Developing a Student-Friendly Repository for Teaching Principles of Repository Management

Palakorn Achananuparp and Robert B. Allen
College of Information Science and Technology
Drexel University
Philadelphia, PA 19104
pkorn@drexel.edu, rba@drexel.edu

Abstract
While several digital repository systems are widely adopted as tools for building digital libraries, they are rather complex or required major customization to be used as tools for teaching principles of repository management. As these tools offer a wide range of functionalities to support all kinds of applications, they are too complex to be used as a basic learning environment. In this paper, we describe a student-friendly repository application currently in development. The major goal is to provide an easy-to-use web interface for teaching principles of repository management. Interactive and responsive environment are provided through AJAX framework. Moreover, we plan to introduce it as a service-based framework in repository management to the students.

1. Introduction

Many programs have begun to offer courses in digital librarianship albeit different curriculum designs. Because existing software can be adopted as tools to teach the principals of digital library, it seems reasonable to just select some of them that might be able to satisfy the instructors' requirements. Choices of software platforms for building digital repositories include general-purpose content management systems (CMSs), or digital library systems, such as Fedora [11], DSpace [13], or Greenstone [15]. While several of these systems are widely deployed and have been used to develop working digital libraries they were too complex or required major customization to be used for teaching. For instance, we had great difficulty extending the basic metadata set in several of the tools to include image metadata.

The difficulty of adopting general-purpose tools for digital library education stems from the usage scenarios in which these tools are designed for. Because the majority of the repositories are general-purpose, they provide a wide range of functionalities that support all kinds of applications, from publishing and organizing websites to building the institutional repositories. However, putting all of these features together tends to make the tools too bloated to be used as a learning environment. We believe that the students will benefit more from a more controlled environment that provides proper scaffolding to help the students learn the concepts of digital repository. Moreover, with a minimal approach to user interface design, the students should be able learn how to use the tool very quickly.

2. Related Work

There are quite a few web-based repository applications as well as general-purpose CMSs. However, these applications were not specifically designed as the tools for teaching principles of repository management. Nevertheless, there are some open-source applications which have similar design and architecture as ours [5][6][9]. For example, Elated [5] is a lightweight, web-

based interface built on Fedora. Their goals are to offer an easy-to-use web interface to Fedora while utilizing Fedora's flexible content models and architecture. MyLibrary [9] aims to help users cope with information overload problem in digital library interface by providing a user-driven, customizable interface.

Several aspects of our tool distinguish it from the others. We have not seen many web-based repository tools that adopt an emerging technology framework like AJAX (Asynchronous Javascript and XML) [7]. Although there are not many studies to evaluate the impact of AJAX on the user's task performance, the usually perceived benefits of AJAX include reducing time spent waiting for data transmission, reducing time spent completing a particular task, reducing steps to complete the task, and improving application responsiveness [14]. In addition, most web-based repository tools only cover common repository functionalities, such as managing repository objects, managing metadata set, searching, etc. In contrast, our tool tries to emphasize the notion of repository services as well. As digital repositories become more interconnected [2], it is important to think about interoperability between repository services and ways in which interconnected repositories can serve patrons.

3. Student-Friendly Technology

Our goal is to build a lightweight, web-based digital repository application that is suitable for an educational as a student-friendly repository, we emphasize on the ease-of-use of the system rather than competing with full-feature repository applications. Specifically, we implement it with AJAX, a rich-internet application (RIA) technology. A flexible and responsive environment can greatly improve the students' task performance and engage them in learning the principles of repository management. We do not intend to replace existing repository software with ours. As the students graduated and entered the workforce, they will have to work on a different tool depending on their organization. Rather, our goal is to help them establish a strong conceptual model of digital repository technology and practice. And, in turn, they will be better prepared to use any existing tools.

The application will be developed in conjunction with our digital library curriculum to ensure that it meets learning objectives and serves as an effective learning environment. We are using an iterative design process in which we obtain feedback from the students based on the system prototype. Our emphasis here is to reach the simplicity in the design yet maintain the major functionality of basic repository. In doing so, we will employ usability techniques such as heuristic evaluation.

