Digital Library Curriculum Development

Module: 5-b: Application Software

(Draft, 03/24/2008)

1. Module name

Application software

2. Scope

This module covers commonly used application software, which are specifically designed for the creation and development of digital library (DL) systems and similar types of collections and services, for example, digital repositories or open access archives.

Note: Section 9 “Body of knowledge” lists multiple technologies used in each application software. Since the technologies evolve and the applications keep being updated, please refer to the documentation on the application software homepages for details of the latest information.

3. Learning objectives

a. Students are able to describe the features and technologies of the DL application software.

b. Students should be able to search and add items to the digital library systems built by the DL application software.

Note: The following optional objective, 3.c, might be achieved through a semester-long class project, which is to develop a DL system using application software. For details, please see ‘Optional semester-long project 12.c’ in the section 12.

c. (Optional) Students are able to both install and configure a DL application software.

4. 5S characteristics of the module

All five S’s are present because a DL application software must have all the components to create minimal digital library, which are explained by 5S framework.

a. Streams: current DL applications are typically designed to deal with various types of data such as multimedia data (e.g., audio, images, videos) as well as text data.

b. Spaces: storage space to store digital contents and the user interface for the DL patrons to communicate with the system are present in the application software.

c. Scenarios: DL application and its patrons interact with each other following a series of steps to achieve tasks.

d. Societies: there are software developers, patrons (who purchase the software), administrators (who will install the software and administer the created DL system). Those groups of people make societies.

e. Structures: DL application software has their architecture, metadata formats used, etc., which have the structure.

5. Level of effort required (in-class and out-of-class time required for students)

To achieve Learning objectives 3.a and 3.b:

a. Out-of-class time:

Preparation for group presentations (Activity a): 5-7 hours (reading the assigned papers or web pages, creating and submitting concept maps individually and preparing group presentation slides)

Review of demos, etc. (Activity b): 0.5-1 hours (visiting the demo sites, trying basic services such as searching, browsing, item depositing or watching a short video tour)

b. In-class time: 1.5 hours (for presentations and question/answer session)

To achieve (optional) Learning objective 3.c:

a. Out-of-class time: it depends on the project. It is expected that this learning objective will be achieved through a semester-long project.

6. Relationships with other modules

The module 5-a: Architecture overview/models should be taught in advance so that the students could have the base knowledge about the DL architectures/models then learn about the application software, which were developed based on those knowledge.

After this module 5-b is taught, 9-a: Project management and 9-b: DL case studies module can be taught to provide students the real-world examples of the projects and DL systems created by the application software.

7. Prerequisite knowledge required (completion optional)

If DL application software is to be installed and configured as an optional learning activity and the instructor would like to supervise and help student groups, some knowledge about the pre-requisite software such as database systems (e.g., MySQL), Linux (e.g., Fedora Core, Ubuntu), HTTP server (e.g., Apache) as well as some knowledge about metadata, digital objects, indexing and collection building might be useful.

8. Introductory remedial instruction

None

9. Body of knowledge

Topic: EPrints (version 3)

1. Overview
    a. It was developed in 2000 as a direct outcome of Santa Fe meeting in 1999, where there was the first meeting of the Open Archives Initiative.
    b. It is commonly used as an institutional repository
    c. It has been developed at the University of Southampton School of Electronics and Computer Science
    d. Version 3 was released in January 24, 2007 whose features are introduced in the section 2 below
    e. Open source under GPL license
2. Features
    a. Duplicate avoidance
    b. Auto complete for entering metadata
    c. Full-text search
    d. Metadata search
    e. Subscriptions
    f. Multi-language support
    g. Optional multi-lingual metadata

