的適應性及競爭力。

分散式的文獻儲存方式，是現今許多企業在隨自身規模成長時所難以避免的，此種現象的存在對於使用者而言皆會產生不必要的困擾。就使用者的角度來說，能從任何地方、從任一個單一的進入點檢索到各地的知識文獻能夠減少查詢文件的時間，進而提升工作效率；就企業的角度來說，將分散於企業各部門或各營運地區等的知識文獻以有效的方式整合呈現給使用者對企業的經營將有莫大的幫助。

本研究的動機便在於提出一套可讓企業更易進行知識交換的機制，以解決在分散式的環境之下，同企業內部門之間知識文獻的傳遞及取得的問題。

1.2 研究目的

目錄服務的發展由來已久。而自目錄服務的存取協定：Lightweight Directory Access Protocol（LDAP）被提出至今也已經過了許久的時間，LDAP 標準已不斷

進化，企業所推出的產品亦已大幅強化其功能。雖然現階段在台灣嘗試將目錄服務納入企業基礎建設的公司仍並不多，但自搭載了 Active Directory 的 Windows 2000 Server 上市以來，目錄服務已逐漸吸引企業主管、軟體開發廠商的目光。目前在市場上已有 Microsoft、Novell、Sun、IBM、Oracle 等軟體大廠推出的目錄伺服器產品，競爭者眾，顯示目錄服務在企業市場將持續發熱。

在本研究中，目錄並不儲存任何知識文獻的全文，其所擔當的角色是一個元資訊（metadata）的儲存者（repository），目錄中記載了使用者及企業知識文件的元資訊。藉由整合 Web 技術，本研究提供一個可讓使用者經由企業內不同部門之入口網站進行知識搜尋、分享的機制。並實際建構出一個系統雛型，使企業間不僅能達到分享知識的需求，更能簡化對於資訊的管理，進一步地擴大企業資訊入口網站的能力，對組織的知識管理形成助力。

目錄服務可以說是一種特別的儲存庫，可用來儲存各式各樣的資訊。與一般使用者常用的資料庫不同的，目錄服務提供了獨特的功能及特性，使其在某些領域的作業上有良好的性能表現。目錄服務的特性簡述如以下幾點[10]：

1. 目錄服務具有相當的彈性：目錄可儲存許多不同種類的資料，使用者可依據標準儲存對應的資料，也可自訂所欲儲存資料的型態。

2. 目錄服務的設計是用在某些特殊的情況：舉例而言，當更新目錄資料的頻率比起讀取目錄中資料的頻率來得低的時候，便是使用目錄服務的好時機，這也是為什麼許多談論目錄服務應用的書籍、雜誌都會以電話簿、通訊錄之例做為目錄服務最典型的應用。目錄服務對於資料的搜尋及讀取做過最佳化，在資料的寫入上則較不擅長，此種特點使得目錄服務適合應用在資料的讀取及搜尋較多而寫入的機會較少之特殊情況。

3. 目錄服務是分散式、階層式的設計：目錄服務是以一個總合的、階層式的命名模式來支援分散式的設計。目錄服務內的資料是以階層式的方式儲存的，有別

於一般的關聯式資料庫，階層式的設計對儲存及展示具階層結構之資料有特別的優勢。

4. 目錄服務所儲存之資料為資料項（entry）導向的資料：如員工資料，每個資料物件之下會有數個屬性值，員工物件所包含的屬性值就可能包括員工姓名、電話、地址、生日等等，而這些屬性值皆可以擁有一個以上的值。

5. 目錄服務具有內建的存取控制機制：目錄服務的存取控制可以下達到物件的屬性層級，這樣的控制可以讓管理者決定一物件是否開放給使用者使用，或是物件下的屬性是否可讀可寫等等。

綜合以上所述，本論文『以目錄服務為基礎達成企業間知識入口網站的知識交換之機制』的研究範圍定義如下幾點：

1. 瞭解目錄服務的源起與發展
2. 研究目錄服務的分類、定義及發展現況
3. 了解目前目錄服務主流的存取協定 LDAP
4. 根據目錄服務的特性，設計可用以支援知識入口網站間知識文獻交換的機制
5. 最後針對本論文不足之處提出未來改進及後續可供繼續研究的方向

1.3 研究方法與步驟

如前文所提，本研究目的在於為企業的知識入口網站間設計一套知識分享的機制。透過此一機制，使企業的知識入口之間能更有效率地存取及定位到對方之知識文獻。為了設計可用的系統雛型，作者首先閱讀國內外有關目錄服務的文獻資料，蒐羅各大網站針對目錄服務應用狀況的最新發展，並藉之以思考、設計適合企業運用的系統。本研究規劃了幾個研究步驟如下：

第一階段：確定研究題目、動機及目的

目錄服務是近幾年才逐漸興起的一項技術議題，其對企業的管理工作有很大的幫助，國外的軟體公司如 Microsoft、Novell 等對其發展不遺於力，而

在國內，目錄服務的相關應用尚未成熟，且較欠缺支援，可說尚在起步的階段。

第二階段：收集相關資料、文獻探討

大量收集及研讀本研究主要會接觸到的技術及觀念等等之相關論文，包括目錄服務、LDAP 等相關領域，目的在於了解目錄服務發展狀況、優劣處所在、及運作方式，進而能進行本研究系統之設計工作。

第三階段：設計系統流程

藉由先前所提之研究基礎，利用目錄服務的特性，規劃設計企業內知識入口網站間知識交換的流程。

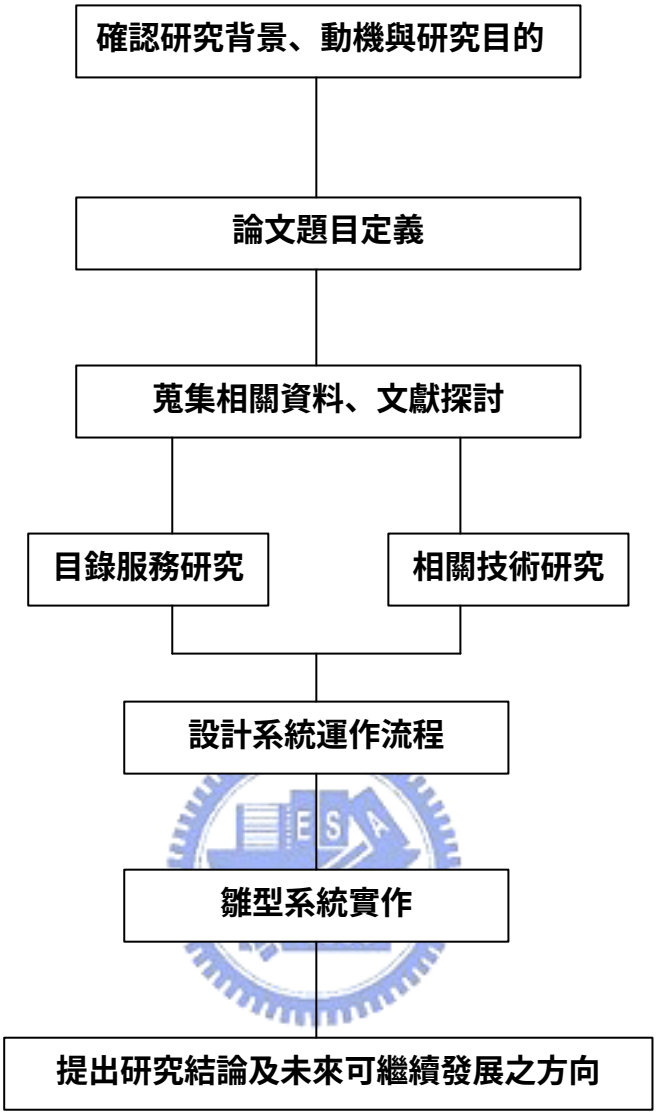
第四階段：雛型系統實作

針對設計出的流程，實作出本研究的雛型系統，藉之可展現本研究所欲表達的概念。

第五階段：提出結論及後續可能發展方向

提出本研究之結論，說明本研究所受到之限制及不足之處，並提供未來可供繼續研究之方向。

本論文之研究方法與流程如圖【1-1】所示。



圖【1-1】研究方法架構與流程圖

1.4 論文架構

以目錄服務所設計成的企業知識交換系統為主題，本論文的結構共分為六章，各章討論之內容分別簡述如下：

第一章：緒論。

本章描述此研究的背景、動機、目的，並說明研究之流程及論文組織架構。

第二章：目錄及目錄服務介紹。

收集、研讀有關於目錄服務、目錄服務產品相關文獻作品。本章討論之重點在於市場上目錄及目錄服務的定義、分類、發展狀況、及其相關組織等等層面。

第三章：目錄服務的存取協定。

了解目錄服務的源由之後，本章進一步探討對目錄服務最重要的角色：存取協定的部份。本章主要介紹現今目錄服務的主流存取協定 LDAP，主要分別以八個 LDAP 之組成元素進行介紹與討論。

第四章：以目錄服務為基礎之知識藍圖系統分析與設計。

根據前二章所述之目錄服務為基礎，本章提出論文的設計概念。敘述本研究之開發環境、系統架構、設計、以及功能等等，並加入本研究所提之知識藍圖與網路常用之搜尋引擎有關之比較與討論。

第五章：企業知識藍圖雛型系統展示。

本章進入實作的部份，根據前一章所設計之架構與概念實作一系統雛型。除將本研究所定義之知識文獻元資料於知識藍圖中的綱要定義列出之外，並說明與展示系統達成之功能。

第六章：結論及未來研究方向。

本章將對本研究所得之成果做出結論，並對本研究所受到的限制及將來可供繼續進行的研究方向提出建議。

第二章 目錄及目錄服務介紹

2.1 目錄服務的歷史淵源

在過去，網路尚不如現在一般普及，主要的應用仍侷限在學術實驗室或企業的區域網路中。由於網路之規模不大，此時的網路或系統管理者普遍利用手動輸入的方式更新、管理網路上新增的節點資料。在這之後，網路節點增加的速度逐漸增快，系統管理員漸漸無法親自到每部機器去維護類似資料，因而發展出利用一個簡單的表格，記載所有有連接至網路之機器的名稱及位址，並將此份表格複製至每一部機器上之方式更新網路節點資料。此種方式雖暫時解決問題，但很快地，節點的資料越來越多，使這些表格變得極端龐大，浪費許多寶貴的資源。

早期的目錄即是應類似上述問題而生的解決方法。其主要的目標在於管理電子郵件、網路節點的名稱轉譯、或是區域網路電腦資源（如印表機、使用者資料等）相關的問題[3]。

2.2 目錄服務的定義及分類

目錄（directory）並非新的名詞，所謂的目錄，以日常生活的觀點來看，早已深入我們的生活當中，最常用來比喻目錄的例子便是如電話簿（yellow pages）或是通訊錄等等，這些目錄蒐集、記載了許多不同的個體（如：人名）的相關簡要資訊（如電話、地址等），有了這些目錄的協助，我們可以便利地找到欲查詢的資料。而由資訊科技中的一項技術這個角度來看，目錄可被視為一個型式較特別的資料庫，用以協助企業中人員透過電腦查詢各項資訊。

