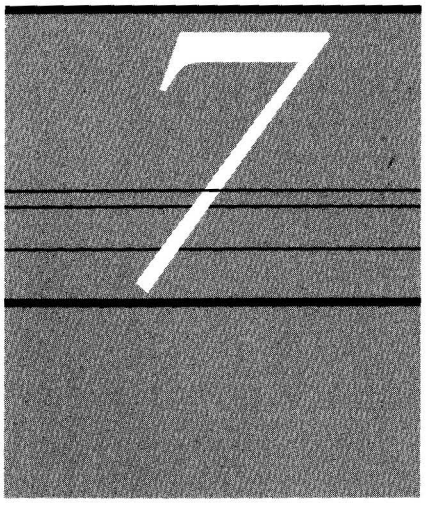
MODS: The Metadata Object Description Schema



This chapter gives an overview of the Metadata Object Description Schema (MODS) element set. Like Dublin Core, MODS is a general metadata scheme intended for description of a wide range of resources, rather than for a specific type of object or in a specific discipline or subject domain. A number of digital libraries that formerly used Dublin Core as their base scheme have moved, and continue to move, to MODS, because of its capacity for richer resource description and because their peer institutions have moved to it. Learning about MODS is valuable for anyone who wants to understand metadata for digital collections, even those who do not or will not use it in practice. To be well-informed about metadata and to be able to critically assess the characteristics and relative strengths and weaknesses of Dublin Core for digital resource description, some level of familiarity with another metadata scheme is needed. A study of MODS serves this purpose, besides being worthwhile in and of itself. It has the additional advantage of providing a good example of a richer XML-based scheme that uses attributes and hierarchically-nested subelements. This kind of scheme is the most common type in the current cultural heritage metadata world. Knowing how to read a MODS record and how to create a MODS record from scratch provides an excellent foundation for learning other XML-based metadata schemes, such as the Visual Resources Core Categories (VRA) version 4.0, covered in the next chapter; Encoded Archival Description (EAD), not covered in this book; and many others.

Having some familiarly with MODS in addition to Dublin Core also allows you to intelligently compare and contrast the characteristics of two different metadata schemes. It allows you to make a more informed assessment of the relative advantages and disadvantages of both schemes for resource description and for general interoperability. It also allows you to learn about mapping from one metadata element set to another, and to experience firsthand the challenges and issues entailed in this process, whether it is done by a human being or by an automated computer process. For all of these reasons, an introductory study of MODS is useful for anyone working with descriptive metadata for digital collections.

IN THIS CHAPTER:

This chapter does not attempt to give a complete picture of the entire MODS scheme. It uses selected elements and attributes to illustrate how MODS works in general and how it compares to Dublin Core. This chapter does not provide complete documentation for MODS. Readers of this book are strongly encouraged to look at the *MODS User Guidelines* (http://www.loc.gov/standards/mods/userguide/). Implementers of MODS will need to read this documentation carefully and consult it regularly. Instructors and students using this book are strongly encouraged to work with the *User Guidelines* for getting some hands-on practice with creating MODS records.

7.1. Introduction and Overview

MODS originated as an abbreviated XML version of the *MARC 21 Format for Bibliographic Data*. MARC is an acronym for *Machine Readable Cataloging* and is the encoding standard used for library catalog data. MODS has a high level of compatibility with MARC and includes a subset of MARC fields, but often grouped differently than in MARC 21. MODS can be used to carry selected data from existing MARC 21 records, but it can also be used for creating original resource description records. The Library of Congress developed and maintains the MODS standard. Unlike MARC, MODS uses language-based tags rather than numeric ones, and it does not assume the use of any specific cataloging code, such as the *Anglo-American Cataloguing Rules* (AACR2) or *Resource Description and Access* (RDA). As an element set, MODS is richer than Dublin Core but simpler than the full MARC bibliographic format. MODS was born in the XML environment, created explicitly as an XML schema. MODS records and documentation express MODS elements in the MODS XML encoding syntax. The MARC background of MODS is evident is some of the elements and attributes and in some of the controlled vocabularies and codes provided within the MODS scheme documentation. In some cases, these are present for the purpose of mapping or converting records from MARC 21 to MODS. Digital collection implementers do not need to know or use all of these MARC-based schemes in order to use MODS as their metadata element set, although a few of them can prove to be very useful and are not any more difficult to learn than those used with Qualified Dublin Core.

