Architecting an Extensible Digital Repository

Anoop Kumar, Ranjani Saigal
Academic Technology
Tufts University
16 Dearborn Road
Medford, MA 02155
{anoop.kumar,ranjani.saigal}@tufts.edu

Robert Chavez
Digital Collections and Archives
Tufts University
Tisch Library-35 Professors
Medford, MA 02155
Robert.chavez@tufts.edu

Nikolai Schwertner
Department of Computer Science
Tufts University
161 College Ave.
Medford Ma, 02155
nikolai.schwertner@tufts.edu

ABSTRACT

The Digital Collection and Archives (DCA) in partnership with Academic Technology (AT) at Tufts University developed a digital library solution for long-term storage and integration of existing digital collections, such as Perseus, TUSK, Bolles and Artifact. In this paper, we describe the Tufts Digital Library (TDL) architecture. TDL is an extensible, modular, flexible and scalable architecture that uses Fedora at its core. The extensible nature of the TDL architecture allows for seamless integration of collections that may be developed in the future, while leveraging the extensive tools that are available as part of individual digital library applications at Tufts. We describe the functionality and implementation details of the individual components of TDL. Two applications that have successfully interfaced with TDL are presented. We conclude with some remarks about the future development of TDL.

Categories and Subject Descriptors

H.3.7 [**Digital Libraries**]: Collection, Dissemination, Standards, System Issues, User Issues.

General Terms

Management, Performance, Design, Security.

Keywords

Digital Library, preservation, Fedora, VUE.

1. Introduction

During the past decade colleges and universities have witnessed an exponential growth in digital information available for teaching and learning. There are many collections of digital objects including images, texts, audios and videos that have great value in a diverse set of fields. As the quantity of information continues to increase and these collections expand, there is need for a repository that can provide appropriate storage and access to all these valuable material in a flexible and extensible manner for the foreseeable future. This need has led many organizations to select a digital library solution that can assimilate current collections and accommodate new materials as they become available.

The Digital Collection and Archives (DCA) in partnership with Academic Technology (AT) at Tufts University has developed a digital library solution that provides for long-term storage and integration of existing digital collections, while leveraging the extensive tools developed by individual digital library projects at Tufts. The Digital Library system developed to serve the needs at Tufts builds on and extends successful models that are currently in vogue in the digital library community. This paper describes the architecture of the Tufts Digital Library (TDL) which is designed to allow assimilation and interoperability of existing Tufts digital libraries while allowing new creators of digital materials to add their content and write new applications for using and managing the material.

The paper describes two applications – Visual Understanding Environment (VUE), a concept mapping application and Tufts Digital Library Search that successfully interface with this architecture to use the content of the repository.

2. Related Work

The architecture for TDL incorporates the concepts described in the emerging standards for trusted digital repositories [5], and complies with the Reference Model for an Open Archival Information System (OAIS) functional and information models for archival information systems [6]. On a practical level this means that we have applied the principles of the "trusted digital repository" and OAIS guidelines in the arrangement of our system architecture [6], matching requirements to system services.

The format to assign persistent IDs or Uniform Resource Names (URN) to objects across digital collections uses the standard specified in the RFCs on URNs [8-10]. OCLC's Persistent Uniform Resource Locators (PURL) [11] was used as a basis to create a "Naming Service" that creates and resolves URNs..

The implementation of Fedora [1-2] by University of Virginia forms the core of the TDL architecture. It provides the "plumbing" or framework for all the components and the services in the architecture. Havard's LDI, [12] uses a model that clearly separates the collections infrastructure, access infrastructure and common services to support their large and unusually decentralized library system. This modular approach has been used a basis to develop the TDL component architecture. We also used the ideas of interoperability, scalability and digital preservation that form the core of the Making of America II [13] project.

The abstract model that represents digital objects is drawn from Lagoze's Warwick framework [3] and Kahn and Wilensky's Framework for Distributed Digital Objects[20]. The model was developed taking into consideration the Digital Repository (DR) Interface proposed by MIT's Open Knowledge Initiative (OKI) [7]. The objects in the repository can be easily accessed and managed by applications that support the OKI-DR interface.

TDL uses the Dublin Core[21] Metadata set for storing fields such as author, title, subject, etc. Administrative metadata, gathered through the METS XML file is used for processing Fedora Objects. Metadata is also acquired and managed as "Datastreams" - a concept that is proposed in the Fedora object model.

The TDL search application uses Lucene [22], which is part of the Jakarta project. It is an open source search engine that provides full text based search, metadata search and advanced search features.