目錄服務顧名思義，是由目錄衍生而出的服務。對於目錄服務，業界或學界並未有標準的定義。以下列出目前正在發展目錄服務的學者、廠商所提之定義，提供一個關於目錄服務較全面的解釋，請見表【2-1】：

表【2-1】目錄服務之定義

定義者	定義
Microsoft	目錄服務提供一個儲存目錄資料的方法，並讓網路上的使用者及管理者能利用到此一資料[12]。
Sun	目錄服務是一個儲存企業和使用者資訊的軟硬體流程的集合，使用者能透過服務利用到這些資訊。目錄服務的組成包括了至少一部目錄服務伺服器以及一個以上的用戶端目錄程式。用戶端的程式能夠存取姓名、電話號碼、地址、以及其他儲存於目錄內的資料[6]。
Oracle	目錄服務是一種具有彈性及特殊目的的資料庫，用來讓應用程式能夠儲存及萃取記錄導向（entry-oriented）的資訊[10]。
IBM	目錄服務是結合了存取方式及相關服務等資訊的儲存庫，它儲存了資源的位置及其他像使用者和伺服器的詳細資訊[8]。
Beth Sheresh & Doug Sheresh	目錄服務以階層式的機制來組織及管理資訊，並以名稱關聯的方法取得資訊[11]。

由各家對目錄服務的定義中可以得知，雖然對目錄服務並無一致標準的解釋，但可看出一個完整的目錄服務是由儲存庫、伺服器、客戶端程式、以及存取協定所構成的，缺一不可。這些組成元件如下圖【2-1】所示：

圖【2-1】目錄服務組成元件

在目錄服務中，最重要的可以說是它的存取協定了，因為它負擔了將客戶端的各種請求送至伺服端的工作，在過去，主要使用的存取協定是 X.500，X.500 協定頗為完整，但由於某些缺點，其並不適合使用在現今的網際網路上，而後專家發展出 LDAP，目前目錄服務存取協定的主流便是 LDAP。關於 X.500 及 LDAP 的關係以及 LDAP 協定的運作方式等方面之討論將於第三章：目錄服務的存取

協定中作完整的解釋。

經過長時間的發展，國外組織在目錄的實作上已有成果。研發目錄服務之廠商在各個領域都有相關實作的產品。Beth Sheresh及Doug Sheresh於其著作[11]中，將目錄服務相關的產品分為如表【2-2】所述幾類：

表【2-2】目錄產品之分類

類型	用途
網路專用目碌	主要用來支援網路作業系統的管理工作。
應用軟體專用目錄	這種目錄裡儲存的是該應用軟體的使用者資料或是軟體的相關設定等等。
特殊用途目錄	此種目錄只用來支援單一用途，像是 DNS 網域名稱系統便是一例。
泛用目錄	支援各種目錄服務的功能
元目錄	此種目錄用來搜集、整合各不同目錄間的資料。

總的來說，目錄服務的軟體大致上可以分為泛用型目錄產品及特殊應用型目錄軟體，泛用型的產品不專注於任何特殊領域，可以應用在所有的情況之上，而特殊用途的目錄服務產品則專精於單一領域的應用。

2.3 相關組織

目錄服務存取協定 LDAP 標準的發展主要是由 IETF（The Internet Engineering Task Force）主導。目前在 IETF 共有三個小組負責推動制定 LDAP 標準的工作，分別是 ldapbis、ldapext、及 ldup。ldapbis（LDAP revision）小組主要負責將 LDAP 相關的 RFCs 推動為網際網路上的標準，其目前正在進行的是 ldapv3 的修改動作；ldup 小組則負責 LDAP 儲存庫的複製（replication, duplication）及更新（update）等 RFC 的撰寫；ldapext（LDAP extension）小組則制定一些延伸性的 LDAP 相關規格，如搜尋結果的呈現方式、或是搜尋重導向（referral）等等。

另一個與目錄服務的推廣息息相關的組織是 OASIS，OASIS 的全名為 Organization for the Advancement of Structured Information Standards。OASIS 是一個非營利性的機構，其成立運作的目的主要在於推動制定電子商務的標準。對於許多 XML 及 Web Service 相關的標準，OASIS 是主要的推動者，在目錄服務方面，OASIS 制定了 DSML 規格，DSML 全文為 Directory Service Markup Language。目前 DSML 最新的版本為 v2，DSMLv1 主要提供一個以 XML 文件來呈現目錄中資料的方式；DSMLv2 則更進一步，提供以 XML 文件來表示目錄的檢索及更新等命令的方法[4]。

1999 年由各大廠商如 Novell, IBM, Oracle 一同成立的 DIF：Directory Interoperability Forum 是為促進不同廠商所實作出產品的互通性所組成，在推動廠商支援如 LDAP 之類的公開目錄標準方面著力甚多。

2.4 現況分析

儘管目錄服務相關技術發展已久，不過其真正受到矚目是在微軟發表 Active Directory 之後。由於目錄服務對企業來說，將會是一項重要的基礎建設，目前在國外，建置目錄服務的相關產品可說是百家爭鳴，而廣為國人所熟知的大廠皆有推出相關的產品。例如微軟的 Active Directory、Novell 的 eDirectory、昇陽微電腦的 Sun One Directory Server（前身為 iPlanet Directory Server）、IBM 的 SecureWay Directory Server（現已改名為 IBM Directory Server）、Oracle 的 Internet Directory 等等。另外也有 OpenLDAP 這一套可免費使用的開放源碼軟體。選擇眾多。

對企業來說，佈署、整合公司中的目錄服務架構這一項基礎建設可說是一項浩大的工程，因此其推行的速度並不快，市場、產品的領導者也尚在渾沌的狀態。目前各家廠商的產品在市場上皆各佔一席之地，難以確定誰將會是目錄服務產品中最後的勝出者。

第三章 目錄服務的存取協定

目錄服務的存取協定目前主要為 LDAP。在 LDAP 被廣為採用前，X.500 是主流的協定。本研究主要利用的是 LDAP 來建置系統，因此不詳述 X.500 的運作細節，而會以較多的篇幅解釋 LDAP 協定。

3.1 X.500

X.500 是 1988 年由 CCITT（Comite Consultatif International Telephonique et Telegraphique or Consultative Committee on International Telephony and Telegraphy）與 ISO 所共同制定。CCITT 目前已改名為 ITU (International Telecommunications Unions)，ITU 曾制定許多知名的標準如 X.25 等等，是一個國際性的機構。X.500 構基於 OSI 之上，制定的目的在支援分散式目錄服務。事實上，X.500 為一組協定的集合，見表【3-1】：

表【3-1】X.500 協定集

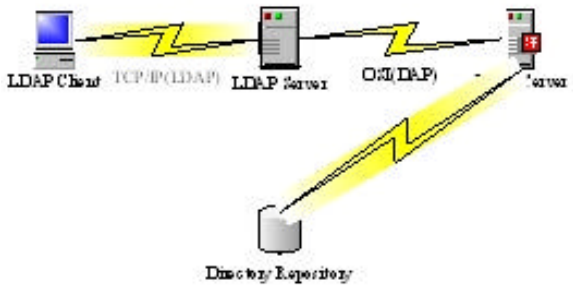
協定	描述
X.500	Overview of Concepts, Models, and Services
X.501	Models
X.509	Authentication Framework
X.511	Abstract Service Definition
X.518	Procedures for Distributed Operations
X.519	Protocol Specifications
X.520	Selected Attribute Types
X.521	Selected Object Classes
X.525	Replication
X.530	Use of Systems Management for Administration of the Directory

X.500 提供了一個稱為目錄存取協定（Directory Access Protocol，DAP）的傳輸規格以及一個具有延伸性的資訊架構，以支援讓目錄能夠儲存任何型式的資訊。雖然 X.500 定義了完整的目錄服務規格，然而由於 DAP 需使用到 OSI 的 7

層來運作，實作時過於龐大且複雜，對於運算能力較差的電腦所需的資源量極多。由於過於龐大，同時也難以在網際網路上運行。因為這個原因，才有後來 LDAP 的發展。

3.2 LDAP

許多文獻都不約而同的指出，有許多大眾誤以為 LDAP 就是目錄服務，此種誤解其來有自。事實上，LDAP 一開始單純只是一個存取斜定，設計來讓運算能力比較差的電腦擁有與 X.500 目錄溝通的能力，如圖【3-1】[3]所示：



圖【3-1】LDAP 初始設計

圖【3-1】中的 LDAP Server 所擔任的是類似閘道器的角色，負責把由客戶端傳來的 LDAP request 的要求轉換為 X.500 request，之後再將其送至 X.500 伺服器，並將傳回的訊息轉換為 LDAP 協定格式。LDAP 此種設計讓一般運算能力較差的電腦可直接透過 LDAP 存取後方 X.500 目錄。而因 LDAP 所使用的協定為 TCP/IP，而不是較為複雜的 OSI，大大地減少運算的成本與實作上的困難度。由於 LDAP 漸漸受到歡迎，此種模式慢慢演變為 LDAP Server 本身便具有目錄，能夠自己直接與目錄溝通，消除了經過 X.500 伺服器這一道手續。因為不再需要用到 OSI 協定堆疊，節省更多的運算資源。LDAP 自發表至今已經是第三版（LDAPv3），表【3-2】及表【3-3】分別是目前 LDAPv2 與 LDAPv3 相關的 RFC 列表：

表【3-2】LDAPv2 RFC 列表

LDAPv2	
RFC Number	Name
1777	LDAPv2
1778	The String Representation of Standard Attribute Syntaxes
1779	A String Representation of Distinguished Names
1959	An LDAP URL Format
1960	A String Representation of LDAP Search Filters

表【3-3】LDAPv3 RFC 列表

LDAPv3	
RFC Number	Name
2251	LDAPv3
2252	LDAPv3:Attribute Syntax Definitions
2253	LDAPv3:UTF-8 String Representation of Distinguished Names
2254	The String Representation of LDAP Search Filters
2255	The LDAP URL Format
2256	A Summary of the X.500 User Schema for use with LDAP

LDAP 創造者之一的 Tim Howes 在 1999 年發表的文章:"LDAP: Use as Directed"[5] 中指出 LDAP 是由八個層面建築而成，而基本上，關於 LDAP 的相關討論，大概都不出於這八個層面之外。以下就是針對 LDAP 這八個層面的詳細探討：

