UC3 Merritt

University of California Curation Center

Merritt Object Modeling

Rev. 0.8 – 2010-08-12

1 Introduction

Information technology and resources have become integral and indispensable to the pedagogic mission of the University of California. Members of the UC community routinely produce and utilize a wide variety of digital assets in the course of teaching, learning, and research. These assets represent the intellectual capital of the University; they have inherent enduring value and need to be managed carefully to ensure that they will remain available for use by future scholars. Within the UC system the UC Curation Center (UC3) of the California Digital Library (CDL) has a broad mandate to ensure the long-term usability of the digital assets of the University. UC3 views its mission in terms of *digital curation*, the set of policies and practices aimed at maintaining and adding value to authentic digital assets for use by scholars now and into the indefinite future [Abbott].

In order to meet these obligations UC3 is developing Merritt, an emergent approach to digital curation infrastructure [Merritt]. Merritt devolves infrastructure function into a growing set of granular, orthogonal, but interoperable micro-services embodying curation values and strategies [Foundations]. Since each of the services is small and self-contained, they are collectively easier to develop, deploy, maintain and enhance [Denning]; equally as important, since the level of investment in and commitment to any given service is small, they are more easily replaced when they have outlived their usefulness. Yet at the same time, complex curation functionality can emerge from the strategic combination of individual, atomistic services [Fisher].

NOTE The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” are to be interpreted as described in RFC 2119 [RFC2119].

2 Object modeling

Digital content curated in a Merritt environment typically will be managed by the Merritt Storage service. The Storage service imposes minimal structural requirements on the digital content that is managed in it; in particular, Storage-managed content is not typed so the service has no understanding of object semantics. Additional requirements are necessary to facilitate the management of object semantics. These requirements will be enforced by the Ingest service and the resulting object semantics will be exposed via the Inventory service.

Merritt defines a *digital object* (often simply referred to as an *object*) as the representation in digital form of a *thing*. The term “thing” is used advisedly in order to be essentially unlimited in scope. For example, Merritt objects can correspond to a variety of important classes of things, some more tangible than others, including:

- Concrete physical objects, such as books, slide images, museum artifacts, films, newspapers, traditional scholarly publications, field or laboratory notebooks, etc.
- Born-digital objects, such as online publications, websites, electronic theses and dissertations, spreadsheets, databases, spreadsheets, etc.
- Aggregate objects, such as curatorially- or administratively-defined sets of objects.
- *Seed* objects, which have a preservation–ready identifier and perhaps some minimal metadata, but for which Merritt currently has no associated content. Seed objects meet the needs of curators to obtain a "placeholder" for in-process content, from the moment to conception to the moment of "birth" (e.g., initial publication or first deposit in Merritt), a process that may span from weeks to years in various workflows.
- Standardized vocabulary objects, perhaps with associated URIs for semantic web compliance, such as canonical format names, Namaste directory types, Checkm schema profiles, licenses, policy statements, etc.

Many other classes of objects are possible as well.

Every object managed in a Merritt repository MUST have a unique primary ARK identifier. This identifier MAY be supplied by the submitting curatorial agent or it MAY be assigned automatically by the Ingest service. In either case, curators have full control over how granularity of the mapping from a unit of curatorial content to one or more objects. For example, a series of 12 annual conference proceedings can be deposited as a single "series" object, as 12 individual "conference" objects, or as 385 individual "paper" objects. UC3 can advise curators on the tradeoffs and methods for defining their assets into objects for submission. Every Merritt object can thus be seen loosely to represent a coherent intellectual *work*, c.f. [FRBR], with coherence being a function of a particular curatorial context.

Formally, a Merritt object is composed of an arbitrary number of *versions*, each representing a discrete state of the object and further composed of an arbitrary number of files, c.f. [Dflat]. In order to represent the significant properties of the components of an object, a Merritt object MUST contain a special component that is an ORE *resource map* [ORE]. A resource map provides a flexible and extensible mechanism for describing aggregations of resources in terms of RDF triples [RDF].

