The DSpace Institutional Digital Repository System: Current Functionality

Robert Tansley, Mick Bass, David Stuve
Hewlett-Packard Laboratories
One Cambridge Center
Cambridge, MA 02142
{robert.tansley, mick.bass, david.stuve}
@hp.com

Margret Branschofsky, Daniel Chudnov,
Greg McClellan, MacKenzie Smith
MIT Libraries
77 Massachusetts Ave
Cambridge, MA 02139
{branschofsky, dchud, gam, kenzie}
@mit.edu

Abstract

In this paper we describe DSpace™, an open source system that acts as a repository for digital research and educational material produced by an organization or institution. DSpace was developed during two years' collaboration between the Hewlett-Packard Company and MIT Libraries. The development team worked closely with MIT Libraries staff and early adopter faculty members to produce a 'breadth-first' system, providing all of the basic features required by a digital repository service. As well as functioning as a live service, DSpace is intended as a base for extending repository functionality, particularly to address long-term preservation concerns. We describe the functionality of the current DSpace system, and briefly describe its technical architecture. We conclude with some remarks about the future development and operation of the DSpace system.

Categories and Subject Descriptors

H.3.7 [**Digital Libraries**]: Collection; Standards; User issues; Dissemination; Systems issues

General Terms

Management, Design, Standardization

Keywords

Digital library, preservation, institutional repository, open archives

1. Introduction

As more and more research and educational material is 'born digital', institutions and organizations are increasingly realizing the need for a stable place in which such material may be stored and accessed long-term. The Massachusetts Institute of Technology is a perfect example of an organization with this need. Much of the material produced by faculty, such as datasets, experimental results and rich media data as well as more conventional document-based material (e.g. articles and reports), is housed on an individual's hard drive or department Web server. Such material is often lost forever as faculty and departments change over time [10].

Providing services for the long-term stewardship of digital material seems a natural extension of the role of MIT Libraries. Although systems such as document management systems exist in this area, many are commercial and proprietary. Both of these factors raise the barrier for long-term preservation, since the hosting institution is reliant on the survival and affordability of the vendor. Since no existing system was a good fit, Hewlett-Packard and MIT Libraries collaborated over two years to create the DSpace digital repository platform [18]. DSpace provides the basic functionality required to operate an institutional digital repository, and is intended to serve as a base for future development to address long-term preservation and access issues. On November 4, 2002, the system was launched as a live service hosted by MIT Libraries, and the source code made publicly available according to the terms of the BSD open source license [15], with the intention of encouraging the formation of an open source community around DSpace. Initial developments in this area have been very promising.

This paper describes the various functional aspects of the DSpace system, followed by a brief overview of the architecture of the system. The paper concludes with some remarks about future plans for the DSpace system.

2. Related Work

DSpace draws on a wealth of previous research and development in the area of digital library systems. Though not a full implementation, the DSpace architecture has roots in Kahn and Wilensky's Framework for Distributed Digital Object Services [9], as well as Arms et al.'s work on digital library architecture [1], [2]. DSpace does not yet support complex dissemination of objects, but future releases will build on existing work on Lagoze et al.'s FEDORA architecture [16], and the prototype implementation of FEDORA at the University of Virginia [19].

Another important piece of work that DSpace draws on is the Consultative Committee for Space Data Systems' Reference Model for an Open Archival Information System (OAIS) [4]. This paper makes use of the concepts and terms defined in that work.

The EPrints system developed at the University of Southampton [6] has many similar features to DSpace, but is optimized to provide access to author-deposited, document-style material, while DSpace provides a platform to begin work on long-term preservation strategies for digital material, including documents and other material used in scholarly research. DSpace's submission user interface in particular draws on experience gained from the design and use of EPrints' submission user interface. Interoperability with EPrints is of course desirable, and can currently be achieved in part through use of the OAI Protocol for Metadata Harvesting (OAI-PMH) [14] to provide a cross-archive access service.

The Greenstone software from New Zealand Digital Library Project at the University of Waikato [7] is another open source digital library tool that has a focus on publishing.

CERN have developed the CERN Document Server Software (CDSware) [3] that is another preprint server.

Commercial document management systems offer some of the required functionality but do not really suit the aim of long-term preservation of the material, partly due to their proprietary nature.

3. Functional Overview

DSpace is designed to operate as a centralized, institutional service. Different *communities* within the institution such as labs, centers, schools or departments can have their own separate areas within the system. Members of these communities deposit content directly via a Web user interface designed to make this depositing as simple as possible. Alternatively the system features a batch item importer for the bulk loading of content.

Each community may also appoint people as 'gatekeepers', who may review and edit submissions before their inclusion in the main repository. The DSpace system then indexes the metadata submitted with the digital item and makes it available according to the access privileges determined by the community.

In order to provide a workable service in the available time, DSpace was developed 'breadth-first'. In other words, each of the basic requirements of an institutional digital repository system was addressed in a relatively simple manner, so that functionality can evolve with the service already in production.

