Int J Digit Libr
DOI 10.1007/s00799-008-0049-6

REGULAR PAPER

Teaching digital library concepts using digital library applications

Jeffrey Pomerantz · June Abbas · Javed Mostafa

© Springer-Verlag 2008

Abstract Many digital library topics may be taught using digital library applications, in the context of project-based digital library courses. Several digital library applications exist, and these applications are used as teaching tools to illustrate the functionality of digital libraries as well as the design decisions that go into them. Using digital library applications as teaching tools provides a valuable learning experience for students, and may provide useful feedback to the developers of DL applications. This paper identifies and explores DL topics that may most effectively be taught using DL applications, in the context of project-based DL courses.

Keywords Digital libraries · Digital library applications · Education · Curriculum

1 Introduction

Hundreds of millions of dollars have been invested in digital library (DL) research since the early 1990s. This research includes a significant body of work on how DLs can aid education, but there has been no parallel investment to support teaching and learning about DLs. In order for development of DLs to continue in the future, it is essential for academic programs to educate information professionals who fully understand the processes by which DLs are developed and their users are supported, as well as the potential of DLs to offer novel information services.

An important component of this educational experience is for students to gain hands-on experience in using, designing, evaluating, and managing DLs. There are several DL applications in existence, and these applications may be used as teaching tools in DL courses. The primary function of DL applications is, of course, to enable the creation of DLs, and not to serve as teaching tools. Nevertheless, using DL applications in this way provides a valuable learning experience for students. In addition, by enabling teaching with DL applications, DL application developers may reap the benefit of a distributed pool of knowledge, expertise, and ideas that can inform future development. Thus, the interaction of DL application development and DL education can benefit both sides.

The purpose of this paper is to identify and explore those DL topics that may most effectively be taught using DL applications, in the context of project-based DL courses. This is not a research paper, but rather a critical consideration of the pedagogy of teaching about DLs. As such, two questions that are two sides of the same coin will be addressed in this paper: What DL-related topics may be taught by making use of DL applications? And: What functions of DL applications may be used to teach what DL-related topics? As a corollary to these questions, a third question will be addressed: What pedagogical techniques lend themselves to teaching DL-related topics using specific functions of DL applications?

Specific DL applications will be used as examples as appropriate during this discussion, but this paper is not a critique or evaluation of DL applications. There is a need for evaluations of DL applications; the only example of such an

J. Pomerantz (✉) · J. Mostafa
School of Information and Library Science, University of North Carolina at Chapel Hill, CB 3360, 100 Manning Hall, Chapel Hill, NC 27599-3360, USA
e-mail: pomerantz@unc.edu

J. Mostafa
e-mail: jm@unc.edu

J. Abbas
Department of Library and Information Studies, Graduate School of Education, State University of New York at Buffalo, 534 Baldy Hall, Buffalo, NY 14260-1020, USA
e-mail: abbasjm@buffalo.edu

Published online: 08 January 2009

evaluation of which the authors are aware is Goh et al. [20]. Goh and colleagues evaluate the functionality of DL applications, with the aim of identifying the "essential categories of features DL software should possess" (p. 361). This paper takes a different approach: rather than evaluating DL applications, this paper focuses on issues that may be raised in a course, using the functionality of DL applications as illustration. Likewise, not all topics that may conceivably be taught in a course on DLs will be addressed in this paper. Rather, only those DL-related topics that can be taught by making use of DL applications will be discussed. A complete discussion of all DL-related topics that need to be taught to appropriately train information professionals to work on DL projects would be too extensive for this paper. Additionally, the authors are currently engaged in projects to articulate the range of these topics and develop educational materials for teaching them [7,28,31].

2 Assignments in DL courses

Most definitions of DLs include the following elements: a DL is a collection of electronic materials, that collection is managed and organized, it is created and managed for (and sometimes by) one or more user communities, and technical and user services are provided that add value to the materials [2,6]. Most DL courses address these elements in some manner.

In two previous studies [30,32], the authors identified the topics taught in DL courses in Library and Information Science (LIS) and Computer Science (CS) programs. These studies revealed that architecture and project management are the topics on which readings are most frequently assigned, in both LIS and CS courses. Other topics that are frequently addressed include collection development, organization of information, services, and preservation. An understanding of these topics is critical for the work of developing and managing DLs, and DL courses in LIS and CS programs are designed to provide students with the skills necessary to conduct this work.

DL course instructors have by and large created and implemented their own courses. Naturally instructors, particularly those at the same institution, share syllabi and assignments. But there has to date been no coordinated effort among DL course instructors to share resources. Indeed, little systematic exploration has been performed of the content of DL courses. Pomerantz et al. [30,32] have begun to explore this issue. This research has identified 40 DL-related courses offered between 2003 and 2006 in ALA Accredited LIS Master's degree programs. Analysis of the syllabi from these courses identified 33 in which descriptions of course assignments were present. Of these, 14 courses (35% of the original 40) were project-based and included an assignment in which students built a DL or a prototype DL. There are two models of project-based DL courses that the authors have identified, though these really are more like two variations on a single theme. In the first of these models, the instructor identifies several small-scale projects for the students in the course, students work in groups, and each group takes on the entire task of planning and implementing one of these small-scale DLs. In the second model, the instructor identifies one large-scale project on which the whole class works: again students work in groups, and each group takes the lead on planning and implementing one aspect (e.g., digitization, metadata, etc.) of the larger DL project. Other literature on DL education has also identified project-based courses as being the primary means by which DLs are taught in LIS programs [26,29].

The task of these project-based courses is often to build a brand-new DL, for a group or an organization that desires but does not currently have one, sometimes on campus and sometimes in the local community. Sometimes these projects have been on the organization's wishlist for some time but are languishing, so to speak, for lack of requisite knowledge or staff time within the organization to take on the planning and implementation. Some examples of projects of this type that the authors have been involved in include the Bentley Snow Crystal Digital Collection of the Buffalo Museum of Science (www.bentley.sciencebuff.org) and the Karpeles Digital Manuscript Library (soistudent.sis.buffalo.edu/lis563/karpeles/). Often these projects are begun in a DL course and then handed over to the group or organization to sustain.

This paper proceeds from the assumption that the DL course being taught is project-based. In a project-based course, a DL application is often used, and this paper presents some topics that the authors suggest must be taught in any DL course, and provides some examples of how these topics may be taught using one or more of the available DL applications.