NOTE An RDF triple is an assertion with the generalized form "*subject predicate object*", where the predicate defines the relation existing between the subject and object.

Merritt resource maps MUST be serialized in Turtle [Turtle].

NOTE The ORE specification currently only defines serializations in RDF/XML, RDFa, and Atom/XML. Thus, Turtle is a non-standard ORE serialization. However, Turtle has the significant benefit of far greater transparency for the human reader.

The minimal Merritt object resource map is:

```
# Minimal Merritt object version resource map
@prefix mrt: <http://uc3.cdlib.org/ontology/mom#>.
@prefix ore: <http://www.openarchives.org/ore/terms/>.
<object-url>
  ore:aggregates                    # Object files
    <file-url>,
    ...,
    <file-url>,
    <resource-map-url>;
  mrt:objectType                    # Object type
    "type";
  mrt:objectRole                    # Object role
    "role";
  mrt:primaryIdentifier             # Object primary identifier
    "primary-identifier".
<resource-map-url>
  ore:describes                     # Resource map declaration
    <object-url>;
  mrt:mimeType                      # Resource map serialization
    "text/turtle".
```

NOTE Resource map examples use the Turtle "," and ";" abbreviation symbols. It is possible, and acceptable, to express functionally equivalent resource maps using more verbose alternative expressions.

The Resource map MUST define all object components and declare the object's type, role, and primary identifier. Note that the resource map MUST also include a reference to itself as a member of the object aggregation.

An object's type indicates its nature within a Merritt curation environment (see Table 1):

Identifier	***Object type***
`MRT-curatorial`	A Merritt curatorial object. Curatorial objects are those representing content originating from external curatorial contributors.
`MRT-system`	A Merritt system object. System objects are those necessary for the internal operation of the Merritt curatorial environment.

Table 1 – Merritt object types

An object's role further indicates its function within a Merritt curation environment (see Table 2):

Identifier	*Object role*	*Object type*
`MRT-class`	A Merritt curatorial class object. Class objects represent arbitrary, but nevertheless administratively- or curatorially meaningful sets of objects.	`MRT-curatorial`
`MRT-content`	A Merritt curatorial content object. Content objects represent individual abstract works.	
	Reserved for future use.	`MRT-system`

Table 2 – Merritt object roles

Additional object types and roles MAY be defined as necessary.

NOTE Merritt object classes can be thought of as "collections", that is, sets of objects aggregated together to meet some administratively- or curatorially-meaningful purpose. However, the term "collection" is widely used in many curation and preservation contexts with similar, but not always equivalent meanings. Therefore the more generic term "class" is used in its stead.

Some of an object's file components may be considered to represent the primary information content of the object, i.e. they are *data*; while other files may be descriptive of object components, i.e. they are *metadata*. Metadata may apply at any component level: the entire object, a particular version, a subset of version files, or a single file. The expression of metadata must conform to the semantics of a specific schema, as indicated by the "`mrt:metadataSchema`" property, and the syntax of the specific encoding and serialization, as indicated by the "`mrt:mimeType`" property.

All metadata relationships SHOULD be specified with appropriate resource map assertions. For example, it is RECOMMENDED that a Merritt object defines a local identifier, resolvable to an appropriate descriptive record in an external discovery system; asserts its membership in appropriate administrative classes; and includes a minimal Electronic Record Citation (ERC) description [ERC]:

```
# Object local identifier, class memberships, and ERC description
@prefix mrt: <http://uc3.cdlib.org/ontology/mom#>.
<object-url>
  mrt:localIdentifier                # Object local identifier
    "local-identifier";
  mrt:hasMetadata                    # Object metadata
    <membership-file-url>,           # Object membership component
    <erc-file-url>.                  # Object ERC component
<membership-file-url>
  mrt:metadataSchema                 # Membership schema declaration
    "MRT-membership";
  mrt:mimeType                       # Membership serialization
    "text/plain";
<erc-file-url>
  mrt:metadataSchema                 # ERC schema declaration
    "ERC";
  mrt:mimeType                       # ERC serialization
    "text/anvl".
```

NOTE Until such time as a formal MIME type for the ANVL [ANVL] is established at the IANA registry, the experimental MIME type "`text/x-anvl`" SHOULD be used.

