

BrainDump #1

**The Need for Best Practices
in Creating
Digital Library Objects**

DRAFT -- 3/22/98

Bernie Hurley

Chief Scientist, UC Berkeley Library

AUTHOR'S NOTE:	1
EXECUTIVE SUMMARY	1
INTRODUCTION	2
THE NEED FOR ADDITIONAL BEST PRACTICES AND COMMUNITY STANDARDS	3
THE TRANSITION FROM THE PRINT TO THE DIGITAL ENVIRONMENT.....	3
HASN'T THE WORLD WIDE WEB SOLVED ALL THESE PROBLEMS?	4
WHAT IS THE ROLE OF THE LIBRARY COMMUNITY IN CREATING THESE STANDARDS?	4
THE DIGITAL LIBRARY SERVICE MODEL	5
AN OVERVIEW OF THE MODEL.....	5
A MODEL FOR DIGITAL LIBRARY OBJECTS	6
<i>Adding Classes and Content to the MoA II Object Model</i>	6
<i>Adding Metadata to the MoA II Object Model</i>	6
<i>Adding Methods to the MoA II Object Model</i>	7
Object Oriented Design (OOD) as Part of the Object Model	7
Defining the Difference between Behaviors and Methods.....	8
Methods as part of the MoA II Digital Object Model.....	9
<i>Building MoA II Archival Objects</i>	10

Author's Note:

Most of this paper has been incorporated into the Making of America II White Paper (<http://sunsite.berkeley.edu/moa2>) commissioned by the Digital Library Federation.

Executive Summary

This paper briefly explores how the *culture of sharing* held by libraries has been enhanced by the creation of automated systems built on community standards such as MARC. It then suggests that libraries will continue their leadership role and define the additional best practices and standards that are required to fully extend this culture into the digital world. One important step will be to develop standards

that will replace print-based documents with digital documents. *That is, to create standards for the content and encoding of digital library objects.* Without standards, these digital objects will be encoded in proprietary formats and then stored in local repositories. The proprietary nature of these documents will not allow libraries to integrate these collections through standard tools that can work across the distributed repositories. In addition, migrating all these proprietary versions of the digital documents to new technologies will be very difficult.

As a starting point, this paper proposes a *Digital Library Service Model* in which *services* are based on *tools* that work with the *digital objects* from distributed repositories. This model recommends that libraries first define the services required for each audience to be supported and then identify the tools that are needed to implement these services. This process should include the identification of the tools' *user level behaviors*¹ (i.e., what the tools do as required by the users).

A *Digital Object Model* that fits within the overall Service Model is then proposed. *The Object Model describes digital library objects, as an encapsulation of content, metadata and methods.* Different classes of objects exist (e.g., diary, photograph, etc.) and the content of each object can be text, digitized page images, photographs, etc. The object also contains metadata that is divided into three types: *Descriptive metadata* used to discover the object; *Structural metadata* that defines the object's internal organization and is needed for display and navigation of that object; and *Administrative metadata* that contains management information (e.g., the date the object digitized, at what resolution, who can access it, etc.). Finally the digital object definition borrows from the popular object oriented design model and includes methods as part of the object. *Methods* are program code segments that allow the object to perform services for tools, such as "get the next page of this digital diary."

Introduction

Libraries have long supported a *culture of sharing* in which materials from one library's local collection have been made available to users of other libraries on an as needed basis. It is this culture that has encouraged the library community's distinguished history of developing standards to enhance the discovery and sharing of print materials (e.g., MARC, Z39.50, ISO ILL protocols, etc.). The implementation of these standards has given rise to automated systems, such as union catalogs and interlibrary loan modules, which aid library users in identifying and acquiring print-based materials from remote collections.

¹ The term "behavior" has a meaning in object oriented design -- objects exhibit *behaviors* (i.e. take actions) through the execution of their methods. We're using this phrase "user level behaviors" in this paper to define "how users describe what tools do for them." That is, their description of the tools' required functionality. *User level behaviors* should map directly into the high-level methods found in a tool. Mapping user level behaviors into high-level methods defines the boundary between functional requirements, as defined by the user, and system design.

The library community's leadership role has continued through their participation in creating best practices and standards for digital collections and content (e.g., EAD, TEI, preservation imaging, etc.). In addition, libraries have worked actively within the broader Internet community to adopt other standards that are used to store and access digital library materials (e.g., TIFF, HTTP, URNs, etc.).