One of the most important considerations in a project-based DL course is selecting the application to use. There are many software applications and tools that may be used to build a DL. Some of these applications are specifically designed for building DLs: for example, Greenstone (www.greenstone.org) and Fedora (www.fedora.info). Some of these applications are designed for similar types of products to DLs, such as digital archives and institutional repositories: for example, CONTENTdm (www.contentdm.com), DSpace (www.dspace.org) and EPrints (www.eprints.org). Greenstone, Fedora, DSpace, and EPrints are all open source applications; CONTENTdm is the only commercial application of the lot, though it has a number of open source extensions. Additionally, several open source tools exist that may be used to construct components of DLs [33]: for example the Apache HTTP Server, Apache Tomcat, and MySQL.

Alternatively, DLs can be built using a wiki, or even with nothing but HTML pages.

Even in courses where one application is used to build a DL, of course, it is still possible to use other applications as examples for discussion. Indeed, comparing DL applications provides an excellent opportunity to compare the different approaches to DL architecture implemented in these applications. Having students use DL applications enables them to experience their broad range of functions and their limitations first-hand. The following sections address some of the functionality specific to various DL applications, the DL-related topics that may be taught using this functionality, and some instructional techniques that may be employed to take advantage of this functionality for teaching these topics.

2.1 DL applications used in DL courses

It is worth pointing out that all of the applications mentioned above are designed for building DLs, and not for pedagogy. There are many applications that are designed specifically to be used in educational settings (e.g., the Mavis Beacon® Teaches Typing series), and many applications that that are designed for other purposes for which there are educational modules (e.g., the Student Versions of SPSS). Neither of these is the case for the DL applications discussed above, however. It is therefore all the more noteworthy that DL applications are used as widely and as successfully as they are in DL courses.

Of the 14 courses identified by Pomerantz et al. [31,32] as being project-based courses, ten mandated or recommended the use of a specific DL application for this assignment. In nine of these courses that application was Greenstone; in one it was CONTENTdm. In another two courses no application was recommended, and the decision of how to construct the DL was left to the students. It is worth noting that even though it is a small sample, two-thirds of the project-based courses used Greenstone for building a DL.

A set of informal conversations with instructors of DL courses in North American LIS and CS programs in 2007 provided some of the reasons that many instructors of DL courses chose to use Greenstone:

- Versions exist for the operating system environments most often present in universities, and it is designed to be cross-platform;
- There is much published literature about it, including a textbook;[1]
- It is easy to set up and can be managed by students who do not have much programming or systems administration expertise;
- It is flexible enough to allow students to make many decisions about the construction of the DL; and
- It is XML-based.
- It is deliberately designed to be used by librarians, with the inclusion of the Greenstone Librarian Interface;

Other DL applications fulfill some of these requirements, but many instructors apparently consider Greenstone to be the best fit for their courses. It is perhaps understandable why DL course instructors do not use CONTENTdm in their courses: as a commercial application, it is likely to be inaccessible to instructors, as there is often a notable lack of funding in higher education for purchasing or licensing software for teaching. Many academic libraries use CONTENTdm to manage their digital collections, and it may be possible for the instructor or the instructor's school or department to negotiate an arrangement with the library to allow students to access to the library's CONTENTdm install. Negotiating this is dependent on the library having extra "seats" for their license, and a willingness to set up an online environment in which students experiment with CONTENTdm; this negotiation is therefore dependent on all parties having a fairly enlightened view of the integration of education and library operations, and thus may not be possible at all institutions.[2] It is less clear why Fedora, DSpace, and EPrints are not used more widely in DL courses, as all are open source applications. It may be that these applications are seen as serving functions that are not fully appropriate for a DL course. EPrints, for example, is designed primarily as a repository for documents, and instructors may wish to use an application that is more flexible. The authors suspect, however, that the last bullet item above, the existence of the Greenstone Librarian Interface, and the overall approach to Greenstone's development with student uses in mind [29,38], is a more critical decision point for instructors than our informal conversations were able to determine.

It is unknown, of course, whether or how many DL course instructors spend the time, either prior to teaching a DL course or during the semester, to conduct a full comparison of all of the available DL applications. This speaks to the need for resources for instructors, evaluating and comparing DL applications and related software tools. Goh et al.'s [20] evaluation of the functionality of DL applications, mentioned

[1] As of this writing, a second edition of the book How to Build a Digital Library is being prepared for Morgan Kaufmann Publishers.

[2] The authors would like to point out that precisely this sort of arrangement was negotiated with the University of North Carolina at Chapel Hill's library. Many thanks are therefore due to the Systems librarians at UNC-CH for their efforts to make this possible, and for their enlightened view of the integration of education and library operations. Although Greenstone is used in the DL course at UNC-CH, CONTENTdm is used to manage many of the library's digital collections.

above, is one useful resource, but more work along these lines is needed. A useful avenue for future research would be to identify the functionality, usability considerations, and other factors that instructors desire in a DL application for use in a course. Such work can inform future development of these DL applications to enable them to be more useful as teaching tools; some suggestions for this are discussed below.

3 DL topics that may be taught using specific functions of DL applications

This section will address those topics that are critical to the development of DLs that may be taught using DL applications. The following section will address topics relevant to DLs that may also be taught using DL applications. Topics addressed in this section are those functions of DL applications without which a DL could not be built or maintained; a DL could exist without the features addressed in the following section, but those features arguably add value to a DL.

3.1 Architecture and digital objects

As mentioned above, architecture is one of the most frequently addressed topics in DL courses. This is only natural, since arguably all other features of a DL are predicated on its architecture. One of the earliest and still one of the best discussions of this topic is Arms' [1] articulation of eight general principles of DL architecture. These eight principles are as follows:

1. The technical framework exists within a legal and social framework,
2. Understanding of digital library concepts is hampered by terminology,
3. The underlying architecture should be separate from the content stored in the library,
4. Names and identifiers are the basic building block for the digital library,
5. Digital library objects are more than collections of bits,
6. The digital library object that is used is different from the stored object,
7. Repositories must look after the information they hold,
8. Users want intellectual works, not digital objects.

These eight principles present only a generic description of DL architecture, however. Some of these principles, for example #1, are generic enough to describe a DL, a website, or any collection of electronic materials. On the other hand, some of these principles, for example #8, are specific enough to provide some guidance for the implementation of functionality in DL applications.