The object local identifier SHOULD be meaningful in the local context of the object's curator. In general, this identifier SHOULD resolve to a more detailed descriptive catalog record for the object external to the Merritt curation environment.

An object can assert membership in an arbitrary number of administrative classes. The class membership mechanism MAY be used to define curatorial collections and other significant administrative object categories. The object membership component SHOULD list all of the classes of which the object is a member.

```
class1
class2
...
```

NOTE An ellipsis ("`...`") indicates arbitrary repetition of the previous element.

The object ERC component MUST minimally define the four REQUIRED Dublin Kernel elements represented in ANVL form:

```
erc:
who: creator
what: title
when: date
where: primary-identifier
[ where: local-identifier ]
[ how: methodology ]
[ why: exposition ]
```

NOTE Brackets `[` and `]` enclose optional elements.

The "`who`" element SHOULD identify a person or party responsible in some manner for the object; the "`what`" element SHOULD define a human-readable title for the object; the "`when`" element SHOULD describe the object's creation date; and the "`where`" element SHOULD specify the object's primary identifier. An additional OPTIONAL "`where`" element MAY specify the object's local identifier, which SHOULD resolve to a more complete descriptive metadata record external to the Merritt curation environment. The OPTIONAL "`how`" element SHOULD express the methodology of the object's creation; and the "`why`" element SHOULD provide an expository note regarding the object's creation.

NOTE Object descriptive metadata should be meaningful in the object's curatorial context. In general, this description should be at the level of a FRBR work, rather than expression or manifestation [FRBR]. This description is intended to provide a useful, but relatively simple overview of the object; in particular, it is *not* intended as a substitute for complete cataloging, which is expected to exist external to the Merritt curation environment at the location provided by the local identifier.

Required Kernel element values that cannot be supplied MUST use one of the pre-defined coded values appropriate to the nature of the omission, for example, "`(:unas)`" (unassigned), "`(:unkn)`" (unknown), or "`(:null)`" (meaningfully empty), c.f. [ERC].

Additional metadata MAY be associated with the object. Metadata semantics MUST be declared using one of the following literal schema identifiers as the object of a "`mrt:hasMetadata`" property. Object producers with latitude regarding metadata file names SHOULD use the reserved file names associated with the various schemas (see Table 3).

Identifier	***File name** (recommended)*	***Metadata schema***
`AES-X098B`	`mrt-aes-x098b.xml`	AES-X098B audio technical metadata.
`CreativeCommons`	`mrt-creative-commons.xml`	Creative Commons rights metadata.
`DublinCore`	`mrt-dublin-core.xml`	Dublin Core metadata.
`ERC`	`mrt-erc.txt`	Electronic Resource Citation descriptive metadata.
`JHOVE2`	`mrt-jhove2.xml`	JHOVE2 characterization metadata,
`MARCXML`	`mrt-marcxml.xml`	MARC bibliographic metadata.
`MIX`	`mrt-mix.xml`	MIX / NISO Z39.87 still image technical metadata.
`MODS`	`mrt-mods.xml`	MODS descriptive metadata.
`MRT-class-members`	`mrt-class-members.txt`	Merritt class member assertions.
`MRT-ingest`	`mrt-ingest.txt`	Merritt ingest metadata.
`MRT-membership`	`mrt-membership.txt`	Merritt class membership assertions.
`PREMIS-object`	`mrt-premis-object.xml`	PREMIS object preservation metadata.
`PREMIS-rights`	`mrt-premis-rights.xml`	PREMIS rights preservation metadata.
`VRA-core`	`mrt-vra-code.xml`	VRA Core cultural heritage metadata.