3.2.1 資訊模式（Information Model）

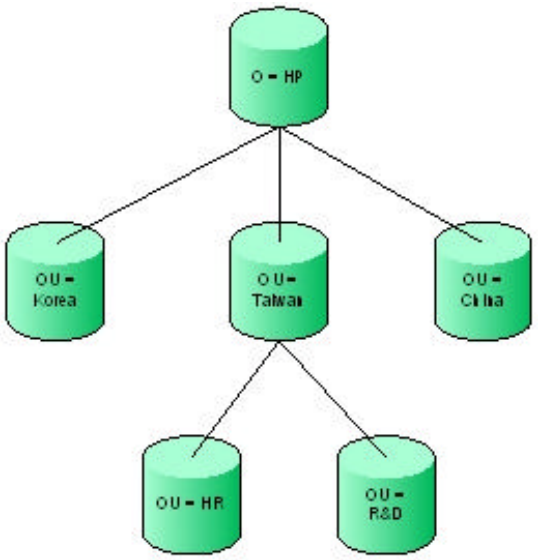
LDAP 的資訊模式決定目錄中儲存什麼樣的資料以及這些資料的行為。LDAP 是由 X.500 衍生而來，在這一方面大部份承襲了 X.500 的規格。LDAP Information Model 的組成大致有以下幾項：目錄資訊樹、資料項、屬性、物件類別等。在 LDAP 的目錄中，資料的結構是階層式的樹狀結構，而非是一般常見的關聯式結構。此種結構稱為目錄資訊樹（directory information tree，DIT）。物件，或稱資料項，乃是 LDAP 中用來表示企業環境中任一物件的資料項目。任何一筆資料項，可能代表一部印表機、一個使用者、或是一個安全政策。一筆資料項是由多個屬性-屬性值對（attribute-value pair）組成的，物件類別（object class）

則是物件的類別名稱，如 Susan 可能是屬於 Person 這個物件類別。所有的物件都會有所屬的物件類別，以決定其擁有的屬性值。舉例來說，屬於 document 這個物件類別的物件便定義了此物件必須要有 documentIdentifier 這個必要屬性，另外還可以有 cn（common name）、abstract、description、documentAuthor、documentAuthorCommonName、documentAuthorSurName、documentLocation、documentPublisher、documentTitle、documentVersion 等等的選用屬性。屬性或是類別大部份都可自訂, 企業在開發適合自身環境的目錄服務時可善用自訂屬性及類別的彈性。

物件類別本身分別有結構（structural）、抽象（abstract）、及輔助（auxiliary）三種型態。結構的物件類別是最常見的物件類別型態，用來實體化(instantiate)物件，抽象型態則類似物件導向觀念內的抽象類別（abstract class），抽象型態的物件類別是不能用來將物件實體化的，但可以定義一個抽象形態的物件類別，再據之以創造另一個實際可用的結構物件類別。輔助型態的物件類別主要不使用在實體化物件的用途上，而是用在提供擴充的功能給其它的結構物件類別，其可附在已被結構物件類別實體化的資料項，擴充該資料項可用的屬性。

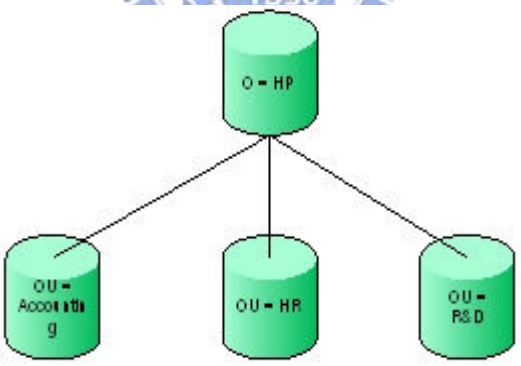
LDAP 目錄中的屬性可以分為兩類，一為使用者屬性、另一則為系統操作屬性。使用者屬性顧名思義為一般使用者或管理者在使用系統時會瀏覽到的屬性；而操作屬性則為目錄內部運作時所使用的屬性。操作屬性一般都是由目錄自己管理，使用者不會干涉到它的運作。

目錄資訊樹是目錄中資料集合後所可展現出的樹狀階層。目錄資訊樹的規劃（planning）及分割（partitioning）極為重要，亦有許多不同的策略可用以處理這個問題。例如可以以地理位置來作目錄樹的設計，其型態應如下圖：



圖【3-2】以地理位置為主的 Partitioning

此圖的組織 HP 之下是以地理位置為分區，分別是 Taiwan、Korea 及 China，而在各區域之下才是部門。圖中 HP 前的 O 為 Organization 的縮寫；OU（organizational unit）則指部門，此二者為 LDAP 目錄中常見的屬性。如同前述，亦有以部門來做 DIT 的分割 partitioning 策略，如下圖：



圖【3-3】以部門為主的 Partitioning

這裡所提的是以地理位置為主及以部門為主的規劃法。而設計的方法很多，如以專案小組為主的設計策略、以企業內硬體設備為主的設計策略等等，一切都依企業的規模、經營方式、商業流程等等考量自行規劃設計。

3.2.2 LDAP 綱要（Schema）

LDAP 綱要在 LDAP 伺服器中擔任的角色便如資料庫中的綱要一般。LDAP

綱要定義了特定伺服器中所能存放的實際資訊以及這些資訊與真實世界中物件的關聯性。綱要是屬性的型態與物件類別定義的集合，定義 LDAP 伺服器可使用的屬性及其規格資訊，可說是所有 LDAP 存放資訊的一套準則。LDAP 伺服器可根據綱要來決定搜尋資料的過濾規則，亦可規範一筆資料項內可以有的屬性資料。舉例來說，以下為 IBM Directory Server 5.1 綱要中對 inetOrgPerson 的定義：

> objectclasses=(2.16.840.1.113730.3.2.2 NAME 'inetOrgPerson' DESC 'Defines entries representing people in an organizations enterprise network.' SUP 'organizationalPerson' STRUCTURAL MAY (audio $ businessCategory $ carLicense $ departmentNumber $ employeeNumber $ employeeType $ givenName $ homePhone $ homePostalAddress $ initials $ jpegPhoto $ labeledURI $ mail $ manager $ mobile $ pager $ photo $ preferredLanguage $ roomNumber $ secretary $ uid $ userCertificate $ userSMIMECertificate $ x500UniqueIdentifier $ displayName $ o $ userPKCS12))

雖看似複雜，實際上卻很容易了解。最初的 objectclasses=表示這是一筆關於物件類別的定義資料，2.16.840.1.113730.3.2.2 為這一個物件的物件識別碼（object identifier，OID），物件識別碼是用來辨識物件的唯一編號，NAME ‘ inetOrgPerson’ 指的是這個物件在目錄中顯示的名稱為 inetOrgPerson，DESC 則是對這個物件類別的描述。跟在描述之候的關鍵字 SUP 指的是 inetOrgPerson 這個物件類別的上層類別（super class），此例中，其上層類別為 organizational person，接下來的 STRUCTURAL 關鍵字表示這是一個 structural 類別，MAY 後面所接的一大串文字則為 inetOrgPerson 所可以使用的選用屬性。

在 LDAPv3 中加入了一個 subschema 物件，用以讓具備不同綱要的目錄（即不同廠商所實作的目錄服務產品）能夠更有效地溝通。subschema 這個物件存在每個

目錄的子樹中，每個目錄的子樹中 subschema 物件定義了其子樹部份的綱要資料。利用 subschema 物件，目錄便可以定義其子樹的綱要，並與其他目錄的綱要交換資料，此一動作發生在在兩個目錄欲交換物件資訊之前。Subschema 物件是個 auxiliary 物件，包含了以下七個屬性：objectClass、attributeTypes、dITStructureRules、nameForms、dITContentRules、matchingRules、matchingRuleUse[3]。

3.2.3 命名模式（Naming Model）

命名模式定義了如何參考以及如何組織目錄資訊的命名慣例。LDAP 中，資料項與資料項之間的關聯是階層性的，舉例來說，最上層的資料項通常代表一個公司、國家、或是省份等等。命名模式就在規範這些上下層關係如何展現。比如，我們可以以 DN: cn=Wu Yung Chou, ou=iim, o=NCTU, c=Taiwan 這樣的一筆資料來描述一個在台灣交大資管所內的一位學生。以這樣的一筆資料項來描述可明顯看出其階層關係。整筆資料是以 DN 為開頭，表示其為目錄中一筆資料記錄的識別名稱，另外，cn=Wu Yung Chou 稱為 RDN（relative distinguished name），一筆 DN 是由眾多 RDN 所組合而成的，如上述 DN 便是由 cn=Wu Yung Chou、ou=iim 等等 RDN 所組合而成，RDN 可以看做一個一個獨立的組成元件，代表了目錄中的一個節點。與資料庫的欄位不同，RDN 允許多值，多值的 RDN 會以加號相隔，如 ou=esc+o=iim。除了這些命名的原則之外，命名模式亦定義了在 LDAP 中 URL（universal resource locator）的表示方式，LDAP 的 URL 可讓使用者透過命令列的工具（command line tool）或是瀏覽器（web browser）的方式直接存取到 LDAP 伺服器中的目錄資料。在 LDAP 中的 URL 表示方法為：

ldap://<host>:<port>/<dn>[?<attributes>[?<scope>?<filter>][<?extensions>]]

<dn>即為前述所討論到的識別名稱，attributes 這一欄為使用者所要 LDAP 伺服器回傳的屬性值，若這部份使用者沒有指定，則伺服器會回傳物件的全部屬

性。Scope 定義了使用者所要查詢的範圍，LDAP 定義了三類的範圍，一是 current entry、一是 one-level、最後是 sub-tree。Current entry 範圍指的是搜尋目前所指定的資料項，one-level 為指定搜尋目前資料項的子資料項，sub-tree 則是尋找目前資料項之下的整棵子樹。Filter 為過濾選項，是 LDAP 的一個強大功能，利用過濾條件的設定，使用者可指定各種過濾條件，更有效地找到自己所需的資料，若未指定過濾條件，則預設會以*為使用者所查詢、請求（request）的值，即是指全部的值；最後的 extensions 則是留給 LDAP 的擴充功能來使用。舉例來說，ldap://localhost:389/ou=esc lab, o=iim?telephone number?sub 這樣一個 URL 所描述的，就是尋找在本機伺服器中，iim 的 esc lab 子樹下所擁有的 telephone number 屬性植，伺服器在接收到由客戶端傳來的請求之後會回傳 ou=esc lab, o=iim 下所定義的所有 telephone number 值。

3.2.4 安全模式（Security Model）

LDAP 對安全性的支援可以由幾個方面來看，一是身份驗證（authentication）、一是存取控制（access control），再來則是資訊真確性（integrity）及私密性的保護（confidentiality）。在實作上，各家廠商所開發的系統安全性並不完全一樣。如 IBM Directory Server 在身份認證方面便支援了 anonymous authentication、basic authentication、及 CRAM-MD5 SASL 等三種認證方式；資訊真確性及私密性的保護則是由支援 SSL 達成。存取控制則是以權限控制表列（access control list，ACL）的方式達成。利用 ACL，管理者可以對目錄中的資料項目的存取權限做精確的設定。下列為 IBM Directory Server 的權限控制表列 BNF 表示法：

<aclEntry> ::= <subject> [":" <rights>]

<aclPropagate> ::= "true" | "false"

<ibm-filterAclEntry> ::= <subject> ":" <object filter> [":" <rights>]

<ibm-filterAclInherit> ::= "true" | "false"

<entryOwner> ::= <subject>

<ownerPropagate> ::= "true" | "false"

<subject> ::= <subjectDnType> ’:’ <subjectDn> |

<pseudoDn>

<subjectDnType> ::= "role" | "group" | "access-id"

<subjectDn> ::= <DN>

<DN> ::= distinguished name as described in RFC 2251, section 4.1.3.