The documentation for each of the different DL applications provides a description of the architecture and data model of that application. The data model of a DL application determines the structure of the digital objects that are stored in the DL. At a high level of analysis, all digital objects have similar characteristics. As Arms [1] points out, a digital object must be stored in the digital library, it must have a unique name to identify it, it must contain some data as its content, and other data must be associated with it to describe its properties (metadata). Once all of these requirements are met, however, it is possible for different DL applications to treat digital objects quite differently.

The most notable of these differences lies in the complexity of the digital objects in a DL. Some DL applications allow the existence of complex objects: that is, objects that are composed of other objects, where these other objects may be objects in their own right. An example of this is a webpage, where images contained within the webpage may also be used in other webpages. In Fedora, for example, each component object is quite simple, being either "mime-typed data or metadata" ([19], Overview). Multipurpose Internet Mail Extensions (MIME) is a set of standards that dictate file formats that may be transmitted over the internet (www.iana.org/assignments/media-types/). Thus, in other words, in Fedora an object may be a file or data about a file. Additionally, every digital object in Fedora has one or more "disseminators," which is functionality that associates services with the object; in other words, every digital object in Fedora has a set of built-in behaviors. In DSpace, on the other hand, objects are not restricted to MIME types, and in fact may be more complex than objects in Fedora. Objects in DSpace may exist at a wide range of granularity, and objects at larger levels may contain one or more objects at smaller levels of granularity. Objects in Greenstone are yet again different. Documents are the central objects in Greenstone, and a document "may consist of several separate files," as in the webpage example above, or a document "may comprise complementary pairs of files—like a Word file and matching PDF version" [8, p. 24]. These are all examples of what Arms [1] means by "intellectual works, not digital objects" (section 8): from the user's perspective, the object is the document, or community, or collection, or other complex object with which it is possible for the user to interact. It is only from the perspective of the application that an object is treated in these various ways. It is only meaningful to the application for an object to be of one or another MIME type; few, if any, users would think of an object in this way.

It is interesting in and of itself that, in a digital environment, the apparently simple concept of "object" is not at all simple, and may be operationalized in many ways. A more important lesson in a DL course, however, is for students to consider in what sorts of environments and for what sorts of content these different definitions of "object" are

applicable and useful. A DL that is created to serve multiple diverse user communities may be most appropriately created in DSpace, since that application supports communities as objects. A scholarly journal, on the other hand, may publish articles in multiple formats—HTML and PDF are common for this purpose – and so Greenstone, which treats complementary pairs of files as a single document, may be the most appropriate platform for a DL. Of course, in the "real world" (that is, outside of the classroom), a study of the needs and requirements of the user communities, and the objects to be included in the collection, would ideally precede and inform any decision about which DL application to use. In a DL course, conducting such a study may not be feasible, but published user studies may be available to serve as examples.

Additionally, in a project-based course with the goal to build a sustainable DL for an organization, the choice of which DL application to use, or for that matter whether or not to use any DL application, is also determined by the organization's resources (i.e., technology skills of the staff, existing technologies/systems and migration and/or interoperability issues, and funding for migrating and sustaining the project). It is useful for students to learn about the preplanning that informs the instructor's decisions about choosing a DL application to work with, as in the "real world" these factors will be critical when a DL application is chosen for the DL project.

Of course any DL must accommodate the object model of the application that it employs. As stated above, several DL courses use Greenstone, so even if Greenstone's object model turns out not to be the most appropriate in principle for the course project DL, the students in that course must adapt. This in itself can be a learning experience. While there are tools that enable the migration of digital objects between DL applications [39], these may not be addressed in DL courses.

3.2 Ingest

Once the object model of the DL is developed and the characteristics of digital objects in the DL are defined, actual digital objects may be created. Unless a DL project is collecting only born-digital materials, digitization is a necessary aspect of any DL project. The functionality of digitization, however, is outside of the scope of DL applications; that is, DL applications possess functionality to enable importing and organization of digital content, but these applications do not possess the functionality of scanners, digital cameras, or other digital capture devices. This discussion will therefore proceed from the point at which materials already exist in digital formats, regardless of how these materials came to be available in digital format.

The ingest functionality has some similarities and some differences across various DL applications. At a very high level, the basic functionality of ingest is identical across all DL applications: a digital object is submitted to the DL, that object is prepared in some way, and once prepared is added to the collection of materials in the database underlying the DL. However, in the details of these steps, the various DL applications differ.

In DSpace, for example, objects to be ingested must be prepared as Submission Information Packages (SIPs), which include the digital object or objects to be ingested, and metadata about those objects. Once the SIP is imported, human reviewers may check the objects and optionally edit the metadata before they are submitted to the DL collection. When the reviewers sign off on the SIP, the ingest continues and a number of automatic functions are performed on the content ([37], Functional Overview). Greenstone and Fedora, on the other hand, do not have a step that allows a review of submitted objects and metadata, though a number of automatic functions are performed by these applications.

A similar set of automatic functions are performed on ingested objects across DL applications. DSpace assigns an accession date to the object; adds provenance data, a checksum for error checking, and a persistent identifier; and finally adds the object to the DL's database and includes the item in the index for any search tool in the DL ([37], Functional Overview). Greenstone, similarly, adds a persistent identifier to the object, and adds the object to the database [4].

The functions performed by different DL applications to perform similar tasks reflect design decisions made by DL application developers. As such, comparing these functions is a useful exercise for students, in order to develop an understanding of the capabilities of different DL applications. Comparing functions also enables an understanding of the design decisions underlying applications, helping students to perceive the DL application developers' intention for the purposes to which the application should be put. For example, the fact that DSpace adds a checksum to objects indicates that the developers believed that error checking and validation of objects was important. In other words, DSpace was designed to support archival efforts for which data integrity is critical.

3.3 Metadata

After ingest of a digital object, the association of metadata with that object is perhaps the most critical function that a DL application supports: without digitization there would be no content for the DL to contain, and without metadata the content in a DL would be difficult or impossible to access. Creation of metadata for a DL strongly influences the interaction that the user will have with the DL: principles of information architecture suggest the metadata scheme influences the searching that the user can do of the content in the DL, and the design of the interface with which the user may browse

this content [34]. An excellent example of this is the Digital Library for Earth System Education (DLESE, dlese.org), which utilizes four simple metadata schemes (Grade level, Resource type, Collections, and Standards) both to organize the materials in the DL, and as a means for users to limit their searches of these materials.