The functional aspects of DSpace can be summarized as follows:

- A **data model** for basic organization of data is defined
- **Metadata** of various types is stored by the system
- The system stores information about users of the system. Some users might not be humans but other computerized systems; hence we call users **e-people**
- While much of the effort is concerned with easing access to an institution's digital material, simply allowing full public access is not always acceptable. Additionally functions such as depositing and reviewing must be restricted to appropriate individuals. Hence the system includes an **authorization** function
- The system must be able to accept incoming material, a process called **ingesting** [4]
- Some communities may require that material or accompanying metadata entering the archive be checked or augmented by designated individuals. This process is called **workflow**
- In order that material in the archive may be cited and accessed using information in a citation, the CNRI **Handle system** [8] is used to assign globally unique, persistent identifiers to archived objects ('items').

- End users should be able to explore and discover the contents of the repository. To this end, DSpace must offer **search** and **browse** functions
- To further increase the possibilities for discovering material in DSpace, metadata is exposed via the **Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)** [4]
- It should be possible to notify end users of the system when new content of interest to them appears in the archive, rather than requiring them to repeatedly access DSpace to check this. DSpace offers an automatic e-mail alerting service called **subscription**
- A **Web user interface** provides access to the above functionality

The following sections discuss each of these functional aspects in detail.

3.1 Data Model

The way data is organized in DSpace is intended to reflect the structure of the organization using the DSpace system. This is depicted in Figure 1. Table 1 shows examples of each type of object.

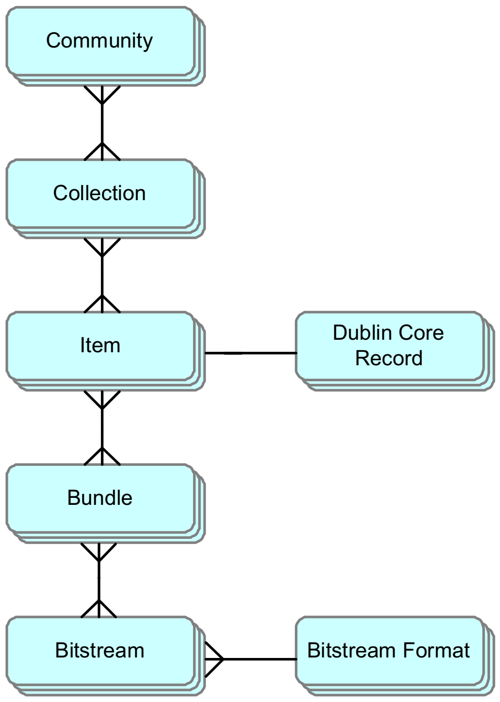


Figure 1: Data Model Diagram

Each DSpace site is divided into *communities*; these typically correspond to a laboratory, research center or department. Communities contain *collections*, which is a grouping of related content. Each collection is composed of *items*, which are the basic archival elements of the archive. Items are further subdivided into *bundles* of *bitstreams*. Bitstreams are, as the name suggests, streams of bits, usually ordinary computer files. Bitstreams that are somehow closely related are organized into bundles, for example HTML files and images that compose a single HTML document.

Object Type	Example Instance
Community	Laboratory for Computer Science; Oceanographic Research Centre
Collection	LCS Technical Reports; ORC Statistical Data Sets
Item	A technical report; a data set with accompanying description; a video recording of a lecture
Bundle	A group of HTML and image bitstreams making up an HTML document
Bitstream	A single HTML file; a single image file; a source code file
Bitstream Format	Microsoft Word version 6.0; JPEG encoded image format

Table 1: Example DSpace Objects

The data model supports multiple inclusion at all levels; that is, an item may belong to more than one collection, and a collection may be in more than one community[1].

Each item has one qualified Dublin Core metadata record. Other metadata might be stored in an item as a serialized bitstream, but we store Dublin Core for every item for interoperability and ease of discovery. The Dublin Core may be entered by end-users as they submit content, or it might be derived from other metadata as part of an ingest process.

In an ideal world, nothing entering a DSpace system should ever be removed; however, practical and legal factors sometimes necessitate this. Items can be removed from DSpace in one of two ways: They may be 'withdrawn', which means they remain in the archive but are completely hidden from view. In this case, if an end-

[1] It should be noted that software has not yet been extensively tested to create or deal with such circumstances. This will be a requirement for MIT in the near future so forthcoming updates will address this.

user attempts to access the withdrawn item, they are presented with a 'tombstone' that indicates the item has been removed. For whatever reason, an item may also be 'expunged' if necessary, in which case all traces of it are removed from the archive.