7.1.1. MODS Implementation Projects

Compared to institutions using Dublin Core as their base scheme, a proportionally much smaller but gradually growing number of digital libraries today have selected MODS as the base metadata scheme for their digital collections. This is especially true among larger academic and research libraries in the United States. Three examples are mentioned here. The MODS Implementation Registry (http://www.loc.gov/standards/mods/registry.php) provides a fuller, but by no means exhaustive, list of some of the institutions currently using MODS.

The Center for Digital Initiatives, Brown University Library (http://dl.lib.brown.edu/) uses MODS as the standard metadata format for its digital repository. Brown maps metadata from other standards, such as MARC, DC, VRA, EAD, various project databases, and other legacy data, into MODS as the common denominator.

The Texas Digital Library (TDL) (http://www.tdl.org/repositories/) includes a repository of electronic theses and dissertations (ETDs). The TDL Metadata Working Group has developed a MODS Application Profile for Electronic Theses and Dissertations (http://www.tdl.org/wp-content/uploads/2009/04/etd_mods_profile.pdf) as well as a set of Guidelines (http://www.tdl.org/wp-content/uploads/2009/04/tdl-descriptive-metadata-guidelines-for-etd-v1.pdf).

The Digital Library Federation (DLF) Aquifer Initiative (http://www.diglib.org/aquifer/) has established the goal of enabling distributed content to be used effectively by libraries and scholars for teaching, learning, and research. The Metadata Working Group of the DLF Aquifer Initiative developed a set of implementation guidelines for MODS (DLF Aquifer Metadata Working Group, 2009). This document provides guidance in using the MODS element set to describe "digital cultural heritage and humanities-based scholarly resources that are to be shared within the Aquifer Initiative and beyond." They are intended to provide a best practice for "rich, shareable metadata that is coherent and consistent." These guidelines, developed to serve Aquifer participants, can be useful to anyone using MODS, and they have in fact now been incorporated into the official *MODS User Guidelines* hosted by the Library of Congress.

7.1.2. MODS Documentation

All of the authoritative documentation for MODS can be found on the official MODS website at http://www.loc.gov/standards/mods/. The current working version of MODS is version 3.4 as of this writing. The MODS v.3.4 XML schema is located at http://www.loc.gov/standards/mods/v3/mods-3-4.xsd. It is strongly recommended that readers of this book look over the range of documentation and information included in the whole MODS website, with a special focus on the "Guidance for MODS record creation" section. This chapter focuses on the *MODS User Guidelines, Version 3* (http://www.loc.gov/standards/mods/userguide/). These Guidelines provide human-readable documentation and guidance for use of all of the MODS elements and attributes. Readers of this book are especially strongly encouraged to read the "Introduction and Implementation" and "General Application" sections of the User Guidelines. When working with MODS records, the core section of the User Guidelines is the "MODS Elements and Attributes" section. It includes links to the complete documentation for every element in the scheme. There is also a handy "Outline of Elements and Attributes in MODS Version 3.4" at http://www.loc.gov/standards/mods/mods-outline.html. Like any living metadata scheme, MODS goes through occasional major changes from one version to another and

more frequent minor changes. Readers should therefore be aware that information in this chapter may not reflect changes to the *MODS User Guidelines* made after this writing. This holds true for all metadata and controlled vocabulary schemes.

7.1.3. MODS XML Structure

Unlike Dublin Core, but like a growing number of metadata schemes, MODS was designed specifically as an XML-encoded element set. This section provides an overview of some especially noteworthy aspects of MODS in XML. First of all, all MODS documents are created according to the MODS XML Schema and validated against it to ensure that they are valid MODS documents. The XML schema gives MODS all of the advantages of XML for data interchange and reuse. A MODS document normally contains a schema declaration that indicates the MODS namespace and the current MODS XML Schema .xsd file. The previous chapter included an example and explanation of this, and the example of a complete MODS record in Section 7.3.1 of this chapter also illustrates this.

7.1.3.1. Container Elements and Subelements

MODS makes use of nested, hierarchical, parent-child elements to bundle together related information, as in the following <originInfo> and <subject> illustrations. MODS often uses **container** (or *wrapper*) elements, such as <titleInfo> and <originInfo>, as parent elements whose purpose is purely to contain, wrap, or bind together their child elements, also called subelements. These *container elements* contain no metadata values themselves. The metadata values can be entered only in their subelements. While roughly half of the MODS top-level elements are container elements, other top-level elements such as <typeOfResource> and <accessCondition> are *not* container elements. They directly contain metadata values without any subelements. The following are examples of two top-level container elements with nested subelements, and one non-container top-level element.

```
<originInfo>
    <place>
        <placeTerm>Washington, D.C.</placeTerm>
    </place>
    <publisher>Library of Congress</publisher>
    <dateIssued>1998</dateIssued>
</originInfo>
<typeOfResource>still image</typeOfResource>
<subject>
    <geographic>United States</geographic>
    <topic>Politics and government</topic>
    <temporal>20th century</temporal>
</subject>
```