The Digital Library Federation (DLF) is pursuing the development of new practices and community standards that will allow the culture of sharing within the library community to prosper in the digital age. For example, the DLF's Making of America II (MoA II) project will actively investigate the creation of possible standards for digital library objects that will populate distributed repositories. This paper explores this need and makes initial recommendations for a process MoA II can use to create standard digital library objects.

This paper is the first in a series of four "braindumps" that explore specific issues that are relevant to developing digital library systems. All four of the papers have been written to stimulate discussion of these issues within DLF and the broader community. The four papers are:

- 1) The Need for Best Practices in Creating Digital Library Objects
- 2) Interface and Repository Considerations for a Distributed Digital Library Architecture
- 3) A Systems Architecture Model for Digital Archival Repositories
- 4) New Tools to Create a "Learning Web" Environment for Digital Libraries

The Need for Additional Best Practices And Community Standards

The Transition from the Print to the Digital Environment

In the print world, libraries have created standards and best practices that have allowed them to share physical objects such as books and journal articles. The standards developed for this environment have been used to aid in the discovery and transport of these physical materials. For example, union catalogs have been built on the MARC standard, Z39.50 has allowed searching across these catalogs and the new ISO ILL protocols support the process of sharing the physical items. However, in this world one "standard" has underpinned the whole process, the physical document – a collection of printed pages. This physical document represents a "well known object" that anyone understands how to use.

The networked, digital environment poses new challenges for the culture of sharing in that there is no physical document. Therefore, we need new best practices and community standards to replace the "physical document standard" with a digital document that can be used in a digital library. Without these practices, each organization creating a digital document will do so in a proprietary

format. The result will be to create islands of digital collections that are only available using the proprietary tools developed to support that individual format.

By developing best practices to create and encode digital documents, we can reach two important goals.

1) *The Use of Standards Tools Across Distributed Repositories*

Digital documents, such as digitized books, will be found in many repositories distributed across the network. If the digital books in this example were all created to a community standards, it would then be possible to develop tools that could discover, display, navigate and manipulate these books in the same manner, regardless of the repository in which they were stored!

2) *Migration*

Migrating digital objects forward across technologies is a large, complicated and as yet, unsolved problem. If each organization that creates a digital document uses its own proprietary format for encoding their materials, the problem is multiplied. Imaging having to write programs to migrate forward different version of digital books created by Stanford, Berkeley, the Library of Congress, etc. However, if all the digital books were encoded to a best practice, the problem becomes somewhat simpler.

Hasn't the World Wide Web Solved All These Problems?

The Web is an extremely powerful tool that has greatly increased *access* to digital materials. We can surf the net as look at any document that has been published in a repository (i.e., web server). However, the Web has done little for the *integration* of materials across these repositories. This is especially true for collections that are stored in databases, where the Web only acts as a user interface. What if we wanted to integrate these documents, either physically into a central repository or virtually by providing standard tools that worked across many repositories? This would be very difficult if all the digital documents, books for example, had been encoded in a proprietary format. It would be similar to attempting to build a union catalog or create a service like Z39.50 without the benefit of the MARC standard.

What is the Role of the Library Community in Creating These Standards?

The Internet community as a whole is active in creating standards that allow for enhanced network services. These standards exist at many levels -- from defining the next generation of the Internet Protocol (IP) to the best practices needed to support electronic commerce. The library community must decide in which of these activities they should participate. For example, the World Wide Web Consortium (W3C) has benefited greatly from library community involvement in the creation of the Resources Definition Framework (RDF). Put (very) simply, RDF is a model for creating and attaching metadata to web resources (e.g., web pages)

which should result in making them easier to discover. Needless to say, the library community will also benefit from its active participation.

In contrast to the collaborative activities listed above, there are other areas in which the library community has the *primary responsibility* for creating best practices. These areas are centered on the specific services, tools and materials that a digital library must provide. As suggested earlier, developing best practices for creating digital objects that have traditionally been collected by libraries is one area of work. However, the networked environment has provided the library community with an opportunity to reach out to other communities and integrate our materials with museum artifacts, GIS data, numeric datasets, etc. In addition, the author believes that the digital library community will have great opportunities in defining best practices for new tools and services that will work with digital library materials. But, this is the topic of the fourth paper in this series.