3.1.1 Bitstream Formats. Each bitstream is associated with one Bitstream Format. Because preservation services are an important aspect of the DSpace service, it is important to capture the specific formats of files that users submit. In DSpace, a bitstream format is a unique and consistent way to refer to a particular file format. An integral part of a bitstream format is an either implicit or explicit notion of how material in that format can be interpreted. For example, the interpretation for bitstreams encoded in the JPEG standard for still image compression is defined explicitly in the Standard ISO/IEC 10918-1. The interpretation of bitstreams in Microsoft Word 2000 format is defined implicitly, through reference to the Microsoft Word 2000 application. Bitstream formats can be more specific than MIME types or file suffixes. For example, application/ms-word and .doc span multiple versions of the Microsoft Word application, each of which produces bitstreams with presumably different semantics.

Supported	The format is recognized, and the hosting institution is confident it can make bitstreams of this format useable in the future, using whatever combination of techniques (such as migration, emulation, etc.) is appropriate given the context of need.
Known	The format is recognized, and the hosting institution will promise to preserve the bitstream as-is, and allow it to be retrieved. The hosting institution will attempt to obtain enough information to enable the format to be upgraded to the 'supported' level.
Unsupported	The format is unrecognized, but the hosting institution will undertake to preserve the bitstream as-is and allow it to be retrieved.

Table 2: Bitstream Format Support Levels

Each bitstream format additionally has a *support level*, indicating how well the hosting institution is likely to be able to preserve content in the format in the future. There are three possible support levels that bitstream formats may be assigned by the hosting institution. The host institution should determine the exact meaning of each support level, after careful consideration of costs and requirements. MIT Libraries' interpretation is shown in Table 2.

3.2 Metadata

Broadly speaking, DSpace holds three sorts of metadata about archived content: Descriptive, administrative and structural metadata.

3.2.1 Descriptive Metadata. Each *Item* has one qualified Dublin Core metadata record. MIT Libraries, and the default configuration shipped with the open source use a derivation of the Library Application Profile set of elements and qualifiers [5]. Institutions with other requirements can easily change this, as DSpace maintains a *registry* of elements and qualifiers, though the system's search functionality and submission UI would not be updated automatically in the present version of the system.

Other descriptive metadata about items, for example MARC records, may be held in serialized bitstreams. *Communities* and *collections* have some simple descriptive metadata (a name, and some descriptive prose), held in the DBMS.

3.2.2 Administrative Metadata. This includes preservation metadata, provenance and authorization policy data. Most of this is held within DSpace's relational DBMS schema. Provenance metadata (prose) is stored in Dublin Core records. Additionally, some other administrative metadata (for example, bitstream byte sizes and MIME types) is replicated in Dublin Core records so that it is easily accessible outside of DSpace, for example via the OAI protocol.

3.2.3 Structural Metadata. This includes information about how to present an item, or bitstreams within an item, to an end-user, and the relationships between constituent parts of the item. As an example, consider a thesis consisting of a number of TIFF images, each depicting a single page of the thesis. Structural metadata would include the fact that each image is a single page, and the ordering of the TIFF images/pages. Structural metadata in DSpace is currently fairly basic; within an item, bitstreams can be arranged into separate bundles as described above. Additional structural metadata can be stored in serialized bitstreams, but DSpace does not currently natively understand this. This will be a very active area of future development of DSpace.

3.3 E-people

Many of DSpace's features such as document discovery and retrieval can be used anonymously, but users must be authenticated to perform functions such as submission, email notification ('subscriptions') or administration. Users are also grouped for easier administration. DSpace calls users *e-people*, since some users may be machines rather than actual people.

DSpace holds the following information about each e-person:

- E-mail address
- First and last names
- Authentication information, such as an encrypted password
- A list of collections for which the e-person wishes to be notified of new items
- Whether the e-person 'self-registered' with the system; that is, whether the system created the e-person record automatically as a result of the end-user independently registering with the system, as opposed to the e-person record being generated from the institutions personnel database, for example.

E-people can be members of 'groups' to make administrator's lives easier when manipulating authorization policies.

READ	The action of knowing of an object's existence, and viewing any metadata associated with it
WRITE	Modifying the metadata associated with an object. This does not include the ability to delete
ADD	The action of adding an object (e.g. an item) to a container (e.g. a collection). In order to submit an item to a collection, an end-user must have ADD permission on that collection
REMOVE	The action of removing an object from a container
WORKFLOW	May participate in a workflow associated with a collection; for example, permission to reject a particular submission from entering the collection

Table 3: Possible Actions in DSpace

3.4 Authorization

DSpace has a flexible authorization system. In order for a user to perform an action on an object, they must have permission; DSpace operates a 'default deny' policy. Permissions do not 'commute'; for example, if an e-person has READ permission on an item, they might not necessarily have READ permission on the bundles and bitstreams in that item.

The actions that the authorization system understands are shown in Table 3. Note that there is no 'DELETE' action. In order to 'delete' an object (e.g. an item) from the archive, one must have REMOVE permission on all objects (in this case, collection) that contain it. The 'orphaned' item is automatically deleted.

Policies can apply to individual e-people or groups of e-people. Additionally they can apply to the 'anonymous' group, which means that anyone can perform the action described by the policy.

