ARTICLES

D-Lib Magazine
September 2005

Volume 11 Number 9

ISSN 1082-9873

StoneD

A Bridge between Greenstone and DSpace

Ian H. Witten*, David Bainbridge*, Robert Tansley†, Chi-Yu Huang*, and Katherine J. Don*

*Department of Computer Science
University of Waikato
Hamilton, New Zealand
{ihw, davidb, chi, kjdon}@cs.waikato.ac.nz

†Hewlett-Packard Labs
Cambridge, MA, USA
<robert.tansley@hp.com>

Abstract

Greenstone and DSpace are widely used software systems for digital libraries, and prospective users sometimes wonder which one to adopt. In fact, the aims of the two are very different, although their domains of application do overlap. This article describes the two systems and identifies their similarities and differences. We also present StoneD [note 1] a bridge between the production versions of Greenstone and DSpace that allows users of either system to easily migrate to the other, or continue with a combination of both. This bridge eliminates the risk of finding oneself locked in to an inappropriate choice of system. We also discuss other possible opportunities for combining the advantages of the two, to the benefit of the user communities of both systems.

1. Introduction

Of the many open source systems for digital libraries, two of the most prominent are Greenstone [1, 2] and DSpace [3, 4]. Greenstone is older and more established internationally; DSpace has a more impressive institutional pedigree. Greenstone emanates from the Department of Computer Science in the University of Waikato, New Zealand, and is developed and distributed in cooperation with UNESCO as part of its *Information for All* program. DSpace was conceived by Hewlett-Packard Labs as a noncommercial product, and has been developed in conjunction with MIT Libraries. Both systems have active open source developer communities.

Many potential users apparently view these systems as competitive, and wonder which one to adopt. The notion of "competition" in a world where products are obtainable at no cost whatsoever seems at first sight rather bizarre. However, a commitment to any comprehensive software system, no matter how tentative, involves some effort in

downloading and installing, and considerably more effort in learning how to use it. Even to simply investigate the capabilities of a system of this nature requires a lot of work. In the case of digital library software, the investment required to create new collections, or adapt existing ones to work with new software, can be very significant. Indeed, traditional libraries build up their collections over centuries.

While there is certainly overlap in what Greenstone and DSpace can do, they have quite different goals and strengths. One aim of this article is to clarify the similarities and differences between them, and map out the natural areas of application of each one. This should help users to understand the differences and make an informed choice between the systems. It should also provide a basis for future developers to combine the features of both. Although creating each system has involved a prodigious amount of programming effort, combining them would not be such a monumental task. One advantage of open source software is that it allows new entrants to stand on the shoulders of established developers, not on their feet.

A second aim is to report on a new technical development that allows users to migrate their collections from Greenstone to DSpace and vice versa. Economic imperatives give designers of commercial systems a strong incentive to lock users in by preventing them from exporting the result of all their work, to increase the cost of migration to another vendor's product. This is a serious practical disadvantage. At least in principle, open source systems are immune because their code is accessible in source form and can be examined, understood, and modified by any competent computer scientist. In practice, however, substantial human investment is required to figure out just how to get the documents and metadata out of a digital library in a usable format. The technical development described in this article allows end users – typically librarians, not computer specialists – to export a collection from Greenstone and import it into DSpace, and vice versa.

This article also introduces other more intimate possibilities for dynamically combining digital libraries built with Greenstone and DSpace.

2. A Tale of Two Systems

The digital library systems Greenstone and DSpace aspire to make it easy for others to build their own digital libraries that offer comprehensive services to users – readers, authors and librarians. This aim stands in sharp contrast to most digital library projects, which are busy creating their own digital libraries. For example, the US National Science Digital Library plans to be the nation's online library of resources for science, technology, engineering, and mathematics education and research. Perseus is an evolving digital library whose primary stated goal is to bring a wide range of source materials to as large an audience as possible. The Chinese Memory Net is an expanding collection that contains thousands of images, originally ones related to the First Emperor of China's terracotta warriors and horses in Xian. There are countless other examples.

Greenstone and DSpace enable librarians to build their own collections, personalize them to their own taste, language, style, etc., and brand them appropriately to reflect the collection's ownership. One might ask why such open source systems exist? In both these cases the motivation is basically philanthropic, but also involves a healthy dose of enlightened self-interest. Working on an open source project is technically exciting. Progress is rapid because other open source modules can be incorporated freely, allowing one to avoid many mundane aspects of commercial programming. The product of research transcends publications and isolated prototypes, and yields a working system with many users. Greenstone is a major research project at its home university. It has opened up new avenues of research for its developers and has helped create an

international reputation for the computer science department there. DSpace is designed as an entrée for HP Labs into the growing open source community, and as a way of generating positive publicity in a powerful and influential forum. Furthermore, its chief developer is passionately interested in the topic of preservation. MIT has participated in the design and development because it wants to use the product as a foundation for collecting and curating the institution's intellectual output [5].

Today, digital library software resides in a space that is located somewhere between research and development. The area is so new that it opens up many research issues. And yet the developers of widely used systems like Greenstone and DSpace enjoy the satisfaction of having users download the software and respond immediately with comments and feedback – something that is rare in academic research environments. Satisfied users fuel the developers of these systems with a great deal of personal motivation.

Note that open source software can create worthwhile commercial opportunities. For example, one of Greenstone's developers has left the university environment to found a company that specializes in designing and building customized digital library solutions using the Greenstone software [note 2], while BioMed Central runs DSpace repositories for institutions [note 3].

Greenstone

Greenstone (http://www.greenstone.org) is a suite of software for building and distributing digital library collections that provides a way of organizing information and publishing it on the Internet or on removable media (e.g., CD-ROM/DVD). It is produced by the New Zealand Digital Library Project at the University of Waikato, and is distributed as open source, multilingual software in cooperation with UNESCO and the Human Info NGO. The dissemination of educational, scientific and cultural information, and particularly its availability in developing countries, is central to UNESCO's goals, and appropriate, accessible technology is seen as a vital tool in this context.

