ALEXANDRIA DIGITAL LIBRARY IN A BOX
USER GUIDE

Alexandria Digital Library Project
http://www.alexandria.ucsb.edu/

Online User Guide
http://www.alexandria.ucsb.edu/doc/ADLUG/Output/index.htm

Alexandria Digital Library in a Box User Guide

1 INTRODUCTION

Welcome to Alexandria Digital Library in a Box (ADLIB). This software lets you publish geospatial digital content for structured search and retrieval over the Web.

Alexandria Digital Library collections can operate in standalone mode for use by individual users, or optionally switch into a distributed mode for web-based information sharing and publication. Distributed means the library's components may be spread across the Internet, as well as coexisting on a single desktop.

This chapter covers:

- What is a Digital Library?
- Why Should I Use ADL Software?
- What is ADL in a Box?

WHAT IS A DIGITAL LIBRARY?

A digital library is a focused collection of digital objects, including text, video, and audio, along with methods for searching, access and retrieval, and for selection, organization, and maintenance[1].

Many digital objects can have a geospatial reference. Geospatial collections are typically heterogeneous in content and can include items as diverse as maps, historical photographs, field data, remotely sensed images or archeological data.

Geospatial & Georeferenced Information

While an Alexandria Digital Library node lets you search using standard library semantic concepts, such as author and keyword, it focuses on *geospatial* and *georeferenced* information.

Geospatial information is directly referenced by longitude and latitude coordinates that locate its footprints on the face of the Earth, such as maps, aerial photographs, and remote-sensing imagery.

Georeferenced information references a geographic location without explicit geospatial representation. Any kind of document that is about a particular geographic place, (*A Tale of two Cities* or *History of the Decline and Fall of the Roman Empire)* is potentially an example of georeferenced information.

1.*How to Build a Digital Library* by Ian Witten and David Bainbridge.

WHY SHOULD I USE ADL SOFTWARE?

ADL has features that make it unique:

- The ability to search for fundamentally different document types in the same way. For example, a search of a specified location will return text documents, maps, or multimedia about the area.
- The ability to search using standard library terms, such as author, title, and keyword.
- A simple query language.
- Support for distributed searches across heterogeneous collections and instances of ADL.

  Distributed searches means you can search different databases at different locations and of different types. ADL has the potential to develop into a worldwide network of databases with geographically referenced information.
- A gazetteer service for translation between place names and geographic coordinates. The large international gazetteer database allows for a less coordinate-dependent client. It also has a standard for the content of a gazetteer and an XML-service to allow you to add new elements to the gazetteer.
- A thesaurus is used to solve semantic ambiguities, such as many words for the same phenomenon.
- The ability to use existing metadata. ADL defines a general architecture that makes it possible to use with totally different types of databases and representation of data. It can be used with a wide variety of databases, often without changing the content or information within of the database.

WHAT IS ADL IN A BOX?

ADL in a Box is a single application package that includes the software needed to install and run the library with any additional software. It uses Open Source software and open protocol standards, such as XML, JDBC, and SQL. The ADL software is freely available and can be installed on all common software and hardware platforms that have a Java virtual machine.

2 QUICK START

Alexandria Digital Library in a Box consists of the following components:

- ADL middleware configured to connect to UCSB's Map and Imagery Lab-hosted collections
- web-based user interface for searching
- Apache Tomcat server
- Apache Ant tool

Overview of setting up a Digital Library

This section provides an overview of the steps required to set up an ADL node. Detailed installation instructions are provided in Chapter 3, "Installing ADL Software." For detailed instructions on adding your collection to ADL, see Chapter 4, "Adding Collections."