3.5 Ingest Process and Workflow

Figure 2 is a simple illustration of the current ingesting process in DSpace. The batch item importer is an application that turns an external Submission Information Package (SIP) [4] (an XML metadata document with some content files) into an "in progress submission" object. The Web submission UI is similarly used by an end-user to assemble an "in progress submission" object.

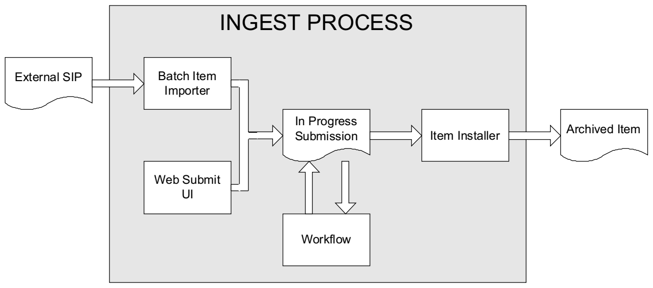


Figure 2: DSpace Ingest Process

Depending on the policy of the collection to which the submission is targeted, a workflow process may be started. This typically allows one or more human reviewers or 'gatekeepers' to check over the submission and ensure it is suitable for inclusion in the collection. Workflows are defined when the collection is first established, set up by the system administrator, and apply to every item submitted to that collection.

When the batch item importer or Web Submit UI completes the InProgressSubmission object, and invokes the next stage of ingest (be that workflow or item installation), a provenance message is added to the Dublin Core which includes the filenames and checksums of the content of the submission. Likewise, each time a workflow changes state (e.g. a reviewer accepts the submission), a similar provenance statement is added. This allows us to track how the item has changed since a user submitted it. (The History subsystem is also invoked, but provenance is easier for us to access at the moment.)

Once any workflow process is successfully and positively completed, the InProgressSubmission object is consumed by an "item installer", that converts the in progress submission into a fully blown archived item in DSpace. The item installer:

- Assigns an accession date
- Adds a "date.available" value to the Dublin Core metadata record of the item
- Adds an issue date if none already present
- Adds a provenance message (including bitstream checksums)
- Assigns a Handle persistent identifier
- Adds the item to the target collection, and adds appropriate authorization policies
- Adds the new item to the search and browse indices
- (Soon) creates and archives an OAIS Archival Information Package, represented in a standard, open format such as METS [12].

3.5.1 Workflow Steps. A collection's workflow can have up to three steps. Each collection may have an associated e-person group for performing each step; if no group is associated with a certain step, that step is skipped. If a collection has no e-person groups associated with any step, submissions to that collection are installed straight into the main archive.

In other words, the sequence is this: The collection receives a submission. If the collection has a group assigned for workflow step 1, that step is invoked, and the group is notified. Otherwise, workflow step 1 is skipped. Likewise, workflow steps 2 and 3 are performed if and only if the collection has a group assigned to those steps.

When a step is invoked, the task of performing that workflow step is put in the 'task pool' of the associated group. One member of that group takes the task from the pool, and it is then removed from the task pool, to avoid the situation where several people in the group may be performing the same task without realizing it.

The member of the group who has taken the task from the pool may then perform one of three actions, shown in Table 4.

If a submission is rejected, the reason (entered by the workflow participant) is e-mailed to the submitter, and it is returned to the submitter's workspace. The submitter can then make any necessary modifications and re-submit, whereupon the process starts again.

If a submission is 'accepted', it is passed to the next step in the workflow. If there are no more workflow steps with associated groups, the submission is installed in the main archive.

One last possibility is that a workflow can be 'aborted' by a DSpace site administrator. This is accomplished using the administration section of the Web UI.

Workflow Step	Possible actions
1	Can accept submission for inclusion, or reject submission.
2	Can edit metadata provided by the user with the submission, but cannot change the submitted files. Can accept submission for inclusion, or reject submission.
3	Can edit metadata provided by the user with the submission, but cannot change the submitted files. Must then commit to archive; may not reject submission.

Table 4: Possible Workflow Steps in DSpace

3.6 Handles

Researchers require a stable point of reference for their works. The simple evolution from sharing of citations to emailing of URLs broke when Web users learned that sites can disappear or be reconfigured without notice, and that their bookmark files containing critical links to research results could not be trusted long term. To help solve this problem, a core DSpace feature is the creation of persistent identifier for every item, collection and community stored in DSpace. To persist identifiers, DSpace requires a storage- and location-independent mechanism for creating and maintaining identifiers. DSpace uses the CNRI Handle System for creating these identifiers [8]. The rest of this section assumes a basic familiarity with the Handle system.

DSpace uses Handles primarily as a means of assigning globally unique identifiers to objects. Each site running DSpace needs to obtain a Handle 'prefix' from CNRI, so we know that if we create identifiers with that prefix, they will not clash with identifiers created elsewhere.