Most DL applications allow DL developers the flexibility to either use an existing metadata scheme, or to create a new scheme. Creating a new scheme, or modifying an existing scheme, requires the DL developers to construct a new or modify an existing Document Type Definition (DTD). This process requires a number of decisions to be made about the description of the objects in the DL. As mentioned above, Arms [1] suggests that digital objects must have unique names, but what other data must be associated with objects in order for them to be found and applied by users? Exploring these decisions can be an informative exercise in a DL course, since different types of objects will require different metadata fields to describe them. An audio file of music, for example, will require different descriptive metadata than the digital version of the score of that same piece of music, which will require still different descriptive metadata from a textual document about that piece of music. Development of a DTD also requires a number of decisions to be made about the user communities for the DL. In exploring these decisions, students can conduct user needs assessments (or read literature about user needs assessments) and review the DTDs from other DLs that serve similar user communities.

Because of the complexity of describing digital objects, the Dublin Core (DC) Metadata Element Set is a useful teaching tool in DL courses. The Dublin Core Metadata Initiative describes the DC Element Set as "broad and generic, usable for describing a wide range of resources" [13]. This flexibility and customizability is precisely what one wants when first introducing the concept of object description to students. Starting with these broad and generic elements, students can then discover where more specific elements are necessary for the particular objects in the DL they are developing, and how to create a DTD to define these more specific elements.

There are, however, many uses of metadata in DLs besides object description, including capturing data about the digitization process, and data to be used for management and administration of the DL itself. While none of the courses identified by Pomerantz et al. [32] used Fedora for DL projects, Fedora provides a good example of an application that makes use of metadata for a variety of functions, and instructors may wish to consider using it for this purpose. When a digital object is ingested into or exported from Fedora, it can be encoded in one of two formats: Fedora Object XML (FOXML), or the Fedora extension of the Metadata Encoding and Transmission Standard (METS, Library of Congress [25]). Students designing a digital library need to address the issue of which XML format is the most appropriate. FOXML was designed specifically for Fedora and it "directly expresses the Fedora digital object model" ([18], Intro. to FOXML). The Fedora digital object model may also be expressed in the Fedora extension of METS, though perhaps less directly, as METS was not created specifically to reflect Fedora's architecture. METS, however, is not an application-specific XML format, and therefore it may be easier to ingest objects from and export objects to other DLs encoded in METS. Students need to consider whether designing this ease of interoperability into their DL is necessary or desirable. Again, in the "real world," decisions about interoperability would be dictated by the needs and requirements of the user community and an organization's resources and existing technologies; in the classroom, discussion of DLs used by similar user communities may serve as realistic examples.

Digital objects may of course be described in both FOXML and METS. But these two formats may be used to encode more metadata about the objects in a DL than simple description. Disseminators were mentioned above, but disseminators are only one type of special digital object in Fedora. Fedora possesses functionality to enable the DL developer to create Behavior Definition (bDef) and Behavior Mechanism (bMech) objects, which are, respectively, the functionality that another digital object can possess, and the web services that must be associated with an object in order for it to possess that functionality ([16], Tutorial #2). For example, the "root format" of a digital object may be a Word document, but a behavior can be associated with this object so that the object is converted to PDF format when viewed. In other words, disseminators and behaviors are metadata, associated with particular digital objects, that "construct and dispatch service requests" [23, p. 128]. Behaviors are a use of metadata that is well beyond simple description; this is metadata used for the purpose of automation of DL services. It is important for students to learn that metadata is flexible in this way. While object description is important—indeed, critical—for the operation of DLs, metadata can be utilized in the service of functionality for objects well beyond what is possible in the physical world.

3.4 Communities

A DL in isolation may be a wonderful work of technical expertise, but without a user community it is little more than a collection of bits. Kilker and Gay [22] discuss the importance of identifying the "relevant social groups" to a DL (i.e., the stakeholder groups that have an interest in the DL). This interest may be as funders, developers, users, librarians, evaluators, or any number of other possible roles.

Stakeholder groups are not necessarily groups of users, though often the communities for which DL applications are designed are user communities. In DSpace, for example, the data model "is intended to reflect the structure of

the organization using the DSpace system," and as such the highest-level object in the data model is the Community ([37], Functional Overview). Communities in DSpace correspond to the largest unit of the organization using the DL (a laboratory, research center or department), and may be subdivided into sub-communities. Communities contain collections of digital objects, and a collection may be contained in multiple communities.

Because DSpace is the only DL application in which communities are explicitly represented in the application's data model (though not necessarily in the DTD), it makes an interesting case study for DL courses. As mentioned above, the DSpace data model contains objects at several levels of granularity: community, collection, item, bundle, bitstream, and bitstream format. A community may contain one or more collections, a collection may contain one or more items, and so on. It is a useful exercise for students to consider what types of objects are necessary or appropriate at lower levels, given the purpose of the DL, and the implications for higher-level objects. Given that a stakeholder group may be represented as a community, to which types of collections should that stakeholder group have access? What types of items are appropriate for inclusion in those collections for that community? What differences might there be between the collections and items appropriate for different stakeholder groups? All of these questions can form the basis for valuable discussion with students in a course on DLs.

3.5 Preservation

The preservation of digital content is a complex undertaking, and many efforts have been undertaken to address various aspects of this problem. One of these efforts is the Open Archival Information System (OAIS) reference model. The Consultative Committee for Space Data Systems [10] defines an OAIS as "an archive… that has accepted the responsibility to preserve information and make it available for a Designated Community" (p. 1–1), for information that requires long-term preservation. A key element of this definition is the Designated Community, which is itself defined as potentially including multiple user communities (p. 1–10).

This model of nested communities is similar to that employed in the DSpace data model. Tansley et al. [35] use this dovetailing of models as a vehicle for discussing DSpace's archival capabilities in terms of the OAIS reference model. They suggest that an item in DSpace fulfills the function of the OAIS Archival Information Package (AIP), in that it contains both the content of the digital object and metadata for the purpose of preservation (for example, metadata elements concerning provenance and rights management), or Preservation Description Information (PDI). Communities and Collections in DSpace also contain preservation metadata, as well as descriptive metadata that may be harvested by automated services (p. 450).