<pseudoDn> ::= "group:cn=anybody" | "group:cn=authenticated" |

"access-id:cn=this"

<object filter> ::= string search filter as defined in RFC 2254, section 4

(extensible matching is not supported).

<rights> ::= <accessList> [":" <rights>]

<accessList> ::= <objectAccess> | <attributeAccess> |

<attributeClassAccess>

<objectAccess> ::= "object:" [<action> ":"] <objectPermissions>

<action> ::= "grant" | "deny"

<objectPermisssions> ::= <objectPermission> [<objectPermissions>]

<objectPermission> ::= "a" | "d" | ""

<attributeAccess> ::= "at." <attributeName> ":" [<action> ":"]

<attributePermissions>

<attributeName> ::= attributeType

(OID or alpha-numeric string with leading

alphabet, "-" and ";" allowed)

<attributePermissions> ::= <attributePermission>

[<attributePermissions>]

<attributePermission> ::= "r" | "w" | "s" | "c" | ""

<attributeClassAccess> ::= <class> ":" [<action> ":"]

<attributePermissions>

<class> ::= "normal" | "sensitive" | "critical"

一個 aclEntry 是由 Subject 加上適當的 rights 所組成的，而 Subject 是由 DN Type 跟 DN（distinguished name）組成，DN Type 有三種，一是 access ID，一為 Group，另一則是 Role。因此，一個 subject 可能長的像這樣子：access ID：cn=Azuritul, o=IIM；或是 group: cn=adminGroup, o=IIM。在這裡 Role 與 Group 很相似，然有稍許不同，若將某個 Role 指派給某個物件（員工、電腦等等…），則通常會有相對應的權限給予這個物件，以讓物件能完成與其被指派的 Role 的相關工作；而對於 Group 的指派，我們不會有這種類似的假設。

Pseudo DN 是用來表示一群擁有同樣性質的 DN，主要是為了使用及管理上的方便而設定的。在 IDS5.1 裡定義了三個 Pseudo DN。第一個是 group:cn=anybody，這個 Pseudo DN 所指的是全部的使用者，所有的使用者都會自動屬於此一個群組；第二個 Pseudo DN 是 group:cn=authenticated，這是任何已通過身份驗證的使用者；最後，是 access-id:cn=this。

存取控制權（rights）的設定可以套用在物件上，亦可以只套用在物件的單一屬性上。而我們也可以不設定任何存取控制權的值，如此可表示不允許對目標物件的所有操作權限。存取控制權的組成包括了三個部份：Action、Permission 及 Access Target。

在 Action 方面，有兩種 Action 可以指派：Grant 與 Deny。顧名思義，Grant 表示給予權限；而 Deny 表示拒絕給予權限。

Permission 方面則可分兩方面來談。前面曾提到，Rights 可套用在物件亦可套用在屬性之上，對於物件的 Permission 是對應到 LDAP 定義的操作的。下表清楚地表現 Permission 與操作之間的關係[7]：

表【3-4】LDAP Operation 與對應所需的 Permission

LDAP 操作	需要的 Permission
ldapadd	add
ldapdelete	delete
ldapmodify	write
ldapsearch	search、read
ldapmodrdn	write
ldapcompare	compare

屬性方面的 Permission 則有讀、寫、搜尋、比對等選項可以選擇：read（r)、write（w)、search（s)、compare（c)。屬性本身有三種安全類別，分別是一般（normal)、敏感（sensitive）及極機密（critical)，在新增或編輯屬性時可指定屬性的安全類別。舉例來說，一個 person 物件的 commonName 屬性可能會設計為一般層級的安全類別，而密碼的屬性則可能設計為屬於極機密之安全類別。

EntryOwner 指的是一筆資料、或一個物件的擁有者，物件的擁有者對於物件擁有完整的權限，也就是說，一旦讓某個人是某個物件的擁有者，則此人除了可對物件做所有的新增、修改刪除等等動作之外，還可以修改、管理此物件的存取控制列表，控制其他人對此物件的存取權限。

以實例說明可以更清楚，以下所列是目錄中 cn=Wu Yung Chou, ou=iim, o=nctu, c=Taiwan 的存取控制列表資料項（ACL entry）之例。仍要注意的是，不同開發廠商所發展的產品在這類資料項的描述上會略有不同。

entryOwner: access-id: iimManager, ou=iim, o=nctu, c=Taiwan

ownerPropagate: TRUE

aclPropagate: TRUE

aclEntry: role: cn=admins, o=nctu, c=Taiwan,

normal:rwcs:sensitive:rwsc:critical:rsc

aclEntry: access-id: cn=Wu Yung Chou, ou=iim, o=nctu, c=Taiwan

object:ad:normal:rwsc:sensitive:rwsc:critical:rsc

aclEntry: group:cn=Anybody:normal:rsc

entryOwner 指的是 cn=Wu Yung Chou, ou=iim, o=nctu, c=Taiwan 這筆資料項的擁有者，資料項的擁有者有較高的權限，其可修改此資料項的存取控制列表，對該物件也有完全的權限。Propagate 類似繼承的觀念，在此例中，ownerPropagate 及 aclProgate 皆設為 true，亦即表示，將這個物件的擁有者和存取控制列表往下傳承，若此物件之下的物件沒有明確設定擁有者或存取控制列表時，會往上找到此物件的上層物件之擁有者及存取控制列表來使用。在 aclPrapagate 之下有三筆 aclEntry。第一筆為 admin，即管理者。此筆資料項所要表達的是管理者這個角色對於 cn=Wu Yung Chou, ou=iim, o=nctu, c=Taiwan 這個物件的中屬於 normal 層級的屬性值有 r（read)、w（write)、c（compare)、s（search）等等的權利，對於屬於 sensitive 的屬性值也有與 normal 層級相同的權利；對於 critical 的屬性則有 r、s、c（read、search、compare）的權利。

3.2.5 功能模式（Functional Model）

此模式主要規範客戶端如何存取及更新存放於 LDAP 目錄上的資料，以及如何操作這些資料。功能模式定義操作資料的方法，LDAP 提供 11 個基本的操作方式，分別是 Bind、Unbind、Search、Modify、Add、Delete、Compare、Abandon、ModifyDN、Unsolicited notification、以及 Extended operation。其功用整理如下：

表【3-5】LDAP 基本操作方式

bind	允許用戶端與伺服器之間交換鑑別資訊
unbind	終止 LDAP 連線
search	允許用戶端向 LDAP 伺服器提出搜尋資料的要求
modify	允許 client 向伺服端要求更改一筆或多筆 Entry
add	允許 client 向伺服端要求新增一筆或多筆 Entry
delete	刪除伺服端的一筆或多筆資料
compare	允許用戶端對 LDAP 資料庫中的一個 Entry 做比較
abandon	用來讓用戶端在操作意外時提出中斷交談的要求

modifyDN	**更改一筆** Entry **的** DN **資料**
unsolicited notification	**由** LDAP Server **所主動發出，通知使用者系統訊息**
extended operation	**用以給予開發廠商增加新功能**

Bind **操作主要是允許用戶端與伺服器之間交換鑑別資訊，目錄伺服器接到由客戶端傳來之請求後會傳回一個** bind response**，用來表示目前用戶端驗證的狀態。若是客戶端** bind **的請求成功，則會回應一個成功的訊息，若是失敗，則會傳回** operationsError、protocolError、authMethodNotSupported、referral、unavailable **等等資訊。**Unbind **則是用來終止一個** LDAP **的連線。**unbind **並未定義伺服器回覆的訊息。也就是說，當發出** unbind **訊息之後，客戶端會假設連線已終止並停止連線。伺服端在接受到** unbind **訊息時亦會假設發出** unbind **訊息的客戶端已經將連線終止，因此會將此客戶端的請求全部結束，並停止連線。**Search **操作允許用戶端向** LDAP **伺服器提出搜尋資料的要求，搜尋的範圍可包含指定的單一資料項、資料項的子資料項、或是資料項下的整棵子樹。**Unsolicited Notification **是由** LDAP **伺服器所主動發出的訊息，主要是用來發出要使用者注意的事項，在** LDAPv3 **的標準中唯一定義的** unsolicited notification **是斷線的通知，亦即是說在伺服器斷線之前會發訊息通知所有使用者。**Extended operation **基本上則是保留用來讓軟體開發廠商為** LDAP **伺服器增加新功能的一種機制**[11]**。**

3.2.6 協定模式（Protocol Model）

LDAP **協定這方面功能主要規範前述幾項模式如何對應到** TCP/IP **上。**LDAP **預設使用主從架構模式，**LDAP **標準並不定義客戶端的請求及伺服器端的回應需以同步方式進行，其運作可以是非同步的。亦即是說，伺服器端的回應及客戶端的請求可以以任何順序進行，只要最後確認所有客戶端的請求傳出後伺服器都有回應即可。**LDAP **協定規定任何支援** LDAP **協定的目錄皆以** 389 **埠為標準之通訊埠。**LDAP **亦定義了當搜尋物件不在** LDAP **伺服器的名稱空間內時，協定會如何**

繼續進行此次的搜尋。除此之外，在 LDAP 第一版及第二版的協定之定義，並未加入 LDAP 伺服器回覆重導向訊息給客戶端的機制。在最新版的 v3 中，為增進伺服器的執行效能及分散式設計的考量，已將此設計加入。LDAP 伺服器因而減少將客戶端的請求指到其它伺服器的工作負荷。

3.2.7 應用程式介面（Application Programming Interface）

應用程式介面為提供給軟體開發者的一個程式開發工具。目前 LDAP 的應用程式介面有標準的 LDAP C API，可予擅長 C 語言的開發者使用。另外還有 ADSI、PerlDAP、JNDI、PHP LDAP API 等等，ADSI 是微軟開發的專屬 API，主要用於其自家產品 Active Directory 身上，亦可用於存取符合 LDAP 標準的目錄產品。PerlDAP 則是 Netscape 的產品，是以 Perl 語言完成的。JNDI 全名為 Java Naming and Directory Interface，是昇陽電腦所開發的一組讓熟悉 JAVA 的開發者使用的 API。下為 PHP LDAP API 範例：

```
<?php
$ds=ldap_connect("localhost");  // assuming the LDAP server is on this host
if ($ds) {
    // bind with appropriate dn to give update access
    $r=ldap_bind($ds,"cn=root, o=My Company, c=US", "secret");
    // prepare data
    $info["cn"]="John Jones";
    $info["sn"]="Jones";
    $info["mail"]="jonj@here.and.now";
    $info["objectclass"]="person";
    // add data to directory
    $r=ldap_add($ds, "cn=John Jones, o=My Company, c=US", $info);
    ldap_close($ds);
} else {
    echo "Unable to connect to LDAP server";
}?>
```