1. Install Java 2 SDK, see http://java.sun.com/j2se/1.4.1/download.html.
2. Download ADLIB software at http://www.alexandria.ucsb.edu/downloads/adllibrary/.
3. Install ADLIB software. See Chapter 3, "Installing ADL Software."
4. Determine what collection-level and item-level metadata will be used. See "Introduction to ADL Metadata" on page 7.
5. Map item-level metadata to buckets in query-translator.py. See "Adding your collection to ADL" on page 9.
6. Set up configuration parameters in bucket99.conf, for example database connection string. See "Adding your collection to ADL" on page 9.
7. Define queries for metadata reports in bucket99.conf (scan, browse, full, access). See "Adding your collection to ADL" on page 9.
8. Format metadata reports by editing report templates (access.xml, full.xml, etc). See "Adding your collection to ADL" on page 9.
9. Test the metadata reports using test-form.html.
   Go to HTTP://[your_host_name]:[port]/adl_library/test-form.html.
   For example:

   ```
   HTTP://myhost.myschool.edu:8080/adl_library/test-form.html
   ```

10. Test webclient and if problems are encountered, enable query log.

3 INSTALLING ADL SOFTWARE

This chapter describes how to install the ADL Standalone software on Windows and UNIX operating systems.

To Install ADL Standalone:

1. Determine the fully qualified domain name of the computer on which you are installing the software. For example:

   ```
   rosarita.alexandria.ucsb.edu
   ```

2. Determine available port numbers for Tomcat operation on your system.

   Two ports are required for Tomcat and Tomcat administration. If Tomcat is already running on your system, you need to ask your System Administrator for available port numbers. If it's not running, you can use the default settings.

3. Install Java 2 SDK, see http://java.sun.com/j2se/1.4.1/download.html.

4. Set the JAVA_HOME environment variable to the directory in which Java is installed.

5. Navigate to http://www.alexandria.ucsb.edu/downloads/adllibrary/ and download:

   ADLStandalone-*yyyymmdd*.tar.gz (*Unix*)

   or

   ADLStandalone-*yyyymmdd*.zip (*Windows*)

6. Untar the file to your home directory (*Unix*) by entering:

   ```
   gunzip
   tar -xvf
   ```

   or

   Unzip the file to the root directory *(Windows)*.

7. It expands to a directory called ADLStandalone. Change your current directory to ADLStandalone.

8. If a version of Tomcat is already running, you need to edit the file library.properties to replace the default port numbers of `catalina.port=8080` and `catalina.admin.port=8005` with the available port numbers from step 2.

   If no other Tomcat is running, skip to step 9.

9 In a command window, enter:

```
adl.sh (Unix)
```

```
adl.bat (Windows)
```

10 In the command window, you will be prompted with the following:

```
FirstRun:
createProperties:
    [echo] No library.properties. This task will help you create one
    [input] server name
```

11 Enter the fully qualified domain name that you determined in step 1. For example:

```
rosarita.alexandria.ucsb.edu
```

12 When the "Build Successful" message is displayed, you can start the ADL server. At the prompt, enter:

```
adl.sh start (Unix)
```

```
adl.bat start (Windows)
```

13 To view the web interface, open a web browser and in the address line, enter:

```
http://[yourHostName]:8080/
```

Where `[yourHostName]` is the fully qualified domain name.

4 ADDING COLLECTIONS

Once you install ADL Standalone, you can add your collections to ADL. This chapter describes how to map your metadata to ADL search buckets and add your collections to ADL. ADL search buckets defines a particular set of ten "buckets" that capture key searchable and descriptive characteristics of widely heterogeneous items. For information on ADL search bucket framework, see Appendix B, "The ADL Bucket Framework."

This chapter covers:

- Introduction to ADL Metadata
- Adding your collection to ADL

INTRODUCTION TO ADL METADATA

Understanding ADL Metadata will help you get started adding your collection to ADL. A collection in ADL consists of a geographically referenced set of items (maps, aerial photographs, satellite images, recordings, etc.), which has:

- **Collection-level metadata** - a standardized description about the entire collection. In the traditional library world, this would be analogous to a description of a specialized library branch, like an Engineering Library.
- **Item-level metadata** - information about each items in the collection, including a unique identifier for each item within the collection. This is analogous to the information on a catalog card in a traditional library card catalog.

Collection-level metadata

Collection-level metadata is an integral part of the ADL philosophy. It helps users determine if the contents of a collection are applicable to a project by providing simple summary information such as title, scope, purpose, and responsible party, as well as aggregated information about the collection such as item counts by format and object type, and histograms describing the spatial and temporal coverage.