3. Designing the Tufts Digital Library

3.1 Need for a New Architecture

Three major discipline specific digital library applications have been developed at Tufts University. Perseus, [15] a digital library in the area of classics has over a million objects and receives about 8 million page hits a month. The collection includes multilingual marked up texts, images audios and videos. The Tufts University Science Knowledgebase or TUSK [17] is a repository of materials specific to the health sciences field. It is widely used by the Medical, Veterinary and Dental schools for teaching a variety of courses. Artifact is a collection of about 2500 images that is used to teach courses in Art History [18]. Tufts also has two substantial collections of digital materials - the Bolles Collection [24] which is a collection of some unique historical maps of London and topographical data and Crime and Punishment [19], which a collection of videos, images and cases used for simulations by Political Science faculty. Table 1 provides details about these collections.

Table 1. Digital Libraries at Tufts

Digital Libraries	Size	Description
Perseus	50 million words	The subset of Perseus Project Classics collection data that the Tufts Digital library is working with is composed primarily of highly structured TEI encoded XML texts of many types, including various forms of Lexica, Grammars, Encyclopedias, ancient and modern language texts.
Bolles Collection	13 million words, 25,000 images, geospatial datasets and multimedia objects	The Bolles collection contains highly structured TEI encoded XML texts, PDF documents, high resolution TIFF images, QuickTime Virtual Reality files.
TUSK	15,000 documents, 125 courses (approximately)	Tufts University Science Knowledgebase (TUSK) contains full-text syllabi, digital slide images, lecture recordings (audio and video) and notes, exam questions, evaluation forms, bibliographies linked to full-text articles, and other resources made available by the faculty of Tufts University**.**
Artifact	2500 images	Artifact contains over 2500 images and corresponding data, with links to the Art History slide collection database containing 120,000 entries. It integrates on-demand viewing and searching with Internet-based adaptations of traditional learning aids, such as flashcards, for review and study
Crime and Punishment	400 images and videos.	The repository contains images in gif format and videos in mov format to support simulations used in Political Science courses.

Perseus, Artifact and TUSK have an extensive set of tools associated with them that allow users to access content in a manner that is most suited to their discipline-specific needs. The collections are continuously expanding adding content in a variety of formats. The current architecture of these libraries is not built to accommodate such expansion. There is a need for development of a new digital library architecture that is modular, scalable and economically viable. The architecture should allow for persistence of objects across collections and reusability of content by multiple applications.

While having a centralized university-wide repository application is an attractive proposition, the diverse classes of digital objects represented in the various collections pose several challenges. The repository must be able to ingest and manage diverse materials and the ingestion and management processes must be able to scale to handle large volumes of content. In addition to the preservation of the objects, the repository architecture must be flexible enough to provide the appropriate hooks so that we can design services that are capable of delivering the content in a user-friendly manner to different user communities. For example, digital objects from the Perseus Project Classics collection that are stored in the repository needs to be disseminated through complex language tools developed by the Perseus Project that link syntax, grammar, and references to particular people and things across the entire collection [4] On the other hand, digital objects from the TUSK collection require a completely different kind of dissemination - one that resembles a courseware environment. The modular and interoperable nature of the TDL architecture allows us to use tools developed within one application such as Perseus to be used on objects that belong to another application such as TUSK. This makes TDL a powerful architecture that can effectively use tools that have been developed by disparate applications.

3.2 System Specifications

A modular system that meets the necessary functional requirements of a long-term digital repository was considered as the most suitable architecture for TDL. It had to be flexible and extensible enough to meet the diverse storage and access needs of data providers and application builders within the Tufts community and in the potential federated digital library community. Principles of "trusted digital repository" [5] and OAIS guidelines were applied in arrangement of system the architecture. Table 2. details the OAIS requirements along with the matching system service.

Table 2. Requirements and system services.

Requirements	System Services
Unique and persistent identification of materials	Naming Service
Use of Archival Information Packages (AIP)	Digital Object Provider (DOP) Service
Use of Submission Information Packages (SIP)	Drop Box, Ingestion Service
Use of Dissemination Information Packages (DIP)	DOP Service
Authentication and integrity checking	DOP Service
Dissemination	Disseminators, Caching Service, Digital Library Application, Search Service
Access	Search Service and other applications

4. TDL Architecture

An architecture made of loosely coupled modular services emerged as the solution that would be best suited to create a flexible, extensible and scalable digital library that could subsume our current digital libraries while allowing for future yet to be determined applications. This model extends the framework provided by the UVA implementation of Fedora which forms the core of TDL.

An architecture which addressed the issue of scalability by defining a number of logical units and their relation in the context of the digital library was devised. HTTP/HTTPS was chosen as the communication protocol of choice between these units. This choice allows use of wide array of server tools in the implementation of each service with the prospect of using the internet as the transport layer. Scalability was the main motivation for minimizing the lines of dependency between the services in the model.

Figure 1. shows the component services that comprise the TDL architecture.

