In Proceedings of TANET 2001 and ELCPE 2001, Vol. 2, pp. 7-12, Chiayi, Taiwan, Oct. 24-26, 2001.
2001 Taiwan Academic Network Conference (TANET 2001),
2001 年台灣區學術網路研討會暨網路學習與繼續專業教育國際會議

以 Metalogy 的詮釋資料為依據的跨圖書館搜尋引擎之設計

陳英祥 **陳哲民** **楊正仁**
Ing-Xiang Chen **Che-Min Chen** **Cheng-Zen Yang**
元智大學資訊工程學系
sean@syslab.cse.yzu.edu.tw linda129@ms19.hinet.net czyang@syslab.cse.yzu.edu.tw

摘要

隨著資訊腳步的快速前進，對於數位圖書館的整體效能而言，發展一個能夠跨越不同數位圖書館的資訊檢索系統顯得日益重要。因為藉由這樣的搜尋系統，使用者能夠透過一個整合性的介面，在不同的數位圖書館中找到相關資訊，而避免使用者分別在各個不同的數位圖書館中做重複查詢的動作。在此篇論文中，我們提出一個具整合性介面的搜尋引擎設計，讓使用者可以透過此介面搜尋不同數位圖書館的資料。本搜尋引擎設計是以建立 XML/DTD 對應關係的方式來達到跨圖書館檢索的功能。利用此方式，我們可以針對不同數位圖書館中不同性質的詮釋資料 (metadata) ，在資料庫中建立一份對應關係，並且此對應關係是對應到系統預設的詮釋資料集 (metadata set)。為達到高擴充性，本系統設計是採用都柏林核心集 (Dublin Core set)。透過這樣的設計理念，使用者可以更快速、更方便地使用不同數位圖書館的資源。

關鍵詞： 數位圖書館、XML、跨圖書館檢索、Metalogy、詮釋資料 (metadata)、都柏林核心集 (Dublin Core set)

1. 簡介

在建構整個數位圖書館的過程中，詮釋資料 (metadata)扮演了一個非常重要的角色，尤其是在資料的管理與組織上更為重要。然而，管理詮釋資料卻不是一件容易的工作，因為管理者需要對於某個特定領域的知識有深入的了解，並且能清楚地將資料分類。此外，管理者還必須訂定詮釋資料中各個物件間複雜的關係，因此管理者需要有好的管理工具來簡化他的工作。

Metalogy [1] 是由「資源組織與檢索之規範」(Resource Organization and Searching Specification, 簡稱 ROSS) [2] 專案計劃小組所發展出來的一套管理系統。藉由使用 Metalogy 系統，不同數位圖書館的詮釋資料便可以依據該圖書館的需求被建構與維護，而且一座數位圖書館在建構的過程中，管理者可以很方便地修改詮釋資料的內容，並且管理者也可以依照不同的資料增加新的修飾詞 (qualifiers)。新的修飾詞加入詮釋資料後，Metalogy 也可以被使用來存入各種多媒體檔案、查詢圖書館資料、匯入或匯出 XML (eXtensible Markup Language) [3,4] 資料以及管理使用者權限。

雖然 Metalogy 的設計主要是管理一座數位圖書館的詮釋資料，但是跨不同數位圖書館做資訊檢索的功能還並未完整的考慮進來。我們也發現到當未來有許多數位圖書館廣泛地分布在不同地方時，跨數位圖書館的資訊檢索對使用者而言便顯得更加重要，因為不同的數位圖書館裡皆保存有其特別的檔案主題，而且這些不同的檔案都是由該領域的專家維護著。例如，一位使用者想要找有關畫家張大千的資料，他可能需要分別到不同的數位圖書館瀏覽、找尋相關畫家的資料以及相關繪畫作品的資料。