Greenstone's aim is to empower users, particularly in universities, libraries, and other public service institutions throughout the world, to build their own digital library collections in the fields of education, science and culture. UNESCO hopes this will encourage the effective deployment of digital libraries to share information and, where appropriate, place it in the public domain.

The key points that Greenstone makes it its core business to support include:

- Design and construction of collections
- Distribution on the web and/or removable media
- Customized structure depending on available metadata
- End-user collection-building interface for librarians
- Reader and librarian interfaces in many languages
- Multiplatform operation.

The liaison with UNESCO and Human Info has been a crucial factor in the development of Greenstone. Human Info began using Greenstone to produce collections in 1998, and provided extensive feedback on the reader's interface. UNESCO wants to empower developing countries to build their own digital library collections – otherwise they risk becoming read-only societies in the information revolution. UNESCO selected Greenstone in 2000, and arranges user testing, helps with internationalization, and mounts courses. Internationalization is a central goal: today the Greenstone reader's interface is available in 35 languages, and the librarian's interface, including all documentation, is

available in 4 (English, French, Spanish, Russian).

Greenstone is issued under the terms of the GNU General Public License. It originated in 1996 [6], and the current production version (Greenstone2) was designed about seven years ago, although it is continually being extended – for example, Greenstone2 now supports the latest version of Open Archive Initiative Protocol for Metadata Harvesting (OAI-PMH) [7] and the relatively recent METS metadata encoding standard [8]. A complete redesign and reimplementation, Greenstone3, has been described [9] and released, informed by experience with the current system and the problems and challenges faced by users, international collection developers, and practicing librarians. Greenstone3 allows documents to be dynamically added to collections; provides more flexible ways to dynamically configure the run-time system by adding new services; lowers the overhead incurred by collection developers when accessing this flexibility to organize and present their content; and modularizes the internal structure. The design is based on widely accepted standards that were unavailable when Greenstone2 was designed. Greenstone2 is still recommended for end-user librarians, while Greenstone3 is an emerging system currently intended for experimental use by computer scientists and information technologists in conjunction with librarians.

DSpace

DSpace (http://www.dspace.org) facilitates the building of institutional repositories that capture, distribute and preserve intellectual output at an institutional level. It is produced by HP Labs and designed in partnership with MIT, who note that much of the intellectual output of professors and researchers is in digital form and potentially ephemeral unless the institution has an aggressive policy for collecting and preserving it. DSpace is designed to help capture and organize everything produced by faculty and staff – digitized versions of lecture notes, videos, papers, and data sets – into an "institutional repository" that will make it available to future generations in its original digital form. Of course, it has applications in government organizations and commercial enterprises too.

DSpace provides a set of tools for helping institutions keep track of their data, organize it in meaningful ways and migrate that data to new formats as old ones become obsolete [5]. It helps establish a system for the curation of the data in a manner that is as automated as possible, in order to handle the increasing volume and complexity of the data being produced.

The key points that DSpace makes it its core business to support include:

- Repositories at an institutional level
- Self-deposit of digital assets by faculty
- End-user interface for depositors
- Assets made available for searching and browsing
- Data retrievable many years in the future
- Institutional commitment to ensure the continued availability of certain named formats.

The liaison with MIT has been a crucial factor in the development of DSpace. A co-development contract between Hewlett-Packard Labs and MIT Libraries was established in March 2000, and MIT publicly launched its institutional repository in November 2002. From the outset, the plan was to create an infrastructure for storing the digitally born intellectual output of the MIT community and to make it accessible over the long term to the broadest possible readership [3].

DSpace supports the Open Archive Initiative Protocol for Metadata Harvesting (OAI-

PMH), and was designed to support interoperation with both other DSpace installations and other OAI-compliant archives. DSpace uses the qualified Dublin Core metadata standard [10] for all collections. Only three fields are required: title, language, and submission date; all others are optional. There are additional fields for document abstracts, keywords, technical metadata and rights metadata. This metadata is indexed for browsing and searching the system, either within a collection or across collections.

The current version of DSpace is available online as an open source Unix program, licensed under the BSD Open Source License. Neither HP nor MIT offer formal support for DSpace. Users are assumed to be institutional, with all the necessary resources to use the system (or the means to outsource this), including hardware running Unix, and a systems administrator to install and configure the system – and, in most cases, a Java programmer who can localize and customize the system [4].

Plans are afoot to develop the next version, which will be a more modular, flexible version into which different modules of functionality can be plugged in to suit different needs, and with refactored storage to enable different approaches to the digital preservation problem to be tested.

Differences

It is certainly not our intention to recommend one of these systems over the other – that would be counterproductive and contrary to the spirit of this article. However, there are some clearly defined situations that quite obviously fall more naturally under the purview of one or the other.

Preservation. The act of creating any digital library collection based on open source software will contribute to the preservation of the material it contains. However, DSpace is explicitly oriented towards long-term preservation, while Greenstone is not. DSpace stores preservation metadata and includes a scheme where institutions commit to ensuring the continued availability of certain named formats [11].

Support infrastructure. DSpace is designed for institutional use, where there are centralized computing facilities and a competent infrastructure for software support. Greenstone is designed to be easy for anyone with basic computer-literacy skills to install, in a laptop, desktop, or institutional environment.

Multiplatform operation (related to the above). Greenstone runs on Windows computers (collections can be built on Windows 95 and up), Unix, and Mac OS/X. DSpace is currently restricted to Unix and OS/X. Note that this is unlikely to be a serious practical disadvantage in typical DSpace usage scenarios.

Author-oriented. DSpace incorporates an interface whereby users (typically authors, though some institutions choose to have librarians do this on behalf of the faculty) can submit documents to the system, and define metadata for them. Greenstone does not.

Librarian-oriented. Greenstone supplies an end-user interface with which collections can be designed, customized, and built. DSpace provides a generic design that can be tailored – but not by typical end users.