The collection developer must always supply the simple summary information; however, in the near future the ADL software will generate the aggregated information automatically. The summary information can be collected or created even before you are ready to load your collection into ADL. Review the Collection-Level Metadata example (http://www.alexandria.ucsb.edu/adl/docs_public/

ADLMetadata_ingest/ADL_CLM_detail.php?collection=drg_24k_ca) to see the required and optional information you need to provide in your collection-level metadata.

ADL collection-level metadata is an XML document that has been rendered into HTML for readability. The ADL-collection-metadata.dtd (http://www.alexandria.ucsb.edu/docs/metadata/ADL-collection-metadata.dtd) (XML Document Type Definition) formally defines the semantics of the collection-level metadata. You may view an example of collection-level metadata rendered in HTML as it appears in the ADL webclient (http://collections.alexandria.ucsb.edu/adl_catalog/metadata.html).

The collection-level metadata XML files need to be in a web-accessible directory (where your local collection data is or will be located). For further detailed information on installing metadata files in ADL, please see: "Adding your collection to ADL" on page 9.

Item-level metadata

Although ADL has specified a content standard and XML encoding for collection-level metadata, collection developers are generally free to use any item-level metadata. The only requirements are:

- Item-level metadata must be mapped to the ADL metadata views. You can view an example of item-level metadata at the bottom of the Collection-Level Metadata example (http://www.alexandria.ucsb.edu/adl/docs_public/ADLMetadata_ingest/ADL_CLM_detail.php?collection=drg_24k_ca).
- Each item must have an identifier that is unique within the collection.

Tips for creating unique identifiers

- Avoid spaces in the identifier.
- If using the filename as an identifier, use the base portion only. Do not use the filename with the extension or type attached. For example, if a SPOT satellite image file is named "santa_barbara.tif," a good identifier would be "santa_barbara" (assuming the name is unique in the collection). This is because ancillary information is often associated with the base filename.

For example, the "santa_barbara.tif" might have additional files containing a georeferencing information in a separate file (often "santa_barbara.tfw"), a header file, "santa_barbara.hdr," or even a metadata file "santa_barbara.meta." Using "santa_barbara.jpg" makes it difficult to associate the information without first removing the extension from the filename.

While removing extensions is simple for a homogeneous collection, the variety of extensions present in heterogeneous collections makes this task more difficult. For holders of a single heterogeneous collection, it may seem trivial to handle extension stripping and mapping. However, for library developers, who handle multiple configurations, extension stripping can become complex, since each collection developer might have a different procedure, standard, and set of file types.

ADDING YOUR COLLECTION TO ADL

To add collections to ADL:

1 Create collection-level metadata by entering the required fields in the Collection-Level Metadata form. When the form is completed, click **Insert Record**. (See http://alexandria.ucsb.edu/adl/docs_public/ADLMetadata_ingest/ADL_CLM_insert.php).

The ADL Metadata Listing is displayed.

2 In the ADL Metadata Listing, locate the name of the collection you just inserted and click on it.

The collection-level metadata you entered in step 1 is displayed in the form.

Important!

If your item-level metadata follows the guidelines of the Appendix A, "ADL Buckets - General Schema," you don't need to map your item-level metadata and can skip to step 9.

3 At the top of the form, click **Add/Edit this Item Level Metadata Information**.

The Item-Level Metadata Insert form is displayed with the name of your collection.

4 Create or map your item-level metadata by following the instructions in the form and the following guidelines.

Guidelines for creating and mapping item-level metadata

In order to map your item-level metadata to the ADL search buckets, you must compare your schema to the ADL Item-level Metadata schema, see Appendix A, "ADL Buckets - General Schema." As you compare your schema to the ADL schema, note the names of each of your fields. Please note that the first five fields are required.