因此，一個能夠搜尋跨不同數位圖書館檔案資訊的搜尋引擎有其必要。藉由這樣的搜尋引擎，使用者便不需要進一步了解各個不同的數位圖書館的網址，也可以避免在不同的數位圖書館網站輸入重複的查詢字串。本搜尋引擎提供了一個整合不同數位圖書館的想法，而且這個搜尋引擎也能處理由不同數位圖書館所提供的詮釋資料，並且透過這個方法，使得無論所輸入的查詢字串是隸屬於哪個資料分類範疇，使用者皆能依據分類的結果，找到許多來自不同數位圖書館裡符合的資訊。

本搜尋引擎最具特色的地方就是它是以 XML/Metalogy 的技術作為基礎，這有別於目前其他的資訊檢索系統，因為他們大多用特有通訊協定的方式來完成，例如 Z39.50 通訊協定 [5] 。我們採用這樣的方式主要有兩個原因：

一、 XML/Metalogy 能提供容易理解的詮釋資料描述及 DTD (Document Type Definition)資訊給搜尋引擎以方便之後進行詮釋資料的檢索。這樣的方法對於搜尋來源不同的詮釋資料提供了很大的彈性。如果有一份新的詮釋資料，它的規格是針對某個特定領域的需求所發展出來，則這一份規格便能夠很容易地被整合在這個 XML/Metalogy 搜尋引擎上。

二、 目前大部分的資訊檢索通訊協定，如 Z39.50 都非常的複雜，而且主要是設計給圖書館專家使用。因此，大部分一般的使用者不知道這個通訊協定的詳細內容，更不知道要如何定義要輸入的查詢項目，所以並不適合使用者直接使用。

我們的搜尋引擎設計簡化了查詢介面的設計，以方便使用者容易操作，此外，此搜尋引擎亦透過 XML 的技術，對於未來的詮釋資料發展提供良好的擴充性。我們以 Java 實做了一個雛形系統，當中結合了資料庫的使用以達到加快搜尋速度的目的，目前這個資料庫表格的欄位是依據都柏林核心集（DC）[6,7]，主要是因為它的擴充性相當的高。雖然這個雛形系統目前只提供基本的查詢功能，但是這樣的雛形系統也替未來跨數位圖書館的分散式檢索奠定良好的基礎。

本論文的其他內容如下：第二節是談論 XML 搜尋引擎設計的相關研究。第三節是描述本搜尋引擎的架構。第四節則是展示本搜尋引擎的雛形架構以及一個查詢的範例。最後，第五節是討論未來的工作及本論文的結論。

2. 相關研究

針對數位圖書館使用效率的提升，跨不同數位圖書館的資訊檢索是非常重要的一個關鍵。從這樣的資訊檢索服務中，使用者便能夠從不同的數位檔案中得到更容易了解的相關資訊，也因為如此，這樣的需求一直以來皆被圖書館界所重視。在1995年時，在Z39.50 (第三版)的資訊檢索協定中，即提出透過網路，在不同種類的資料庫上進行搜尋與檢索。Z39.50是一個功能強大的通訊協定，能提供摘要檢索的功能，因此使用者可以進行查詢而不需要知道該資料庫的架構。然而，這樣一個功能強大的通訊協定卻有一個缺點，因為它整體功能太過複雜，因此在實做上也相對地變得很困難 [8]。

在 1999 年美國 Santa Fe 的會議，曾討論到有關學術電子出版品檔案的合作計劃，之後成立 Open Archives Initiative（OAI）。其主要的任務是發展與制定標準以促進電子出版品的傳播，進而提昇學術研究的交流 [9,10]。在 2001 年，OAI 提出 Arc，即用來達成跨檔案搜尋的一套系統 [11]。我們的搜尋引擎設計也近似 Arc 的設計架構，但是有部分地方不同。例如，在 Arc 的 harvester 系統設計上是利用 OAI 通訊協定來擷取數位檔案，但是在我們的設計上，則並不考量使用一個特別的通訊協定，而是使用一個詮釋資料對應關係的控制介面來完成跨異質數位圖書館的查詢，這是因為我們透過建立對應關係後，即可在索引資料庫（Index Database）上，查詢由各個數位圖書館所轉出的詮釋資料，並提供超連結讓使用者進一步存取該檔案。