Built-in metadata standard (related to the above). DSpace imposes a single metadata standard on all collections. Greenstone provides a widely used standard (Dublin Core) but also allows collection-builders to use their own metadata scheme either by extending an existing one in an ad hoc manner or by defining an entirely new one using a metadata set editor.

Distribution on removable media. Those who create Greenstone collections can write them to a self-installing CD-ROM that operates on all Windows systems (even obsolete ones right down to Windows 3.1/3.11, still in use in developing countries).

Dynamic collections (historically related to the above). In Greenstone [note 4], adding documents to a collection normally involves rebuilding the full-text index and browsing structures (though rebuilding can be scheduled to take place automatically), whereas DSpace operates incrementally (though operations on recently-added documents like extracting text or producing image thumbnails are processed in batch mode).

International users. Greenstone provides interfaces for readers in 35 languages, including many minority ones, and has a scheme that helps language maintainers keep the interfaces up to date when new interface features are added [12].

Note that both systems are continually evolving, and these features can change rapidly. For example, Greenstone can indeed accommodate dynamic collections by using a different search engine from the default one. Although this is probably beyond the technical capabilities of the librarian-level users that Greenstone targets, a user interface enhancement could easily rectify this. Conversely, although the default DSpace configuration is currently restricted to UNIX, it would not be hard to modify it for other operating systems. And there are some DSpace installations in languages other than English.

The difference between the two systems is largely explained by the environments in which they are designed to operate. DSpace is designed for the institutional setting, where members of faculty submit their documents to a common system that enforces common standards. Its model envisages "communities" (e.g., schools, departments, centers, labs, and programs) that contain one or more "collections" of digital "items" [4]. Greenstone is designed to allow non-specialist users to produce single, individualized, collections. Its model envisages a "librarian" who is creating collections from existing "resources" (comprising both "items" and metadata resources) and distributing them over the Web or on removable media, possibly in an international setting [13].

3. Building Bridges

An obvious way to transfer a digital library collection from DSpace to Greenstone or vice versa is to use the Open Archive Initiative Protocol for Metadata Harvesting (OAI-PMH), which is an application-independent interoperability framework. As noted above, both systems can serve a collection over OAI-PMH, and both systems can ingest information in this form.

However, there are some disadvantages to OAI-PMH-level integration, as discussed below. And even if it were satisfactory from a practical point of view, the intention of this article is to explore other possibilities for deeper integration and cooperation. The exploration will shed light on the differences between the two systems, and their strengths, and may suggest new directions of research. Let us consider the various levels at which Greenstone and DSpace might co-operate.

OAI-PMH-level integration

As the name implies, OAI-PMH is based on metadata harvesting, and both Greenstone and DSpace are concerned not just with metadata but with the documents themselves. Metadata records may contain a link to the resource to which they refer, but the nature of this link is beyond the scope of OAI-PMH. The protocol description suggests the use of

the Dublin Core *Identifier* element for this purpose, and the Dublin Core standard recommends that the resource be identified by means of a string or number conforming to a formal identification system such as the Uniform Resource Identifier (URI) (including the Uniform Resource Locator, URL; the Digital Object Identifier, DOI; and the International Standard Book Number, ISBN).

Establishing interoperability at this level gives a degree of integration that is broad but not deep. Although the metadata will be transferred correctly, the documents themselves will not – not without implementing a special convention outside OAI-PMH that unifies the way the two systems utilize the *Identifier* element. Furthermore, both DSpace and Greenstone have the notion of a single document being present in alternative formats or "versions." Since all Dublin Core fields are repeatable, it is possible to use multiple occurrences of *Identifier* metadata to point to these alternatives. But again, practical difficulties arise because this is not part of the OAI-PMH standard, and different systems do things in different ways.

In a small experiment we switched on OAI support for our DSpace server and downloaded a set of metadata records from it using Greenstone's *importfrom* command, which was then digested into a Greenstone collection. The newly built collection was configured to display the metadata in a table, with a hyperlink pointing back to the source document on any URL-based identifier that appears in the record.

METS-level integration

Another route to integration is through the METS standard. With its combined metadata and document container approach, METS offers a deeper form of interchange. It seems to be an attractive option because both systems already have the capability to export to this format.

The METS standard uses a meta-description approach to describe what constitutes a "work" in a digital library. Although this approach is very flexible, it has the disadvantage that different systems may use structures that cannot logically be mapped into each other. This is indeed the case with DSpace and Greenstone. For instance, DSpace supports a hierarchical form of metadata that can be attached at the document level, whereas in Greenstone each item of metadata is flat, but metadata can be attached to individual sections within a document. These differences are reflected in the METS files the two systems generate. To support METS interchange, such differences must be reconciled.

We explored this option, but ultimately decided to implement a bridge based around the native interchange formats (described below). This offers a middle ground solution between the two extremes of OAI and METS.

Joint distribution of DSpace and Greenstone

The idea of releasing a joint distribution of the current production versions of Greenstone and DSpace originally sparked this project. However, this has not been attempted because the systems make different assumptions about the installation process (see "Support infrastructure" above). DSpace runs only under Linux, while many – probably most – Greenstone users employ Windows. Also, a joint distribution *per se* would merely enable users to run the two systems side by side, which is not very useful unless there is some meaningful interaction between the two – such as that described below.

DSpace import/export for Greenstone

We have implemented StoneD, a bridge between DSpace and Greenstone that allows a collection of documents to be exported from DSpace and imported into Greenstone and vice versa. This provides a far greater level of integration than an OAI-PMH export-import – for example, it deals with documents as well as metadata, and resolves the problems of multiple representations of a single document. It works within Greenstone2, the current production version that is recommended for end-user librarians, and is included in the current release of the software. This bridge is described in Section 4.

Greenstone access to the DSpace document database

Rather than converting the DSpace document database into a form more congenial to Greenstone and vice versa, as StoneD does, one could imagine the Greenstone runtime system consulting the DSpace document database directly. This would make the most sense for Greenstone3 because, like DSpace, Greenstone3 uses a relational database for metadata (Greenstone2 does not, because it was designed to be able to serve collections even on primitive computers).