Presently, Handles are assigned to communities, collections, and items. Bundles and bitstreams are not assigned Handles, since over time, the way in which an item is encoded as bits may change, in order to allow access with future technologies and devices. Older versions may be moved to off-line storage as a new standard becomes de facto. Since it is usually the *item* that is being cited, rather than the particular bit encoding, it only makes sense to persistently identify and allow access to the item, and allow users to access the appropriate bit encoding from there.

The Handle system also features a global resolution infrastructure; that is, an end-user can enter a Handle into any service (e.g. Web page) that can resolve Handles, and the end-user will be directed to the object (in the case of DSpace, community, collection or item) identified by that Handle. In order to take advantage of this feature of the Handle system, a DSpace site must also run a 'Handle server' that can accept and resolve incoming resolution requests. All the code for this is included in the DSpace source code bundle.

Handles can be written in two forms:

```
hdl:1721.123/4567
http://hdl.handle.net/1721.123/4567
```

The above represent the same Handle. The first is possibly more convenient to use only as an identifier; however, by using the second form, any Web browser becomes capable of resolving Handles by means of a proxy server run by CNRI. An end-user need only access this form of the Handle as they would any other URL. It is possible to enable some browsers to resolve the first form of Handle as if they were standard URLs using CNRI's Handle Resolver plug-in, but since the first form can always be simply derived from the second, DSpace displays Handles in the second form, so that it is more useful for end-users.

It is important to note that DSpace uses the CNRI Handle infrastructure only at the 'site' level. For example, in the above example, the DSpace site has been assigned the prefix '1721.123'. It is still the responsibility of the DSpace site to maintain the association between a full Handle (including the '4567' local part) and the community, collection or item in question. This is done internally with a database table.

3.7 Search and Browse

DSpace allows end-users to discover content in a number of ways, including:

- Via external reference, such as a Handle
- Searching for one or more keywords
- Browsing though title, date and author indices

Search is an essential component of discovery in DSpace. Users' expectations from Web search engines are quite high, so a goal for DSpace is to supply as many search features as possible. DSpace's indexing and search module has a very simple API which allows for indexing new content, regenerating the index, and performing searches on the entire corpus, a community, or collection. Behind the API is the Java freeware search engine Lucene. Lucene gives us fielded searching, stop words, stemming, and the ability to incrementally add new indexed content without regenerating the entire index.

Another important mechanism for discovery in DSpace is the browse. This is the process whereby the user views a particular index, such as the title index, and navigates around it in search of interesting items. The browse subsystem provides a simple API for achieving this by allowing a caller to specify an index, and a subsection of that index. The browse subsystem then discloses the portion of the index of interest. Indices that may be browsed are item title, item issue date and authors. Additionally, the browse can be limited to items within a particular collection or community.

3.8 OAI Support

The Open Archives Initiative has developed a Protocol for Metadata Harvesting (OAI-PMH) [14]. This allows sites to programmatically retrieve or 'harvest' the metadata from several sources, and offer services using that metadata, such as indexing or linking services. Such a service could allow users to access information from a large number of sites that are collated in a central catalog.

DSpace exposes the Dublin Core metadata for items that are publicly (anonymously) accessible. Additionally, the community and collection structure is also exposed via OAI-PMH's 'sets' mechanism. OCLC's open source OAICat framework is used to provide this functionality [13].

DSpace's OAI-PMH service exposes deletion information for withdrawn items.

While OAICat supports resumption tokens, DSpace does not, simply because of time and resource constraints on the development effort. The need for flow control in the harvesting will increase as the amount of content in DSpace archives increases, so a future release will include resumption token support.

3.9 Subscriptions

As noted above, end-users (e-people) may 'subscribe' to collections through the Web user interface in order to be alerted when new items are added to those collections. Each day, end-users who are subscribed to one or more collections will receive an e-mail giving brief details of all new items that appeared in any of those collections the previous day. If no new items appeared in any of the subscribed collections, no e-mail is sent. Users can unsubscribe themselves at any time.

3.10 History

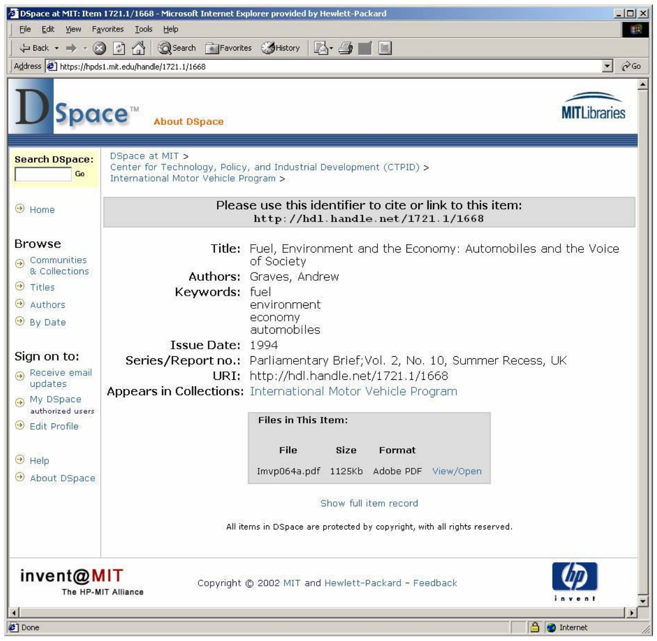
While provenance information in the form of prose is very useful, it is not easily programmatically manipulated. The History system captures a time-based record of significant changes in DSpace, in a manner suitable for later 'refactoring' or repurposing.