Fedora, on the other hand, was not designed with the OAIS reference model explicitly in mind, but in the past few years the Fedora Preservation Services Working Group has made a concerted effort to "recommend enhancements to the Fedora repository service… for new preservation-support services" [15]. This attention to preservation by the Fedora development community emerged in large part out of the Fedora Project and the Preservation of University Records Project, undertaken by the Digital Collections and Archives at Tufts University and Manuscripts and Archives at Yale University, and funded by a National Historical Publications and Records Commission (NHPRC) electronic records research grant [14]. This project sought to answer the question: "Can a Fedora repository, surrounded by the proper preservation policies, tools, and Fedora services, serve as the basis of a trustworthy preservation system?" (Digital Collections and Archives [12]). The conclusion that this project came to was that Fedora can, in fact, be used as the basis of a trustworthy preservation system, due to its flexibility in managing digital objects and the policies and services associated with these objects. This project made a number of recommendations that would forward Fedora's usefulness as the basis of a preservation system, and the Fedora Preservation Services Working Group has continued that effort.

DSpace and Fedora provide useful case studies of different approaches that DL applications may take to preservation. The architecture of DSpace was in part designed around the OAIS reference model, and as such approaches preservation as a task that can be implemented via algorithms and policies. Fedora has been found to be useful for preservation, but this was not a design consideration, and is rather a happy unintended consequence of its treatment of all actions on digital objects as services.

4 DL-related topics that may be taught using DL applications

The previous section focused on the functionality of DL applications, and used that as the basis for discussing the topics that may be taught using that functionality. This section takes the reverse approach. This section addresses some DL-related topics, the functionality of various DL applications that may be used to teach these topics, and some instructional techniques that may be employed to do so.

4.1 Use of open source tools

An extremely important aspect of a DL course—or any course that utilizes software applications—is providing students with exposure to and experience with open source tools. As

discussed above, there are several open source applications and tools for building DLs or components of DLs [33]. There are likewise many open source tools that are frequently used by information managers to provide access to semi-structured (e.g., HTML and XML) and unstructured textual and multimedia content. One basic tool for building such systems is web server software. The most common open source tool of this type is the Apache HTTP Server. It is a modular tool that allows the server manager to select specific components to be integrated at install time, in addition to its core service that involves serving HTTP requests generated from clients' browsers. Details of the functionality of the Apache software will not be discussed here, but some key facts about Apache and related tools are important to point out. First, this class of software that is capable of serving HTTP requests originating from common web browsers forms the backbone of the current web, and so to develop DLs that operate on the web it is essential that students acquire some level of familiarity with these tools. Second, there are some tools associated with the web server software that are becoming increasingly important, including Apache Tomcat and MySQL: the former is important for script-based expanded services that are triggered based on HTTP requests, and the latter is important for offering database services. Third, incorporation of "hands-on" experiences with tools that are actually used in the information profession is likely to make DL courses more exciting and effective for the students (at least this has been the experience of the authors of this article).

It is important to point out that the IT manager and her staff in the institution where DL courses are taught must be involved in planning and launching these courses. The authors recommend a set of dedicated hardware platforms for teaching these DL courses. Server-oriented software (such as Apache) can "open up" systems to the internet in a manner that makes them vulnerable to security breaches and undesired exploitations. The hardware platforms used to teach must be "isolated" properly from the outside network to ensure that experimental-level learning by the students can take place without security compromises.

4.2 Services

Given that stakeholders are central to DLs, services for these stakeholders are one of the most critical aspects of a DL. Just as different communities in DSpace may contain different collections and items, so too may different stakeholder groups find different services to be relevant and useful.

As Pomerantz et al. [30] suggest, certain DL-related topics are defined differently in the fields of LIS and CS, and one of these areas of difference is Services. The documentation from the various DL applications almost uniformly treats services as algorithmic, as they are generally treated in the field of computer science. For example, the DSpace documentation [36] discusses handle resolvers and metadata harvesting as services, and the Fedora documentation [17] discusses ingest and search tools as services. Services of these types are automated, provided by an algorithm either to a human user (as with search services) or to another automated process (as with harvesting services). In the field of LIS, on the other hand, services are generally thought of as provided by a human to another human. Reference and readers' advisory services are classic examples of these.

There is no DL application that the authors are aware of that contains functionality to enable human-intermediated services, though, to be fair, these are not the functions for which DL applications are designed, and other software applications exist that do enable these functions. Human-intermediated services add value to a library, physical or digital, and it is worthwhile for students to consider the types of value that may be added to a DL. Given the limitations of DL applications, however, as well as the complexity of implementing human-intermediated services, it may not be feasible for students to gain first-hand experience with such services in the context of a DL course. Other courses exist in LIS curricula, and hopefully in CS curricula as well, in which students may gain experience with human-intermediated services.

While a DL course cannot fully introduce students to human-intermediated DL services, the distinction between these services and automated services is an important one to address in DL courses, as it is illustrative of the disconnect that often occurs between individuals and organizations primarily concerned with information technology and those primarily concerned with users. The practice of librarianship in the modern age is heavily dependent on technology: searching databases, collecting statistics on materials usage, even providing reference service are only a few of the many functions for which today's librarians need to be technologically literate. New students in LIS programs are sometimes resistant to learning technical subject matter, as, being new to the field, they may not yet understand the degree to which modern librarianship is technological. Overcoming this resistance lies in exposing students to those aspects of the field of LIS for which technical expertise is required. Even if an LIS student does not end up working in a technical capacity in a library, she still needs to be able to communicate with those who are, in order to understand, manage, and evaluate the work of those individuals.