- Does each item have a unique identifier with no spaces? See "Tips for creating unique identifiers" on page 8.This is a required field.
- Does each item have a Title? Which fields are used to create the Title? This is a required field.
- Does each item have bounding box coordinates? This is a required field.
- If not, you must create decimal coordinates.
- Does each item have a Date or Date Range?
- This consists of the start date and end date in the format of yyyymmdd. The start and end date can be the same.
- Does each item have Assigned Terms? This is from an authority list, such as the Library of Congress Subject Headings?
- Does each item have any subject-related text? This is used for free-text searching.

- Does each item have a Type, such as map, aerial photo? This is a required field.
- Does each item have a Format listed, such as tiff, jpeg, gif? This is a required field.
- Does each item have an Originator? You can have multiple originators.

5 When the form is complete, click **Send Information**.

The ADL Metadata Listing is displayed.

6 Click on your collection name.

The Item-Level Metadata Insert form with your collection-level and item-level metadata is displayed.

7 Click **View Collection Level Metadata XML**. Copy the URL of this XML file. You will need it to complete the bucket99.config file.

8 Click your browser Back button to return to the ADL Collection Level Metadata form for your collection. The information on this page will be used to create the configuration files in step 10. You may want to print this page or bookmark it.

9 Locate your data objects, thumbs, and browse images.

Your data objects—the actual images that you will be serving out through ADL, need to be located on a web server. You will also need a small thumb version (no larger than 128 pixels on a side) and an easy-to-view "browse" version (no larger than 512 pixels on a side).

We suggest you locate your data objects, thumbs and browse images in a directory called `../collections/your-collection-name/distribution_forms/data (thumb, browse)`. But if your data, browse, and thumbs already exist in their native location, you may prefer to leave them where they are. In either case, you will point to their location via the access.xml and browse.xml reports, which are created in step 10.

10 Create the collection configuration files.

Note

We are working on automation of this process. For now, contact Rudolf rnottrott@alexandria.ucsb.edu for assistance in creating these files.

- bucket99.conf (configuration file)
- query-translator.py (mapping search bucket field names)
- scan.xml (fields required to list result set)
- access.xml (access report, link to data)
- browse.xml (link to thumb and browse images)
- full.xml (full report, complete item level metadata record)
- user.properties (database login id and password)

11 Copy configuration files created in step 10 above into ADLStandalone/collections/[your-collection-name].

[your-collection-name] is the name you entered in the Collection- Level Metadata form.

12 Go to http://[your_host_name]:8080/adl_library/admin/configuration.jsp.

You should see your collection in the list of Running Collections.

13 Click on the HTML link for your collection. This creates the metadata.html file. Copy the URL to this metadata.html file into C:\ADLStandalone\hierarchies\collection_opml.xml (per the example in the comments at the top).

14 Test the metadata reports using test-form.html.
Go to HTTP://[your_host_name]:[port]/adl_library/test-form.html.
For example:

```
HTTP://myhost.myschool.edu:8080/adl_library/test-form.html
```

If problems are encountered, the error messages should indicate which configuration file is causing the error.

15 To view the web interface, open a web browser and in the address line, enter:

```
http://[yourHostName]:8080/
```

Where `[yourHostName]` is the fully qualified domain name.

16 Click **Geospatial Search**. Test the web interface by submitting searches. If problems are encountered, enable the query log, which should indicate which configuration file is causing the error.

5 METADATA ADMINISTRATION

VIEWING COLLECTIONS

To view metadata for existing collections, go to the ADL Metadata Listing page at http://alexandria.ucsb.edu/adl/docs_public/ADLMetadata_ingest/ADL_CLM_master.php.

ADDING COLLECTIONS

To add or create metadata for new collections, go to the Collection-level Metadata form at http://alexandria.ucsb.edu/adl/docs_public/ADLMetadata_ingest/ADL_CLM_insert.php.