在其他相關的研究方面，在 1999 年 Lin 和 Lu 發展了 Harp 數位圖書館分散式查詢系統 [12]。藉由使用 Harp，傳統開放式的圖書館與結構化的資料庫皆可利用 HarpSQL 以及用於查詢傳統圖書館分類的 SQL-like 查詢語言進行查詢。Harp 對於傳統的圖書館提供了一個整合的查詢介面，但是本搜尋引擎設計架構裡的查詢/檢索管理者設計卻不同於 Harp 的設計架構，在 Harp 的設計當中，一個 HarpSQL 伺服器所扮演的角色就像負責儲存與處理中間查詢結果的查詢代理人程式（query agent），但是它並不是搜尋引擎，因為它不能夠從不同的數位圖書館中收集與儲存所有的詮釋資料，而且它並未考量到在詮釋資料上做搜尋的功能。

METALICA [13] 則是採用類似 MetaCrawler [14] 的方式以提供一個整合的使用者介面進而達到跨檔案搜尋的功能。在 METALICA 上，藉由整合 integrator, mediator 及 wrapper 程式，此系統即可提供跨檔案查詢的服務。但是，使用者過去所查詢過的詮釋資料並不會保留到未來的查詢紀錄裡，因為 METALICA 系統是直接擷取遠端其他來源查詢所得的結果，它沒有自己專屬的索引資料庫。從詮釋資料處理的觀點來看，METALICA 的設計和我們所採用的方式並不相同，其中 METALICA 沒有建立屬於自己的索引資料庫，因此它的查詢服務品質得倚賴目前仍不甚穩定的網路情況。

無論是任何一種已經發展出來的架構，都必須面對同樣的一個問題：一個由分散式查詢服務所得到結果的品質，必須高度倚賴該數位檔案資訊來源的品質。這也是我們在發展搜尋引擎的資料來源選擇上，決定採用 Metalogy 詮釋資料管理系統的主要原因。

3. 系統架構

在搜尋引擎設計的過程中，如何提供給不同性質的數位圖書館一個整合的搜尋介面一直是我們主要的考量因素。目前，我們採用一個 XML/DTD 對應的方法。因為 Metalogy 能提供 well-formed 且內容、標籤定義完整的 XML 詮釋資料，所以此系統採用 Metalogy 所轉出的詮釋資料作為設計的基礎，將來不管是何種 XML 資料來源，只要該詮釋資料能夠定義完整的 DTD，本系統皆能夠延伸給該詮釋資料使用。