The databases involved are different – DSpace uses PostgreSQL or Oracle, and Greenstone uses MySQL – but both implementations operate in a portable manner and use Java Database Connectivity (JDBC) to insulate their code from any particular database system. Thus it would be easy to arrange for Greenstone3 to access the DSpace metadata database, perhaps by specifying in a configuration file that it should use PostgreSQL rather than MySQL technology. Furthermore, if the DSpace collection had a full-text index it should be possible for Greenstone to use it too, because DSpace uses Lucene [14] for indexing and Greenstone3 can employ a variety of full-text search engines, including this one.

This mode of operation assumes that DSpace is present (though not necessarily running) on the same computer system as Greenstone – or at least that the two share a file space. A Greenstone3 installation is a network of modules that communicate in terms of XML messages; so-called "service" modules provide the core functionality. Two new services could give Greenstone direct access to the DSpace collections. One would communicate with the database in which the metadata is stored; the other would retrieve the files from DSpace's data store. It is likely that some of DSpace's Java classes could be retargeted for this purpose.

In this scenario, users would be able to employ Greenstone3's cross-collection search facility to jointly search Greenstone collections and DSpace repositories.

Greenstone access to DSpace services

An alternative to sharing information at the database level is to share at the service level. Again we assume Greenstone3 rather than Greenstone2. The Greenstone service modules could communicate with DSpace servlets and therefore access collections anywhere – not just on a computer that has Greenstone installed and configured to use local DSpace collections. This would enable Greenstone users to peruse DSpace collections in different repositories on different computers. A similar modification could be made in the other direction, with Java code on the DSpace side interacting either through servlets or the SOAP protocol [15] to support searching and browsing facilities compatible with DSpace functionality.

It appears that DSpace servlets return raw HTML, which the Greenstone service module would have to parse. Alternatively, DSpace could be modified to provide an option for the servlet to present its output in XML rather than HTML format, an option that

Greenstone already incorporates because it has proved useful in many other contexts. Then, the two systems would be able to exchange structured data at runtime.

As in the previous scenario, users could jointly search Greenstone collections and DSpace repositories.

DSpace access to Greenstone

Much the same discussion above can be applied in the other direction. While some of the terminology changes, the ideas remain the same, as does the end result: seamless integration of Greenstone collections within a DSpace site. Manipulating the inheritance hierarchy in DSpace, new Java classes could be introduced that access Greenstone3 functionality. This could be accomplished at the servlet level, taking advantage of the XML output option, or more directly through the Greenstone message passing mechanism over SOAP.

Joint distribution of DSpace and Greenstone3

Although we decided above not to pursue a joint distribution of DSpace and Greenstone2, an analogous venture with Greenstone3 could prove worthwhile. Greenstone uses the InstallShield software for easy installation, and this would simplify the installation process for DSpace, which currently requires specialist computer knowledge. Both systems run as servlets and could share the Tomcat implementation of the Java Servlet technology [16]. They could also share a relational database. As noted above they have made different choices, but both use JDBC to insulate themselves from the details of the particular database system.

Users could employ either system's collection-building tool – the DSpace document submission mechanism in Figure 1 or the Greenstone Librarian Interface applet in Figure 5 – on top of either system. The joint release should include examples in both systems, and make it easy for users to pick the features that suit them best.

4. StoneD

Figures 1-6 are browser snapshots showing a worked example that demonstrates the bi-directional link we have implemented. To anyone who is familiar with the individual systems, the layout and structure of the figures are essentially unremarkable. The snapshots in Figures 1 and 2 look much like any other DSpace installation; likewise for the Greenstone snapshots in Figures 3 and 4. It is what underlies them that is of interest here. We first describe the technical developments that constitute the bridge between the two systems, and then walk through the example shown in the figures.

Figure 1. Adding a document to DSpace.

**Figure 2. Having deposited the document,
DSpace returns to the community's home page.**

Exporting from Greenstone to DSpace

DSpace uses a documented internal archive format for batch importing of records, and for exporting records or collections to other DSpace systems. The format is straightforward:

one directory per document, with a manifest specifying all files that make up the document, a metadata file specified using Dublin Core, a handle file [17] containing the DSpace identifier, and the files themselves.

Greenstone uses an "importing" phase to bring all documents and metadata into the system, converting them from their source formats if necessary, followed by a "building" phase that creates the full-text indexes and browsing structures used by the reader. The internal format for storing documents and metadata long predates the METS standard for encoding descriptive, administrative and structural metadata for objects in a digital library [18]. However, following development of that standard, an option has been added to Greenstone's importing phase that makes Greenstone use METS as an alternative internal format. (The particular form that Greenstone uses has been accepted as a METS Profile by the METS Board.)

It is natural, therefore, to add a similar option to the importing phase that converts the input documents and metadata into the DSpace batch import format. This is accomplished by running the Greenstone import program with the *-saveas DSpace* flag. (In fact, this feature had already been suggested by users on the Greenstone mailing list.) For each document, the importing procedure builds a network of data structures that represents it in the form of a document object model. Then this structure, which is independent of any file format, is traversed to generate a specific set of files in accordance with the designated target file format. Support for DSpace merely required the inclusion of an additional traversal routine.

This functionality was also wrapped into a new Greenstone command called *export*. In reality, this command is just a cosmetic wrapper to the more general *import* script, with arguments tuned to this task – but the *export* program is included because it seems counterintuitive to have to run a program called "import" to perform the task of exporting.

Because Greenstone collections can use any metadata standard, it was necessary to provide a crosswalk facility to map the metadata into Dublin Core, if it is not in that form already. The same operation is needed when exposing Greenstone collections over the OAI-PMH protocol, and this previously implemented mechanism is reused for DSpace export. A metadata crosswalk in Greenstone is a text file that specifies source metadata elements and the corresponding term in the destination set. Prefix notation is used to specify metadata namespaces. For example, *gils.abstract→dc.description* maps "abstract" metadata in the Government Information Locator Service format to Dublin Core's "description" field. To provide convenient abbreviations for a variety of common situations, additional syntax supports defaults, wildcard matching, and collection-specific overrides.