6 GLOSSARY

AVHRR	Advanced Very High Resolution Radiometer: an image sensor on an earth satellite.
Bounding box	A bounding box is a regular polygon that is parallel to the equator. It is used to represent, in a general way, the location of a geographic area. A bounding box is represented by two latitude and two longitude values.
Channels	Multiple images captured by imaging sensors of the same location, at the same time, but at different wavelengths of light.
Collection-level metadata	A standardized description about a collection.
Contains	*(A matching operation for spatial areas.)* When you set the search option for the query area to “contains,” spatial matches are limited to those collection objects whose spatial footprints are totally contained within the query area. See also Overlaps.
Datasets	Groups of data objects that may be described as a single unit. for example, the Encyclopedia Britannica.
Footprint	Also called Location. Each data object in the ADL collections has a representation of the area(s) it is about. This "location" is represented by latitude and longitude coordinates in the following order: north latitude west longitude sourth latitude east longitude. Currently, the footprints either points or Bounding boxes.
Gazetteer	Place name index showing an earth feature such as a city along with its geographic location.
Georectified	Spatial data that has been computer processed so as to match Earth locations exactly.
Image Map	A map that has its features presented photographically rather than hand or machine drawn.
Item-level metadata	Information about items in a collection, including an identifier that is unique within the collection.
Location	See Footprint.

Metadata	Information about a data object; a bibliographic catalog record describing the data unit.
Overlaps	*(A matching operation for spatial areas.)* When you set the search option for the query area () to "overlaps," spatial matches are made to any collection object whose spatial footprint overlaps the query area; that is, where any of the objects in the collection(s) have footprints that occupy the same area as the query area. See also Contains.
Query area	Area of interest drawn on the map.
Spatial matching	When you specify a query area, the latitude/longitude coordinates of this area are stored as part of the query. Each data object in the ADL collections is described by a spatial footprint that represents the geographic area that the data object is about. When the query is sent to the database(s), the query area is compared to the data objects' footprints, according to the functioning matching operator (Contains or Overlaps), and those data objects that match are potentially members of the result set for the query.
Spatial relevance	This is a measure of the comparative size and overlap of two spatial areas—the query area and the spatial footprint of the collection object. The higher the value, the more similar the areas are in terms of size and overlap. This value can be used to sort the result sets according to their spatial relevance to the query area.
Thesaurus	A classified list of terms in a particular field for use in indexing and information retrieval. For writers, a thesaurus is a tool like Roget's Thesaurus with words grouped and classified to help select the best word to convey a specific nuance of meaning. For the information storage and retrieval functions of digital libraries, a thesaurus is a set of controlled vocabulary (preferred terms) for representing concepts (topics, subjects) that includes hierarchical, equivalent, and associated relationships among the terms.
Tiles	Data that has been cut into "pages" so they can be viewed one at a time or stitched together.
Topographic maps	Maps that show contour lines of equal elevation in addition to other Earth features.

Appendix A
ADL Buckets - General Schema

```
CREATE TABLE ADL_fields_buckets (
collection                      varchar(25)     NOT NULL,
unique_identifier               varchar(30)     NOT NULL,
title                           varchar(100)    NOT NULL,
last_updated                    timestamp(14)   NOT NULL,
w_b_coord                       float(9,3)      NOT NULL,
e_b_coord                       float(9,3)      NOT NULL,
n_b_coord                       float(9,3)      NOT NULL,
s_b_coord                       float(9,3)      NOT NULL,
other_coords                    varchar(255)
subject_related_text            varchar(255)
assigned_terms                  char(1)
assigned_terms_authority_list   varchar(50)
originator                      varchar(255)    NOT NULL,
type                            varchar(25)     NOT NULL,
frmt                            varchar(25)     NOT NULL,
beginning_date                  date            NOT NULL,
ending_date                     date            NOT NULL,
scale                           int(11)
mapname_quadname                varchar(25)
other_info                      varchar(255) )
TYPE=MyISAM
```

Appendix B
THE ADL BUCKET FRAMEWORK

Greg Janée, James Frew, Linda L. Hill, Terence R. Smith
{gjanee, frew, lhill, smithtr}@alexandria.ucsb.edu

The Alexandria Digital Library Project, as part of its Digital Earth Prototype (ADEPT) effort, is developing a distributed digital library for georeferenced information. Key features of the library include a distributed, peer-to-peer architecture; services supporting federation and interoperation of collections and items; personalized "learning spaces" in which collections of pedagogical materials can be built, shared, organized, and incorporated into curricula; and explicit support for both very large and very small (i.e., personal, local) geospatial collections.