3.1 Repository Module

There are two main modules in our system: *the repository module* and *the administration module*. The repository module allows students to access items in the repository. This is an extension of the service we have developed for collecting image descriptions from students in a course of *Content Representation* [1]. The students can view the items and associated metadata. In addition, the students can modify information in the metadata given proper account privilege. A simple metadata search interface is also provided.

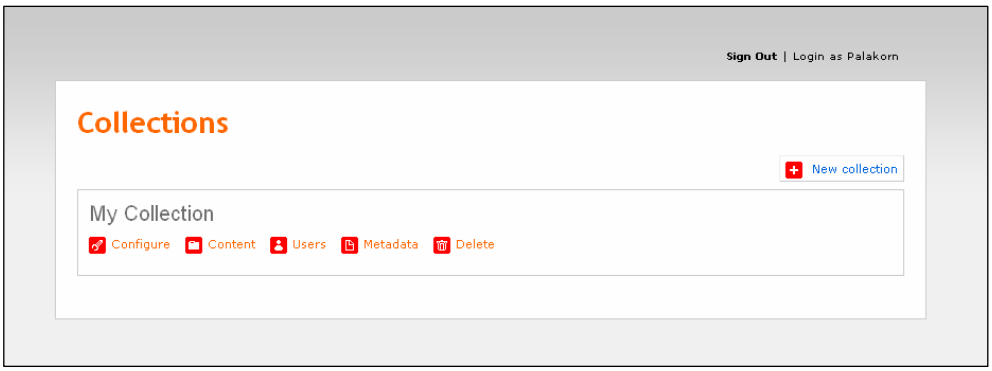

Figure 1. The main collection management page

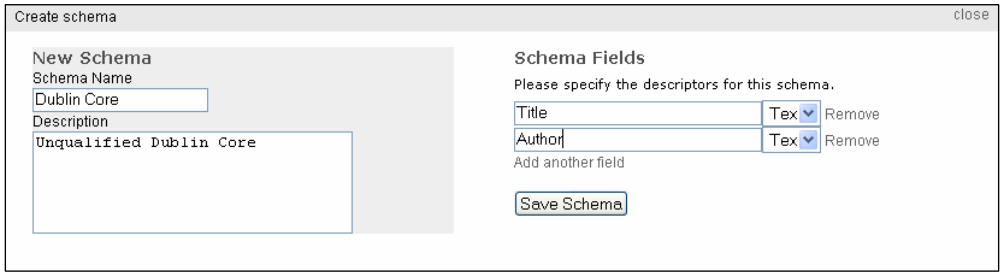

Figure 2. Metadata schema editor

3.2 Administration Module

The administration module provides a user interface to manage the repository. The major functionalities of the administrative module are structured around the collection-level tasks. To this end, the system provides four subcomponents: *collection manager*, *content manager*, *user manager*, and *metadata manager*. Each component provides basic functionalities to manage objects in a corresponding component.

To illustrate administrative workflow, consider the following example scenario. First, a new registered user enters login information to access the administrative interface. After a successful authentication, the user is redirected to a collection management page. The user creates a new collection by providing basic information, such as collection title, description, storage location, viewing privilege, and managing privilege. Once the new collection has been created, the user has a few options. The user can choose to upload one or more items onto the collection via collection submenu. Or, the user can add a new set of users and assign specific privilege to the users. Alternatively, the user could choose to create a default metadata schema for this collection.

4. Service Management

Through the service management interface, we will emphasize the roles of service-oriented architecture to the students. The service management interface allows the students to make specific services locally available in each collection as well as globally available across collections. An important question is the kinds of services we should include in the system. As there are numerous of repository services being implemented in various frameworks, there is the need to consolidate them into logical categories. Table 1 shows the example services

across various frameworks. Generally, the most common services across various frameworks are related to repository access and indexing.