Note that some subelements have their own subelements, as in the case of <place> in the preceding example, which has a <placeTerm> subelement. MODS <place> is itself a container subelement, with actual

place names allowable only in its child <placeTerm> element. Recall also from the previous chapter on XML that line breaks and indentations are typically used to make the XML easier for humans to read, but that XML itself is indifferent to these, and XML processors ignore them.

7.1.3.2. Element Attributes

MODS also makes use of XML attributes to refine the meaning or scope of an element, to designate an authority or encoding scheme used for an element value, and for other similar purposes. Recall that attributes are included within an element's start tag. They follow the *attributename="attributevalue"* pattern, as in the example <placeTerm **type="text"**>. The source of controlled vocabularies are usually designated in MODS by the use of an *authority* attribute, such as <subject **authority="lcsh"**>. This indicates that the content of that MODS subject element is taken from the *Library of Congress Subject Headings*. A single element may include more than one attribute within the start tag, but it may contain only those that have been established in the MODS XML Schema. The examples below include several uses of attributes within MODS elements, indicated by bold font.

```
<originInfo>
    <place>
        <placeTerm authority="marccountry"
         type="code">dcu</placeTerm>
        <placeTerm type="text">Washington, D.C</placeTerm>
    </place>
    <publisher>Library of Congress</publisher>
    <dateIssued>1977-2002</dateIssued>
    <dateIssued encoding="marc" point="start">
     1977</dateIssued>
    <dateIssued encoding="marc" point="end">
     2002</dateIssued>
</originInfo>

<language authority="iso639-2b">eng</language>

<subject authority="lcsh">
    <geographic>United States</geographic>
    <topic>Politics and government</topic>
    <temporal>20th century</temporal>
</subject>
```

In the first example above, notice the use of both a coded and a textual form for place of origin (publication), and the ability to explicitly encode beginning and ending dates in a date range by using the *point* attribute with values of *start* and *end*. In the third example, the Library of Congress subject heading with two subdivisions, *United States--Politics and Government--20th Century*, has each segment broken into a separate subelement designating the type of heading or subdivision that it is. They are placed within a single <subject> element because they are all part of a single subject string consisting of a heading and two subdivisions. MARC 21 does the same thing using its own tagging system: for example, *651 _1 $a United States $x Politics and government $y 20th century*, an instance of three subfields within a single subject field.

While most attributes are specific to individual elements, MODS does include a few attributes to be used throughout the MODS schema, specifically language and related attributes, date attributes, and linking attributes (http://www.loc.gov/standards/mods/v3/mods-userguide-generalapp.html#list). If elements are incorrectly nested, or if attributes are incorrectly applied, the MODS record will not validate against the MODS XML Schema.

7.1.4. Flexibility in MODS Level of Detail and Granularity

MODS gives implementers a great deal of freedom to be as simple or as detailed as they wish in the use of many MODS subelements, attributes, and vocabularies. As with Dublin Core, MODS itself mandates almost nothing, and implementers create their own local application profiles, or use a shared, consortial profile, documenting their local MODS application decisions. Like Dublin Core, this includes such aspects as which MODS elements are required versus optional, which elements are repeatable, and the order in which they are used in XML records and/or displayed locally.

MODS records can therefore range from the very simple, at the same level of detail as a Simple Dublin Core record, to the complex, with much greater detail than Qualified Dublin Core, depending on local resource description needs. MODS gives implementers a greater range of options for breaking metadata into smaller, separately tagged chunks. The important thing for any implementer to keep in mind is what you want your metadata to *do*. MODS allows you, for example, to tag a person's given name and family name in separate subelements. This would be useful if you want to machine process these pieces of information separately. If not, you do not need to make your MODS metadata that finely granular.

The Digital Library Federation's Aquifer Project has published some MODS Guidelines Levels of Adoption, included here in a sidebar. This document outlines five levels of adoption of the MODS guidelines, ranging from the loosest to the strictest, for institutions contributing records to the shared Aquifer repository. The focus is on the usability of aggregated MODS records for end users. For each level, specific MODS elements are listed. Even for non-Aquifer participants, looking at these levels can be useful for getting a sense of the range of possibilities in using MODS and for understanding that an institution does not have to use every available subelement and attribute in order to implement MODS.