If so directed, Greenstone extracts certain metadata fields automatically. For instance, it might interpret the value specified in the <title> tag of an HTML document as *Title* metadata, or set *Language* metadata by heuristically analyzing the document's full text. The resulting metadata is artificially scoped by prefixing it by "*ex*." to distinguish its namespace from all others. Greenstone uses exactly the same crosswalk technique to manipulate extracted metadata as well.

Standardized crosswalks have been defined for a variety of formats. Transformations from MARC to Dublin Core and from GILS to Dublin Core are provided with Greenstone. End users can extend these and add more as their needs dictate.

Exporting from DSpace to Greenstone

Greenstone uses modules called "plugins" to import documents and metadata in different formats, and to associate metadata with the appropriate documents. Plugins can manipulate the stream of filenames that governs what is included in the collection [19]. Plugins that import documents can perform format conversion internally or take advantage of existing conversion utilities. Metadata can be read from the input documents or from separate metadata files, or it can be computed from the documents themselves. New plugins can be written for novel situations.

The natural way to enable Greenstone to process DSpace collections was to write a plugin that digests a collection in its entirety by reading the DSpace archive format. This plugin seeks out DSpace manifest files and their associated Dublin Core records, which are in XML format. These records are parsed to form a partial document object model. This is attached to a list of the filenames that are specified in the manifest as primary forms of the document. Nothing more is necessary: Greenstone's existing mechanism for binding metadata to documents automatically performs the rest of the work.

To place this in a specific context, consider a single PDF document in DSpace. When exported, the manifest lists just one file: a copy of the PDF document. Accompanying it is an XML file that encodes the document's Dublin Core metadata. To import it into Greenstone, the collection should be configured to use both the DSpace plugin *and* the PDF plugin. This is trivial to arrange: Greenstone collections routinely use multiple plugins. During the import process the DSpace plugin causes the metadata in the Dublin Core record to be associated with the named PDF document, but it is not until the PDF plugin processes the primary document that the information is actually bound to the file and the full document object is formed.

To support more complex scenarios some further features are needed. DSpace distinguishes between files that play primary and secondary roles. For example, for an HTML web page with supporting image files, the HTML file is marked as primary and the images secondary. The Greenstone plugin must block secondary files to prevent them from being treated as independent documents with the same metadata attached. The plugin infrastructure already has the ability to manipulate the stream of filenames being processed, and the required blocking is straightforward to arrange.

To cover the possibility of a DSpace collection including a document type that is not supported by Greenstone, an existing plugin called *UnknownPlug* is used as a catchall [note 5]. This generic plugin was designed for the situation where there is no existing native plugin for a particular file format. It functions by attaching any stipulated metadata to the raw source document.

It is our experience that different collections often trigger different design needs, even when they use the same fundamental document formats and metadata. Consequently Greenstone allows plugins to have options that are specified in a collection's configuration file. The DSpace plugin supports several options, some of which are mentioned in the example that follows.

Example

The setting for our small (fictitious) example is a university music department that makes use of DSpace and Greenstone to store essays and other documents that relate to its research interests. Figure 1 shows a user interacting with the DSpace submission process to add a new item to the communal repository: an interview with Bob Dylan by British DJ John Peel, for which there is an MP3 audio recording and a written transcription in Word format. The user has performed two steps in a sequence that garners a variety of information. On completion of the sequence, the user returns to the community's home

page in Figure 2, which shows the new contribution on the right as a recently added document. From this page the user can browse the collection by metadata such as title and author, or perform a search on the metadata fields.

Using the *export* feature of DSpace combined with Greenstone's DSpace plugin, the music department hosts a parallel Greenstone version of the same collection. Because of the static nature of Greenstone2, this is set up as an automated process that runs hourly. An alternative would be to arrange for anyone in the user community to rebuild the collection from Greenstone's Librarian interface. Indeed, a trivial addition to DSpace would enable Greenstone to rebuild the collection immediately upon deposition of a new work. In all cases rebuilding takes place behind the scenes: the old version of the collection is served until it is silently superseded by the new one, and Greenstone's document identifier system ensures that users never notice the changeover – except that new documents may appear when they refresh their web browser.

In Figure 3 the user has accessed the Greenstone home page for the same collection. Greenstone collections can easily be customized by end users, but in this default case the two versions of the collection offer essentially the same features. Here, users can search and browse by title and author just as in DSpace, although the interface layout differs. Figure 4 shows the page accessed by clicking the *titles a-z* button in the navigation bar. From here the various source documents can be viewed, in multiple representations where available. The icon for the Word transcript of the Bob Dylan interview is displayed to the left, and the MP3 icon is shown under "also available as."

Figure 3. Home page for the Department of Music collection in Greenstone.