TDL was explicitly designed to facilitate the business processes that are associated with the creation and use of the library. The design of the component services was done in conjunction with the design of the business processes associated with each service. Each component was designed to effectively support the corresponding business process and interface appropriately with other components.

The architecture is comprised of five basic services.

- **Drop Box and Ingestion Service** provides a conduit for objects to be uploaded into TDL. This does the validation and tagging of the objects as part of the preprocessing and then ingests the objects.

- **Naming Service** creates a unique persistent identifier which is the Universal Resource Name (URN) for the object. The service also resolves URNs.
- **Fedora Repository Service** provides management of and access to named digital objects
- **Indexing and Search Service** indexes the digital objects and provides a search mechanism.
- **Application Creation Service** provides a mechanism for external applications to interface with the repository.

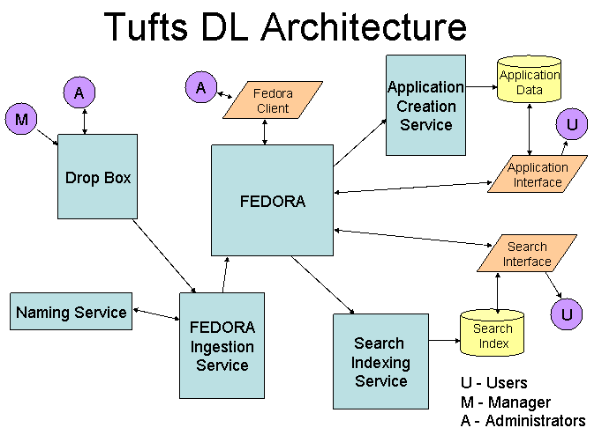


Figure 1. TDL Architecture

4.1 Drop Box and Ingestion Service

The "Drop Box" as the name suggests is a location where users can place digital objects that need to reside in TDL. It provides temporary data storage during the pre-processing phase. The drop box contains a template file provided by the archivists. The template file has basic metadata which is associated with all objects. The drop allows for association of additional metadata with the objects. The drop box also tests for validity of object types.

The Ingestion Service automatically collects the objects from the Drop Box. It validates the Fedora object schema and waits for archivists to perform content quality review before approving or rejecting objects based on archival standards. It calls the Naming Service to obtain an URN for approved objects. It takes the content, binds it with the associated metadata and prepares the METS object, which is then ingested into Fedora. It gets the PID (Persistent Identifier) from Fedora and calls the Naming Service to associate PIDs with URNs. Finally it informs the contributors about the success or failure of the attempt to ingest the object.

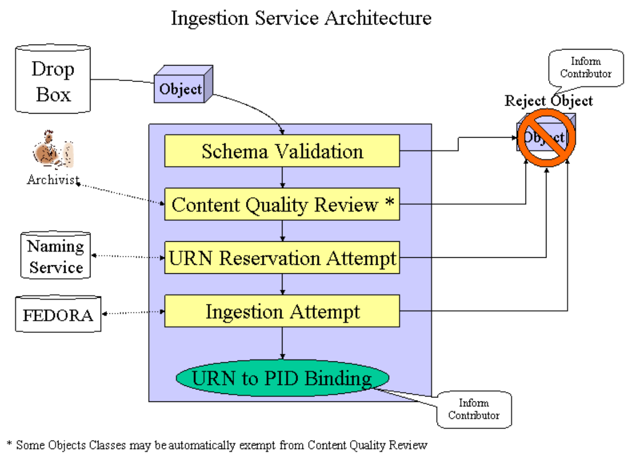


Figure 2. Ingestion Service

4.2 Naming Service

Fedora provides a very limited system for referencing objects. Every object in Fedora is assigned a PID in the format: "string:number". This makes it difficult to track and reference objects uniquely across collections. Furthermore objects may move between Fedora servers creating a need for an identifier that is uniquely associated with the object, independent of the repository in which it resides. The Naming Service creates a URN. It also creates a binding between the URN and the Fedora PID and provides a resolution service to locate the object.

The convention developed for the TDL URN is as follows:

tufts:school name:owner:[collection:]item name.

The first field of the URN created through this service is always *tufts*. The second field is ***school name*** which is unique for any school (or organizational entity) registered through this service. If an object is not associated to any school, it is allocated to the *default **school name***. An optional field – ***collection*** is provided by the projects. Collection helps further classify repositories in a project. The project/repository owners/contributors can provide ***item name*** or it will be created by the service. The URN formed by combining these four fields is guaranteed to be unique.

4.3 Fedora Repository Service

The Fedora Repository Service forms the core of TDL. The key features of the architecture are: (1) support for heterogeneous data types; (2) accommodation of new types as they emerge; (3) aggregation of mixed, possibly distributed, data into complex objects; (4) the ability to specify multiple content disseminations of these objects; and (5) the ability to associate rights management schemes with these disseminations.

