DRAFT

California Digital Library
Technical Architecture and Standards

March 10, 2000

Reviewed and updated annually

Table of Contents

Table of contents

1. About This Document

1.1 Charge

The California Digital Library is responsible for developing and providing access to the shared collections of the University of California and the application of appropriate digital technologies to influence and support innovations in scholarly communication. The digital technologies employed will include a mix of distributed and centralized systems which must follow architectural guidelines and standards to ensure interoperability between systems, consistent levels of system performance and coherent searching and access methods.

The California Digital Library Technology Architecture and Standards Workgroup is responsible for recommending to the University Librarians and other appropriate University bodies for review, architectural guidelines and standards for the University of California shared Digital Image Collections. These guidelines should be consistent with industry and other University standards, provide a framework that facilitates the creation of integrated systems and provide the flexibility to foster innovations in scholarly communication.

Technology architecture and standards must evolve as both needs and technology evolve. This document will be reviewed at least annually, and will be revised as needed to reflect the evolution of the CDL systems, new national and international standards, developments and best practices in other digital library initiatives, and new capabilities provided by technology innovation.

1.2 Guiding Principles

The following principles guide the development of this architecture and related standards:

- **Service Driven** - The architecture for the CDL must be driven by the services it provides and the tools required for delivering the service.
- **Open Architecture** - The architecture must be open, extensible and support interoperability among heterogeneous, distributed systems.
- **Scalability** - The architecture must be robust, scalable and reliable in a high transaction rate, production setting supporting thousands of patrons with a wide variety of backgrounds and information needs.
- **Persistance** – The architecture must ensure persistant access to collections of the CDL, addressing such issues as naming, digital archiving, and digital preservation.
- **Privacy** - The architecture must be sensitive to privacy issues and support both anonymous and customized access to resources.

- **Practicality** - The architecture will represent a flexible and practical approach to standards, recognizing the need to balance the level of information collected with economic constraints.
- **Modularity** – The architecture will represent a mix of new technology and legacy pieces, all of which must interoperate while evolving at different rates.
- **Timeframe** - This work is in support of planning for system migrations in the next year as well as planning for a technology generational timeframe of approximately 3 to 5 years.
- **Client Support** - The architecture will support a baseline level of services which can be accessed with common desktop computing configurations and software. Certain higher level services will require proprietary clients, but the support of these clients will be determined by the CDL Tools and Services group.
- **Infrastructure** - The architecture will rely upon University of California computing infrastructure services whenever possible such as authentication, distributed file systems, and system monitoring.

By adhering to these principles, the architecture and standards will facilitate the creation of reliable digital library services that are deployed in a cost-effective manner, and which are capable of being preserved and sustained over long time frames.

1.3 How to Use This Document

This document is intended as a guide for all in the University of California and its partners who are designing, implementing, or contracting for components of the California Digital Library, or for systems that expect to interoperate with the CDL. The document provides an overview of the CDL technical architecture and standards. It sets the context for both detailed implementation guidelines and for specific projects over a 3-5 year timeline. From this document and the related guidelines, CDL and campus project teams may discover:

- How to integrate existing systems into the CDL
- How to build a system that will integrate with the CDL
- What minimum data standards are recommended for content to be part of the CDL
- What communication standards and guidelines are recommended for interacting with CDL content and services
- How to create local delivery mechanisms to access CDL content and services

- What services are required from non-UC vendors and partners

Detailed implementation information for each of the major architectural components defined in this document will be provided in separate documents. As of November 1999, the CDL Digital Image Collection Standards, which provide implementation details for the creation of CDL-compatible digital objects, has been completed. That document is available at the following URL:

http://www.ucop.edu/irc/cdl/tasw/Current/current.html

As they are completed, additional implementation guides will be placed at the same location, beginning with information on creating and maintaining persistent identifiers for digital objects stored within CDL-compatible digital libraries.