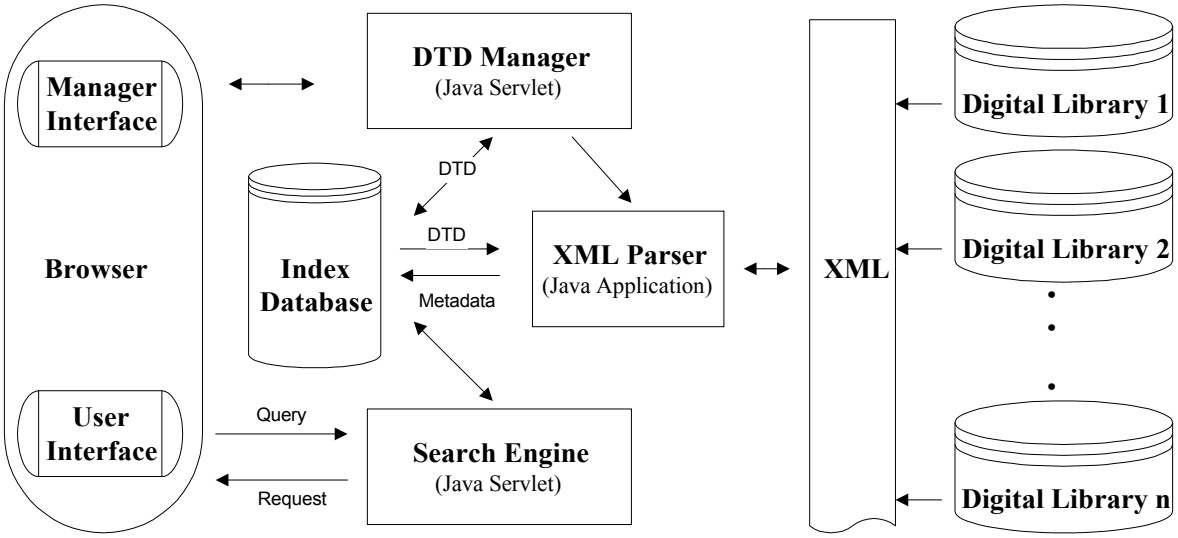

圖 1. 搜尋引擎的系統架構圖

圖 1 中描述了整個系統架構的設計。此搜尋引擎主要分為三個主要的模組：搜尋引擎模組、索引資料庫模組及 Metadata/DTD 管理者模組。藉由使用 XML/DTD 的方法，我們的設計有以下兩個優點：

一、 可跨不同 DTD 檢索：
系統管理者可透過管理者界面將該 DTD 對應到的元素 (element)填入 DTD 關係對應的表格中；本系統依據管理者建立的對應關係，把不同 DTD 的文件剖析，並且把剖析後所拮取的詮釋資料存入索引資料庫，使用者再透過簡易的界面來找到符合使用者需求之文件。雖然，目前一些傳統的圖書館無法提供 DTD 或詮釋資料的資訊，但是我們仍可透過一個外部的轉換程式來協助這些圖書館將資料轉換成合適的格式，而且這並不會使得搜尋引擎的設計變得更複雜。

二、 可新增不同的搜尋服務：
網際網路世界的不斷改變，新的搜尋服務也需要能夠被獨立地加入到搜尋引擎，並且對於原本數位圖書館提供的資料來源不會造成衝突。本搜尋引擎便是可以依據不同使用者的需求提供多變化的搜尋服務。

使用者透過一致的使用者介面送出需求後，他能夠依據所輸入的請求 (request)得到相關的搜尋結果，這些搜尋所得的結果是直接由搜尋引擎設計裡的索引資料庫 (Index Database)提供，之後，使用者也能夠透過由搜尋結果裡，相關訊息所提供的超連結而獲得更詳細的資訊。

以下我們將介紹每個模組的功能及其運作流程：

一、 搜尋引擎模組是在圖 1 中的 Search Engine。這是由一個整合的使用者介面以及一些 Java servlets 程式所組成，當此尋引擎模組接收到一個來自於使用者介面所給的請求 (request)時，它便會搜尋索引資料庫並且傳回符合的相關資訊。

二、 索引資料庫模組是在圖 1 中的 Index Database。此模組主要是負責儲存來自各個不同數位圖書館的詮釋資料。在這個資料庫裡，這些不同的詮釋資料都會被對應到一組特定的詮釋資料集。為了整體擴充性的考量，我們採用簡易都柏林核心集 (simple DC)當作我們系統的預設詮釋資料集。當然這些對應的關係也是被儲存在這個索引資料庫模組裡，而這些對應關係則是由 Metadata/DTD 管理者模組負責管理與維護。

三、 Metadata/DTD 管理者模組是在圖 1 中的 DTD Manager。其主要進行以下三個工作：

(1) 由於詮釋資料是由數位圖書館以 XML/DTD 的格式所提供，因此管理者介面主要的工作之一，即負責管理一個來自遠端數位圖書館所提供的 XML/DTD 與本系統預設的都柏林核心集的對應關係。