JNDI API 的使用方式則較為複雜[9]：

```
import java.util.Hashtable;
```

```
import java.util.Enumeration;
import javax.naming.*;
import javax.naming.directory.*;

public static void main(String[] args) {
Hashtable env = new Hashtable(5, 0.75f);
env.put(Context.INITIAL_CONTEXT_FACTORY, Env.INITCTX);
env.put(Context.PROVIDER_URL, Env.MY_SERVICE);

try {
DirContext ctx = new InitialDirContext(env);
SearchControls constraints = new SearchControls();
constraints.setSearchScope(SearchControls.SUBTREE_SCOPE);
NamingEnumeration results
= ctx.search(Env.MY_SEARCHBASE, Env.MY_FILTER, constraints);
while (results != null && results.hasMore()) {
SearchResult si = (SearchResult)results.next();
System.out.println("name: " + si.getName());
Attributes attrs = si.getAttributes();
if (attrs == null) {
System.out.println("No attributes");
} else {
for (NamingEnumeration ae = attrs.getAll();
ae.hasMoreElements();) {
Attribute attr = (Attribute)ae.next();
String attrId = attr.getID();
for (Enumeration vals = attr.getAll();
vals.hasMoreElements();
System.out.println(attrId + ": " + vals.nextElement()))
;}}
System.out.println();
}
} catch (NamingException e) {
System.err.println("Search example failed.");
e.printStackTrace();
}}}
```

3.2.8 LDAP 資料交換格式（LDIF）

LDAP 資料交換格式（LDAP Data Interchange Format）是一種純文字的資料格式，可用來描述存在目錄中的物件資料、匯入或匯出存在目錄中物件的相關描述資料，或是在 LDIF 檔中指定要對目錄中物件之屬性所做的修改，再透過圖形化的管理介面（GUI）或命令列的工具將更新匯入至到目錄中。簡言之，LDIF 之於目錄便如 Doc 之於 Microsoft Word 一般，是其預設流通的檔案格式。LDIF 中的描述須遵循一定的格式撰寫，最基本的格式如表【3-6】所示：

表【3-6】LDIF 基本格式

dn:	<distinguished name>
objectClass:	<object class>
objectClass:	<object class>
<attribute type>:	<attribute value>
<attribute type>:	<attribute value>

Dn 於前述章節便提到，為物件的識別名稱，因此在 LDIF 中是必須指定的，另外也至少須指定一個物件所屬的物件類別。而屬性型態等等，則是當所屬之物件類別需要其定義時才須指定。表【3-7】為上述範例的欄位解釋：

表【3-7】LDIF 基本欄位解釋

欄位	解釋
dn:<distinguished name>	**指示物件的識別名稱**
objectClass: <object class>	**指定資料項所屬的物件類別，物件類別決定了物件會擁有什麼樣的屬性**
<attribute type>	**屬性名稱，屬性名稱決定了可允許使用的屬性值。**
<attribute value>	**屬性值欄位。**

除了上述八個層面之外，還有值得一提的便是目錄的儲存庫：DIB，DIB 為 Directory Information Base 之縮寫，其為儲存所有物件的地方。DIB 為目錄服務中的資料庫。有關要從 DIB 這個資料儲存處中儲存或取出物件的方法是依開發

軟體廠商不同而異。由於目錄服務的運作大部份是用於接受使用者的搜尋、讀取等請求，因此有許多廠商針對其搜尋引擎做最佳化，以提升對客戶端請求的可靠度。LDAP 標準對 Directory Information Base 這方面的功能著力不多，如同步運作、複製、分割等等的設計均未被標準提及。不過為了將來不同廠商所開發的目錄軟體間的相互操作整合（interoperability）之考量，此問題可望在不久之後解決[11]。

第四章 以目錄服務為基礎之知識藍圖系統分析與設計

經由前述章節的介紹，對於目錄服務之特性、組成及架構都有了進一步的認識。由本章開始，本研究將分析開發雛型系統所需的開發工具、系統須擁有的基本功能，同時也對於此系統的架構、流程、及概念做一完整的說明。

4.1 知識藍圖之系統開發環境

本研究為以目錄服務為基礎，為分散於企業不同地區或部門之知識入口網站的資訊交換提出一個可行的雛型系統。本系統的開發工具主要選擇了下列幾項：

一、 目錄服務伺服器：本研究中最重要的角色為目錄服務的伺服器，其負責了系統中大部份物件的儲存、管理、設定等工作。在這項伺服器的選擇上考量了幾個因素：

1. 跨平台性：現今企業中充斥著異質性的作業平台，從微軟的 Windows 95/98、NT/2000/XP、Linux、FreeBSD、UNIX、(IBM AIX)等等，各處理不同目的的工作。為了能夠更方便、簡易地在這些平台上運作，選擇一個具有跨平台功能的伺服器是很重要的。

2. 符合工業標準：LDAP 及 X.500 是目前目錄服務的主要標準。本研究是以 LDAP 協定為主，因此軟體必須要符合 LDAP 的規範。LDAP 是一個開放的標準。使用遵從開放性標準的軟體，軟體開發者能夠較容易得到相關實作上的幫助。

3. 功能性：目錄伺服器需能提供足夠的延伸功能。如本研究前數章所述，LDAP 標準的制訂仍持續在進行中，這個標準仍有許多值得持續改善、進步的空間。舉例來說，目錄伺服器間的複寫（replication）便尚未得到一個一致的標準，但複寫功能又是目錄伺服器不論在處理負載

平衡、備援、分散處理、資訊安全等情況的重要功能，因此這部份各家伺服器業者應有一可供使用的複寫機制，以解決相關問題。

4. 相依性：此處所指的獨立性是指伺服器與其他系統間的相依性，本研究以相依性越低，也就是獨立性高的目錄為首要選擇。主要原因在於獨立性高的目錄伺服器不論在佈署、安裝及設定上均較為簡易，步驟較少，同時在設計系統架構時的考量較為單純。

5. 介面友善度：雖說在許多目錄管理方面的功能，如新增、刪除資料物件等等，使用命令列工具已足夠，但是要更方便地管理目錄伺服器的話，一個容易使用的使用者界面仍是必須的。設計良好的使用者介面不僅能讓初學者快速熟悉伺服器的管理，亦能讓管理者以較符合人類直覺的方式管理目錄中所儲存的資料。同時，對於新增入目錄的資料邏輯結構也會有較清楚的認識。

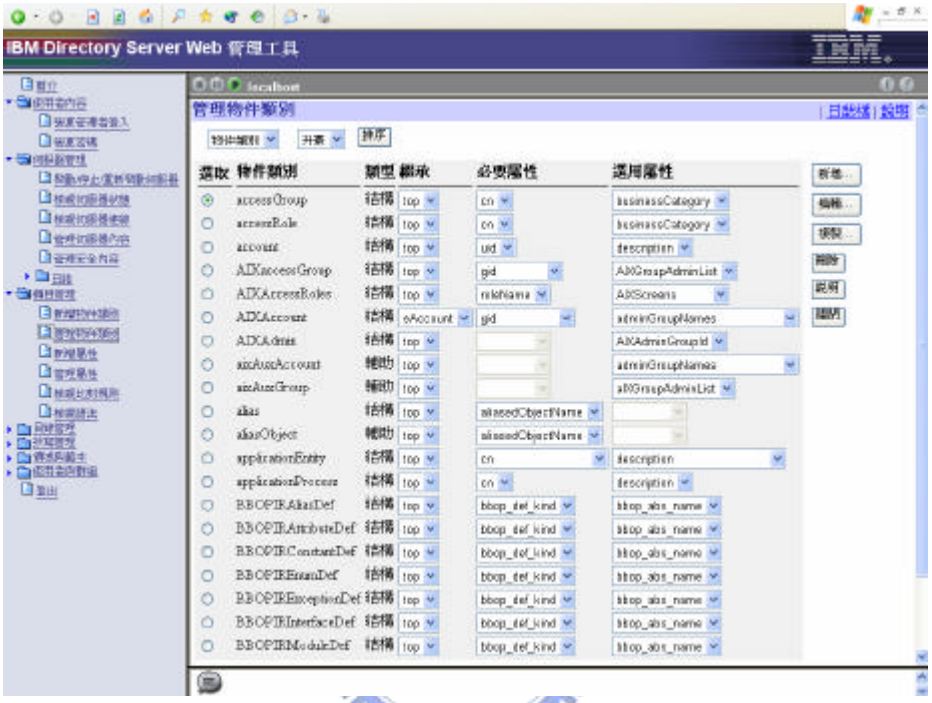
目前各家軟體大廠皆已有目錄伺服器軟體相關產品投入市場，簡要情況可見下表：

表【4-1】軟體大廠開發之目錄服務伺服器

開發者	產品
Sun	SunOne Directory Server 5.1
Oracle	Oracle Internet Directory
Microsoft	Active Directory
IBM	Directory Server 5.1（Former: SecureWay）
Novell	eDirectory
Open Source Group	OpenLDAP

本研究所使用的主要目錄服務伺服器為 IBM Directory Server 5.1。IBM Directory Server 原本稱為 IBM SecureWay Directory，在近一次的發行做了名稱的修改，以後都以 IBM Directory Server（後文皆以 IDS 稱之）做為產品名稱。IDS 是一個以 LDAP 為基礎的目錄伺服器，其支援眾多的平台，包括了 AIX、Linux、

Solaris、Windows、HP-UX 等等，同時它也是免費的。IDS5.1 的後端儲存系統是 DB2 Universal Database 8.1，其具備了一個以瀏覽器為主的 Web 介面管理工具。圖【4-1】為 IBM Directory Server 所附之系統管理工具，系統管理者可藉由此工具進行新增、刪除目錄物件或修改目錄綱要等管理工作。



圖【4-1】IBM Directory Server 5.1 管理介面

二、作業系統：在作業系統的選擇上主要是受到目錄伺服器所支援的平台所影響，由於 IBM Directory Server 支援平台廣泛，因此不須在作業系統上多費心神。本研究使用 Windows XP 做為主要的作業系統。

三、網頁伺服器：在系統的實作中，使用的語言為 JSP，所以須用到網路及應用伺服器以處理客戶端程式的請求，本系統所使用的是 Apache Tomcat。

四、開發語言：IDS 5.1 支援 C LDAP API、JNDI 等等介面，因系統欲使用 Web 的考量，因此選擇 JSP（JAVA Server Pages），開發時可在 JSP 中使用 JNDI 介面與 IDS 進行溝通。