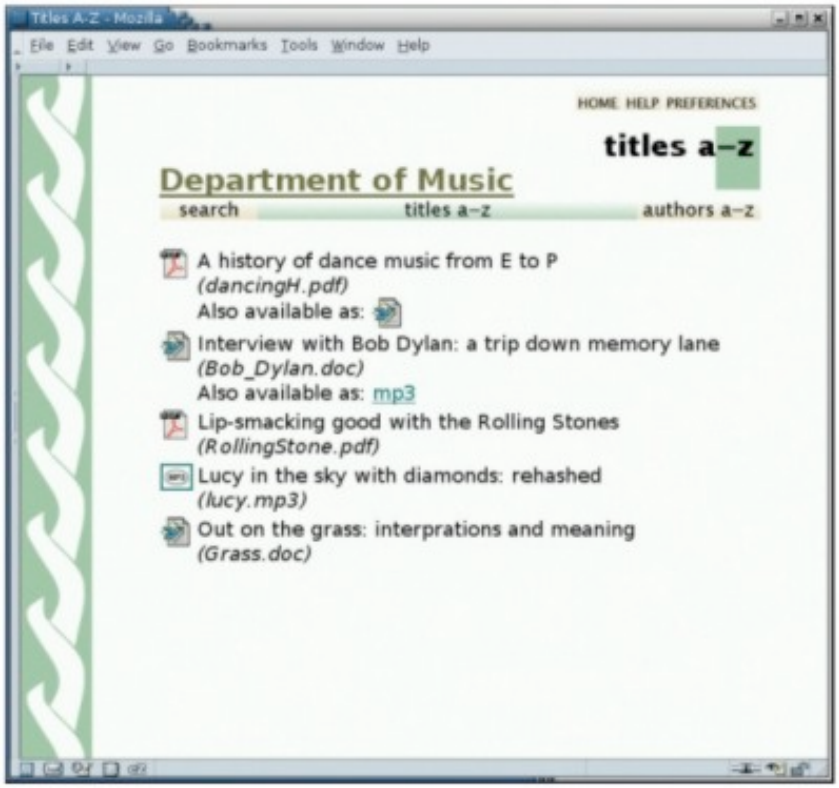

Figure 4. Browsing titles in Greenstone.

This superficial visual difference reflects a more fundamental disparity in how the two systems work. DSpace treats the PDF and MP3 versions of the interview as equal representations of the same work, and metadata is associated with the work as a whole. Greenstone regards the PDF version as the primary document, to which the metadata is attached, while the MP3 file fulfills a secondary role (an "associated file" in Greenstone terminology).

Greenstone indexes the text extracted from the primary document. By default DSpace does not offer full text indexing (although any collection can be configured to provide it). When users add documents, DSpace treats each work as a black box and makes no attempt to peek inside. Readers normally locate documents by searching and browsing metadata. Users supply metadata manually when they work through DSpace's interactive process for depositing documents. In contrast, the role of indexing is central to Greenstone, and considerable effort is invested in plugins that process individual document formats – with the fringe benefit of the possibility of automatically generated (and consequently probably unreliable) metadata. This approach is better matched to a batch workflow. Each system can support the other type of workflow, but doing so does not play to its strengths.

In the context of our music department example, this difference means that Greenstone users can search the full text of any document to locate an item of interest. However, DSpace users can peruse a list of recently added items, a notion that is less natural in a collection that is built afresh each time. We emphasize that with a little effort each system could be configured to add the missing facility if desired.

Behind the scenes, Greenstone uses a collection configuration file to encapsulate the collection's design. Figure 7a shows pertinent excerpts of this file for the music department's collection. (This is unlikely to be of any interest to members of that department!) A list of plugins is specified that forms a pipeline for processing, with *DSpacePlug* occurring before the likes of *PDFPlug* and *WordPlug* so that the metadata it extracts is available when the other plugins encounter their respective document types.

DSpacePlug has the option *xfirst_inorder_ext* with an argument that dictates that the PDF version of a work should be chosen ahead of any Word version as the principal document to index, and a Word version should be chosen ahead of an MP3 file.

The configuration file specifies two browsing features using the *classify* command: one for titles and the other for authors. This gives the Greenstone version comparable features to the DSpace version. Greenstone's configurable design would allow much more end-user tailoring of browsing and indexing options. Although the Greenstone configuration file is a succinct way of showing how a collection works behind the scenes, ordinary Greenstone users never encounter this file in its raw form. Collection design is handled through the Librarian interface.

To demonstrate the connection between DSpace and Greenstone traveling in the opposite direction, we reuse the same document set and metadata. The process results in Greenstone and DSpace collections that are virtually identical to those in Figures 1- 4. Even so, there are interesting intricacies in how this was accomplished. For example, we assume that author metadata is entered in Greenstone as *dc.Creator*, which we believe most users would probably do in this context, rather than *dc.Contributor*, which is DSpace's standard.

In Figure 5, the collection designer is using Greenstone's Librarian Interface to digest the documents, accompanied with metadata, and to shape the collection. Having selected Dublin Core as the metadata set and dragged the source documents into the collection area (not shown), the designer is in the process of adding metadata. She is using the Enrich tab, selecting each primary document in the file tree displayed in the left-hand panel of Figure 5 and entering metadata values on the right. A value for *dc.Title* has already been entered; the author *John Peel* is in the process of being added as *dc.Creator* (if desired, *dc.Contributor* could equally well have been used). Here we encounter another difference: the mismatch between unqualified Dublin Core (Greenstone) and qualified Dublin Core (DSpace). The mismatch is resolved when transferring the collection from Greenstone to DSpace by utilizing the metadata mapping facility, which works within a single metadata set as well as across two different ones. When the export script is finally initiated, a metadata mapping file that includes the line *dc.Creator→dc.Contributor^author* serves to match up these values.

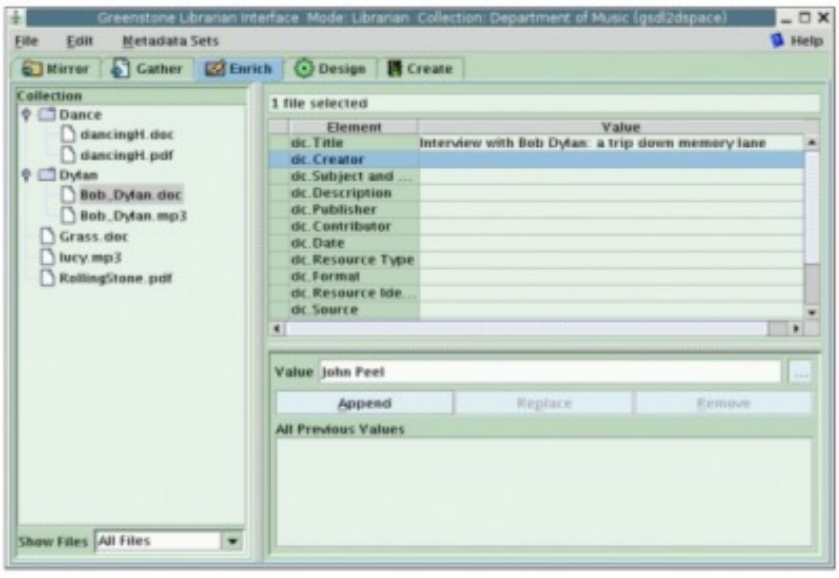

Figure 5. Augmenting a document with metadata using Greenstone's Librarian interface.