Currently, the History subsystem is explicitly invoked when significant events occur (e.g., DSpace accepts an item into the archive). The History subsystem then creates RDF data [20] describing the current state of the object. The RDF data is modelled using the ABC Model [11], an ontology for describing temporal-based data, and stored in the file system. Some simple indices for unwinding the data are available.

3.11 Web User Interface

Built on Java Servlet and JavaServer Page technology, DSpace's Web user interface allows end-users to access DSpace via their Web browsers. The user interface consists of the following:

- On-line help
- Community and collection home pages, configurable by individual communities. Recent arrivals in the collection are displayed, as well as convenient subscribing and depositing controls
- Searching and browsing
- Item pages, which display the basic metadata associated with an item. The full Dublin Core and bitstreams contained in an item may be accessed from this page (subject to authorization). DSpace does not yet have a sophisticated dissemination mechanism; we are investigating relevant work such as FEDORA [16] to address this. Figure 3 shows an example item display page


DSpace at MIT: Item 1721.1/1668 - Microsoft Internet Explorer provided by Hewlett-Packard
File Edit View Favorites Tools Help
Back Search Favorites History
Address https://hpds1.mit.edu/handle/1721.1/1668
Go
DSpace™
About DSpace
MITLibraries
Search DSpace:
Go
Home
Browse
Communities & Collections
Titles
Authors
By Date
Sign on to:
Receive email updates
My DSpace authorized users
Edit Profile
Help
About DSpace
invent@MIT
The HP-MIT Alliance
invent
Done
Internet

DSpace at MIT >
Center for Technology, Policy, and Industrial Development (CTPID) >
International Motor Vehicle Program >

Please use this identifier to cite or link to this item:
`http://hdl.handle.net/1721.1/1668`

Title:	Fuel, Environment and the Economy: Automobiles and the Voice of Society
Authors:	Graves, Andrew
Keywords:	fuel environment economy automobiles
Issue Date:	1994
Series/Report no.:	Parliamentary Brief;Vol. 2, No. 10, Summer Recess, UK
URI:	http://hdl.handle.net/1721.1/1668
Appears in Collections:	International Motor Vehicle Program

Files in This Item:

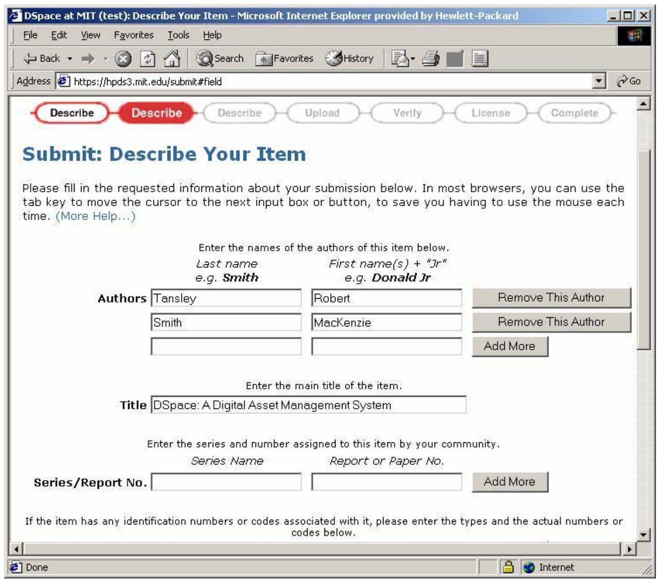
File	Size	Format	
Imvp064a.pdf	1125Kb	Adobe PDF	View/Open

Show full item record

All items in DSpace are protected by copyright, with all rights reserved.

Copyright © 2002 MIT and Hewlett-Packard - Feedback

Figure 3: Item Display Page


DSpace at MIT (test): Describe Your Item - Microsoft Internet Explorer provided by Hewlett-Packard
File Edit View Favorites Tools Help
Back Search Favorites History
Address https://hpds3.mit.edu/submit#field
Go
Describe
Describe
Describe
Upload
Verify
License
Complete
Remove This Author
Remove This Author
Add More
Add More
Done
Internet

Submit: Describe Your Item

Please fill in the requested information about your submission below. In most browsers, you can use the tab key to move the cursor to the next input box or button, to save you having to use the mouse each time. (More Help...)

Enter the names of the authors of this item below.

	Last name e.g. **Smith**	First name(s) + "Jr" e.g. **Donald Jr**
Authors	Tansley	Robert
	Smith	MacKenzie

Enter the main title of the item.