2. Scope and Purpose

2.1 Broad Scope:

This document is intended to guide the development of digital library technologies, services, and resources for the University of California. Its scope is limited to defining the key architectural components of a distributed digital library, and standards for those components. Specific details of implementation are not addressed at this level, but may be addressed by other documents produced by the same working group, or successors. For example, this document states the need for standardization of image file formats and the metadata associated with digitized images. That need arises from the University Libraries' need for interoperability, distributed access, and long-term availability of valuable digital assets. The details of what those standards are, and how they might be met, are addressed in the CDL Digital Image Collections Standards document.

2.2 Consistent Model:

The most important result of adherence to the architecture and standards proposed here will be the use of a consistent model for the creation of digital objects. That consistency is the core of interoperability among digital library implementations, and is the best way to protect the University's investment in data capture. If the digital objects and associated metadata are compliant with these standards, then they can be used in broader contexts than the specific electronic resource for which they were created.

2.3 Economies of Scale:

Any tools developed to help support the creation of digital library assets that follow these architectural standards will be applicable across all similar University efforts. The Making of America 2 project provides a good example of this benefit. Since the CDL adopted the MOA2 metadata standard, the extensive work that UC Berkeley did to develop tools to aid in the capture of metadata as images are scanned can be reused for other projects that make use of the same metadata structure. Underlying infrastructure elements such as file systems, database servers, directories and authentication services can also be used to support multiple projects across the entire institution, provided that all the projects use the same basic architecture.

2.4 Interoperability:

It is critical that digital library resources be interoperable. Digital objects that are created within one context may be equally useful, or even more so, in another context. A watercolor painting produced by a Japanese-American internee and digitized as part of the JARDA project might also be of great interest to someone assembling an art history resource. If that painting was digitized using the architecture and standards proposed here, it will be possible to access the same base object from two entirely separate resources with completely different user interfaces. There will be no need to recapture the data or metadata, and no need to store a separate copy of the master image file. Without a consistent set of standards, such interoperability becomes much more difficult and costly.

2.5 Integration of user services

The end user's perception of the degree to which systems are integrated is largely controlled by the underlying architecture. When users are required to maintain different usernames and passwords for access to different services, they tend to perceive a limited level of integration. When navigational tools vary widely from one resource to another, that also has a negative impact on the user's perception of coherence within a distributed digital library. Adherence to the overall architecture and use of common services such as persistent identifiers, authentication, metadata management, etc. allows a developer to present the end user with a better integrated suite of applications

3. Overall Architecture

The major components of the architectural model for the California Digital Library are described below. It describes 9 major functional areas that, taken together, provide the necessary components to build robust, scalable and interoperable digital library applications and services.

Deconstructing an application into these functional areas allows the developer to better identify opportunities to promote interoperability among CDL-hosted services. It is also a useful exercise for highlighting functional areas where it may be possible to make use of existing infrastructure or other components instead of developing and maintaining them from scratch. For example, an application that requires authentication of members of the UC community might take advantage of the existing UC X.509 certificate service and the associated directory service, instead of building its own access control lists internally. A project that will involve digitizing large numbers of images could use the CDL image metadata standards to allow other applications that share the same standards to interoperate with the resulting digital objects.

- **Client Desktop**
  - World Wide Web browser
  - Telnet client
  - Application-specific desktop client

- **Service-specific functionality**
  - User profiles
  - Application services
    - Searching
    - Analysis
    - Result set management

- **Integrating components**
  - Citation linking
  - Search management
  - Format & display normalizing

- **Server tools**
  - Middleware (web/database integration)
  - Database
  - Custom tools
  - Mediators and converters (Dynaweb, MIX...)

- **Protocols and infrastructure**
  - Transport
    - HTTP
    - TCP/IP
    - Telnet
    - Storage Request Broker
  - Information management
    - Z39.50
    - ODBC, Corba
    - SDLIP
  - Directory services
  - Security
    - X.509 authentication
    - SSL, Kerberos, PGP, SSH...