五、瀏覽器：微軟的 Internet Explorer 為目前最廣為接受的網路瀏覽器，在本系

統中，其擔任的角色為客戶端預設的瀏覽工具。

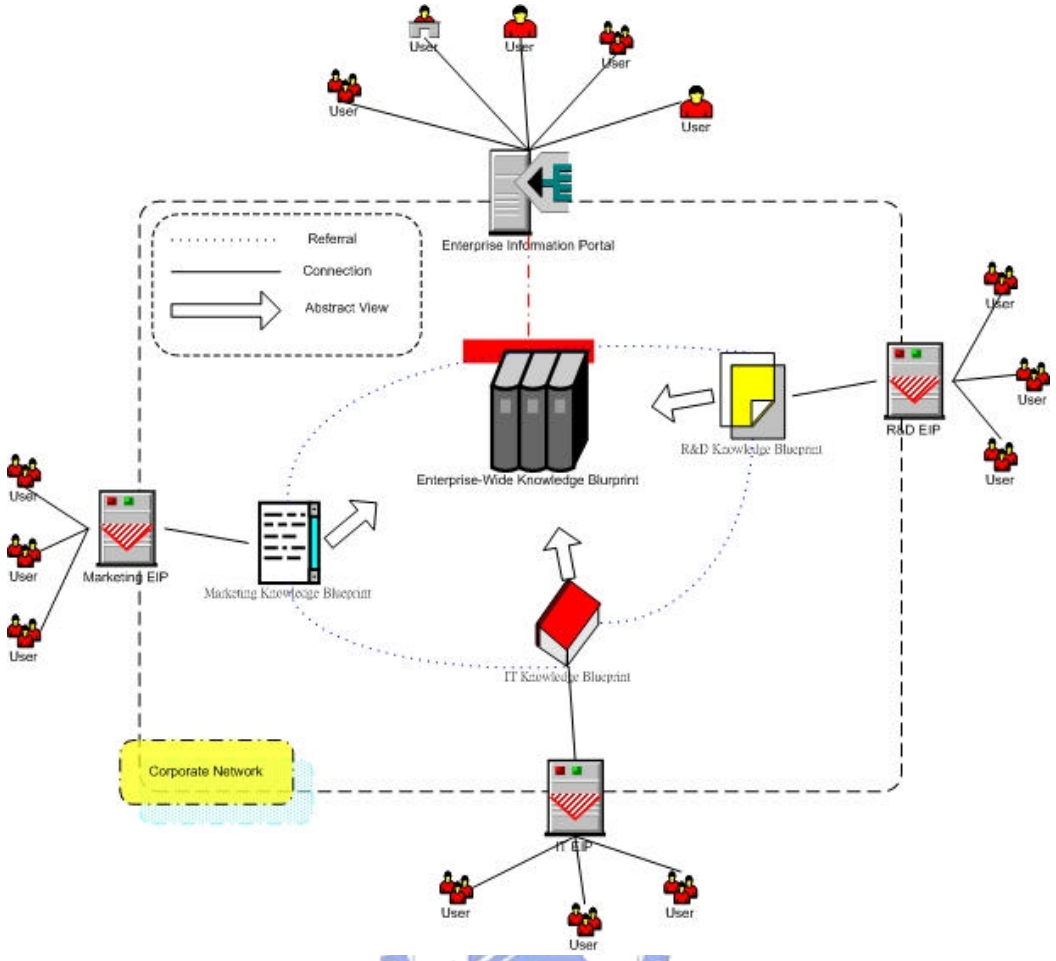
表【4-2】所列軟體為全開發工具的簡表：

表【4-2】開發工具列表

作業系統	Windows XP
目錄伺服器	IBM Directory Server 5.1
網路伺服器	Apache Tomcat
開發工具	JSP、JNDI、HTML
客戶端	Internet Explorer

4.2 知識藍圖系統概念與架構

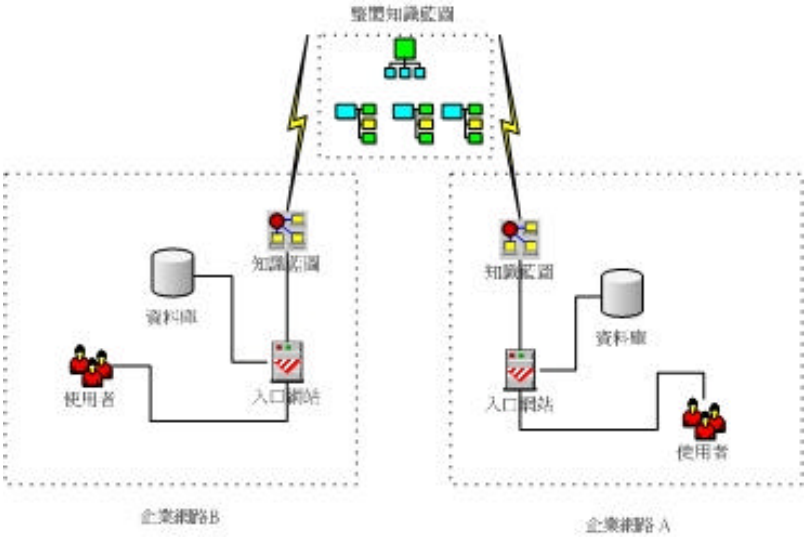
對於企業而言，掌握知識是企業在嚴苛的競爭環境中致勝的利基，而要有效運用知識，將員工的知識（不論是隱性或是顯性）形諸於文件的知識文獻便是企業不可或缺的資產。本研究之知識文獻所指，乃是對企業之競爭力有所提昇、幫助之文獻。而目錄所扮演的角色是一個知識文獻的藍圖，此一藍圖為本研究之核心。知識藍圖儲存了知識文獻的元資訊，知識文獻的元資訊包括了如文獻作者、文獻標題、位置、創造日期等等可讓使用者對文獻內容有更多了解的資訊。使用者的資料則包括如使用者的部門、姓名等等基本資料。企業知識藍圖可以說是企業內知識文獻的通訊錄，企業可決定依部門別建立知識文獻藍圖或是建立一個整體的知識藍圖。使用者若是要存取不同位置的藍圖，則可透過藍圖的重導向（referral）機制完成，使用者不會察覺到在藍圖後端所發生的動作。下圖為企業內使用者透過知識入口網站連結到各知識藍圖的示意圖：



圖【4-2】知識藍圖使用示意圖

由示意圖中可發現，藉由各獨立的藍圖之間的重導向關係，可形成一個大型的、抽象的藍圖，使用者只須進入其部門的入口網站，便有機會可存取到所有部門的知識藍圖。此種做法可立即得到的好處，在於藍圖的使用者毋須分別至各部門或具有合作關係之企業的網站來蒐集資料，而僅需進入一個地方，即可得到所有的結果。不但減輕了使用者的負擔，蒐集、找尋知識文獻的時間效率也可提升，提供使用者在知識定位方面的協助。

企業知識入口網站在這裡的角色，為一個中介者。使用者對知識藍圖的登入、檢索、閱覽等等工作，均是利用知識入口網站的介面完成。透過知識入口，使用者可輕易地使用知識藍圖，而不須理會知識藍圖背後的運作機制。對一般使用者而言，整個知識藍圖的架構便應如圖【4-3】所表示一般：



圖【4-3】知識藍圖整體之架構圖

在藍圖的建立方面，如前文所述，LDAP 目錄伺服器為藍圖的儲存者，對於組織中已有的文件，使用者或管理者可透過 EIP 所提供的 Web 介面將文獻的元資訊存入知識藍圖之內，亦可以搭配元目錄之相關技術，將分散在企業各處的資料庫、檔案系統內的文獻元資訊，收集回至藍圖內；而對於新建立的文獻，則在將文獻儲存至資料庫或檔案系統之時，便可順道將文獻的元資訊存到知識藍圖之中。

以上所述為系統整體之概念與架構，下一小節將說明知識藍圖系統所具有的基本功能。

4.3 知識藍圖功能介紹

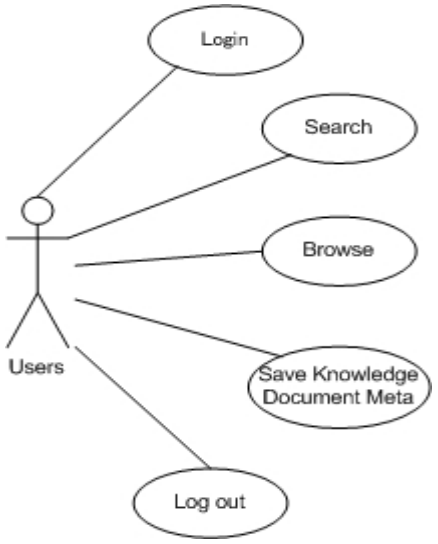
知識藍圖系統的設計，主要是由企業知識入口網站的 Web 介面呈現。在介面上須呈現的藍圖功能表列如下：

表【4-3】知識藍圖功能表

功能名稱	功能說明
登入	驗證使用者身份以登入知識藍圖系統，辨識成功後可進入知識藍圖系統進行檢索工作。

搜尋	依使用者要求，進行部門藍圖或企業藍圖間知識文獻之搜尋。
瀏覽	依使用者要求，瀏覽本部/企業，或跨部門/跨企業之知識文獻。
儲存藍圖物件	讓使用者能輸入欲上傳的知識文獻之元資訊。
登出	使用者使用完畢後進行登出，切斷使用者與藍圖的連線。

我們可以使用 UML（Unified Modeling Language）中的使用案例圖清楚地表示系統功能需求，請參見下圖：

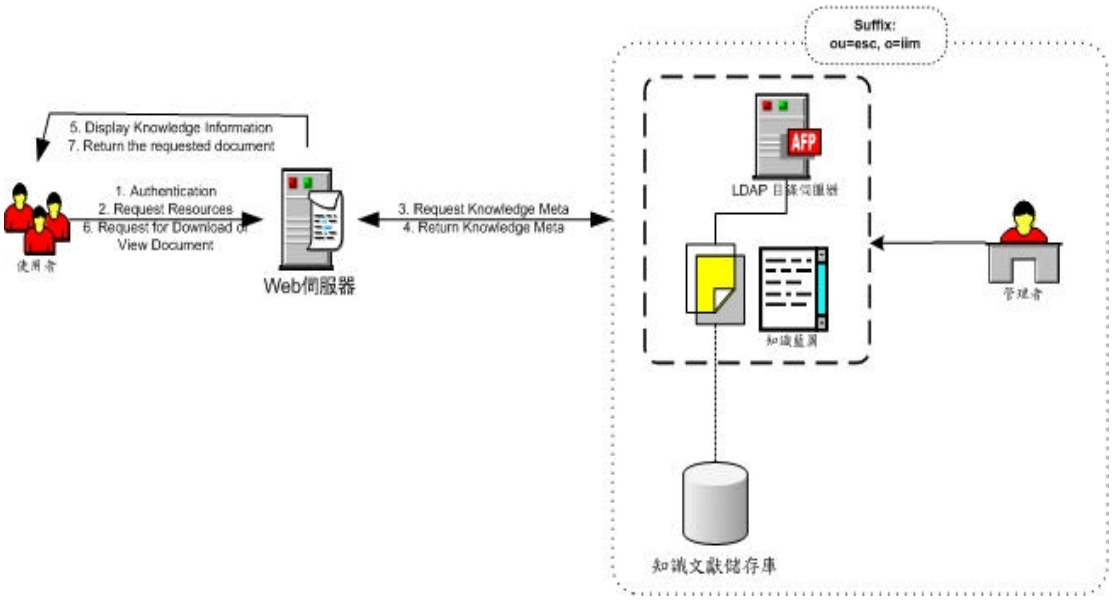


圖【4-4】系統使用案例圖

在這個使用案例圖中的 Users 泛指所有與知識藍圖系統發生互動的使用者，其餘五個主要的使用案例則是表【4-3】所述的功能。

4.4 知識藍圖之運作流程

圖【4-5】所示為使用者查詢企業單一部門之知識藍圖的情況。使用者首先需登入企業知識入口網站。身份驗證通過之後，則進入知識藍圖系統的功能選單。由功能選單中可以決定所要進行的工作，分別是檢索、瀏覽、儲存藍圖物件及登出。其步驟如下圖所示：