The following sections describe TDL's implementation of the repository model, objects, behaviors and disseminators and custom modifications.

4.3.1 Repository Model

TDL's implementation of Fedora is a modification of the implementation developed by the University of Virginia. Modifications were necessary to create a fast and efficient production system. Figure 3. details the different components of the repository model.

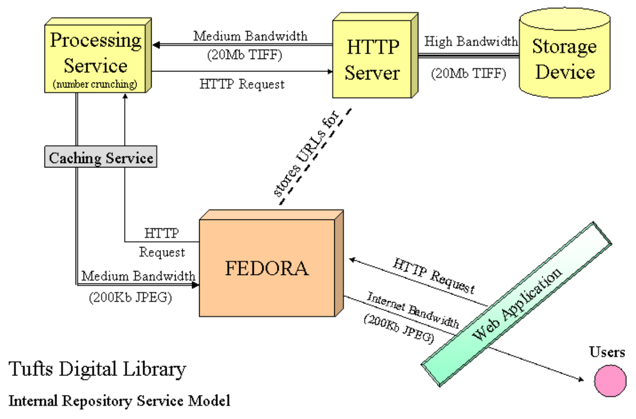


Figure 3. The Repository Model

4.3.2 Objects, Behaviors and Disseminators

Each object in the repository is identified with a particular content-type. Consistent with the Fedora model each content type in the repository has a set of associated behaviors and disseminators. Following is the list of content types that are supported in TDL.

- TUFTS_STD_IMAGE
- XML_TO_HTMLDOC
- TUFTS_BINARY_FILE
- TUFTS_VUE_CONCEPT_MAP
- TUFTS_COLLECTION

All content types contain disseminators supported by Fedora's Behavior Definition (bdef) *fedora-system:3* and *demo:277*, which is a Behavior Definition to support indexing. The *fedora-system:3* bdef supports few basic disseminators like *getObjectProfile, viewObjectProfile, getMethodIndex, viewMethodIndex, getItemIndex, viewItemIndex, getItem, getDublinCore, viewDublinCore* [14]. Additional Behavior Definitions and Disseminators were linked to the content types to make the objects usable by the applications. Tables 3-7 show the association of content types with Fedora behaviors and disseminators that have been developed for TDL.

Dissemination	**Description**
getThumbnail	Returns thumbnail sized image (120 x 120 pixels).
getImage	Returns image in jpeg or gif format.
getStandard	Gets a screen size of the image (650-850 pixel width).
getResized	Returns image with specified width and height.
getZoomedImage	Returns image specified magnification.
getImageTile	Returns a tile of image specified by location x, y and dimensions width and height.

Table 3. Dissemination Index for TUFTS_STD_IMAGE

Dissemination	**Description**
getXML	Returns the content of document in raw XML format.
getTOC	Returns the Table of Content
getInfo	Returns basic information about the document.
getDocument	Returns the document in browse-able format.
getChunk	Returns the specified chapter from the document.

Table 4. Dissemination Index for XML_TO_HTMLDOC

Dissemination	**Description**
getFile	Returns the binary file.

Table 5. Dissemination Index for TUFTS_BINARY_FILE

Dissemination	**Description**
getConceptMap	Returns the concept Map generated and used by VUE.
getManifest	Returns the manifest file describing the content in VUE concept map.
getResource	Returns the specified resource used by VUE concept map.

Table 6. Dissemination Index for TUFTS_VUE_CONCEPT_MAP

Dissemination	Description
getCollType	Given the collection and object type, returns the objects in the collection of specified type.
getCollResources	Given a collection ID get a list of the objects in that collection (i.e. texts, images, audio, video, etc.).

Table 7. Dissemination Index for TUFTS_COLLECTION

4.3.3 Implementation Challenges

Certain enhancements had to be made to the out-of-the box Fedora to increase the speed and efficiency of disseminations.

4.3.3.1 Caching Vs. Preprocessing

The processing of most source files within the digital library is a computationally intensive process. Initial tests indicated that the system would not scale well with all the processing done in real time. A demand-driven caching service and a full preprocessing upon ingestion mechanism were considered as possible solutions.

Full preprocessing requires significant computational power and large quantities of storage. The branching complexity of the parameterized disseminators spanned an infinite tree of possibilities. Though there are no off-the-shelf products that support full preprocessing, preprocessing of small subset of frequently used disseminations was possible. Partial preprocessing was used in conjunction with Squid [25], which is an on demand caching service to increase the speed and efficiency of disseminations.

4.3.3.2 Internal Cache Vs. External Cache

TDL uses an internal caching mechanism, where the data/disseminations are cached within the repository. Unlike an external cache, where data is cached by applications that interface with the repository, internal cache allows full control over the caching process and guarantees that the cache will always reflect the true state of the objects. Internal cache cannot be bypassed by applications using TDL thus blocking malicious code that may potentially overload processing server. In addition the TDL internal caching server preprocesses parts of collections, monitors and logs traffic, isolates some processing services for maintenance without service interruptions, etc.