    (The benefits of the new features for administrators, developers, researchers, institutions, depositors, etc. are introduced below - excerpt from Eprints homepage at http://www.eprints.org/software/v3/)

    a. Repository managers

i. Lower the barrier for your depositors while improving metadata quality and the value of your collection (with metadata auto-completion feature)

b. Depositors

i. Time saving deposits (with metadata auto-completion)
ii. Import data from other repositories and services using the scripts provided by the software homepage
iii. Auto-complete-as-you-type for fast data entry

c. Researchers

i. Optimized for Google Scholar
ii. Works with bibliography managers
iii. Works with desktop applications and new Web 2.0 services
iv. RSS feeds and email alerts keep you up-to-date

d. Developers

i. Tightly-managed, quality-controlled code framework
ii. Flexible plug-in architecture for developing extensions

e. Webmasters

i. Easily integrate reports, bibliographic listings, author CVs and RSS feeds into your corporate web presence

f. Institutions

i. High specification repository platform for high visibility, high quality institutional open access collections
ii. Conform with research funding agency's open access mandates

3. Content types
   a. Text
   b. Multimedia (image, audio, video)
4. Technologies used
   a. Unix-like OS (e.g., Linux)
   b. Written in Perl (allows rapid development and modification)
   c. XML (for import/export of data, partial configuration)
   d. Apache server with mod_perl installation
   e. MySQL database
   f. Unicode (UTF-8 encoding)
   g. OAI-PMH support

Topic: DSpace

1. Overview:
    a. It was developed as a collaboration between MIT libraries and Hewlett Packard Research Lab
    b. Research institutions use it to build various digital archives such as institutional repositories, learning object repositories, eTheses, electronic records management, digital preservation, publishing, etc.
    c. It is available to anyone free of charge under the BSD open source license
    d. DSpace federation coordinates the planning, research, development and distribution of DSpace. It also advocates for digital archiving initiatives open access to research literature
2. Features
    a. Long-term preservation supported
        i. There are three types of data formats (supported, known and unsupported types)
        ii. For all three types, DSpace does bit preservation: the preserved file remains exactly the same over time – not a single bit is changed
        iii. For supported type, DSpace does functional preservation: the file changes over time so that the material can be immediately usable as the same way it was originally while the physical media and digital formats change
    b. Interoperability
        i. It can export digital content with its metadata in XML-encoded file or METS
        ii. DSpace Java API can be customized to allow the interoperation with other systems
        iii. Handle System from CNRI is assigned to each digital item as a persistent identifier
    c. Support for Open Archives Initiative's Protocol for Metadata Harvesting (OAI-PMH)
        i. DSpace supports OAI-PMH v.2.0 as a data provider
        ii. OAI support was implemented using OCLC's OAICat
        iii. Institutions running DSpace can turn on and off OAI and choose to register as a data provider or not
3. Content types
    a. Text (articles, preprints, technical reports, theses, data sets, books, etc.)
    b. Multimedia (image, audio, video)
4. Standards
    a. Well-defined APIs for interoperability with other systems
    b. CNRI handles for persistent identifiers
    c. X.509 certificate-based access control
    d. Dublin Core metadata for digital objects
    e. OAI-PMH for metadata harvesting/providing
    f. METS profile can be used to export digital items
5. Technologies used

a. Operating system: Linux, Solaris, HP/UX, etc.
b. Server: Apache, Tomcat, OpenSSL
c. Indexing/searching: Lucent
d. Database system: PostgreSQL, JDBC
e. Data item id: CNRI Handle System
f. Jena (RDF history system)
g. Java, JSP, Servlets
h. JUnit (testing) and Log4j(logging)