- **Object metadata (EAD, Dublin Core, Open Archives, CDL, A&I, FGDC...)**
  - Descriptive
  - Structural
  - Administrative

- **Persistent identifiers**

- **Digital objects**
    - Structured text (SGML, XML)
    - Semi-structured text (HTML, TeX)
    - Unstructured text
    - Images
    - Proprietary objects

- **Storage**
    - HPSS
    - AFS/NFS
    - Local file systems

4. Specifics

The **Client Desktop** consists of the software and supporting infrastructure (network connection, DNS resolution, authentication mechanism, etc.) that need to be installed on a client workstation to access a specific CDL application or service. In most cases this will be either a properly configured World Wide Web browser or an application-specific local client program.

Service-Specific Functionality contains those elements that are used to build the screens that are presented in the top layer, and handle the interactions with the end user at the client workstation. This might be as simple as a set of static HTML pages, or as complex as a major web application written in Java or C. In some cases where the client desktop is based on locally installed software (as opposed to a web browser, for instance), the service-specific functionality will be combined with the client desktop package.

Integrating Components are distributed tools that are used to tie disparate networked resources into a coherent digital library collection or resource. Citation linking might provide live navigation paths within a document that allow the user to peruse related materials. Format and display normalizing services take information from sources that do not use compatible storage or display formats and convert them into a common format for delivery to the application, user interface and presentation layers. Search management handles problems such as broadcasting a search to multiple data sources.

Server Tools are used to manage programmatic access to underlying data sources. These might include web/database integration middleware such as Cold Fusion or Web Objects. Direct use of database queries would also fit into this category, as would custom built tools for specific purposes. Mediators and format converters such as Dynaweb or MIX can support services provided by both integrating components and service-specific components.

Protocols and Infrastructure are the most important elements of the architecture for maintaining interoperability among diverse digital library applications and services. These include transport protocols such as HTTP, information management protocols like Z39.50, and infrastructure relating to directory services, authentication and security.

Object Metadata is the information used to search through and select appropriate digital objects from the underlying data sources. It is also used to organize digital objects for presentation, and to manage the objects for preservation and access. The metadata used by a given application might be based on standards such as the Dublin Core or the Encoded Archival Description, or it might be in a proprietary abstracting & indexing or bibliographic database. The methods in the protocol and infrastructure layer need to be able to identify and manage the information in the metadata. There are generally three types of metadata associated with an object: descriptive, structural, and administrative. Descriptive metadata is used to categorize the object, and is the primary metadata used for search and retrieval. Structural metadata relates a given object to other associated objects, such as pages in a scanned book or photographs of a specific event. Administrative metadata is used to store the size, format, digitizing parameters, usage restrictions, and other information needed to manage the storage, delivery, and maintenance of the object.

Persistent identifiers are probably the least well-developed major component of the digital library architecture. Some mechanism needs to exist to reliably resolve well-known and long-lived identifiers into specific current locations for digital objects.

Digital Objects are the underlying core content of the digital library. These are the items which are identified by persistent identifiers, described by the metadata, discovered through the protocol and infrastructure which provides access to the tools. They are then combined in meaningful ways by the integrating components and delivered by applications and user interfaces to the desktop. Digital objects can be structured text such as XML or SGML, semi-structured text like HTML or TeX, or unstructured text including plain ASCII files, and text embedded within PostScript or PDF files. Other types of digital objects might include images (discussed more fully in the CDL Digital Image Collections Standards at http://www.ucop.edu/irc/cdl/tasw/Current/current.html) or various types of standardized objects such as Open GIS files. In addition, there may be proprietary digital objects created by specific applications which do not adhere to any general standard. Native format word processor or spreadsheet files might fit into this latter category.

The final element in the digital library architecture is **Storage**. Digital objects can be stored in a distributed networked environment using tools such as HPSS, AFS or NFS, or they can be kept on local filesystems. Networked storage systems for digital objects add complexity and, potentially, overhead to the process of retrieval.