A key component of this architecture is a framework—the ADL "bucket" framework—for creating homogeneous views of heterogeneous metadata. The ADL bucket framework supports aggregation of semantically similar, strongly typed metadata fields into high-level "buckets" for the specific purposes of providing higher-level search and description functions. Notably, in mapping metadata fields to buckets, the mappings themselves are formally and explicitly represented by the system, and metadata semantics are preserved and carried throughout. The framework defines standard representations and bucket types, including sophisticated types such as the "geospatial" bucket type which supports description of geospatial regions and searching using spatial operations. The framework also defines a particular set of 10 buckets that have proven successful in capturing key searchable and descriptive characteristics of widely heterogeneous items, while having sufficient global applicability as to be easily populated. For example, the "Originator" bucket is defined to contain textual values (specifically, names) related to the origin of an item. One item may map the tuple (MARC 100 [Personal name], "Mark Twain") to the Originator bucket, while another may map the tuple (FGDC 1.1/8.1 [Citation/Originator], "U.S. Geological Survey") to the same bucket. In general, an item may map any number of typed (field, value) pairs to a bucket. The ultimate effect of such mappings is that clients of the library can choose to 1) operate at the high level of buckets (e.g., a client can simultaneously search multiple collections by Originator), or 2) operate at the native metadata level (e.g., search by and retrieve MARC 100 fields in a collection that supports such native metadata).

Several other approaches to metadata interoperability have been pursued in the past, and it is useful to compare the bucket framework to them. There have been comprehensive metadata standards that attempt to capture most of the nuances and breadth of a domain [MARC, FGDC]. There have been minimalist, high-level metadata standards [Dublin Core]. Other approaches have focused on automated translation of metadata [Stanford Infobus]. And still other approaches emphasize explicit representation of, and operation on, metadata semantics [RDF, Semantic Web].

The ADL bucket framework combines many aspects of these different approaches in a new and novel way. Like the Semantic Web, the bucket framework maintains representations of metadata semantics. But unlike the Semantic Web, the bucket framework provides a mechanism for aggregating metadata into higher-level, uniform abstractions. The bucket framework is notably similar to unqualified Dublin Core in that it provides a standard set of high-level fields. But it differs just as notably: to Dublin Core the bucket framework adds search-oriented fields, strong typing, and well-defined, rich search semantics. The bucket framework provides a refinement mechanism similar to qualified Dublin Core, but unlike the latter, the refined fields are not mandated by the system, but rather are simply discoverable in a standard way. Finally, the bucket framework is similar to automated metadata translation systems in that metadata is mapped from one form to another, but the bucket framework differs from the Stanford Infobus approach in that it is focused on the simpler problem of mapping arbitrary metadata to a common, higher-level form, as opposed to mapping arbitrary forms of metadata to one another.

The ADEPT logical view of the library is that of 1) many, distributed, collections of heterogeneous items, and 2) a central collection discovery service that clients can use to locate relevant collections. ADEPT employs a standard three-tier architecture in which clients connect to collections through a middleware server, which acts as a common access point and as a kind of broker. The middleware provides SDLIP-like search and retrieval services as well as services related to collection development. The ADL bucket framework is a common thread running throughout the tiers that provides a uniform view of the library. Specifically, at the lowest level individual items within collections map their native metadata to buckets. Collections accumulate and index the mapped metadata, thereby providing a search capability over their contents at both the bucket level and at the native metadata level. Collections also aggregate information and statistics about the metadata mappings employed by the items contained in the collection, and include such information in collection-level metadata for the benefit of clients and the collection discovery service. The middleware provides standard representations of arbitrary metadata and metadata mappings. The central collection discovery harvests collection-level metadata and builds a master index.

The ultimate effect of the ADL bucket framework is that clients of the library can, without relying on any *a priori* knowledge or out-of-band agreements, successfully search arbitrary collections at a uniform, high level. Furthermore, the types of search provided are far more capable than just text-based search. At the same time, the richness and semantics of the underlying native metadata are entirely preserved, and are discoverable and exploitable by clients in an entirely regular way.

INDEX