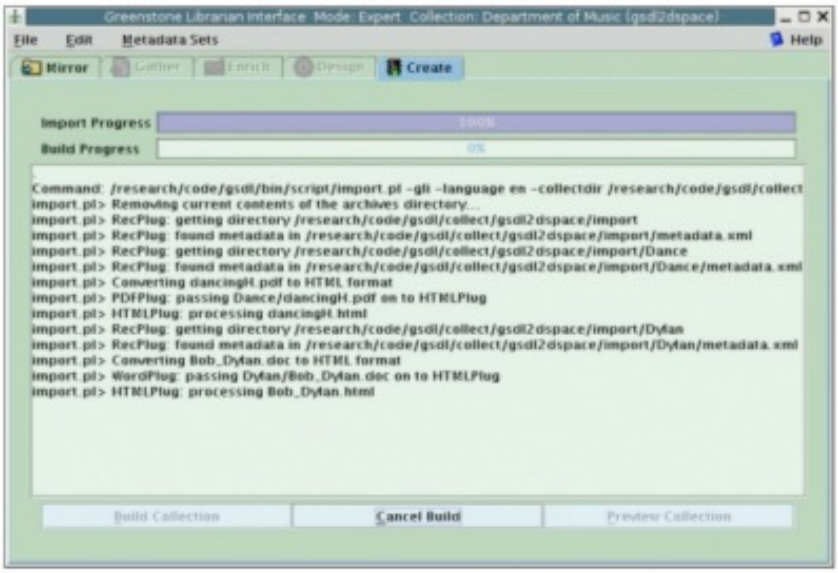

Figure 6. Building the collection in the Librarian interface.

But first there is one further design element to consider. When the Greenstone collection was created, a default list of plugins was incorporated. Plugins for PDF and Word documents are included in this default, but one for MP3 needs to be added. To bind the different media representations of the same work together into one item in the collection, we exploit a Greenstone convention and give them the same root filename with different file extensions. This is done before dragging the documents into Greenstone, and is activated using the *-associate_ext plugin* option. (If that option was not set, they would be treated as different works.)

The interaction (not shown) to do this is simple. Figure 7b shows pertinent excerpts from the collection configuration file that results. For documents that are available in several different formats, DSpace allows one to be nominated as the primary document. Greenstone can impose a preference order on document versions. In this case we put the plugins in the order *pdf*, *doc*, *mp3*; and then assign *-associate_ext doc,mp3* for the PDF plugin and *-associate_ext mp3* for the Word plugin. When a PDF document is processed, this option instructs the plugin to check for files with the same root and extension *.doc* or *.mp3*, and associate them with this document. This has the necessary side-effect of blocking these files from being processed by other plugins. Any document that ends up being processed by the Word plugin must necessarily have no PDF counterpart (otherwise it would have been blocked), which is why that plugin's *-associate_ext* option only specifies *mp3*. The shared root file naming convention is useful in a variety of other situations too.

```
#...
indexes    document:text document:dc.Title
           document:dc.Contributor^author
defaultindex      document:text

plugin     ZIPPlug
plugin     GAPlug
plugin     DSpacePlug -first_inorder_ext
           pdf,doc,mp3
plugin     WordPlug
plugin     PDFPlug
plugin     MP3Plug
plugin     ArcPlug
plugin     RecPlug -use_metadata_files

classify   AZList -metadata dc.Title
classify   AZList -metadata dc.Contributor^author
           -buttonname Creator
#...
```

(a)

```
#...
indexes    document:text document:dc.Title
           document:dc.Creator
defaultindex      document:text

plugin     ZIPPlug
plugin     GAPlug
plugin     PDFPlug -associate_ext doc,mp3
plugin     WordPlug -associate_ext mp3
plugin     MP3Plug
plugin     ArcPlug
plugin     RecPlug -use_metadata_files

classify   AZList -metadata Title
classify   AZList -metadata dc.Creator
           -buttonname Creator
#...
```

(b)

Figure 7. Greenstone configuration file (abridged)
(a) for files imported from DSpace, (b) for files to be exported to DSpace.

Figure 7b illustrates another wrinkle. Since author metadata is now entered as *dc.Creator*, this term should be used when stipulating the index and *AZList* classifier (which creates an alphabetical index). Again, in practice this is accomplished in the more supportive environment of the Librarian Interface, which displays the available choices as a selectable list of items.

Now we are ready to press the *build* button in the Create tab, shown in Figure 6. On completion, this yields a collection visually identical to that shown in Figures 3-4. One of the options on the Librarian Interface's *File* menu is *Export*. Upon start-up, the system interrogates the *export* script described earlier for a list of known export file formats, and these formats are dynamically added into the interface. Activating *File→Export* produces a popup that lists the available formats – currently METS and DSpace. Choosing DSpace, browsing to a suitable metadata mapping file (if required), and pressing the *export* button produces a set of files that can be transferred to a DSpace installation and imported in batch mode. The resulting collection is shown in Figures 1-2.

5. Conclusions

Greenstone and DSpace are both designed to help third parties set up their own digital libraries. However, they represent rather different perspectives and have different, and in many ways complementary, goals and strengths. One goal they share is to be flexible, and both can be customized and modified at many different levels – including the programming level, since they are open source systems. This gives the ultimate flexibility and yields significant advantages over closed-source systems. Of course, this very flexibility makes fair comparison tricky.