4.4 Indexing Service and Search Engine

TDL provides an indexing and search mechanism for accessing its content. Users may access the content using the no-frills user interface provided by this service. External applications can interface with the hooks specifically provided by the service for this purpose. The search application uses Lucene 1.3 which is a widely used open-source search engine supported by the Jakarta project.

To provide a way for the search engine to index the objects within TDL, a series of disseminators which expose the textual content of an object in a search engine friendly format were developed. This customization allows use of additional types of search not supported by the basic Fedora search.

These disseminators are polymorphic in nature allowing the search engine to index all digital library objects in a type-neutral manner. Every object supports the *getForIndexing* and *getPreview* disseminators which serve as hooks for the search engine. The indexing process does not differentiate between document-types since all of them provide the same indexing disseminators. All objects subscribe to the same indexing behavior definition. The disseminators are implemented in different ways across the different object types through type-specific behavior mechanisms.

4.4.1 Indexing

The disseminator *getForIndexing* is accessed via HTTP GET or SOAP request. It returns HTML page containing the metadata and content of digital objects. Following example demonstrates the use of disseminator.

Example

Disseminator call:
http://hosea.lib.tufts.edu:8080/Fedora/get/tufts:1/demo:277/getDocumentForIndexing/

HTML page return on above call:

```
<html>
   <head>
      <meta http-equiv="Content-Type"
content="text/html; charset=utf-8">
      <meta name="title" content="John
Holmes, March 13, 1935">
      <title>John Holmes, March 13,
1935</title>
      <meta name="subject"
content="Faculty">
      <meta name="subject" content="Holmes,
John A. (1904-62)">
      <meta name="subject" content="John A.
Holmes, papers">
      <meta name="subject" content="People">
      <meta name="subject"
content="Portraits">
      <meta name="date" content="1935">
      <meta name="type" content="image">
      <meta name="format"
content="image/tiff">
      <meta name="identifier"
content="tufts:1">
      <meta name="relation"
content="collection:TuftsHistory">
      <meta name="relation"
content="text:2000.04.0015">
   </head>
   <body><img
src="../../demo:60/getThumbnail/"></body>
</html>
```

The Dublin Core metadata returned by the disseminator that performs indexing is in the header of HTML document. This enables the indexing feature to be easily used by any search engine that supports metadata search. A successful implementation was carried out using Google's search engine.

This type of indexing enables not only metadata and full-text search but also advanced searches such as the ability to use metadata filters and date based search.

Indexing for all digital objects in the repository can be done every night or by specific request. Indexing can also be performed on collections or smaller sets objects.

4.4.2 *Processing Search Results*

Hits returned on conducting any type of search (metadata, full-text or advanced) have a tagged reference to the *getPreview* disseminator. All the digital objects have the *getPreview disseminator* to display the snapshot. Like all disseminators, *getPreview*, can be accessed by HTTP GET/POST or by using a SOAP request.

4.5 Application Creation Service

An important design requirement for TDL was to allow current digital library applications to easily interface with TDL and provide access to the content in the digital library within their own environments in a seamless fashion. The Application Creation Service was designed to address this need. Current applications like Perseus can interface with this service to allow their tools to disseminate the content that resides in TDL. This mechanism allows application providers to combine the power the repository system with their individual applications. The service has been designed not only to support current application but also to accommodate the needs of future yet-to-be-defined applications like course management systems, learning tools, portals etc.

5. Applications Accessing TDL Content

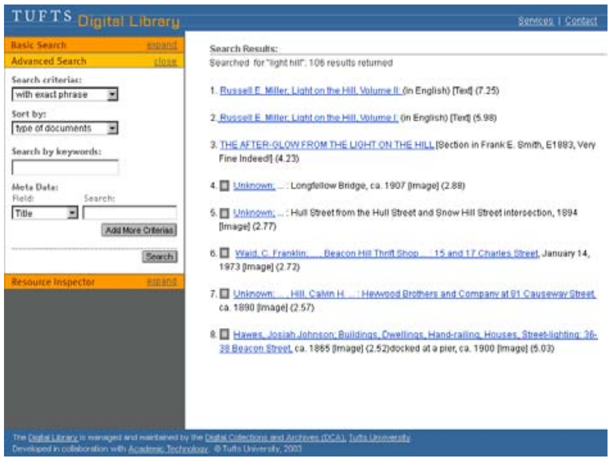
Two applications - Tufts DL Search and Visual Understanding Environment (VUE) have been developed at Tufts that use the content from TDL.