Topic: Greenstone

1. Overview
    a. It was developed and distributed as an international cooperative effort established in 2000 by the University of Waikato with UNESCO and Human Info NGO, “New Zealand Digital Library Project.”
    b. Its aim is to empower users especially in the universities, libraries and public service institutions to build their own digital libraries.
    c. It is a suite of software that has ability to build new digital library collections and provide services for them.
    d. Open source under General Public License (GPL)
2. Features
    a. Installation of GSDL
        i. It runs on Windows, Unix/Linux, and Mac OS/X. It can be installed easily by using the ready-to-use binaries which is included in the distribution (but some functionality is limited).
        ii. It might be installed on a laptop for personal use (built-in web server), or run on the main web server (Apache or Windows IIS).
    b. Collection building
        i. It can harvest documents over OAI-PMH to include them in a collection
        ii. Full text tagging is supported for hierarchical document browsing
        iii. Automatic text extraction and indexing are provided
        iv. Data compression is supported
        v. Metadata
            1. Automatic extraction of simple metadata
            2. Explicit metadata via classifiers
            3. Used for browsing and searching
        vi. Multiple languages supported via Unicode
    c. Browse and search provided
        i. Full text search
        ii. Metadata field search
        iii. Either Boolean or ranked (when indexed with MG indexer)
        iv. Search history, search term highlighting, etc.
    d. Presentation
        i. Search results formatting available
        ii. Homepage customization available

    e. Collection administration
        i. Adding new documents (batch operation)
        ii. Usage monitoring
        iii. Security issues
    f. Interoperability
        i. Any Greenstone collection can be exported to DSpace
        ii. Any DSpace collection can be imported into Greenstone
        iii. Any collection can be exported to METS (in the Greenstone METS Profile) and Greenstone can ingest documents in METS form
    g. Customizable, extensible
        i. New document and metadata formats can be accommodated by writing 'plug-ins' in Perl
        ii. New metadata browsing structures can be implemented by writing 'classifiers.'
        iii. User interface can be customized using 'macros' written in a simple macro language
        iv. CORBA protocol allows agents (e.g., written in Java) to use all the facilities associated with document collections
3. Architecture
    a. Receptionist
        i. Provide user interface
        ii. User input accepted
        iii. Page generation
        iv. Send to appropriate collection server
    b. Collection server
        i. Collection content management
        ii. Search/filter information
        iii. Return results
        iv. Handle multiple collections
    c. Metadata supplied by communities
4. Content types
    a. Text
    b. Multimedia (image, audio, video)
5. Standards
    a. Dublin Core metadata for digital items
    b. Z39.50 client-server protocol for searching and retrieving information from remote computer databases.
    c. Support for OAI-PMH both as a client and a server
    d. Unicode for multiple language support
6. Technologies used
    a. Greenstone runs on all versions of Windows and Unix/Linux and Mac OS-X.
    b. Apache HTTP server
    c. Source code in C++ (experimental Greenstone v.3 is written in Java) and Perl available
    d. Greenstone provides a choice of three indexing tools

i. MG is the default indexer. It does section level indexing and the searches can be either Boolean or ranked. For phrase searching, Greenstone does 'AND' search on all the terms.
ii. MGPP (MG plus plus, new version of MG). It does word level indexing, which provides fielded, phrase and proximity searching. Boolean searches can be ranked. Document/section levels and text/metadata fields are all handled by the one index. It's a bit slower compared to MG when large data is to be indexed considering MGPP does word level indexing.
iii. Lucene was added for incremental collection building, which cannot be provided by MG and MGPP. It handles field and proximity searching but only at a single level for example, complete documents or individual sections but not both. It also provides single-character wildcards and range searching.

e. Multiple GNU software are integrated
   i. Apache web server
   ii. Perl
   iii. wget to download html pages from the web
   iv. XML::Parser used to read and write internal XML documents
   v. Stemmer for English document
   vi. CVS for version control
   vii. GDBM for database
   viii. and many more

Topic: CONTENTdm

1. Overview

a. It was conceived by the Center for Information Systems Optimization (CISO) at the University of Washington. It was then taken over and extended by the Online Computer Library Center (OCLC).
b. It is commercial software.
c. Its users are universities, public libraries, government entities, museums, non-profit organizations, etc.
d. It is 100 percent web compatible so the servers and collections can be administered remotely. There could be maximum of 50 'acquisition stations', which are remote locations for items and their metadata entry. Those data entered through the acquisition stations are stored and provided by the central CONTENTdm server.
e. Collection sharing is supported.

   i. Collections can be added to OCLC WorldCat catalog system so that the user collections can be part of WorldCat's 80 million record global catalog.
   ii. CONTENTdm functions as OAI data repositories for the users who want their metadata available for harvesting.

iii. Its Multi-Site Server allows users to query multiple CONTENTdm servers from a single user interface.