However, they tend to facilitate the re-use of digital objects in multiple contexts and in support of multiple top-level services.

The architecture proposed here is by no means the only way to develop and deploy a digital library application. However, there are some significant advantages to this approach. This architecture tends to separate the underlying object from its presentation within the context of a specific application. This makes it easier to use the same objects in different ways. This architecture also abstracts the metadata which describes and locates an object from the object itself. This facilitates the creation of new virtual collections and search mechanisms. Segregating tools of general utility from applications and user interfaces used for specific purposes also tends to encourage the re-use of generic infrastructure components, enhancing interoperability and the perceived level of integration among the various resources that comprise a digital library. The use of standardized digital object models and storage architectures make it easier to provide for the preservation and long term access to the content of the digital library.

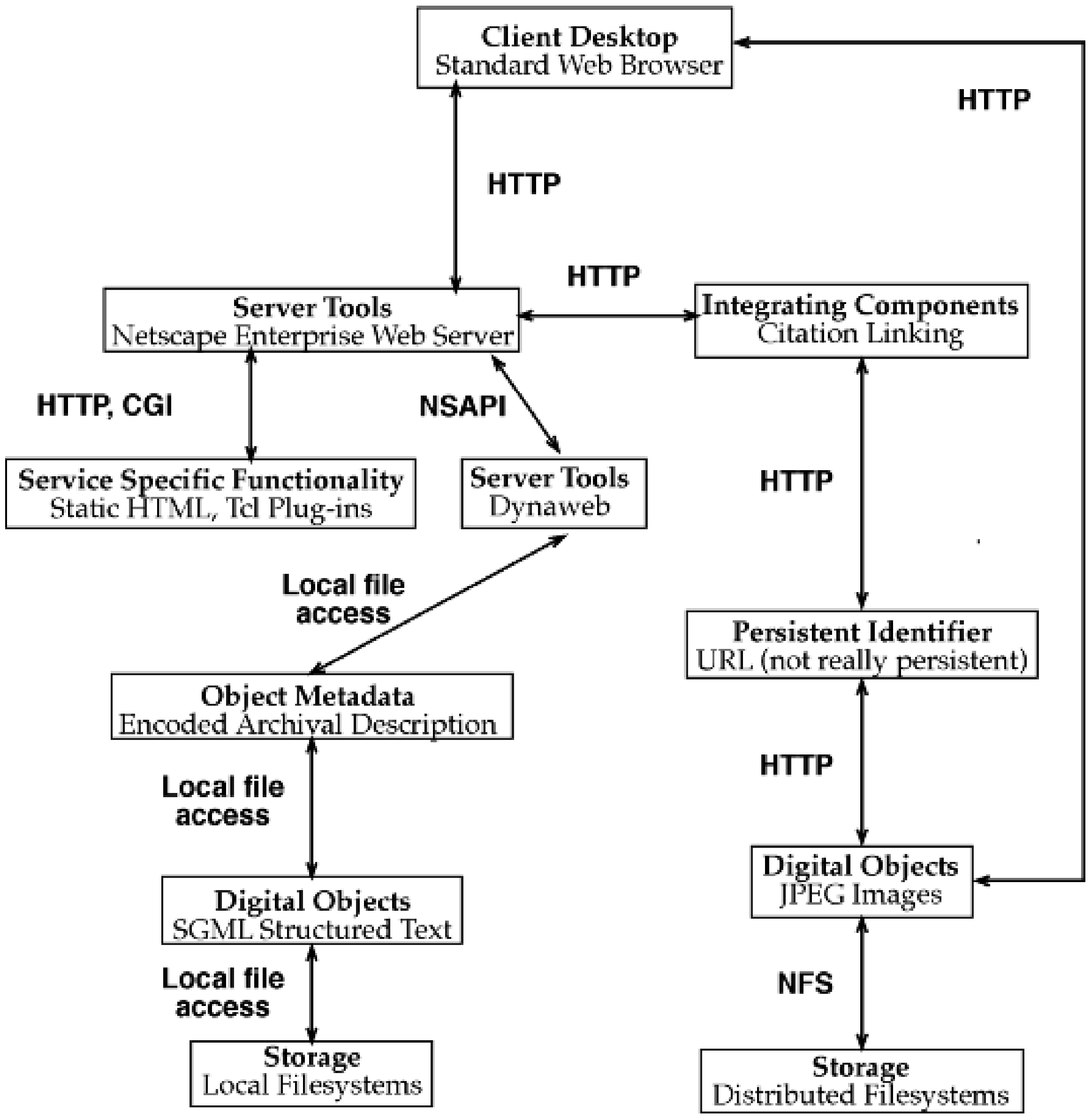
5. Examples

It might be useful to examine a few existing digital library applications to see how they make use of this architecture. These concrete examples should make the inter-relationships among the various architectural components clearer.

5.1 The Online Archive of California (OAC)

The Online Archive of California (OAC) is a collection of finding aids for special collections. It is accessible at http://www.oac.cdlib.org. Browse the California Heritage collection for a good introduction to the OAC content. A special collection is a group of related materials - the papers and correspondence of Ansel Adams, photographs and paintings of Japanese-American internment camps, MGM movie posters, Case tractor manuals, or the original musical scores of Beethoven. The finding aid is a document that describes and catalogs the contents of the special collection, so that a researcher can determine which collections might have materials of interest, and then locate those materials within the collection itself. The OAC publishes these finding aids on the Web, and in many cases includes digital images of the actual collection objects within the on-line version of the finding aid.