5.1 Tufts DL Search

The requirements and specifications of the search application were initially conceived out of our research and experimental implementations of the search applications developed by the Perseus Project [16, 23]. The primary purpose of this first implementation of the search application is to promote resource discovery within the digital repository for a wide audience, i.e. the user community of the digital library application. Since the Tufts repository will always contain an ever increasing and diverse set of digital objects and object types it was important to develop a system that would take advantage of the rich object metadata that is encoded in the Fedora Repository Service and indexed by the repository's search engine. Every object in the digital repository supports disseminators that serve as hooks for the search engine. Once these disseminators are activated all objects in the digital repository are indexed in a type-neutral manner and essentially create large pool of data against which basic information discovery queries using any search application can be run.

In order to discover and query objects in the digital repository through the Tufts Digital Library generic search application was developed that provides two initial levels of searching capabilities: a "basic search", and an "advanced search." The basic search function provides a means for searching either the full-text or the metadata of digital objects in the repository. The advanced search provides a means for searching both full-text and metadata of digital objects in the repository. The advanced search also provides an interface for field-based searching of metadata, as well as various types of results sorting. Figures 5 and 6 show screen shots of advanced search and the search result page respectively.

The general functions of the Perseus search engines have been be adapted to the new digital repository and there are plans to continue to develop and implement specific aspects of the Perseus Project work such as clustered searching, bi-directional searching, language related search tools, and DTD tag based searching of full-texts. The behavior centric nature of the Fedora repository facilitates the process of building such search applications because Fedora provides the hooks needed to disseminate data from different objects types.

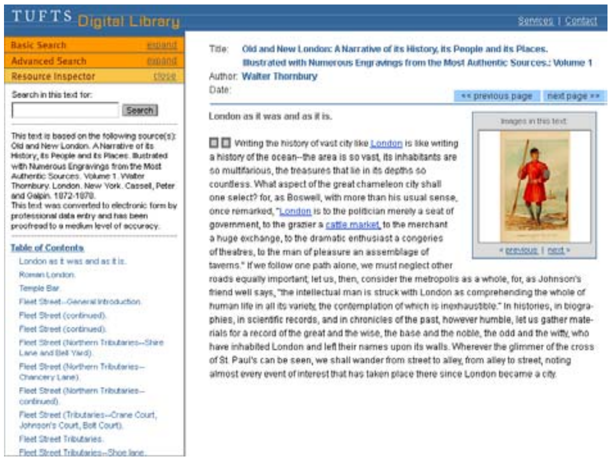


Search Results:
Searched for "light hill": 106 results returned

1. Russell E. Miller, Light on the Hill, Volume II: (in English) [Text] (7.25)
2. Russell E. Miller, Light on the Hill, Volume I: (in English) [Text] (5.98)
3. THE AFTER-GLOW FROM THE LIGHT ON THE HILL [Section in Frank E. Smith, E1883, Very Fine Indeed!] (4.23)
4. Unknown; ... : Longfellow Bridge, ca. 1907 [Image] (2.88)
5. Unknown; ... : Hull Street from the Hull Street and Snow Hill Street intersection, 1894 [Image] (2.77)
6. Waid, C. Franklin; ... Beacon Hill Thrift Shop ... : 15 and 17 Charles Street, January 14, 1973 [Image] (2.72)
7. Unknown; ... Hill, Calvin H. ... : Heywood Brothers and Company at 91 Causeway Street, ca. 1890 [Image] (2.57)
8. Hawes, Josiah Johnson; Buildings, Dwellings, Hand-railing, Houses, Street-lighting; 36-38 Beacon Street, ca. 1865 [Image] (2.52)docked at a pier, ca. 1900 [Image] (5.03)

The Digital Library is managed and maintained by the Digital Collections and Archives (DCA), Tufts University
Developed in collaboration with Academic Technology. © Tufts University, 2003

Figure 5. Screenshot of Tufts DL Search



This text is based on the following source(s): Old and New London. A Narrative of its History, its People and its Places. Illustrated with Numerous Engravings from the Most Authentic Sources. Volume 1. Walter Thornbury. London. New York. Cassell, Peter and Galpin. 1872-1878.
This text was converted to electronic form by professional data entry and has been proofread to a medium level of accuracy.

Table of Contents
- London as it was and as it is.
- Roman London.
- Temple Bar.
- Fleet Street--General Introduction.
- Fleet Street (continued).
- Fleet Street (continued).
- Fleet Street (Northern Tributaries--Shire Lane and Bell Yard).
- Fleet Street (Northern Tributaries--Chancery Lane).
- Fleet Street (Northern Tributaries--continued).
- Fleet Street (Tributaries--Crane Court, Johnson's Court, Bolt Court).
- Fleet Street Tributaries.
- Fleet Street Tributaries--Shoe lane.

Title: Old and New London: A Narrative of its History, its People and its Places. Illustrated with Numerous Engravings from the Most Authentic Sources.: Volume 1
Author: Walter Thornbury
Date:

London as it was and as it is.