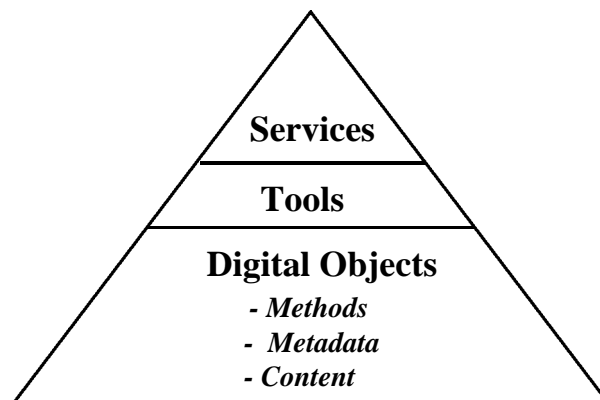
The Digital Library Service Model

AN OVERVIEW OF THE MODEL

The *Digital Library Service Model* being recommended for the MoA II Testbed Project has three layers (Figure 1), one each for *services*, *tools* and *digital objects*. In this model, services are performed through tools that discover, display, navigate and manipulate digital objects from distributed repositories.

This paper also proposes a *Digital Object Model* that fits within the overall Service Model. The Object Model defines digital objects, which are the foundation of the Service Model, as an encapsulation of content, metadata and methods.

Figure 1: *Digital Library Service Model*



1) *The Service Layer*

The service layer is comprised of a *suite of tools* that is created to support the needs of a particular audience. For example, scholars would be

comfortable using sophisticated electronic finding aids to locate and view digital archival materials such as photographs or diaries. However, fifth-graders with less rigorous information needs, may require simpler tools to discover and view these items.

2) *The Tools Layer*

This layer contains the tools that act on the digital objects at the request of the user. For example, a tool may be created to display and navigate a diary. Any tool itself is actually a *suite of behaviors*. That is, behaviors represent actions the tool can take on behalf of the user. In our digital diary example, behaviors could include actions such as turn to the next page, the previous page, jump to chapter three, translate this page to French, etc.

3) *The Digital Object Layer*

This layer contains the actual digital objects that populate distributed network repositories. Objects of the same *class* share encoding standards that encapsulate their content, metadata and methods – a full explanation of this concept follows in the next section. Separate digital object classes could be defined for books, continuous tone photographs, diaries, etc.

A MODEL FOR DIGITAL LIBRARY OBJECTS

Digital library objects form the foundation layer of the Digital Library Service Model, as described in the previous section. We can now create a Digital Object Model for these objects that will fit within the overall Service Model.

Adding Classes and Content to the MoA II Object Model

The MoA II object model defines *classes* of digital archival objects (e.g., diaries, journals, photographs, correspondence, etc.). As expected, each object in a given class has content that is a digital representation of a particular item. The format of the content can be digitized page images, ASCII text, numeric datasets, etc. The following examples describe three classes of archival objects, along with their content format.

- ***Photograph*** made up of a single digitized TIFF image;
- ***Photo Album*** made up of 30 photograph objects;
- ***Diary*** made up of 200 digitized TIFF page images and textual transcriptions.

The Object Model starts by defining classes of archival objects, where each object has content that is an electronic representation of a particular archival item of that class.

Adding Metadata to the MoA II Object Model

For the purposes of this discussion, we will consider metadata as separate from content. Metadata is data that in some manner describes the content. DLF has identified three type of metadata:

1) *Descriptive Metadata* is used in the discovery and identification of an object. Examples include MARC and Dublin Core records.

2) *Structural Metadata* is used to display and navigate a particular object for a user and includes the information on the internal organization of that object². For example, a given diary has three volumes: volume one is comprised of two sections called *dated entries* and *accounts* respectively; the entry section has two-hundred entries; entry twenty is dated August 4th, 1890 and starts on page fifty of volume one.

3) *Administrative Metadata* represents the management information for this object: the date it was created, its content file format (JPEG, JTIP, etc.), rights information, etc.

We can now add metadata to our model by stating that any given class of archival object encapsulates both the content and metadata, where the metadata is used to discover, display, navigate, manipulate and learn more about a particular object's management information.

One final note on metadata recognizes that the distinction between the types of metadata is not absolute. For example, chapters are part of the structure of a book, but chapter headings may be indexed to aid in the discovery of this item, thus filling one of the roles of descriptive metadata. In fact, the text of a book itself could be indexed and used for discovery.

Adding Methods to the MoA II Object Model

Methods are a concept defined within the object oriented analysis and design paradigm. Therefore, it would be useful to begin by reviewing concepts to be used in this paper that originate from object oriented design.

Object Oriented Design (OOD) as Part of the Object Model

Object oriented design has become very popular, as can be seen by the widespread use of related programming languages like C++ and Java. Some of the reasons for this popularity also makes OOD an attractive addition to the Digital

² Structural metadata could exist in various levels of complexity. The diary example above represents a rich structure that may be created for an important work and would include a transcription of the digitized, handwritten pages. The structure of the dairy could be encoded in this transcription and the structural metadata could be extracted from the same. On the other extreme, a dairy could exist with only enough structural metadata to turn the pages.