Table 1. Summary of Service Frameworks

Service framework	Examples of services
Fedora Services Framework [11]	- Open URL Access Point - Fedora Search - Fedora Workflow - Directory Ingest - Preservation Integrity - OAI Provider - Federation PID Resolution - Preservation Monitoring - Event Notification
The JISC Information Environment Architecture [12]	- Authentication and authorization - Service registries - Metadata schema registries - Identifier services - Institutional profiling services - Terminology services
The OCKHAM Initiative's Testbed Services [9]	- Interoperation service - OPI-PMH-to-Z39.50 searching service - Alerting service - Browsing service - Conversion service - Cataloging service - Pathfinding service
The Making of America (MoA) II Digital Library Service Model [8]	Three layers: services, tools, and digital library objects. *Services* are provided through *tools* that discover, display, navigate, and manipulate *digital objects* from distributed repositories.
California Digital Library (CDL) Common Framework [3]	- Ingest, Indexing, Access (basic), Administration and Account Management - Search and Browse, Capture (for defining web crawls, harvests and other content gathering mechanisms), Rights Management, Collection Management, Metasearch

We endorse the Digital Library Federation (DLF)'s analysis of the need to establish service framework for digital libraries [4]. Particularly in the educational context, having a shared model would provide a foundation to help the students understand the issues more effectively than merely introducing assorted services.

To our knowledge, there are currently no dominant service frameworks. Rather, most frameworks are still a work-in-progress. The example provided in DLF progress report only illustrates abstract service framework that describes the framework in context of *business requirement*, *business process*, and *business function*. Thus, we plan to explore different ways to classify repository services and incorporate the final service framework/taxonomy into our system as well. For example, we might categorize based on person/organization involved – *repository-to-user services* vs. *repository-to-repository services*. A user searching to find particular resources in repository would be classified to fit the first service scenario, while a repository providing metadata records through OAI protocol would be classified as the latter. Our current plan for the first iteration of the tool is to cover at least one repository-to-repository service, i.e. OAI data provider service.

5. Conclusion

We have described a student-friendly repository application currently in development. The major goal is to provide an easy-to-use web interface for teaching principles of repository management. To achieve that, we provide an interactive and responsive environment using AJAX framework. In addition, we plan to introduce a service-based framework in repository management to the students.

6. Acknowledgements

This project is based supported by the Institute of Museum and Library Services (IMLS) Grant RE-05-05-0085-05 for developing a Model Curriculum for the Management of Digital Information. Tony Moore and Dr. Xia Lin worked with us in the initial stages of this project and we thank them for their advice.

7. References

[1] Achananuparp, P., McCain, K. W., and Allen, R. B. (2006) A tool for teaching principles of image metadata generation. In *Proceedings of JCDL 2006*. ACM Press, New York, NY, 341.

[2] Besser, H. (2002) The Next Stage: Moving from Isolated Digital Collections to Interoperable Digital Libraries. *First Monday*, 7(6). http://firstmonday.org/issues/issue7_6/besser/index.html

[3] CDL Common Framework. http://www.cdlib.org/inside/projects/common_framework/

[4] Dempsey, L. and Lovoie, B. (2005) DLF Service Framework for Digital Libraries. May 17. http://www.diglib.org/architectures/serviceframe/dlfserviceframe1.htm

[5] Elated: a general-purpose web-based client for the Fedora Repository System. http://elated.sourceforge.net/

[6] Fez. http://sourceforge.net/projects/fez

[7] Garrett, J.J., Ajax: A new Approach to Web Applications. http://www.adaptivepath.com/publications/essays/archives/000385.php

[8] Hurley, B. J., Price-Wilkin, J., Proffitt, M., and Besser, H. (1999) The Making of America II Testbed Project: A Digital Library Service Model. http://www.clir.org/pubs/reports/pub87/contents.html

[9] Morgan, E. L., Frumkin, J., and Fox, E. (2004) The OCKHAM Initiative - Building Component-based Digital Library Services and Collections, *D-Lib Magazine*, November, 10(11).

[10] MyLibrary. http://dewey.library.nd.edu/mylibrary/

[11] Payette, S. and Lagoze, C. (1998) Flexible and Extensible Digital Object and Repository Architecture (FEDORA), *ECDL,* 41-59.

[12] Powell, A. (2005) A service-oriented view of the JISC Information Environment, November.

[13] Smith, M., et al., DSpace: An Open Source Dynamic Digital Repository, *DLib Magazine*, 9(1), 2003.

[14] White, A. Measuring the Benefits of Ajax. http://www.developer.com/java/other/article.php/3554271

[15] Witten, I. H., Bainbridge, D., and Boddie, S. J. (2001) Power to the People: End-user Building of Digital Library Collections. In *Proceedings of JCDL*.