One technology-based frontier for library services that can be introduced in a DL course is Library 2.0. While there is still no widespread agreement as to the precise meaning of the term, Habib [21] provides the following useful definition: "Library 2.0 describes a subset of library services designed to meet user needs caused by the direct and peripheral effects of Web 2.0 services" (p. 22), where Web 2.0 services leverage the efforts of users themselves, either through users'

direct participation (e.g., "read/write web" applications such as blogs and social networking services) or secondary harnessing of collective intelligence (e.g., Google's PageRank algorithm). To a certain extent, Library 2.0 is the merger of the LIS and CS approaches to service: the algorithmic harnessing of human effort. Miller [27], for example, describes several services that harness the "library as platform," building on the services provided by the online catalog and other pre-existing library services. One of the attractive aspects of Library 2.0 is that applications are generally developed to be "lightweight," so that anyone with some knowledge of programming can create a new service. Indeed, both OCLC and Talis have recently held competitions to promote the development of new lightweight library applications , referred to as "mashups" since these generally integrate multiple web services to create a new service. The winner of the 2006 OCLC competition adds functionality to searches in online catalogs and other library databases (www.oclc.org/research/announcements/2006-09-28.htm); the winner of the 2006 Talis competition adds personalized views of library data into the user's Google home page (blogs.talis.com/panlibus/archives/2006/09/mashing_up_the_4.php). Thus far, there are no library mashups of which the authors are aware that have been developed to utilize DL content or tools. But who better to develop such tools than LIS and CS students? And what better venue in LIS and CS curricula than in a DL course in which to deliver the message to students that they can change the face of library services?

4.3 Use of DLs in cyberinfrastructure

In 2002, a Blue Ribbon Advisory Panel to the National Science Foundation chaired by Dr. Daniel Atkins investigated the emerging developments in cyberinfrastructure, and published a report in early 2003 [3]. Atkins et al. define cyberinfrastructure as "infrastructure based upon distributed computer, information and communication technology," and necessary for a knowledge economy (p. 5). The cyberinfrastructure report became a blueprint for a number of new NSF programs under the new Office of Cyberinfrastructure (OCI), and influenced a broad range of scholarly groups in the classic sciences, the social sciences, and the humanities. In this report a framework is presented in which DLs play a central role, and it is suggested that there is an inherent relationship between DLs and knowledge communities. A knowledge community, broadly defined, is any community containing members with mutual interest in using data and tools to investigate research problems by following the typical scholarly phases: origination of research questions, search for data and evidence, experimentation and exploration, analysis, and publication and dissemination.

Some of the services required by knowledge communities are closely related to core DL services. These include classic operations such as searching and presentation of content. Knowledge communities also place new constraints on expanding core DL services, however, which demand closer scrutiny and attention by DL educators. Some of these new services include: (1) ability to upload data, sometimes directly by users, (2) support for a wide variety of formats covering both experimental data and research articles, (3) data manipulation capabilities that range from common statistical operations to simulations and real-time graphical rendering of content, and (4) community or social network services to promote interactions and leverage the collective knowledge of the scholarly user group. This report also suggests that the development of cyberinfrastructure will have an impact on science and engineering education, through the need to develop environments and resources for teaching and mentoring, and scientific collaboration (p. 27). The growing interest in scholarly institutional repositories, for example, is a promising area for studying and understanding the linkages between knowledge communities and DLs, and may influence advances in DLs that clarify their role in the cyberinfrastructure framework. Institutional repositories are also a fertile area for projects in DL courses, providing an environment in which students may experiment with and develop new services for specific knowledge communities.

4.4 Feedback to DL application developers

All organizations that develop software, whether corporate or nonprofit, have some mechanism for users to contact the organization for technical support. Many corporations have the means to maintain a group that performs technical support, often by telephone, though increasingly via instant messenger applications. On the other hand, nonprofit organizations, open source communities, and university consortia – in other words, organizations such as those that develop the most commonly-used DL applications—often do not have sufficient means to maintain a technical support group. As a result, technical support must be accomplished by other means. The developers of Greenstone, DSpace, and Fedora all maintain at least two electronic mailing lists (commonly referred to as listservs): one for general discussions about the applications and technical support, and another for technical discussions among software developers. Maintaining listservs for these purposes shifts some of the responsibility for providing technical support from the application developers to the community of users of the applications, thus enabling more support questions to be answered.

There is no doubt, however, that DL application developers monitor these listservs closely, as these venues can be a rich source of feedback about bugs in the software, solutions to problems, and suggestions for functionality desired by the user community. Given their value to DL application developers, it would also be useful for students in DL courses

to monitor these listservs. Pomerantz et al. [32,30] did not find any DL courses in which students are required to monitor DL support listservs, nor are the current authors aware of any discussion in the literature of DL coursework of this nature. Nevertheless, for students to monitor DL support listservs may be useful for expanding their understanding of the application, and issues involved in its implementation and use.

Students in a DL course are, by definition, just beginning to learn about implementing DLs, and are likely to encounter problems and questions. These problems and questions are likely to have been encountered by others, however, and students may be able to find solutions by monitoring these listservs and their archives; in the event that a solution cannot be found, students can post a new question to a listserv. Further, instructors of DL courses presumably have also evaluated and reviewed the available documentation and technical support prior to selecting a DL application for use in a course. The instructor is therefore in a position to know about any issues with the available documentation and to build this sort of problem-solving into the course. By encouraging students to join in the community of users on a listserv, instructors are in a position to engage students in what Lave and Wenger [24] call "legitimate peripheral participation": engaging with a community of practice as newcomers while working to gain more familiarity with the knowledge of the community.

Conversely, by posting questions to listservs, students working on DL projects can be a valuable source of information for DL application developers. As noted above, the primary function of DL applications is not to serve as teaching tools. But given that DL applications—particularly Greenstone—do serve as teaching tools, it behooves DL application developers to encourage students' use and experimentation. As newcomers to DL development, students are likely to encounter problems and questions common to new DL developers, and DL application developers can monitor these questions as an "early warning" system for problems and questions that they are likely to be asked by others. This kind of early warning system would be particularly useful upon the release of new versions of DL application software, and as a guide for topics that are not addressed clearly in the application's documentation. As newcomers to DL development, students are also in a position to be extremely creative about the design and implementation of DLs, and this creativity may also be useful to DL application developers for ideas about functionality for future versions of the application and environments for promoting the use of the application.

5 Discussion and conclusion

As was mentioned above, not all topics that may be taught in a course on DLs have been addressed in this paper; only those topics that can be taught by making use of DL applications have been discussed. This is, however, a fairly wide range of topics, covering DL development from design to implementation, and covering both system-centered and user-centered approaches to DLs. Table 1 summarizes the above discussion exploring those DL topics that may most effectively be taught using DL applications, in the context of project-based DL courses, and presents brief answers to the questions posed in the Introduction to this paper. The order of topics in Table 1 is different than that presented above, however, as Table 1 presents these topics in the approximate order in which they might be addressed in a DL course.