Writing the history of vast city like London is like writing a history of the ocean--the area is so vast, its inhabitants are so multifarious, the treasures that lie in its depths so countless. What aspect of the great chameleon city shall one select? for, as Boswell, with more than his usual sense, once remarked, "London is to the politician merely a seat of government, to the grazier a cattle market, to the merchant a huge exchange, to the dramatic enthusiast a congeries of theatres, to the man of pleasure an assemblage of taverns." If we follow one path alone, we must neglect other roads equally important; let us, then, consider the metropolis as a whole, for, as Johnson's friend well says, "the intellectual man is struck with London as comprehending the whole of human life in all its variety, the contemplation of which is inexhaustible." In histories, in biographies, in scientific records, and in chronicles of the past, however humble, let us gather materials for a record of the great and the wise, the base and the noble, the odd and the witty, who have inhabited London and left their names upon its walls. Wherever the glimmer of the cross of St. Paul's can be seen, we shall wander from street to alley, from alley to street, noting almost every event of interest that has taken place there since London became a city.

Figure 6. Screenshot of Tufts DL Search displaying preview of object.

5.2 VUE

5.2.1 Concept Map to Content Map

Digital Libraries have enabled academic communities to access large amounts of digital information which may be used as part of course materials. As the quantity of digital information continues to grow, there is a need for flexible tools to help people organize, make sense of and apply digital information to teaching, learning and problem solving. Concept maps have had a long and fruitful history in education as they provide visual representations of complex ideas or process, their contributing processes and the ways in which these elements are connected or related. Visual Understanding Environment (VUE) [4] extends the functionality of traditional concept mapping applications by allowing users to map against and draw from persistent collections of content contained within digital libraries. This integrated digital library application transforms ordinary concept maps into content maps. In support of teaching, it provides faculty with a visual means to semantically structure and create pathways through digital materials in a manner most consistent with the ideas and concepts they must communicate to their students. Figure 7 presents a screenshot of VUE.

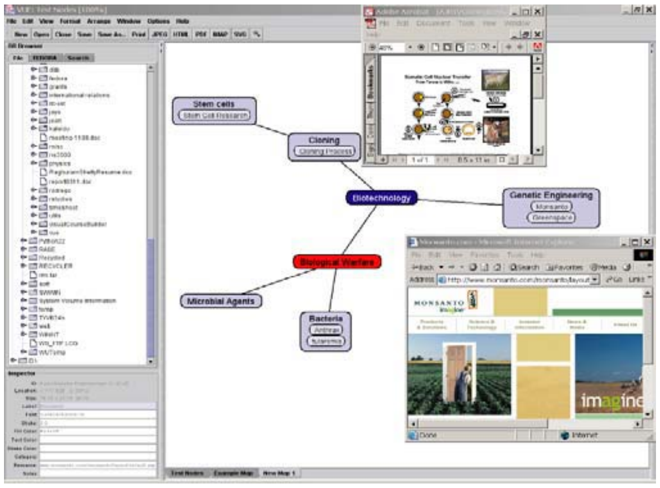


Figure 7. Screenshot of VUE

5.2.2 VUE : connecting to TDL

The technical design of VUE incorporates all relevant application programming interfaces (API) developed at MIT as part of the Open Knowledge Initiative (OKI) to maximize the utility of this tool within environments beyond Tufts. The VUE architecture is shown in the figure 8. The repository management subsystem is at the heart of VUE, as it provides interaction with persistent collections of digital materials, local and remote file systems. VUE implements the File and Content Management APIs to support communication between the VUE, file systems and digital repositories.

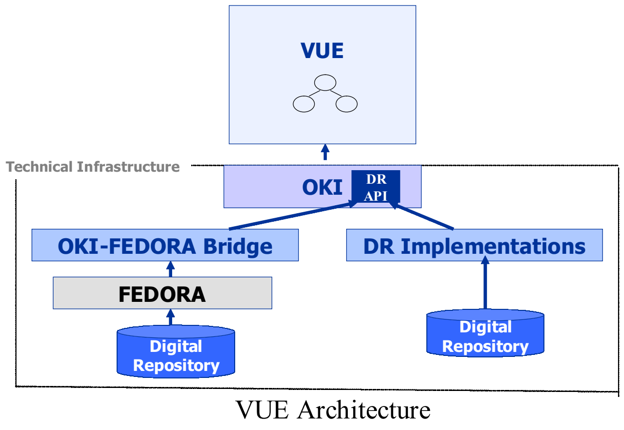


Figure 8. VUE Architecture

Users of VUE can add and modify content of TDL based on the privileges they have on resources. Authentication and Authorization is performed through Tufts LDAP by an implementation of OKI's Authentication and Authorization APIs. Authorization part will be moved to repository level when Fedora implements it in their next release.