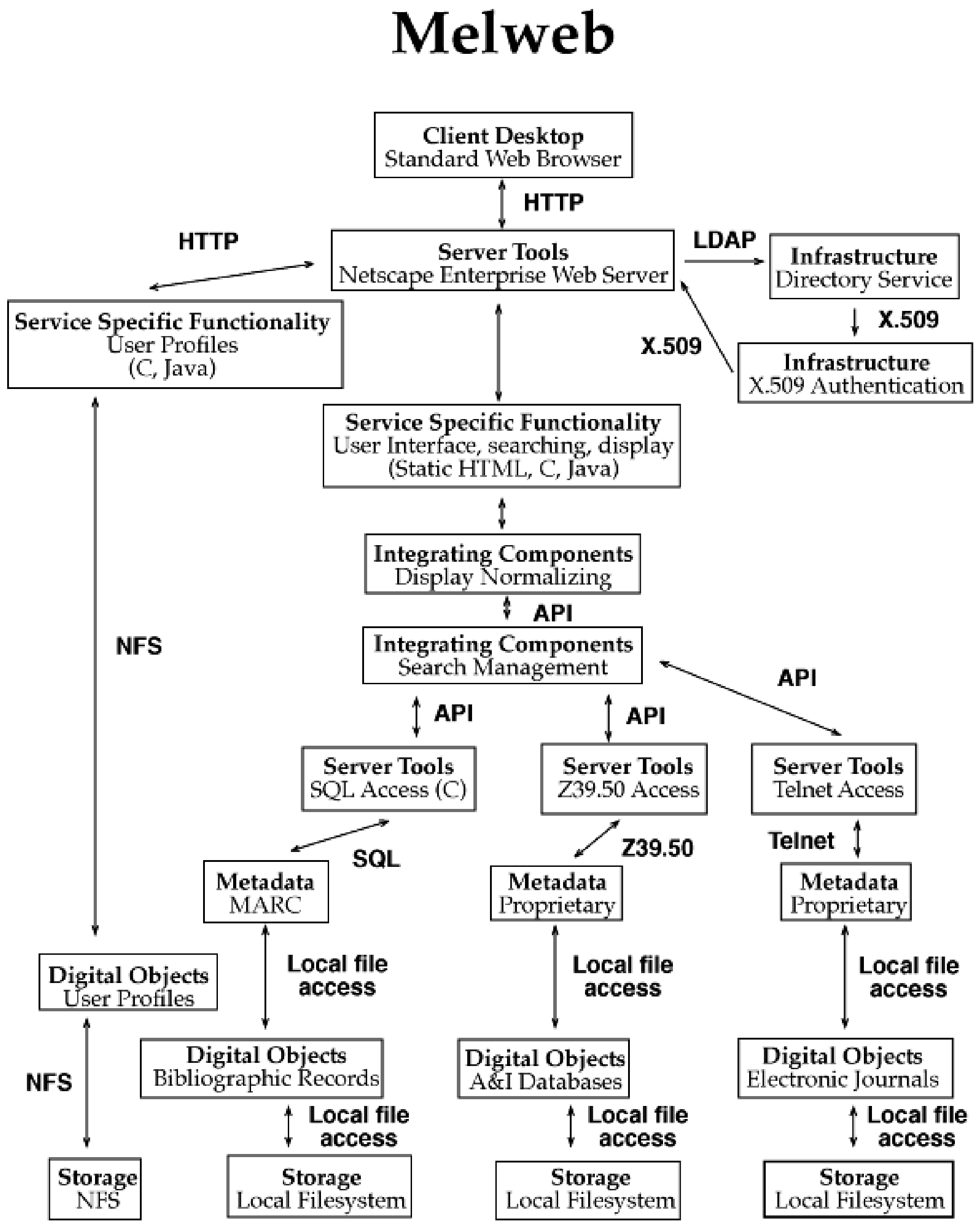
Online Archive of California



The OAC uses the World Wide Web and any standard web browser for the client desktop. The application and user interface components are constructed using a combination of static HTML pages and TCL scripts running as NSAPI (NetScape Application Programming Interface) plug-ins to the HTTP server. Integration is provided by a very simple form of citation linking which relies upon defined SGML entities, and HTTP is used to create live links to the digital proxy images of the objects in the collection. The Dynaweb tool handles the conversion of SGML into HTML for delivery to the client browser. HTTP is used as a transport mechanism, and all metadata is stored as Encoded Archival Descriptions, an SGML DTD specifically designed for finding aids. The underlying digital objects are a mixture of structured (SGML) text and images. The objects are stored on a combination of both local filesystems and distributed storage accessed through HTTP.

5.2 Melweb

The Melweb application provides access to a number of library resources, including the Melvyl catalog of UC library holdings, the Medline database of medical journal articles, and a number of other A&I and bibliographic databases, some of which include full text articles. Melweb can be seen at http://www.dbs.cdlib.org.



Melweb uses a standard web browser as the user display. The application and user interface are built with a combination of C and Java applications and applets. The primary integration function is display normalizing. Search results from disparate sources are presented in a common format. Custom SQL access tools written in C are used to retrieve information from Sybase databases. Melweb uses HTTP, Z39.50 and Telnet to communicate with clients and data sources. It also can use X.509 certificates to provide authenticated users with access to licensed databases, even from non-UC IP addresses. The primary metadata used by Melweb is the MARC record, developed by the Library of Congress. Most of the digital objects served up by Melweb are unstructured text, images, or proprietary formats from remote vendor databases. Melweb relies on NFS filesystems for its own object storage, and local filesystems for vendor objects.

5.3 Alexandria Digital Library (ADL)

[Greg, maybe you could try to put Alexandria, with the proprietary Java application client, into this context and see if it flies as a third example?]

6. Specific Architectural Components

This section describes specific implementations or tools for each of the major components of the CDL technical architecture. These are intended to provide the basis for standardization and interoperability among CDL and co-library applications. Some of the architectural components described here are important to maintaining interoperability, such as data formats, inter-application communication tools, and protocols. Others are specific tools now in use by the CDL for application development and delivery. These are not really intended as recommendations, but rather are included so that co-libraries that are working on resources that might be transferred to the CDL at some future date can have an understanding of the software, systems, and skills that presently exist within the CDL Technologies unit. Where appropriate, these CDL-specific current practices are identified and distinguished from more general recommendations for all UC digital library development.