Table 3 – Metadata schemas

NOTE Within a Merritt curation environment all file system names beginning with "`merritt`" or "`mrt`" (on a case-insensitive basis) are reserved.

Additional metadata schemas MAY be defined as necessary.

2.1 Collections

Each Merritt class ("collection") is represented by a class object of type "`MRT-class`". The name of the class is specified by the "`what`" element in its associated ERC file "`mrt-erc.txt`". A Merritt object asserts its membership in some set of classes by enumerating the class identifiers in a membership file, "`mrt-membership.txt`" file. The class identifiers are the primary ARK identifiers of the class objects. Alternatively, a Merritt class object can assert that some set of objects are members of itself by enumerating the object identifiers in a members file, "`mrt-class-members.txt`" file. Object identifiers are the primary ARK identifiers for the objects. Class membership can be established through either mechanism, or both, although it is not necessary to use both. Thus, an object MAY be added to a class without any modification to the object itself by adding

the object identifier to the class members file, "`mrt-class-members.txt`", and an object MAY be added to a class without any modification of the class object itself by adding the class identifier to the object's membership file, "`mrt-membership.txt`".
Every Merritt object is implicitly a member of two classes ("collections"):

- The class corresponding to the Ingest service submission profile under which the object was submitted.
- The class corresponding to the service level agreement under which the object was submitted, which is retrievable from the relevant submission profile.

The class object for the submission profile class SHOULD define a local identifier that is the URL of the profile in the Merritt registry:

`http://uc3.cdlib.org/registry/ingest/profile/`*`profile`*

The class object for the service level agreement class SHOULD define a local identifier that is the URL of the agreement in the Merritt registry:

`http://uc3.cdlib.org/registry/ingest/sla/`*`agreement`*

3 Ontological classes and relationships

Merritt makes use of ontological classes and properties defined in the following namespaces (see Table 4).

Prefix	***URI***	***Ontology***
`cc`	`http://creativecommons.org/ns#`	Creative Commons rights metadata.
`dc`	`http://purl.org/dc/elements/1.1/`	Dublin Core metadata elements.
`dcterms`	`http://purl.org/dc/terms/`	Dublin Core metadata terms.
`ore`	`http://www.openarchives.org/ore/terms`	ORE aggregation terms.
`mrt`	`http://uc3.cdlib.org/ontology/mom#`	Merritt classes and properties.
`rdf`	`http://www.w3.org/1999/02/22-rdf-syntax-ns#`	RDF vocabulary terms.
`rdfs`	`http://www.w3.org/2001/01/rdf-schema#`	RDF Schema vocabulary terms.
`xsd`	`http://www.w3.org/2001/XMLSchema`	XML Schema datatypes.

Table 4 – Merritt namespaces

NOTE The acronym "mom" stands for Merritt object model.

Additional namespaces MAY be defined as necessary.

Merritt defines a number of its own ontological classes and properties, all in the "`mrt`" namespace (`http://uc3.cdlib.org/ontology/mom#`).

3.1 Merritt classes

The following Merritt classes are defined.

mrt:Component	
rdfs:label	Component
rdfs:comment	Abstract Merritt class for object model components: object, version, file.
rdf:type	rdf:Class

mrt:File	
rdfs:label	File
rdfs:comment	Merritt class for files. A file is a formatted octet stream.
rdf:type	rdf:Class
rdfs:subClassOf	mrt:Component

mrt:Object	
rdfs:label	Object
rdfs:comment	Merritt class for objects. An object is the representation in digital form of an abstract work. An object is composed of an arbitrary number of versions.
rdf:type	rdf:Class
rdfs:subClassOf	mrt:Component
rdfs:seeAlso	mrt:Version

mrt:Version	
rdfs:label	Version
rdfs:comment	Merritt class for versions. A version is a set of files that collectively represents a discrete state of an object. A version is composed of an arbitrary number of files.
rdf:type	rdf:Class
rdfs:subClassOf	mrt:Component
rdfs:seeAlso	mrt:Object
rdfs:seeAlso	mrt:File