Project-based DL courses provide the opportunity for instructors to use DL applications as teaching tools. This approach provides the opportunity for students to use these applications in a realistic fashion. This active learning approach [5] to a DL course allows students to gain experience that will be valuable for their future as information professionals. Engaging with the community of DL developers and implementers is a valuable experience for students' future work on digital library projects.

In a recent study, Choi and Rasmussen [9] identified knowledge of DL architecture and software as one of the most important areas of knowledge that digital librarians need to perform their work. Many of the other skills and areas of knowledge that Choi and Rasmussen identify are also covered in DL courses; several have been discussed here. Similarly, Croneis and Henderson [11] identified many of the topics discussed above in job descriptions for digital librarian positions. Having taken a DL course is useful for a professional seeking work on a digital library project, but having gained relevant skills by having participated in building a DL is potentially even more marketable.

To date, however, DL course instructors have by and large created and implemented their own courses. Naturally instructors, particularly those at the same institution, share syllabi and assignments. But there has to date been no coordinated effort among DL course instructors to share resources. The authors of this paper are currently engaged in projects to articulate the range of topics that could or should be included in DL courses, and to develop educational materials for teaching such courses. These resources will be shared among DL course instructors, and will hopefully encourage more instructors to offer DL courses at their institutions.

An important part of educational materials on any topic must be ideas and suggestions for pedagogical approaches. These can only be suggestions, however, as teaching style is highly personal, and every instructor has a unique approach to presenting material in the classroom or online. Even when adopting a syllabus from a colleague, it is common practice for an instructor to adapt the syllabus to fit his or her unique teaching style. Consequently, this paper has identified topics that may be taught using DL applications, but has not dictated

Table 1 DL-related topics and suggestions for DL course instructors using DL applications

DL topics that may be taught using DL applications	Functions of DL applications that may be used to teach DL topics	Instructional techniques that may be used to teach DL topics
Identification of stakeholder groups	DTD and data model development	Identification of how users and groups are represented in the application's data model and DTD
		Needs assessment
Open source tools	Access and security settings: not functions of DL applications, but functions of server operating systems that enable DL applications	Experiences with Apache HTTP Server, Apache Tomcat, MySQL, and other tools for providing access to web content
DL architecture	The object model used by the application for digital objects	Comparison of object models across applications
		Discussion of object models appropriate to different environments and content types
		Conducting or reading about user and community needs assessment
Preparation of digital objects	Ingest functionality	Comparison of ingest functionality across applications
		Discussion of the design decisions underlying ingest functionality and implied application purposes
Design of a metadata schema for digital objects	DTD development, assignment of metadata to digital objects	Brainstorming on uses of metadata
		Construction of a new or modification of an existing DTD
		Brainstorming on expansion of the Dublin Core Element Set
Provision of services to users	Algorithmic services enabled by the application: e.g., metadata harvesting and search tools	Brainstorming on adding value to digital objects and services that may be provided to enable this
Preservation	Data model and policies	Identification of preservation functionality in the application's design of digital objects, policies, and services
Cyberinfrastructure	Presentation, navigation, and data manipulation functionality; support for user communities	Brainstorming on applications' capabilities to support knowledge communities
Software development process	Technical support and community support mechanisms	Mechanisms for legitimate peripheral participation

or even made many recommendations for how that teaching should be accomplished.

Sharing educational materials is important, but it is only one step in advancing DL education. In addition to lesson plans, instructors need resources that they can use in their courses. A DL application "sandbox," for example, in which students gain access to and experiment with installed versions of the various DL applications, could be a valuable teaching tool. Whether these sorts of resources should be provided by individual institutions for their own students, or shared among the community of LIS and CS programs, is a question that this community should discuss. In order to educate information professionals who fully understand the processes by which DLs are developed and their users are supported, as well as the potential of DLs for affording novel information services, instructors and academic programs must reach out to DL developers, and provide students with the best possible resources and teaching.

Acknowledgments The authors wish to thank the many instructors of digital library courses for sharing details of their courses: Stephen Downie, Laurie Gemmill, Bryan Heidorn, Stacy Kowalczyk, Cavan McCarthy, Jerome McDonough, and Candy Schwartz. Thanks to Cathy Blake and Cal Lee for discussions on the technological dependence of librarianship and methods for introducing this to students. Particular thanks to Barbara Wildemuth for extensive feedback on early versions of this paper.