圖【4-5】系統流程

4.5 心得與討論

目前廣為接受的網路搜尋引擎如 Yahoo、Google **等等，同樣可以一個單一介面讓使用者輸入關鍵字查詢位在世界各地的相關資料。然而，搜尋引擎所查詢出的資料太過龐大。以輸入關鍵字：**LDAP **為例，**Google **便傳回一百八十萬筆資料，**Yahoo **亦傳回一百六十萬筆之資料。而由這些搜尋引擎所得之資料除了與** LDAP **有關的知識文獻之外，亦包括廠商的行銷文件、**LDAP **的相關軟體等等，使用者難以從中過濾對自己有用之資訊。除此之外，搜尋引擎無法查詢到受企業內網路保護的知識文獻。綜合以上所述，作者將搜尋引擎與知識藍圖之差異整理如表【**4-4**】：**

表【4-4】 知識藍圖與搜尋引擎之差異

差異類別	知識藍圖	搜尋引擎
介面	單一介面	單一介面
查詢範圍	企業內網路	整個 Internet，無法進入企業內網路
查詢內容	知識文獻	各種與關鍵字相關的文件或軟體
資訊組織度	有組織的	未經過組織，使用者難以從中過濾自己所需要之資訊

第五章 知識藍圖雛型系統展示

本章針對本研究所提出的架構作介紹，內容包括物件及屬性的設計及系統功能展示。本研究之雛型系統利用 IBM Directory Server 5.1 建立知識藍圖，為儲存知識文獻之元資訊，於 IBM Directory Server 的綱要中新增一個 DocMeta 物件。DocMeta 物件的資訊及屬性的詳細設計分列如下。

5.1 DocMeta 物件屬性表

在目錄中，要存放資料之前，首先就是要確認綱要中對資料物件及屬性定義的規範，這些規範包括：允許儲存什麼型態的物件類別、資料項、資料項的屬性值或是屬性的語法等等。本研究於目錄中的綱要定義了一個新的物件類別：DocMeta，用以儲存企業中知識文獻的相關元資訊，以下提供本研究中所使用資料屬性型態之簡表及詳細列表。

表【5-1】DocMeta 物件所含之屬性簡表

DocMeta所含屬性簡表				
屬性名	**容許多值**	**比對規則**	**檢索規則**	**必要/選用屬性**
DocFileName	no	caseIgnoreMatch	**相等/大約**	Mandatory
DocTitle	no	caseIgnoreMatch	**相等/大約**	Mandatory
DocAuthor	yes	caseIgnoreMatch	**相等/大約**	Mandatory
DocDate	no	caseIgnoreMatch	**相等/大約**	Mandatory
DocType	no	caseExactMatch	**相等**	Mandatory
DocCategory	yes	caseIgnoreMatch	**相等/大約**	Mandatory
DocDescription	yes	caseIgnoreMatch	**相等/大約/子字串**	Mandatory
DocTranslator	yes	caseIgnoreMatch	**相等/大約**	Optional
DocPublisher	yes	caseIgnoreMatch	**相等/大約**	Optional
DocKeywords	yes	caseIgnoreMatch	**相等/大約**	Optional
DocLocation	no	caseExactMatch	**相等**	Mandatory

5.2 DocMeta 物件屬性完整列表

表【5-2-1】DocFilName 屬性定義表

DocFileName	
屬性名稱	DocFileName
說明	The file name of the document
OID	DocFileName-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	128
容許多值	no
比對規則	相等=caseIgnoreMatch
必要/選用屬性	Mandatory
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	一般
索引規則	相等、大約

表【5-2-2】DocTitle 屬性定義表

DocTitle	
屬性名稱	DocTitle
說明	The title of the document
OID	DocTitle-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	500
容許多值	no
比對規則	相等=caseIgnoreMatch
必要/選用屬性	Mandatory
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	Normal
索引規則	相等、大約

表【5-2-3】DocAuthor 屬性定義表

DocAuthor	
屬性名稱	DocAuthor
說明	Describe the author of the documents
OID	DocAuthor-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	300
容許多值	yes
比對規則	相等=caseIgnoreMatch
必要/選用屬性	Mandatory
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	Normal
索引規則	相等、大約

表【5-2-4】DocDate 屬性定義表

DocDate	
屬性名稱	DocDate
說明	State the creation date of the document
OID	DocDate-oid
上階屬性	none
語法	Generalized Time Syntax
屬性長度	20
容許多值	no
比對規則	相等=caseIgnoreMatch
必要/選用屬性	Mandatory
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	Normal
索引規則	相等、大約

表【5-2-5】DocType 屬性定義表

DocType	
屬性名稱	DocType
說明	Indicate the type of the document, like doc or xml or html...etc.
OID	DocType-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	5
容許多值	no
比對規則	**相等**=caseExactMatch
必要/選用屬性	Mandatory
DB2 **表格名稱**	Default
DB2 **直欄名稱**	Default
安全類別	Normal
索引規則	**相等**

表【5-2-6】DocCategory 屬性定義表

DocCategory	
屬性名稱	DocCategory
說明	Describe the class of the document.
OID	DocCategory-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	128
容許多值	yes
比對規則	**相等**=caseIgnoreMatch
必要/選用屬性	Mandatory
DB2 **表格名稱**	Default
DB2 **直欄名稱**	Default
安全類別	Normal
索引規則	**相等、大約**

表【5-2-7】DocDescription 屬性定義表

DocDescription	
屬性名稱	DocDescription
說明	The abstract of the document
OID	DocDescription-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	1000
容許多值	yes
比對規則	相等=caseIgnoreMatch
必要/選用屬性	Mandatory
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	Normal
索引規則	相等、大約、子字串

表【5-2-8】DocTranslator 屬性定義表

DocTranslator	
屬性名稱	DocTranslator
說明	Specifies the translators of the document
OID	DocTranslator-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	128
容許多值	yes
比對規則	相等=caseIgnoreMatch
必要/選用屬性	Optional
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	Normal
索引規則	相等、大約

表【5-2-9】DocPublisher 屬性定義表

DocPublisher	
屬性名稱	DocPublisher
說明	Specifies the publisher of the document. Here the publisher means the people who post the document.
OID	DocPublisher-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	128
容許多值	yes
比對規則	相等=caseIgnoreMatch
必要/選用屬性	Optional
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	Normal
索引規則	相等、大約

表【5-2-10】DocKeywords 屬性定義表

DocKeywords	
屬性名稱	DocKeywords
說明	Keywords of the document
OID	DocKeywords-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	128
容許多值	yes
比對規則	相等=caseIgnoreMatch
必要/選用屬性	Optional
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	Normal
索引規則	相等、大約

表【5-2-11】DocLocation 屬性定義表

DocLocation	
屬性名稱	DocLocation
說明	Specifies the URL or other location of the document. So the requestor can use it to get the exact document.
OID	DocLocation-oid
上階屬性	none
語法	Attribute Type Description Syntax
屬性長度	128
容許多值	no
比對規則	相等=caseExactMatch
必要/選用屬性	Mandatory
DB2 表格名稱	Default
DB2 直欄名稱	Default
安全類別	Critical
索引規則	相等

各欄位型態的解釋如下：

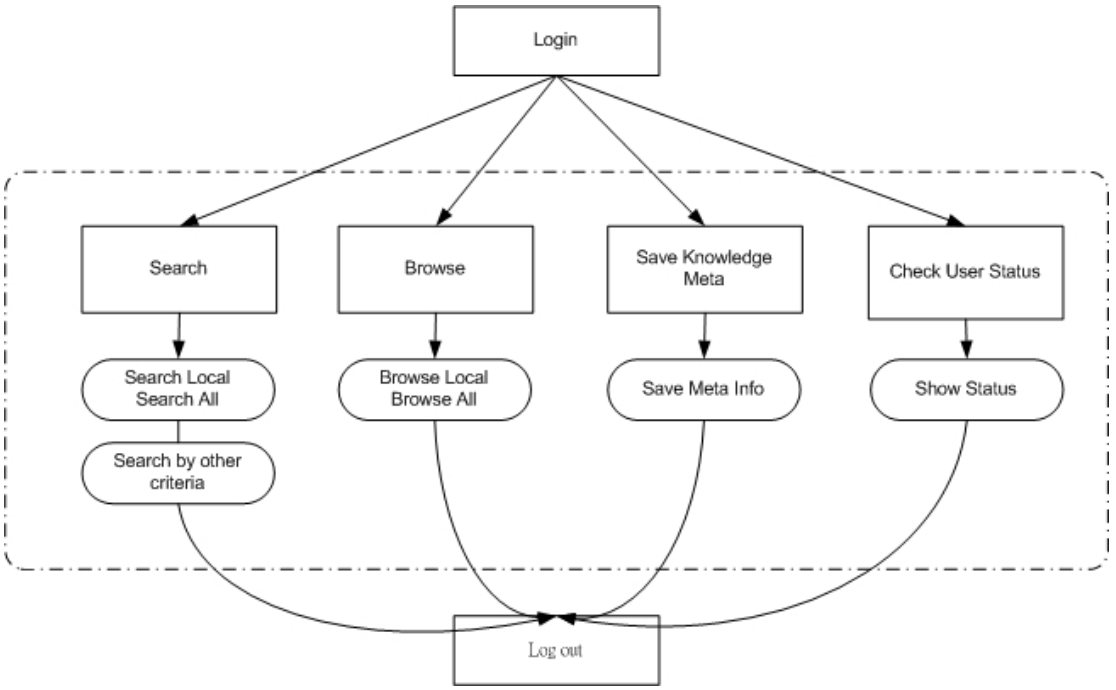
表【5-2-12】屬性之欄位型態解釋

欄位型態解釋	
屬性名稱	顧名思義，在目錄中所顯示的屬性識別名稱。
說明	描述屬性的功用。
OID	目錄中規定所有的屬性都要有一組 OID（Object Identifier），即物件的識別碼，若沒有識別碼則可以屬性名後面加上"-oid"表示這是物件的 oid。
上階屬性	屬性與屬性之間可以有繼承之關係。
語法	描述構成此屬性值的語法規則。
屬性長度	此屬性所允許值的最大長度。
容許多值	用以表示此屬性是否是個多值的屬性。
比對規則	比對規則為搜尋時的比對提供指引。比對規則可分三類，一是 Equality、一是 Ordering、一是 Substring。
必要/選用屬性	必要屬性一定要有值，選用屬性則可有可無。
安全類別	屬性有三種安全類別：Normal、Sensitive、Critical，使用者可依需求決定該屬性為哪一類別，進而限制使用者的存取。