MODS Requirements

- **Order of Elements**
  - "The order of elements in the MODS schema does not assume display order. A style sheet is used to control display order of MODS records."
- **Element Repeatability**
  - "All MODS top-level elements are repeatable except <recordInfo>."
- **Mandatory Elements**
  - "No element is mandatory in a MODS record, however, every MODS record requires at least one element. Applications may wish to develop profiles specifying mandatory elements as needed."
  - "The DLF/Aquifer guidelines specify a profile for sharable metadata in the DLF/Aquifer Summary of Requirements and Recommendations table that indicates required, recommended and optional elements."

(Library of Congress, 2010.)

7.2. MODS Elements: An Overview with Examples

MODS has 20 top-level elements as of this writing. Some are very simple, while others are more complex and are able to carry and encode a much richer set of distinctions and specificity than Dublin Core, even Qualified

Dublin Core. Recall that some of the top-level MODS elements contain actual metadata values, while others are container elements which themselves contain only subelements; these subelements contain the metadata values, as explained and illustrated in section 7.1. Table 7.1 lists the top-level MODS elements and their immediate subelements. No second-level subelements or any of the element attributes are included in this summary overview table. Top-level elements without any subelements have the characters --- in the second column of the table.

The following sections of this chapter constitute a brief introduction to each of the 20 top-level MODS elements, with examples of each. Most of the illustrative examples are the same as those used in the Chapters 3 and 4 sections on Dublin Core, but expressed here in the MODS

Table 7.1. MODS Top-Level Elements and Subelements

Top-Level Element	Subelements (first level only)
titleInfo	title | subTitle | partNumber | partName | nonSort
name	namePart | displayForm | affiliation | role | description
typeOfResource	---
genre	---
originInfo	place | publisher | dateIssued | dateCreated | dateCaptured | dateValid | dateModified | copyrightDate | dateOther | edition | frequency
language	languageTerm | scriptTerm
physicalDescription	form | reformattingQuality | internetMediaType | extent | digitalOrigin | note
abstract	---
tableOfContents	---
targetAudience	---
note	---
subject	topic | geographic | temporal | titleInfo | name | geographicCode | genre | hierarchicalGeographic | cartographics | occupation
classification	---
relatedItem	titleInfo | name | typeOfResource | genre | originInfo | language | physicalDescription | abstract | tableOfContents | targetAudience | note | subject | classification | relatedItem | identifier | location | accessCondition | part | extension | recordInfo
identifier	---
location	physicalLocation | shelfLocator | url | holdingSimple | holdingExternal
accessCondition	---
part	detail | extent | date | text
extension	---
recordInfo	recordContentSource | recordCreationDate | recordChangeDate | recordOrigin | languageOfCataloging | descriptionStandard

Source: Adapted from "Outline of Elements and Attributes in MODS Version 3.4," http://www.loc.gov/standards/mods/mods-outline.html. Network Development & MARC Standards Office, Library of Congress.

DLF Aquifer MODS Guidelines Levels of Adoption

1. **Minimum for participation:** Allows users to cite the resource.
   - The minimum for participation level defines the information necessary for the most basic indexing of records.
2. **Minimum for doing anything useful:** Allows users to perform basic searches and filtering.
   - This second level of adoption represents a minimum that will allow an institution's resources to be incorporated meaningfully into an aggregation.
3. **Allows more advanced functionality:** Allows users to browse and group search results.
   - The third level of adoption allows the more advanced discovery features expected of an aggregation of records from the high caliber of institution that participates in the Aquifer initiative.
4. **Adopt all required guidelines (and some recommended):** Allows users to perform more precise searches.
   - The fourth level of adoption, like the third, represents more advanced functionality than that found in traditional aggregations. Meeting this fourth level would allow the introduction of very precise searching capabilities across a wide variety of resources.
5. **Completely adopt all recommendations:** Allows users to effectively evaluate resources.
   - The fifth and highest level of adoption includes information a user would review to make a final evaluation as to whether the resource is relevant to his or her needs. Often this information is only present on the contributing institution's site, but its inclusion in shared records helps enhance the user's experience for the aggregation.

(Abbreviated from DLF Aquifer *MODS Guidelines Levels of Adoption*, 2009, added by Jenn Riley, lasted edited by Tom Habing, June 30, http://wiki.dlib.indiana.edu/confluence/display/DLFAquifer/MODS+Guidelines+Levels+of+Adoption. Network Development & MARC Standards Office, Library of Congress.)