Title: DSpace: A Digital Asset Management System

Enter the series and number assigned to this item by your community.

	Series Name	Report or Paper No.
Series/Report No.		

If the item has any identification numbers or codes associated with it, please enter the types and the actual numbers or codes below.

Figure 4: Deposit Interface Screen

- 'My DSpace', where authorized users can deposit material, perform any workflow tasks they have been assigned, and manage their automatic e-mail alert subscriptions. Figure 4 shows an example of the depositing interface

- Administration section, consisting of pages intended for use by central administrators. Presently, this part of the Web UI is very basic so users of the administration subsystem need to know what they are doing! In the future, as this subsystem is improved, it will be possible to pass more responsibility for administration to individual communities as the administration UI becomes more developed.

Although the entire system is designed to be easy to modify to suit an institution's needs, the Web UI has been designed to be particularly so:

- Since JavaServer Pages are mostly HTML with small pieces of Java embedded code, they are easy to modify without touching the business logic code in the Servlets.
- Institutions are also likely to have existing electronic authentication infrastructure. By implementing a Java interface and altering a configuration parameter, DSpace can be made to use this local authentication infrastructure. DSpace is shipped with a simple e-mail address/password authentication module, and the MIT implementation, which understands X509 certificates.

4. System Architecture

The main code of DSpace is implemented in Java, and runs on any UNIX-like system such as Linux or HP-UX. It makes use of several third-party open source systems:

- PostgreSQL, an open source relational database system
- Jakarta Tomcat Java Servlet container
- Apache HTTPD server, for optional SSL and X509 certificate support

Alternative tools may be used in place of these; for example Tomcat may be replaced with Caucho's 'Resin' application server. In order to minimize the barrier to adoption, however, by default DSpace makes use of an entirely free, open source tool stack.

The main DSpace system is organized into three layers, each of which consists of a number of components. Figure 5 depicts this.

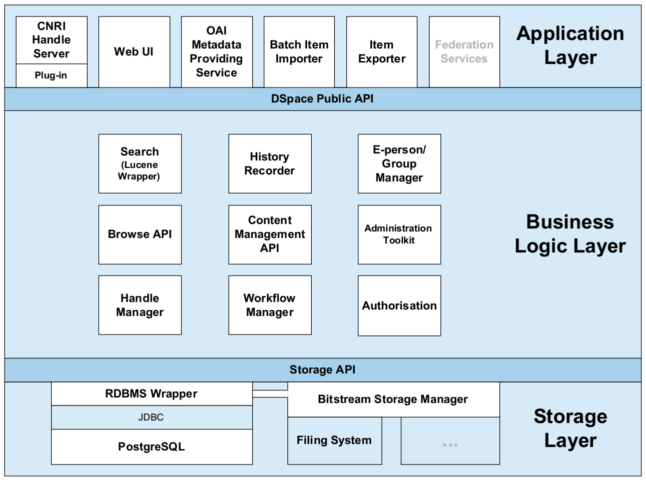


Figure 5: DSpace System Architecture

The storage layer is responsible for physical storage of metadata and content. It consists of a 'wrapper' around JDBC for database access, and a simple bitstream storage and retrieval API called the bitstream storage manager. Presently, the bitstream storage manager is very lightweight and simply stores bitstreams in a file system. This can easily be modified, and will shortly be extended to cover multiple file systems to enable larger volumes of content to be stored.

The business logic layer deals with managing the content of the archive, users of the archive (e-people), authorization, and workflow.

The application layer contains components that communicate with the world outside of the individual DSpace installation, for example the Web user interface and the Open Archives Initiative protocol for metadata harvesting service.

Each layer only invokes the layer below it; the application layer may not use the storage layer directly, for example. Each component in the storage and business logic layers has a defined public API. The union of the APIs of those components is referred to as the Storage API (in the case of the storage layer) and the DSpace Public API (in the case of the business logic layer). These APIs are in-process Java classes, objects and methods. These could be exposed as Web services via a suitable component in the application layer.

Since each component has a clearly defined API, they may be modified and replaced individually, without requiring extensive modification of the rest of the system.

It is important to note that each layer is trusted. Although the logic for authorizing actions is in the business logic layer, the system relies on individual applications in the application layer to authenticate e-people correctly and securely. If a 'hostile' or insecure application were allowed to invoke the Public API directly, it could very easily perform actions as any e-person in the system.

The reason for this design choice is that authentication methods will vary widely between different applications, so it makes sense to leave the logic and responsibility for that in those applications.

5. Conclusions and Future Work

DSpace has been a ‘breadth-first’ attempt to start addressing a growing and unfulfilled need of academic institutions and other organizations. MIT Libraries has been running DSpace as a live service at MIT for several months, and several other institutions have successfully installed and started to run DSpace. MIT Libraries remain committed to maintaining the DSpace service and software.

Naturally, DSpace does not currently address all of the issues of long-term preservation and access of digital material; however it serves as a useful basis for developing and deploying solutions to those issues. Already, an open source community is forming around DSpace. This exciting development bodes well for the future development and impact of DSpace. In addition, two further pieces of work are already under way to enhance the DSpace system.