2. Features (based on http://www.oclc.org/contentdm/about/default.htm)

   a. It supports both text documents and multimedia. For example, it builds documents, books and other multiview and multipage materials. It can also present video and audio files with related transcripts.
   b. By using the batch import tools, it can import images and metadata quickly and easily as well as text files for full-text searching.
   c. By utilizing the compound object import wizard, CONTENTdm can import multiple compound objects, such as newspapers, in batches. I also can queue multiple compound objects and process them during off-hours to not slowdown the system use.
   d. It supports JPEG2000, which is a format for high-quality and large format images without a browser plug-in.
   e. To prevent unwanted copying of images it manages, CONTENTdm has three different options for image rights: band, brand or watermark. Band uses a band of color and words (in here, a 'band' means a layer in a digital image. The term originally came from electrical engineering field to represent a range of wavelengths or colors). Brand uses icons and words. Watermark uses grayscale images.
   f. For digitized text documents, CONTENTdm provides an integrated Optical Character Recognition (OCR) capability for full-text searching. Users will be able to search words in the digitized text in addition to searchable metadata fields within your collections. When viewed, items prepared with this feature will display highlighted search terms within the digitized document image.
   g. To index subjects of various still images (so that they can have consistent and uniform metadata), CONTENTdm uses the *Library of Congress Thesaurus for Graphical Materials I* (TGM I), which provides a controlled vocabulary to describe activities, objects, types of people, events or places. Proper noun names of those are excluded. As an option, you can develop your own controlled vocabulary to index images.
   h. It provides customizable user interfaces—Create predefined queries and customized interfaces to collections.
   i. Its flexible search features include Dublin Core and Latin-1 character set support, Boolean search and advanced search option. Advanced search option provides search-by-fields, across all fields, by proximity, and across one or many collections. CONTENTdm also auto-generates the search terms based on the existing metadata.

3. Content types

   a. Text
   b. Multimedia (e.g., image, video, audio)

c. Compound objects (items which consist of multiple views. For example, two-sided objects such as postcards, brochures, ticket stubs, or six-sided objects such as images of a chair seen from six different directions)
    a. CONTENTdm allows the users to define compound objects so that all the views of a compound object can be retrieved.
d. Null data type support for the items not yet in the system
e. URL data type support allows lengthy video and audio files stored in the streaming media server to be accessed through CONTENTdm.

4. Standards and technologies

a. CONTENTdm is fully compliant with OAI-PMH v.2.
b. Its default metadata templates are Dublin Core and Visual Resource Association (VRA) Core. Collection admins can still add their own descriptions.
c. It is Z39.50 (client-server protocol to access and retrieve information in remote computers) compatible through ZCONTENT, open source software developed by the Univ. of Utah Marriott Library. ZCONTENT allows users to access the collections of CONTENTdm and download items.
d. XML is used for all the internal structure description. For example, it is used to export the metadata descriptions in order to work with other systems that have different metadata standard.

10. Resources

Note: Feel free to read about features, technologies and (optionally) installation and configuration manuals as well as the assigned portion in the software homepages.

- Eprints 3
    - Reading for students
        - Read ‘Introducing EPrints 3’ and watch short QuickTime video clips at http://www.eprints.org/software/v3/
- DSpace
    - Reading for students
        - Visit DSpace homepage at http://www.dspace.org/ and read ‘About DSpace’ under ‘New to DSpace?’ on the top left pane.
    - Advanced reading for students (optional) and instructors
        - DSpace architecture review group, “Toward the next generation: Recommendations for the next DSpace Architecture”, January 24, 2007. http://wiki.dspace.org/static_files/0/0e/DSpace-recs.pdf
- Greenstone
    - Readings for students
        - Ian H. Witten and David Bainbridge, A brief history of the Greenstone Digital Library Software, at

        - http://wiki.greenstone.org/wiki/gsdoc/others/Greenstone_history.htm
        - Katherine J. Don, David Bainbridge, and Ian H. Witten, The design of Greenstone 3: An agent based dynamic digital library, at http://www.greenstone.org/docs/greenstone3/gs3design.pdf
    - Advanced readings for students (optional) and instructors
        - Ian H. Witten and David Bainbridge. (2003). How to build a digital library. Morgan Kaufmann.
- CONTENTdm
    - Readings for students
        - Visit http://www.oclc.org/contentdm/about/default.htm and read the topics under 'About' on the left pane.