6.1 Interoperability

Client Desktop

The basic client desktop is expected to offer network connectivity at a minimum of 56 KBPS, with a web browser capable of managing SSL encrypted communication, X.509 certificates, and Java. Standard web browsers are the preferred presentation tool for California Digital Library applications and services. Specialized client packages that are installed on the local desktop raise significant support and maintenance issues. Such specialized packages should be considered interim solutions, pending improvements in basic desktop packages such as web browsers that would allow delivery of equivalent functionality without requiring the installation of special software. While this is a CDL-specific current practice, it is strongly recommended for all digital library development efforts.

6.2 Service-specific Functionality

The CDL does not currently have a specific recommendation for development of service-specific functionality. A number of approaches appear to be equally valid, including the use of C, C++, Java, Python, Perl, or web development environments like Zope or Cold Fusion. The key issue is to adhere to sound software design and programming practices, so that the applications and user interfaces are robust and maintainable, particularly over long periods of time. Currently CDL is using C, Java, Perl and Cold Fusion. Cold Fusion is being used for rapid development of applications, but may present some scaling problems.

6.3 Integrating Components

There are no current recommendations for solutions that integrate disparate resources into a unified presentation. A number of technologies are under investigation. Citation linking architectures such as SFX www.dlib.org/dlib.april99/van_de_sompel/04van_de_sompel-pt2.html) and OpCit (www.cogsci.soton.ac.uk/~harnad/citation.html) offer some interesting directions. The CDL is experimenting with a search replication tool called Searchlight (searchlight.cdlib.org/cgi-bin/searchlight). Format normalizing is a key feature of the Typed Object Model (TOM) at tom.cs.cmu.edu/intro.html.

6.4 Server Tools

The CDL currently uses Cold Fusion for web/database integration and limited-scale web application development. Zope (www.zope.org) is also under investigation, and looks very interesting. A proprietary package called Dynaweb is being used to perform SGML content management and web delivery, but may be replaced in the near future.

6.5 Protocols and Infrastructure

HTTP is recommended for most data transport purposes. Z39.50 is recommended for distributed bibliographic queries. ODBC and JDBC are recommended for access to SQL databases. RMI and CORBA are suggested for management of distributed objects. X.509 certificates are recommended for distributed authentication. LDAP is recommended for directory services. Storage Resource Broker is under investigation for access to remote HPSS storage arrays, but most CDL applications currently use NFS or local disk for file storage.

6.6 Object Metadata

All object metadata schema should include at minimum information that can be mapped to the Dublin Core elements. The following more specialized metadata schema are recommended for specific types of underlying digital objects:

EAD	-	Finding aids
CDL Image Metadata	-	Images

Open Archive Initiative	-	Author-published scholarly articles
MARC	-	Bibliographic records

6.7 Persistent Identifiers

No specific technology has been identified to provide persistent identifiers to the UC library community. A pilot implementation is in the early planning stages, and should result in a recommendation. Key features of this implementation will include:

- Single namespace for all UC Persistent IDs (PIDs)
- Single naming authority
- User interface for creation of individual mappings from PID to resource
- Programmatic interface for bulk management of PIDs and mappings

6.8 Digital Objects

See the previously referenced document 'CDL Digital Image Collections Standards' for both format recommendations and suggested structural and administrative metadata for images. Structured text should be encoded with XML or SGML. XML is preferred. Unstructured text should be stored as Unicode 8-bit text files. Proprietary digital objects such as word processor files should be stored as binary objects wrapped with XML markup. The structural and administartive metadata recommendations for images should also apply to text objects. Recommendations for descriptive metadata are under development.

6.9 Storage

There is no recommendation for storage architecture, but distributed filesystems such as AFS and HPSS are preferable to local storage. Local storage is acceptable, but should be provided in an environment that ensures maximum data integrity and availability over time.