Communication with the repository happens through Fedora- OKI Bridge. The object model of resources in VUE which is an implementation of OKI's dr.Asset interface closely maps to the object model in TDL.

VUE is developed in Java and the communication happen through SOAP calls that are supported by TDL. Thus the modular approach and support for SOAP and HTTP GET/POST bindings by TDL makes the integration very simple.

6. Conclusion and Future Work

Tufts University's quest to architecting an extensible digital library that can support current digital library applications and future needs resulted in a design that uses the Fedora architecture at its core. While Fedora is an excellent digital repository architecture, it needs customization and enhancements to produce a complete digital library solution that addresses a wide variety of needs.

A range of services developed and implemented as part of the Tufts Digital Library provide the necessary infrastructure to manage, disseminate and create persistent collections of a large number of digital objects. A customized caching implementation was used to enhance the basic Fedora repository service that helped augment the speed and efficiency of dissemination.

TDL is designed to interface with applications that need to use the content of the repository. VUE and TDL Search are examples of applications that have been able to successfully use the content of the repository.

In the future we hope to use TDL as the repository to host objects from Perseus, TUSK and Artifact and to allow for seamless integration with the existing applications. Other services such as authentication and authorization which are proposed in future releases of Fedora will be integrated into TDL as an when they become available.

7. References

[1] Thornton Staples and Ross Wayland: Virginia Dons Fedora: A Prototype for Digital Repository. *D-Lib Magazine 6,* 7/8 (July/August 2000)

[2] Sandra Payette and Carl Lagoze: Flexible and Extensible Digital Object Repository Architecture, in Christos Nikolau and Constantine Stephanidis, eds., *Research and Advanced Technologies for Digital Libraries: Proceedings of Second European Conference, ECDL '98, Heraklion, Crete,Greece, September 21-23, 1998, G. Goos, J. Hartimis and J. vanLeeuwen, eds., Lecture Notes in ComputerScience, 1513 (Berlin: Springer, 1998)* http://www.cs.cornell.edu/payette/papers/ecdl98/Fedora.html

[3] Carl Lagoze: The Warwick Framwork: A container Architecture for Diverse sets of Medata, *D-Lib Magazine* 2(7/8), 1996.

[4] David Kahle, Anoop Kumar, Ranjani Saigal: Visual Understanding Environment, *Syllabus Fall Conference 2003.*

[5] Trusted Digital Libraries: Attributes and Responsibilities, RLG, May 2002.

[6] *Reference Model for an Open Archival Information System (OAIS)*. Blue Book. Issue 1. January 2002

[7] Open Knowledge Initiative. http://web.mit.edu/oki/

[8] URN Syntax. http://web.mit.edu/rfc/rfc2141.txt

[9] Functional Requirements for Uniform Resource Names. http://web.mit.edu/rfc/rfc1737.txt

[10] A trivial Convention for using HTTP in URN resolution. http://web.mit.edu/rfc/rfc2169.txt

[11] PURLs. http://www.purl.org/

[12] Harvard University Library LDI. http://hul.harvard.edu/ldi/index.html

[13] Making of America II. http://sunsite.berkeley.edu/moa2/

[14] Fedora Project. http://www.Fedora.info

[15] Gregory Crane, Clifford E. Wulfman, Lisa M. Cerrato, Anne Mahoney, Thomas L. Milbank, David Mimno, Jeffrey A. Rydberg-Cox, David A. Smith, and Christopher York. Towards a cultural heritage digital library. In *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries, JCDL 2003*, pages 75-86, Houston, TX, June 2003

[16] Gregory Crane, David A. Smith, and Clifford E. Wulfman. Building a hypertextual digital library in the humanities: A case study on London. In *Proceedings of the First ACM+IEEE Joint Conference on Digital Libraries*, pages 426-434, Roanoke, VA, 24-28 June 2001.

[17] Bruce A. Metz, Mary Y. Lee, Susan Albright, and Tarik Alkasab, "Transforming Medical and Health Science Education at Tufts University", *EDUCAUSE Quarterly* (Volume 24, Number 4)

[18] Artifact Project, Tufts University http://artifact.tufts.edu

[19] Crime and Punishment. http://at.tccs.tufts.edu/apps/candp/

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

[21] Dublin Core. http://www.dublincore.org/

[22] Lucene. http://jakarta.apache.org/lucene/docs/index.html

[23] Anne Mahoney. Explicit and implicit searching in the Perseus digital library. In Einat Amitay, editor, *Information Doors: Pre-Conference Workshop at the Eleventh ACM Conference on Hypertext and Hypermedia*, 2000.

[24] Bolles Collection http://yngve.lib.tufts.edu/4000.01/

[25] Squid. http://www.squid-cache.org/