11. Concept maps (created by students)

Note: IHMC Cmap Tools is an open source client tool to create concept maps. CmapServer enables the users to collaborate and share concept maps anywhere on the internet. Both software can be downloaded freely for educational purposes from http://cmap.ihmc.us/download/index.php

12. Exercises / Learning activities

a. Individual learning activity: Interacting with software demos

Prior to the class session, each student should complete the following activities. Students may work individually or together.

- (EPrints demo) Try searching and browsing. You need to create an account if you want to try depositing an item. Examine the metadata fields when you enter the metadata while depositing an item.
    - Demo site at http://demoprints3.eprints.org
- (DSpace demo) Interactive demo for students
    - Learn how to submit an item at http://libraries.mit.edu/dspace-mit/build/dspace-demo.html
    - Try searching and browsing at http://dspace.mit.edu/
- (Greenstone demo)
    - Demo page for searching for an item at http://diglib.auburn.edu/gsdlr/cgi-bin/library?site=localhost&a=p&p=about&c=demo&ct=0&l=en&w=utf-8
- (CONTENTdm) Watch the four minute tour video clip at
    - http://www.oclc.org/contentdm/tour/tour.htm

b. Group presentations on specific application software

Note: These group presentations will substitute for a formal lecture by the instructor. The instructor should be prepared to fill in gaps or make corrections if any of the presentations are incomplete or misleading.

- During the previous class, students form into four groups. Each group chooses DL application software for their group presentation. Readings are assigned from the Resources list in section 10.
- The students in a group should work together to create their presentation slides explaining the features and other information of the software such as services, technologies and standards used.
- In the class, each group gives a presentation about their application software followed by a question and answer session. Each of the four presentations should be allowed 15-20 minutes.

c. Optional semester-long project (group activity)

- Step 1: Students form a group and meet with clients who want to have a customized DL system developed.
- Step 2: The clients give the student groups specifications of the DL systems they want.
- Step 3: Each student group explores different DL application software to find the most appropriate application to meet their client's needs.
- Step 4: Student group installs the application software chosen in step 3, including installation of any pre-requisite software. For example, to install Eprints 3 in a linux machine, the Perl programming language along with its multiple modules, MySQL database and Apache server should also be installed in advance.
- Step 5: The installed application software is customized. For example, the students might configure the subject classification system as the Library of Congress (LOC) system or ACM classification, change the appearance of the user interface, modify the metadata fields used, etc.
- Step 6: The client verifies the installed DL application software is appropriately configured to meet their needs.
- Step 7: Student groups begin to create collections by adding the items provided by their clients to develop a DL (e.g., adding a group of pictures to create the Digital Library of Native American History or the Virginia Digital Museum of Cars, etc.).
- Step 8: Student group members make sure all the services of the developed DL system work well.
- Step 9: Clients evaluate the developed DL system and the student group refines it based on the feedback.

13. Evaluation of learning objective achievement

Note: Since the learning objectives and the learning activities are in one-to-one mapping relationships, the performance and the quality of the learning activities achievements are evaluated as the means to evaluate the learning objectives of this module.

a. Individual concept maps on specific application software

- After the class, each individual student creates one or more concept maps for the different application software packages and submits them to the instructor. The concepts maps are expected to demonstrate the student's overall understanding of all four software packages introduced in this module.
- The concept maps should be evaluated in terms of their comprehensiveness (did they include all the major concepts covered in the module?), their richness (were the concepts well-connected?), and their organization (was there a clear depiction of the concepts and their relationships?).

b. Group presentations on specific application software

The group presentations described in section 12 could be graded, to evaluate students' learning.