This article has compared and contrasted the two systems' goals in terms of the core business that they aim to support, and compared their features in terms of their natural domain of operation. A crude caricature of the difference is that Greenstone supports individually designed collections of different kinds of documents and metadata in an international setting – epitomized by completely static collections on CD-ROM or DVD – whereas DSpace supports institutions in their struggle to capture and disseminate the intellectual output of an institution and preserve it indefinitely – epitomized by its use by MIT Libraries, who helped pioneer its development. However, each system is highly flexible and customizable to meet a wide variety of needs.

There are fertile opportunities for crossover between the two systems. As well as being of great practical benefit to users, studying these opportunities sheds light on many practical issues of interoperability between different digital library systems. Standard interoperability frameworks include OAI-PMH, which focuses on interoperability of *metadata* alone, and METS, which is a general framework that focuses on interoperability of document and metadata *containers*. Neither of these provides a sufficient mechanism for a satisfactory bridge between Greenstone and DSpace, at least not without establishing elaborate further conventions on top of these basic standards.

What we have actually implemented in StoneD is a facility for exporting collections from DSpace and importing them into Greenstone, and vice versa. It is included in the current release of Greenstone2, the production version of this digital library software. This article has described the mechanism, along with an extended practical example.

We have also investigated more intimate linkages, which fit better within the framework of Greenstone3, a new variant that is under active development and has been released as an "alpha" version. One possibility has DSpace and Greenstone sharing their document and metadata database; another has Greenstone accessing active DSpace services to

present a DSpace collection through the Greenstone interface. In both cases users would be able to search across Greenstone and DSpace collections, and in the second scenario Greenstone users would have access to DSpace collections on different computers. However, neither of these possibilities has been implemented. If they were, the effort might culminate in a joint Greenstone/DSpace release.

We regard it as healthy that different open source systems are being developed for digital libraries. We hope that library implementers will find this work helpful in clarifying the differences between these systems and mitigating the long-term effect of any decision they make as to which to use today. We also hope that future developers can build upon this work and on the enormous open source code base that these two systems represent.

Acknowledgements

We gratefully acknowledge the stimulating environment of the New Zealand Digital Library Project that supported this work, and the helpful suggestions of John Rose, Dr T.B. Rajashekar, and Tim Elphick.

Notes

1 Pronounced *Stone-Dee*.

2 Stefan Boddie, DL Consulting, <http://www.dlconsulting.co.nz>.

3 Open Repository home page, <http://www.openrepository.com>.

4 This applies to Greenstone2 only. Collections in Greenstone3 can be dynamic.

5 The converse problem is not an issue, because DSpace does not look inside its document files except when directed to create a full-text index, and then it simply skips over files that are in formats it cannot process.

References

[1] Witten, I. H., Bainbridge, D. and Boddie, S.J. (2001). "Greenstone: Open-source digital library software." *D-Lib Magazine* 7(10) (doi:10.1045/october2001-witten).

[2] Witten, I.H. and Bainbridge, D. (2003) *How to build a digital library*. Morgan Kaufmann, San Francisco.

[3] Smith, M. (2003) "DSpace: An open source institutional repository for digital material." *D-Lib Magazine* 8(10) (doi:10.1045/october2002-inbrief#SMITH).

[4] Smith, M., Bass, M., McClella, G., Tansley, R., Barton, M., Branschofsky, M., Stuve, D. and Walker, J.H. (2003) "DSpace: An open source dynamic digital repository." *D-Lib Magazine* 9(1) (doi:10.1045/january2003-smith).

[5] "MIT's DSpace experience: a case study." (http://www.dspace.org/implement/case-study.pdf).

[6] Witten, I. H., Cunningham, S.J. and Apperley, M. (1996) "The New Zealand Digital Library Project." *D-Lib Magazine* 2(11) (doi:10.1045/november96-witten).

[7] "The Open Archives Initiative protocol for metadata harvesting." (http://www.openarchives.org/OAI/openarchivesprotocol.html).

[8] Cundiff, M.V. (2004) "An introduction to the Metadata Encoding and Transmission Standard (METS)." *Library Hi Tech* 22(1): 52-64.

[9] Bainbridge, D., Don, K.J., Buchanan, G.R., Witten, I.H., Jones, S., Jones, M. and Barr, S.I. (2004) "Dynamic digital library construction and configuration." *Proc European Digital Library Conference*, Bath, England.

[10] Weibel, S. (1999) "The state of the Dublin Core metadata initiative." *D-Lib Magazine* 5(4) (doi:10.1045/april99-weibel).

[11] Tansley, R., Bass, M. and Smith, M. (2003) "DSpace as an Open Archival Information System: Current Status and Future Directions." *Proc European Conference on Digital Libraries*, Trondheim, Norway. pp 446-460.

[12] Bainbridge, D., Edgar, K.D., McPherson, J.R. and Witten, I.H. (2003) "Managing change in a digital library system with many interface languages." *Proc European Conference on Digital Libraries*, Trondheim, Norway.

[13] Witten, I. H. and Bainbridge, D. (2005, in press) "Creating digital library collections with Greenstone." *Library Hi-Tech*.

[14] Hatcher, E. and Gospodnetic, O. (2004) *Lucene in Action*. Manning Publications.

[15] Mueller, J. (2001) *Using SOAP*. Que.

[16] Brittain, J. and Darwin, I.F. (2003) *Tomcat: The definitive guide*. O'Reilly.

[17] Kahn, R. and Wilensky, R. (1995) "A framework for distributed digital object services." (http://www.cnri.reston.va.us/home/cstr/arch/k-w.html).

[18] "METS: An overview and tutorial." (http://www.loc.gov/standards/mets/METSOverview.v2.html).

[19] Witten, I. H., Bainbridge, D., Paynter, G.W. and Boddie, S. (2002) "The Greenstone plugin architecture." *Proc Joint Conference on Digital Libraries*, Portland, Oregon.

doi:10.1045/september2005-witten