索引規則[7]	**屬性使用到索引規則的話，目錄會為該屬性建立索引，如此一來能使搜尋的速度加快，索引規則有五類，分別是：相等、排序、大約、子字串跟逆向索引。** **相等（**equality**）會套用下列的搜尋作業：** • equalityMatch '=' **例如：** "cn = John Doe" **排序（**ordering**）會套用下列的搜尋作業：** • greaterOrEqual '>=' • lessOrEqual '<=' **例如：** "sn >= Doe" **大約（**approximate**）會套用下列的搜尋作業：** • approxMatch '~=' **例如：** "sn ~= doe" **子字串（**substring**）會套用採用子字串語法的搜尋作業：** • substrings '*' **例如：** "sn = McC*" "cn = J*Doe" **逆向（**reverse**）會套用下列的搜尋作業：** • '*' **子字串** **例如：** "sn = *baugh"

5.2 系統功能說明與展示

圖【5-1**】顯示使用者登入雛型系統後可選擇之功能以及操作進行流程。**



圖【5-1】系統功能操作流程

登入系統：登入功能置於系統首頁，使用者輸入帳號及密碼，成功後便可登入知識藍圖的檢索系統，若輸入資料不正確，則顯示錯誤訊息。

圖【5-2】知識藍圖登入畫面

經核對正確的使用者登入後，則顯示系統主畫面，功能選項皆位於上方的橫向目

錄之內，選項包括有檢索、瀏覽及登出。使用者若未經登入便直接選擇功能選項，系統會將該使用者導回至登入首頁。

圖【5-3】登入系統後歡迎畫面

瀏覽功能：在瀏覽功能中，使用者可瀏覽知識藍圖中自己有權觀看之所有知識文獻物件的元資訊，使用者可選擇瀏覽全部藍圖、或是只瀏覽單一的藍圖。

圖【5-4】選擇知識藍圖畫面

圖【5-5】展示了選擇瀏覽藍圖內全部知識文獻功能後的畫面。

圖【5-5】瀏覽藍圖內全知識文獻畫面

如圖【5-6】所示，在搜尋方面，可依藍圖位置、作者、文獻類別、文獻標題、關鍵字、文獻日期、文獻摘要等等屬性值進行搜尋。搜尋後同樣會依文獻標題、作者、關鍵字等欄位顯示出可供存取之知識文獻。

圖【5-6】知識藍圖的檢索畫面

第六章 結論

6.1 研究貢獻

隨著三層式架構的成熟和網路環境的發展，企業內部門的知識文獻通常是儲存於檔案系統或資料庫。而隨著企業規模成長，或是與其他公司締造合作關係，企業利用網際網路存取知識文獻的需求將日漸殷切。以 Web 介面搭配資料庫讓使用者可進行文獻的存取是目前最普遍的方式。然而當文獻儲存的位置分散在許多處之時，要同時查詢存取各地資料庫內之文獻會有一定的困難度。本研究提出一個以 LDAP 目錄為基礎所規劃的知識藍圖，藉由企業入口網站 Web 介面的輔助，企業內使用者可以一個單一介面搜尋、存取分散於企業內各部門之文件。

除此之外，以基於 LDAP 的目錄來架構系統尚存在幾項優點：

1. LDAP 是一個開放式的目錄服務協定標準，且眾多的軟體大廠如 Microsoft、IBM、Oracle、Novell、Sun 等等皆有推出相關的產品，開放源碼方面也有 OpenLDAP 此一伺服器可以選擇，不虞有失去技術支援之疑慮。

2. LDAP 目錄服務支援集中式（centralized）及分散式（distributed）的資料儲存，若企業決定以集中的方式儲存目錄資料，則可佈署單獨一台目錄伺服器做為單一的儲存點；若要使用分散式的設計，則可佈署多台目錄伺服器，每一台負責處理之資料內容可以是相同的（運用複寫的技術）或是經過分割後才將資料分別存放在不同的伺服器之上。為因應今日高度的競爭環境，企業內部基礎建設常需保持相當的彈性，LDAP 目錄服務伺服器儲存資料之方式可分散可集中，企業可依需求規劃符合自身環境的服務。

3. 一般關聯式資料庫所擅長的領域是在異動（transaction）的支援及資料的寫入上；另一方面，目錄在資料的讀取及搜尋上已經過最佳化，對企業來說，文件通常是用來讓使用者讀取及搜尋的，利用目錄儲存文獻的元資訊這種做法可讓

使用者的查詢等待時間大幅縮短。張賢安[2]於其研究指出，LDAP 的子字串搜尋較 SQL 快上許多：

表【6-1】Results of sub-string search 資料來源：[2]

	LDAP	SQL
Single match（milliseconds）（eq. Cn=1112）	17.38	32.964
Right side sub-string match（milliseconds）（eq. Cn=*112）	98.42	1255.146
Left side sub-string match（milliseconds）（eq. Cn=111*）	102.302	1170.196
Sub-string match（milliseconds）（eq. Cn=*11*）	106.412	1193.978

綜合之前所述，本研究提出以目錄服務為基礎的知識共享機制，運用了目錄服務可以一個單一的邏輯架構呈現分散在不同地區、位置的資訊之優點，建構一知識藍圖，藉由單一的 Web 介面，提供使用者一個易於使用的系統。

6.2 限制與未來研究建議

本研究仍有許多的不足與限制，可以在未來做為繼續進行研究及努力的目標。首先，針對資料庫中的分散式資料庫技術便是為了解決本研究所述資料零散儲存於不同地理位置的問題而生。由於作者對於分散式資料庫之技術熟悉度不足，因此無法將本系統所提出之機制與分散式資料庫的差異之討論納入文中。未來可將知識文獻的元資料儲存於分散式資料庫中，並就系統效能、易用性及整合性等議題與本研究所提出之知識藍圖系統做進一步之比較與討論。

此外，在本論文的實作中，企業不同目錄藍圖中知識文獻的元資料的綱要定義是一致的。然而若未經過規劃，同企業內的不同部門或分公司可能會給予元資料不同的欄位定義。有上述情況發生時，知識藍圖間事先便要經過一道綱要定義交換的手續，以確保彼此對於資料的定義是相同的。

XML 可說是目前文件交換的熱門標準格式，目錄服務在這方面亦有了 DSML（Directory Service Markup Language）為延伸。DSML 目前已到了第二版。未來可將欲儲存入目錄伺服器的文件轉換為 DSML 格式而非傳統的 LDIF 格式，也可以利用 DSML 對 LDAP 伺服進行如搜尋、讀取等等的操作。利用 DSML 可使一般的 Web 應用程式更方便地存取目錄伺服器。

元資料的相關研究目前正方興未艾。妥善利用元資料描述知識內容，並藉由網路技術呈現，可以讓使用者所查詢、檢索之資料更具意義、更切合需求。元資料在知識文獻這方面尚未有相關標準，不過在其他領域已有如都柏林核心集（Dublin Core）及 Text Encoding Initiative（TEI）等等標準出現，未來對於企業內的知識文獻若有相對應的標準，則可套用之，以增進系統間的相互操作性。

Reference

[1] 陳光明，「企業資訊入口網站設計之研究」，國立交通大學資訊管理研究所，碩士論文，民國 91 年 6 月。

[2] 張賢安，「在分散式環境中目錄服務的議題、設計及其效能」，國立清華大學資訊工程研究所，碩士論文，民國 88 年 6 月。

[3] Cox, Nancy. Directory Services Design, Implementation, and Management. Digital Press. 2002.

[4] "Directory Services Markup Language v2.0 Specification." Organization for the Advancement of Structured Information Standards. 2001.

[5] Howes, Tim. "LDAP: Use as Directed." Retrieved September 1, 2002 from the World Wide Web：
http://www.networkmagazine.com/article/DCM20000502S0039

[6] "IBM Application Framework for e-business – Directory Services." Retrieved March 15, 2002 from the World Wide Web：
http://www-3.ibm.com/software/network/directory/library/whitepapers/directory.htm

[7] IBM Directory Server Version 5.1 Administration Guide. IBM. 2002.

[8] "iPlanet Directory server Deployment Guide Version 5.1." Sun Microsystems. 2001.

[9] "JNDI API." Sun Microsystems, Inc. 1999.

[10] Mesaros, P., Michael. "Introducing Oracle Internet Directory." Retrieved December 21, 2002 from the World Wide Web：
http://www.oracle.com/ip/integrate/collateral/index.html?590.html

[11] Sheresh, Beth and Sheresh, Doug. Understanding Directory Services. New Riders Publishing. 2000.

[12] Windows 2000 Server Documentation. “Introduction to Active Directory.” Retrieved August 12, 2002 from the World Wide Web：
http://www.microsoft.com/windows2000/en/server/help/

附錄 A 本文所使用的物件類別及屬性之綱要定義

objectClasses
{
(DocMeta-oid NAME 'DocMeta' DESC 'The object that describe the Metainfo of each document.' SUP top MUST (DocAuthor $ DocCategory $ DocDescription $ DocFileName $ DocLocation $ DocTitle $ DocType) MAY (DocDate $ DocKeywords $ DocPublisher $ DocTranslator))
}

attributeTypes {
(DocAuthor-oid NAME 'DocAuthor' DESC 'Describe the author of the documents.' EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{300})
(DocCategory NAME 'DocCategory' DESC 'Describe the class of the documents.' EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{128})
(DocDate-oid NAME 'DocDate' DESC 'State the creation date of the document' EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.24{20} SINGLE-VALUE)
(DocDescription-oid NAME 'DocDescription' DESC 'The abstract of the document' EQUALITY 2.5.13.5 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{1000})
(DocFileName-oid NAME 'DocFileName' DESC 'The Filename of the document.' EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{128} SINGLE-VALUE)
(DocKeywords-oid NAME 'DocKeywords' DESC 'Keywords of the document' EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{128})
(DocLocation NAME 'DocLocation' DESC 'Specifies the URL or other location of the document. So the requestor can use it to get the exact document.' EQUALITY 2.5.13.5 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{128} SINGLE-VALUE)
(DocPublisher-oid NAME 'DocPublisher' DESC 'Specifies the publisher of the document. Here the publisher means the people who post the document.' EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{128})
(DocTitle-oid NAME 'DocTitle' DESC 'The Title of the document.' EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{500} SINGLE-VALUE)
(DocTranslator-oid NAME 'DocTranslator' DESC 'Specifies the translator of the document' EQUALITY 2.5.13.2 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{128})
(DocType-oid NAME 'DocType' DESC 'Indicate the type of the document, like doc or xml or html...etc.' EQUALITY 2.5.13.5 SYNTAX 1.3.6.1.4.1.1466.115.121.1.3{5}

SINGLE-VALUE)
}

IBMattributeTypes {
(DocAuthor-oid EQUALITY APPROX)
(DocCategory EQUALITY APPROX)
(DocDate-oid EQUALITY APPROX)
(DocDescription-oid EQUALITY SUBSTR APPROX)
(DocFileName-oid DBNAME ('DOCFILENAME' 'DOCFILENAME') LENGTH
128 EQUALITY APPROX)
(DocKeywords-oid EQUALITY APPROX)
(DocLocation ACCESS-CLASS CRITICAL EQUALITY)
(DocPublisher-oid EQUALITY APPROX)
(DocTitle-oid EQUALITY APPROX)
(DocTranslator-oid EQUALITY APPROX)
(DocType-oid EQUALITY)
}