(2) 第二項任務就是呼叫此模組裡的 Java 程式來進行剖析 (parse)的工作，先從資料庫裡存取 DTD 對應的資訊，並對從遠端數位圖書館所提供的 XML 文件剖析，之後再將所剖析好的結果儲存在索引資料庫裡，以便接下來的搜尋工作。

(3) 第三項任務是呼叫程式重複地收集來自遠端數位圖書館的資訊，並且像其他搜尋引擎設計一樣，進行索引資料庫更新。

表 1 指出在索引資料庫中對應儲存表格的結構，這個表格是用來記錄其他數位圖書館的詮釋資料/DTD 與本系統預設的簡易都柏林核心集的對應關係結構。

表 1. DTD 的屬性及欄位對應表格

Field	*Type*	*Null*	*Key*	*Default*	*Extra*	*Privileges*
DTD	Varchar (50)	YES		NULL		select, insert, update, references
Type	Varchar (255)	YES		NULL		select, insert, update, references
Format	Varchar (255)	YES		NULL		select, insert, update, references
Title	Varchar (255)	YES		NULL		select, insert, update, references
Description	Varchar (255)	YES		NULL		select, insert, update, references
Subject	Varchar (255)	YES		NULL		select, insert, update, references
Creator	Varchar (255)	YES		NULL		select, insert, update, references
Contributor	Varchar (255)	YES		NULL		select, insert, update, references
Publisher	Varchar (255)	YES		NULL		select, insert, update, references
Date	Varchar (255)	YES		NULL		select, insert, update, references
Identifier	Varchar (255)	YES		NULL		select, insert, update, references
Source	Varchar (255)	YES		NULL		select, insert, update, references
Relation	Varchar (255)	YES		NULL		select, insert, update, references
Language	Varchar (255)	YES		NULL		select, insert, update, references
Coverage	Varchar (255)	YES		NULL		select, insert, update, references
Rights	Varchar (255)	YES		NULL		select, insert, update, references
Check	Char (1)	YES		NULL		select, insert, update, references

4. 雛形系統

我們實作了一個能夠提供基本查詢功能的雛形系統，此搜尋引擎模組與 Metadata/DTD 管理者模組是由 Java 程式語言所設計實作，索引資料庫的部分是採用一個版權開放使用的資料庫系統, MySQL。現階段 XML 資訊是由 JAXP (Java API for XML Parsing)封包 (package)中的 Simple API for XML (SAX)剖析器 (parser) [15] 來負責做剖析與處理，因為 SAX 提供了事件導向 (event-driven)和循序存取的機制。透過 SAX 所提供的這些方法可讓 XML 文件在剖析時，即對每個事件進行處理，而不需等到整個文件讀進記憶體後才作處理，因此使用 SAX 加強了對 XML 文件讀取與控制的功能。此外，採用 Java 語言更使得這個雛形系統具有跨平台的優點。