3.2 Merritt properties

The following Merritt properties are defined.

mrt:hasPreview	
rdfs:label	hasPreview
rdfs:comment	Merritt property asserting that an object has a preview representation. The preview representation should present a concise summary of the object, such as a thumbnail for an image or an abstract for a document.
rdf:type	rdf:Property
rdfs:domain	mrt:Object
rdfs:range	mrt:File

mrt:hasMetadata	
rdfs:label	hasMetadata
rdfs:comment	Merritt property asserting that an object component has metadata. Metadata provide further information about the component.
rdf:type	rdf:Property
rdfs:domain	mrt:Component
rdfs:range	mrt:File

mrt:isDerivativeOf	
rdfs:label	isDerivativeOf
rdfs:comment	Merritt property asserting that an object component is a derivative of another component. Note that both components must be of the same type. For example, a file can only be a derivative of another file.
rdf:type	rdf:Property
rdfs:domain	mrt:Component
rdfs:range	mrt:Component

mrt:localIdentifier	
rdfs:label	localIdentifier
rdfs:comment	Merritt property asserting that an object has a local identifier. A local identifier should be meaningful in the object's curatorial context and may resolve to a descriptive metadata record for the object in a discovery service external to a Merritt curation environment.
rdf:type	rdf:Property
rdfs:domain	mrt:Object
rdfs:range	xsd:string

mrt:metadataSchema	
rdfs:label	metadataSchema
rdfs:comment	Merritt property asserting that a file contains metadata conforming to a particular schema.
rdf:type	rdf:Property
rdfs:domain	mrt:File
rdfs:range	rdf:Literal

mrt:mimeType	
rdfs:label	mimeType
rdfs:comment	Merritt property asserting that a file has a particular MIME type. MIME types must either be registered in the IANA registry or be experimental types.
rdf:type	rdf:Property
rdfs:domain	mrt:File
rdfs:range	xsd:string

mrt:objectRole	
rdfs:label	objectRole
rdfs:comment	Merritt property asserting that an object has a particular role.
rdf:type	rdf:Property
rdfs:domain	mrt:Object
rdfs:range	xsd:string

mrt:objectType	
rdfs:label	objectType
rdfs:comment	Merritt property asserting that an object is of a particular type.
rdf:type	rdf:Property
rdfs:domain	mrt:Object
rdfs:range	xsd:string

mrt:primarydentifier	
rdfs:label	primaryIdentifier
rdfs:comment	Merritt property asserting that an object has a primary identifier. A primary identifier is the unique and persistent identifier used by a Merritt curation environment to identify the object.
rdf:type	rdf:Property
rdfs:domain	mrt:Object
rdfs:range	xsd:string

4 File system expression

Object components may originate from the object's producer, an object consumer, or be generated automatically by a Merritt service. The primary obligation of a Merritt curation environment is to ensure the long-term viability of producer-supplied content. There is a somewhat lower obligation regarding system-generated content, which presumably can be regenerated as necessary; and consumer-supplied content, which by definition originates outside of the object's primary curatorial context. The distinction between component provenance is explicitly represented in the file system representation of an object, c.f. [Dflat].

```
consumer/
        [ consumer-supplied-files
          ... ]
producer/
```

```
        [ producer-supplied-files
          ... ]
    system/
        [ mrt-erc.txt ]
        [ mrt-ingest.txt ]
        [ mrt-membership.txt ]
          mrt-object-map.ttl
        [ system-generated-files
          ... ]
```

By convention the file "`mrt-object-map.ttl`" is the Merritt object resource map automatically generated by the Merritt Ingest service at the time of submission.