Library Service Model. In particular, object oriented design actually models users' behaviors, making it easier to more accurately translate their needs into system applications. This advantage will be discussed in more detail in the following section.

There is another important advantage for considering OOD. In object oriented design, a digital object conceptually *encapsulates* both content and methods. An object has content, as expected, but it also contains segments of program code called *methods*. These methods are part of the object and can be used by developers to interact with the content. For example, a developer can ask a digital book object named Book1 for page 25 by executing that object's *get_page()* method and specifying page 25. This method call may look something like *Book1.get_page(25)*.

There are a number of advantages in making methods part of the object, but probably the most important is that these basic program segments do not have to be re-invented by every developer creating a new tool³. Instead, the developer can have the tool ask the object's existing method to perform the work needed. The ability for tool developers to "reuse" these methods makes the development of new tools faster and easier. Since tools directly support the end user in this model, we want to encourage their development as much as possible!

Defining the Difference between Behaviors and Methods

One great advantage of the object oriented design approach is that it models users' behavior with methods. Therefore, this paper will now introduce a clear distinction between user level *behaviors* and *methods*. Simply put, behaviors are how users' describe what tools can do for them. For example, zoom in on this area of a photograph, show me this diary, display the next page of this book, or translate this page to French. Methods are how system designers describe what tools can do for a user.

One important purpose for distinguishing differences between user level behaviors and methods is to put in place a process where the library community can engage their users in a dialogue on what services and tools they require, down to the behaviors they need in each tool. Software engineers can then map the user behaviors into sets of methods that are required to perform the necessary functions. The line between behaviors and methods represents the transition from user requirements to system design.

³ *Technical Note:* It is worth noting that the above digital object model is only a conceptual model. In fact, complete objects made up of metadata, data and methods would not sit in a repository waiting for use. Instead, they are created as needed. That is, the parts of the objects (i.e., methods, metadata and content) are assembled from different areas of persistent store located anywhere on the network. Using the object oriented model does not require a repository to use specific object technologies like object oriented databases. Relational databases, for example, could be used for the persistent storage.

The following are example user level behaviors that might be relevant to a digital library object class of type diary.

- Show me the organization of this diary (e.g., it may have three volumes, each of which includes a section on dated entries, accounts and quotes)
- Show me the first page of Volume 1
- Show me page 3, the next page or the previous page
- Show me the fourth journal entry
- Show me the first entry for August 1890
- Show me more entries on the same topic as this one
- Show me entries that are separated by gaps of more than 10 days
- Show me entries that have these words in them
- Bookmark this entry
- Annotate this entry
- Share these entries with my colleagues

In each of the above cases, these users level behaviors would have to be mapped into a series of methods that performed the behavior. A short example may help illustrate the mapping that occurs between behaviors and methods. Imagine a user level behavior that is described as, “show me this diary.” This simple request could actually be mapped into a series of three methods that would: 1) fetch the table of contents for this particular diary; 2) display it in one frame of a Web browser that will be used to navigate the diary and; 3) fetch the first page of the diary and display it in a second frame.

Methods as part of the MoA II Digital Object Model

We can now add methods to the Object Model. At this point, it is important to note the close relationship between methods and metadata. In most cases, the methods require that appropriate metadata to be present if they are to perform their functions⁴.

The MoA II Object Model includes methods that are conceptually encapsulated along with content and metadata within an object of any given class, where the methods are used by tools to retrieve, store or manipulate that objects content. Methods often need the object’s metadata to perform their functions.

⁴ The methods that are part of an object will tend to be the ones most used across sets of tools. Tools themselves will have methods and therefore, will need access to the metadata and content of the objects. We expect that every object will have a base set of methods that can provide the tools with any metadata or content that is required.

Building MoA II Archival Objects

The final step in building a digital library object is to encapsulate the methods, metadata and content (i.e., data) into a digital library object⁵. The metadata and content must be *encoded* in a standard manner for objects in a given class. This is required so the methods (programs) that are defined for each class can work across all objects in that class.

⁵ Technical note: While the content and metadata need to be encoded in a standard manner, they do not necessarily have to be stored together. In fact, the three different types of metadata do not need to reside together. This is due to the fact that objects only come into existence on an as needed basis. Therefore, the object can be “assembled virtually” from persistent storage when required.