Starting with seven other research institutions (Cambridge University in the UK, the University of Toronto in Canada, and Columbia University, Cornell University, Ohio State University, and the Universities of Rochester and Washington in the USA) MIT is establishing a federation of DSpace partners to explore the issues around deploying the DSpace service at different locales. Much of this work will involve exploring organizational issues, as well as maintaining and enhancing the functionality of the DSpace code base.

Hewlett-Packard Laboratories, the World-Wide Web Consortium and MIT are also collaborating in another project called SIMILE [17]. The SIMILE work involves exploring the use of RDF and Semantic Web techniques to address two problems. The first is how to achieve interoperability among diverse metadata schemas, and the digital works to which they are applied. The second problem is to do with individuals interacting with DSpace: Given this diverse metadata, how does DSpace allow individual users to find and organize information relevant to them?

The success of the DSpace work to date, combined with the strong interest in the digital library community and the exciting new work being undertaken, give us great confidence in our ability to contribute to the fields of the preservation and management of digital research and education material, and the field of open access to such material.

6. Acknowledgments

We would like to thank the following for their previous contribution to the DSpace work: Mary Barton, Peter Breton, Peter Carmichael, Bill Cattey, Eric Celeste, Joyce Ng and Julie Harford Walker. We would also like to thank the staff at MIT Libraries and the members of our early adopter communities for their valuable input.

7. References

[1] Arms, William Y.: Key Concepts in the Architecture of the Digital Library, *D-Lib Magazine*, July 1995. http://www.dlib.org/dlib/July95/07arms.html

[2] Arms, William Y., Blanchi, Christophe, and Overly, Edward A.: An Architecture for Information in Digital Libraries, *D-Lib Magazine*, February 1997. http://www.dlib.org/dlib/february97/cnri/02arms1.html

[3] CERN Document Server Software (CDSware). http://cdsware.cern.ch/

[4] Consultative Committee for Space Data Systems, Reference Model for an Open Archival Information System (OAIS), CCSDS 650.0-R-2, Red Book, Issue 2, July 2001. http://ccsds.org/documents/pdf/CCSDS-650.0-R-2.pdf

[5] Dublin Core Library Application Profile. http://dublincore.org/documents/2002/09/24/library-application-profile/

[6] GNU EPrints Software. http://software.eprints.org/

[7] The Greenstone Digital Library Software http://www.greenstone.org/

[8] Handle System Overview. http://www.ietf.org/internet-drafts/draft-sun-handle-system-10.txt

[9] Kahn, Robert and Wilensky, Robert: A Framework for Distributed Digital Object Services, May 1995. http://www.cnri.reston.va.us/home/cstr/arch/k-w.html

[10] Koehler, Wallace: Web Page Change and Persistence-A Four-Year Longitudinal Study. In *Journal of the American Society for Information Science and Technology* 53, 2 (December 2001) 162—180.

[11] Lagoze, C. and Hunter, J. "The ABC Ontology and Model", in *Proceedings of the International Conference on Dublin Core and Metadata Applications 2001,* Tokyo, 2001, 160-176.

[12] Metadata Encoding and Transmission Standard (METS). http://www.loc.gov/standards/mets/

[13] OCLC Research OAICat Open Source Project. http://www.oclc.org/research/software/oai/cat.shtm

[14] Open Archives Initiative. http://www.openarchives.org/

[15] Open Source BSD License. Available at http://www.opensource.org/licenses/bsd-license.php

[16] Payette, Sandra and Lagoze, Carl: Flexible and Extensible Digital Object and Repository Architecture, in Christos Nikolau and Constantine Stephanidis, eds., *Research and Advanced Technologies for Digital Libraries: Proceedings of the Second European Conference, ECDL '98, Heraklion, Crete, Greece, September 21-23, 1998, G. Goos, J. Hartmanis, and J. van Leeuwen, eds., Lecture Notes in Computer Science, 1513 (Berlin: Springer, 1998)* http://www.cs.cornell.edu/payette/papers/ECDL98/FEDORA.html

[17] SIMILE: Semantic Interoperability of Metadata and Information in unLike Environments. http://web.mit.edu//simile/

[18] Smith, MacKenzie, Barton, Mary, Bass, Mick, Branschofsky, Margret, MacClellan, Greg, Stuve, David, Tansley, Robert and Walker, Julie H: DSpace: An Open Source Dynamic Digital Repository. *D-Lib Magazine* 9, 1 (January 2003). http://www.dlib.org/dlib/january03/smith/01smith.html

[19] Staples, Thornton and Wayland, Ross: Virginia Dons FEDORA: A Prototype for a Digital Object Repository. *D-Lib Magazine* 6, 7/8 *(*July/August 2000). http://www.dlib.org/dlib/july00/staples/07staples.html

[20] W3C Resource Description Framework (RDF). http://www.w3.org/RDF/