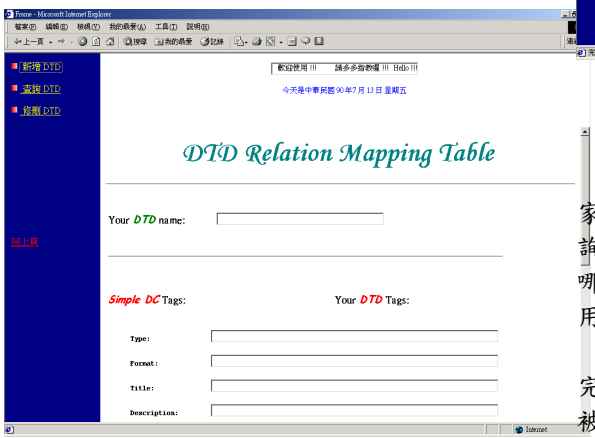

圖 2. XML/DTD 管理者介面模組的 DTD 對應關係管理介面

圖 2 是展示 XML/DTD 管理者介面的操作畫面，管理者可以透過這個介面增加新的對應關係以及進一步地管理對應關係的表格。

圖 3 是展示管理者為某一個特定的數位圖書館所建立好的一個對應關係，管理者也可透過這個介面進行修改的動作。

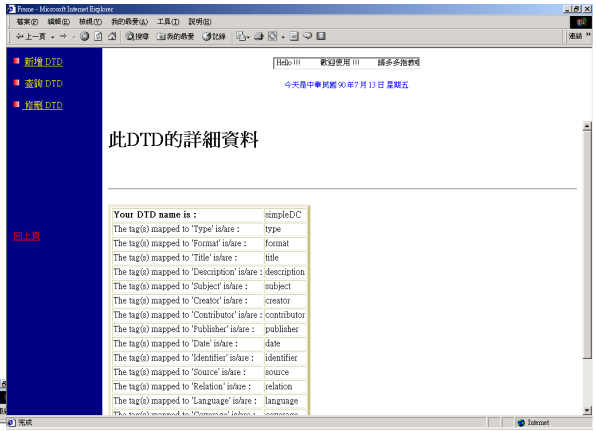

圖 3. 管理者依據某數位圖書館的需求所建立的範例

圖 4 是展示一個使用者找尋有關晉朝名書法家王羲之相關資訊的範例。使用者只需要輸入查詢字串，並不需要知道所輸入的關鍵字會出現在哪個專業的圖書館中。在送出這樣的請求後，使用者可以得到如圖 5 所顯示的搜尋結果。

這些詮釋資料若能夠依據 XML/DTD 的格式完整地訂定好，在剖析後，他們便能夠很適當地被建立在索引資料庫裡，而且也能夠藉由資料庫中索引的機制來促進搜尋效能。雖然我們目前測試的查詢速度良好，但是，當資料量非常龐大時，若直接在索引資料庫上進行全文檢索將使搜尋速

度大為降低，目前我們將搜尋效能的問題列入未來研究的議題。

圖 4. 使用者想要查詢有關晉朝名書法家王羲之的相關資訊

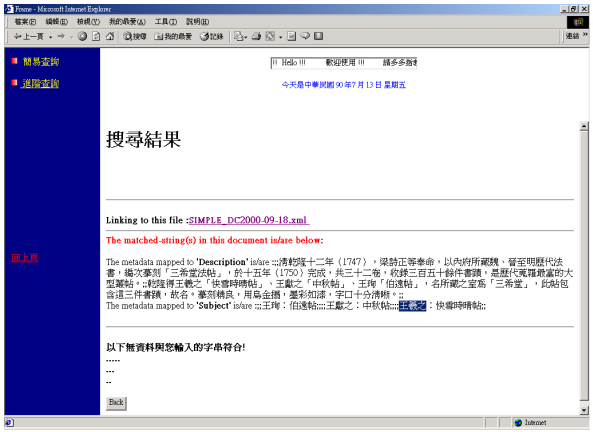

圖 5. 搜尋引擎傳回的搜尋結果

5. 結論與未來工作

在這篇論文中，我們提出一個搜尋引擎的設計架構，此搜尋引擎設計的目標是要透過 Metadata/XML 的方式以達到跨數個不同的數位圖書館做資訊檢索之目的。因此，使用者得以透過使用一個整合的介面來進行跨數位圖書館搜尋相關資訊。

本搜尋引擎設計有三個優點：第一，因為它採用一個 XML/DTD 的對應方式來管理詮釋資料的內容，因此整個系統架構設計相當地簡單。相對地，整個開發建構跨圖書館檢索搜尋引擎的成本也得以減少。第二，此系統對於新的需求服務具有很好的擴充性，雖然我們沒有採用一些功能強大的資訊檢索通訊協定，但是這類多用途的搜尋服務卻可以藉由使用 XML/DTD 的方式來達到。第三，使用者並不需要知道要到何處去尋找資訊，因為本搜尋引擎提供了一個方便的使用者介面，使用者可以透過這樣的一個介面來找到許多儲存於不同數位圖書館的相關資訊。