- Group presentations might be evaluated in terms of their comprehensiveness (did they include the important features and characteristics of the software?), their clarity (did they explain the software in a way that it could be distinguished from the alternative software packages?), and the quality of the presentation (e.g., slide quality, presentation style, use of time, and Q/A session).

c. Optional semester-long project

- Each instructor may develop a different method for evaluating the learning achieved through the project. We might suggest that points be assigned as follows.
  - The DL application software is incorrectly installed and not working (0 points)
  - DL application software as well as all the pre-requisite software is correctly installed (3 points)
  - DL application software is installed and fully configured (6 points)
  - The DL system is fully configured and a collection(s) is created with the data provided by the clients (10 points)
  - All the features of the new DL system are fully functional (15 points)

14. Glossary

Application software is a complete, self-contained program that performs a specific function directly for the user. This is in contrast to system software such as the operating system kernel, server processes, libraries which exists to support application programs and utility programs. – Dictionary of Computing –

API (Application Programming Interface) The interface (calling conventions) by which an application program accesses operating system and other services. An API is defined at source code level and provides a level of abstraction between the application and the kernel (or other privileged utilities) to ensure the

portability of the code. An API can also provide an interface between a high level language and lower level utilities and services which were written without consideration for the calling conventions supported by compiled languages. In this case, the API's main task may be the translation of parameter lists from one format to another and the interpretation of call-by-value and call-by-reference arguments in one or both directions. – Free On-Line Dictionary Of Computing –

CORBA is the acronym for Common Object Request Broker Architecture, OMG's open, vendor-independent architecture and infrastructure that computer applications use to work together over networks. Using the standard protocol IIOP, two application programs that are based on CORBA but developed by different vendors, on different operating systems, programming languages can interoperate with each other. – Object Management Group (OMG) –

Open URL is a type of URL that contains resource metadata for use primarily in libraries. The National Information Standards Organization (NISO), has developed OpenURL and its data container (the ContextObject) as international ANSI standard Z39.88. On 22 June 2006, OCLC was named the maintenance agency for the standard.

Dublin Core metadata element set is a standard for cross-domain information resource description. It provides a simple and standardized set of conventions for describing things online in ways that make them easier to find. Dublin Core is widely used to describe digital materials such as video, sound, image, text, and composite media like web pages. Implementations of Dublin Core typically make use of XML and are Resource Description Framework based. Dublin Core is defined by NISO Standard Z39.85-2007

Z39.50 is a client server protocol for searching and retrieving information from remote computer databases. It is covered by ANSI/NISO standard Z39.50, and ISO standard 23950. The standard's maintenance agency is the Library of Congress. Z39.50 is widely used in library environments and is often incorporated into integrated library systems and personal Bibliographic Reference software. Interlibrary catalogue searches for interlibrary loan are often implemented with Z39.50 queries.

OAI-PMH (*Open Archives Initiative Protocol for Metadata Harvesting*) is a protocol developed by the Open Archives Initiative. It is used to harvest (or collect) the metadata descriptions of the records in an archive so that services can be built using metadata from many archives.

XML (**Extensible Markup Language**) is a general-purpose markup language. It is classified as an extensible language because it allows its users to define their own tags. Its primary purpose is to facilitate the sharing of structured data across different information systems, particularly via the Internet.

SOAP (Service Oriented Architecture Protocol) is a protocol for exchanging XML-based messages over computer networks, normally using HTTP/HTTPS. SOAP forms the foundation layer of the Web services stack, providing a basic messaging framework that more abstract layer can build on.

15. Additional useful links

16. Contributors

Seungwon Yang (Virginia Tech, development), Eunyee Koh (Texas A&M, evaluation), Uma Murthy (Virginia Tech, evaluation), Jeff Pomerantz (UNC, evaluation), Sanghee Oh (UNC, evaluation), Barbara Wildemuth (UNC, evaluation)