The full form of an object resource map is:

```
# Merritt object version resource map
@prefix mrt: <http://merritt.cdlib.org/terms#>.
@prefix ore: <http://www.openarchives.org/ore/terms/>.
<object-url>
  ore:aggregates                    # Object files
    <file-url>,
    ...,
    <file-url>,
    <resource-map-url>;
  mrt:objectType                    # Object type
    "type";
  mrt:objectRole                    # Object role
    "role";
  mrt:primaryIdentifier             # Object primary identifier
    "primary-identifier" [ ;
  mrt:localIdentifier               # Object local identifier
    "local-identifier" ] [ ;
  mrt:hasMetadata                   # Object metadata
  [ <ingest-file-url> ] [ ,         # Object ingest component
    <membership-file-url> ] [ ,     # Object membership component
    <erc-file-url> ] [ ,            # Object ERC component
    <metadata-file-url> ] [ ,       # Object metadata component
    ... ] [ ;
  mrt:hasPreview                    # Object preview component
    <preview-file-url> ] .
<resource-map-url>
  ore:describes                     # Resource map declaration
    <object-url>;
  mrt:mimeType                      # Resource map serialization
    "text/turtle".
[ <class-members-file-url>          # Note: for class objects only!
  mrt:metadataSchema                # Class members schema declaration
    "MRT-class-members";
  mrt:mimeType                      # Class members serialization
```

```
        "text/plain". ]
    [ <membership-file-url>
        mrt:metadataSchema                  # Membership schema declaration
          "MRT-membership";
        mrt:mimeType                        # Membership serialization
          "text/plain". ]
    [ <erc-file-url>
        mrt:metadataSchema                  # ERC schema declaration
          "ERC";
        mrt:mimeType                        # ERC serialization
          "text/anvl". ]
    [ <metadata-file-url>
        mrt:metadataSchema                  # Metadata schema declaration
          "schema";
        mrt:mimeType                        # Metadata serialization
          "mime". ]
      ...
```

References

[Abbott] Daisy Abbott, *What is Digital Curation?* April 3, 2008 <http://www.dcc.ac.uk/resource/briefing-papers/what-is-digital-curation/>.

[Denning] Peter J. Denning, Chris Gunderson, and Rich Hayes-Roth, "Evolutionary system development," *Communications of the ACM* 51:17 (December 2008): 29-31.

[Dflat] UC3, *Dflat: A Simple File System Convention for Digital object Storage*, 2010.

[ERC] J. Kunze and A. Turner, *Kernel Metadata and Electronic Resource Citations*, April 28, 2009 <http://dublincore.org/kernelwiki/FrontPage?action=AttachFile&do=get&target=ercspec.html>.

[Fisher] David A. Fisher, *An Emergent Perspective on Interoperation in Systems of Systems*, CMU/SEI-2006-TR-003, ESC-TR-2006-003, March 2006 <http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06tr003.pdf>.

[Foundations] UC3, *Digital Preservation Program: Foundations*, 2010.

[FRBR] IFLA Study Group on the Functional Requirements for Bibliographic Records, *Functional Requirements for Bibliographic Records – Final Report*, UBCIM 19 (February 2009; München: *K*. G. Saur, 1998).

[Merritt] UC3, *Merritt: An Emergent Approach to Digital Curation Infrastructure*, 2010.

[MIME] IANA, *MIME Media Types*, 2010 <http://www.iana.org/assignments/media-types>.

[ORE] Carl Lagoze, Herbert Van de Sompel, Pete Johnston, Michael Nelson, Robert Sanderson, and Simeon Warner, eds., *ORE User Guide – Primer*, October 17, 2008 <http://www.openarchives.org/ore/primer>.

[RDF] Graham Klyne and Jeremy J. Carroll, eds., *Resource Description Framework (RDF): Concepts and Abstract Syntax*, W3C Recommendation, February 10, 2004 <http://www.w3.org/TR/rdf-concepts/>.

[RFC2119] S. Bradner, *Key Words for Use in RFCs to Indicate Requirement Levels*, BCP 14, RFC 2119, March 1997 <http://www.ietf.org/rfc/rfc2119.txt>.

[Turtle] David Beckett and Tim Berners-Lee, *Turtle – Terse RDF Triple Language*, January 14, 2008 <http://www.w3.org/TeamSubmission/turtle/>.