References

1. Arms, W.Y.: Key concepts in the architecture of the digital library. D-Lib **1**(1) (1995). doi:10.1045/july95-arms
2. Arms, W.Y.: Digital Libraries. The MIT Press, Cambridge, MA (2000)
3. Atkins, D.E., Droegemeier, K.K., Feldman, S.I., Garcia-Molina, H., Klein, M.L., Messerschmitt, D.G., et al: Revolutionizing Science and Engineering Through Cyberinfrastructure: Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure. Arlington National Science Foundation, VA. http://www.nsf.gov/cise/sci/reports/atkins.pdf (2003)
4. Bainbridge, D., McKay, D., Witten, I.H.: Greenstone Digital Library: Developer's Guide. http://www.greenstone.org/manuals/gsdl2/en/html/Develop_en_all.html (2004)
5. Bonwell, C.C., Eison, J.A.: Active learning: creating excitement in the classroom. ASHE-ERIC Higher Education Report No. 1. The George Washington University, School of Education and Human Development. Washington, DC. ED336049 (1991)
6. Borgman, C.L.: What are digital libraries? Competing visions. Inf. Process. Manage. **35**(3), 227–243 (1999). doi:10.1016/S0306-4573(98)00059-4
7. Brancolini, K.R., Mostafa, J.: Developing a digital libraries education program: JCDL 2006 workshop report. D-Lib **12** (7/8) (2006). doi:10.1045/july2006-brancolini
8. Buchanan, G., Bainbridge, D., Don, K.J., Witten, I.H.: A new framework for building digital library collections. In: Marlino, M., Sumner, T., Shipman, F. (eds.) Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 23–31. Association for Computing Machinery, New York. doi:10.1145/1065385.1065392 (2005)
9. Choi, Y., Rasmussen, E.: What is needed to educate future digital librarians: a study of current practice and staffing patterns in academic and research libraries. D-Lib **12**(9) (2006). doi:10.1045/september2006-choi
10. Consultative Committee for Space Data Systems: Reference Model for an Open Archival Information System (OAIS), CCSDS 650.0-R-2, Red Book, Issue 2. http://public.ccsds.org/publications/archive/650x0b1.pdf (2002)
11. Croneis, K.S., Henderson, P.: Electronic and digital librarian positions: a content analysis of announcements from 1990 through 2000. J. Acad. Librariansh. **28**(4), 232–237 (2002). doi:10.1016/S0099-1333(02)00287-2
12. Digital Collections and Archives: Tufts University, & Manuscripts & Archives, Yale University . Analysis of Fedora's Ability to Support Preservation Activities, version 1.0. http://repository01.lib.tufts.edu:8080/fedora/get/tufts:UA069.004.001.00011/bdef:TuftsPDF/getPDF (2006)
13. Dublin Core Metadata Initiative (DCMI): Dublin Core Metadata Element Set, Version 1.1. http://dublincore/documents/dces/ (2006)
14. Fedora and the Preservation of University Records: Yale Unversity, Manuscripts and Archives Department; Tufts University, Digital Collections and Archives. http://dca.tufts.edu/features/nhprc/index.html (2006)
15. Fedora Preservation Services Working Group: Working Group: Preservation. http://www.fedora.info/wiki/index.php/Working_Group:_Preservation (2007)
16. Fedora Project: Fedora Tutorial #2: Getting Started: Creating Fedora Objects and Using Disseminators. http://www.fedora.info/download/2.2/userdocs/tutorials/tutorial2.pdf (2005)
17. Fedora Project: Fedora System Documentation: Fedora Release 2.2. http://www.fedora.info/download/2.2/userdocs/ (2007)
18. Fedora Project: Introduction to Fedora Object XML (FOXML), Fedora Release 2.2. http://www.fedora.info/download/2.2/userdocs/digitalobjects/introFOXML.html (2007)
19. Fedora Project: Overview: The Fedora Digital Object Model. http://www.fedora.info/download/2.2/userdocs/digitalobjects/objectModel.html (2007)
20. Goh, D.H.-L., Chua, A., Khoo, D.A., Khoo, E.B.-H., Mak, E.B.-T., Ng, M.W.-M.: A checklist for evaluating open source digital library software. Online Inf. Rev. **30**(4), 360–379 (2006). doi:10.1108/14684520610686283
21. Habib, M.C.: Toward Academic Library 2.0: Development and Application of a Library 2.0 Methodology. Unpublished Master's thesis, University of North Carolina at Chapel Hill, Chapel Hill, NC. http://hdl.handle.net/1901/356 (2006)
22. Kilker, J., Gay, G.K.: The social construction of a digital library: a case study examining implications for evaluation. Inf. Technol. Libr. **17**(2), 60–70 (1998)
23. Lagoze, C., Payette, S., Shin, E., Wilper, C.: Fedora: an architecture for complex objects and their relationships. Int. J. Digit. Libr. **6**(2), 124–138 (2006). doi:10.1007/s00799-005-0130-3
24. Lave, J., Wenger, E.: Situated Learning: Legitimate Peripheral Participation. Cambridge University Press, New York (1991)
25. Library of Congress: METS: Metadata Encoding & Transmission Standard. Official Web Site http://www.loc.gov/standards/mets/ (2007)
26. Ma, Y., Clegg, W., O'Brien, A.: Digital library education: the current status. In Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 165–174. Association for Computing Machinery, New York. doi:10.1145/1141753.1141786 (2006)
27. Miller, P.: Coming together around library 2.0: a focus for discussion and a call to Arms. D-Lib **12**(4) (2006). doi:10.1045/april2006-miller
28. Mongin, L., Fu, Y., Mostafa, J.: Open archives data service prototype and automated subject indexing using D-Lib® archive content as a testbed. D-Lib **9**(12) doi:10.1045/december2003-mongin(2003)
29. Nichols, D.M., Bainbridge, D., Downie, J.S., Twidale, M.B.: Learning by building digital libraries. In: Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 185–186. Association for Computing Machinery, New York. doi:10.1145/1141753.1141788 (2006)
30. Pomerantz, J., Oh, S., Wildemuth, B.M., Yang, S., Fox, E.A.: Digital library education in computer science programs. In Larson, R., Rasmussen, E., Sugimoto, S., Toms, E. (eds.) Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 177–178. Association for Computing Machinery, New York (2007)
31. Pomerantz, J., Wildemuth, B., Fox, E.A., Yang, S.: Curriculum development for digital libraries. In: Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 175–184. Association for Computing Machinery, New York. doi:10.1145/1141753.1141787 (2006a)
32. Pomerantz, J., Oh, S., Yang, S., Fox, E.A., Wildemuth, B.: The core: digital library education in library and information science programs. D-Lib **12**(11) (2006b). doi:10.1045/november2006-pomerantz
33. Rhyno, A.: Using Open Source Systems for Digital Libraries. Libraries Unlimited, Westport, CT (2004)
34. Rosenfeld, L., Morville, P.: Information Architecture for the World Wide Web. O'Reilly & Associates, Inc., Sebastopol, CA (2002)
35. Tansley, R., Bass, M., Smith, M.: DSpace as an open archival information system: current status and future directions. In: Research and Advanced Technology for Digital Libraries: 7th European Conference, ECDL 2003 Trondheim, Norway, August 17–22, 2003 Proceedings,vol. 2769/2004, pp. 446–460.Springer, Berlin. doi:10.1007/b11967 (2003)
36. Tansley, R., Bass, M., Branschofsky, M., Carpenter, G., McClellan, G., Stuve, D.: DSpace system documentation: contents. MIT and Hewlett Packard, Cambridge, MA. http://dspace.org/technology/system-docs/ (2005)

37. Tansley, R., Bass, M., Branschofsky, M., Carpenter, G., McClellan, G., Stuve, D.: DSpace system documentation: functional overview. MIT and Hewlett Packard, Cambridge, MA. http://dspace.org/technology/system-docs/functional.html (2005)
38. Witten, I.H., Bainbridge, D.: A retrospective look at Greenstone: lessons from the first decade. In: Larson, R., Rasmussen, E., Sugimoto, S., Toms, E. (eds.) Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 147–156. Association for Computing Machinery, New York. doi:10.1145/1255175.1255204 (2007)
39. Witten, I.H., Bainbridge, D., Tansley, R., Huang, C., Don, K.J.: StoneD: a bridge between Greenstone and DSpace. D-Lib **11**(9) (2005). doi:10.1045/september2005-witten