未來還是有下列問題需要進一步地探討，第一個問題是數位圖書館所提供來源資料的品質控制問題，雖然我們提供了一個 Metadata/DTD 管理者介面，而且管理者可以透過這樣的機制以人工控制的方式來定義相對應的關係，但是數位圖書館的原始資料來源仍然扮演了一個極重要的角色。雖然使用 Metalogy 當作資料來源幫助我們解決這樣的問題，但是當我們要整合一些傳統的圖書館時，這些問題仍然無法避免。

第二個問題是當我們在定義對應關係時會產生一些困難，目前都柏林核心集在資料庫上當作對應的基礎使用看似很合適，然而，為了支援不同性質的數位檔案，如何建立這樣對應的機制應當再更深入地做探討。

第三個問題是搜尋速度的問題，這也是所有搜尋引擎設計共同面對的一個問題。目前的搜尋引擎雛型還未在大量資料環境中進行測試，因此當實際應用上時，這樣的搜尋效能便需要再進一步討論與改進。未來為了能夠實際測試真實環境下的情況，我們將考慮架設實用的數位圖書館環境，以便進一步研究相關發展及需要。

6. 誌謝

我們特別要感謝 ROSS 計劃和陳昭珍教授等人提供 Metalogy 詮釋資料管理系統相關資料以協助我們的研究，謹在此表示感謝。

7. 參考文獻

1. 陳昭珍、陳雪華、陳光華，「數位圖書館與博物館 metadata 管理系統 - Metalogy 之設計」，*TANET 2000 台灣區網際網路研討會*，民 89 年，頁 492-499。

2. *Resources Organization and Searching Specification*. http://ross.lis.ntu.edu.tw/.

3. *World Wide Web Consortium.* “Extensible Markup Language (XML)”, http://www.w3.org/XML/ .

4. T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler. “Extensible Markup Language (XML)1.0 (Second Edition).” *In W3C Recommandation* http://www.w3.org/TR/2000/REC-xml-20001006.

5. http://www.loc.gov/z3950/agency.

6. S. Weibel, J. Kunze, C. Lagoze, and M. Wolf. “Dublin Core Metadata for Resource Discovery.” *RFC 2413*, http://www.ietf.org/rfc/rfc2413.txt

7. *Dublin Core Metadata Initiative.* http://purl.org/DC/.

8. W. Y. Arms. *Digital Libraries.* MIT Press, 1999.

9. C. Lagoze and H. Van de Sompel. “The Open Archives Initiative: Building a Low-Barrier Interoperability Framework.” *In Proceddings of the Joint Conference on Digital Libraries 2001 (JCDL 2001),* pp. 54-62, Roanoke, VA, June 2001.

10. The Open Archive Initiative. http://www.openarchives.org/.

11. X. Liu, K. Maly, M. Zubair, and M. L. Nelson. “Arc – An OAI Service Provider for Cross-Archive Searching.” *In Proceddings of the Joint Conference on Digital Libraries 2001 (JCDL 2001),* pp. 65-66, Roanoke, VA, June 2001.

12. E. Lim and Y. Lu. “Harp: A Distributed Query System for Legacy Public Libraries and Structured Databases.” *ACM Transactions on information Systems,* Vol. 17, No. 3 pp. 294-319, July 1999.

13. B. Schmitt and A. Schmidt. “METALICA: An Enhanced Meta Search Engine for Literature Catalogs.” *In Proceedings of the Second Asian Digital Library Conference (ADL’99),* pp. 142-160, Taipei, Taiwan, 1999.

14. E. Selberg and O. Etzioni. “The MetaCrawler Architecture for Resource Aggregation on the Web.” *IEEE Expert,* Vol. 12, No. 1, pp. 11-14, January-Febrary 1997.

15. Sun. "Java API for XML Parsing 1.0 Tutorial", http://java.sun.com/xml/tutorial_intro.html