UC Curation Center
CDL UC3 Merritt
California Digital Library

UC Curation Center / California Digital Library

Merritt: An Emergent Micro-services Approach to Digital Curation Infrastructure

Rev. 0.6 – 2010-03-25

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), one of five programmatic areas of the California Digital Library (CDL), has a broad mandate to ensure the long-term usability of the University's digital assets.

2 Digital Curation

Digital curation is the set of policies and practices aimed at maintaining and adding value to a body of trusted digital content for use now and into the indefinite future [Abbott]. Traditionally, preservation and access have been considered as disparate activities. They should, however, properly be seen as complementary functions: preservation focused on ensuring use *over time*, while use depends upon preservation up to a *point in time* [Rusbridge]. Curation is thus an ongoing process of management and enrichment at all stages of the lifecycle of a digital asset [Higgins] and should facilitate the alignment with the scholarly lifecycle (see Figure 1). While curation is not solely a technical undertaking – curation success is, for example, highly dependent on important human competencies, analysis, and decision making – a robust infrastructure in which to manage valuable digital content efficiently and effectively is nevertheless a necessary foundation.

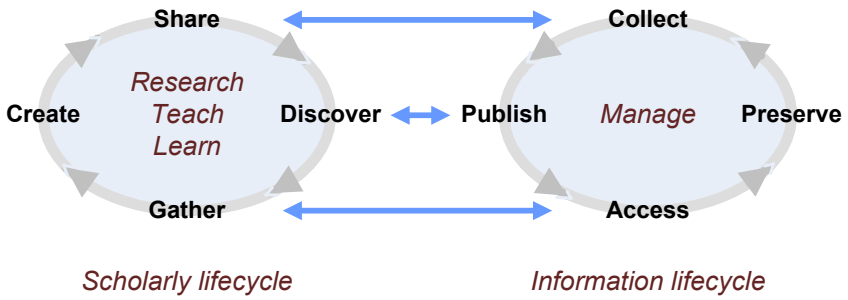

Figure 1 – Alignment of scholarly and information lifecycles

3 Merritt

Merritt is an approach to digital curation based on the ideas of *micro-services* and *emergent behavior*. Merritt devolves technical infrastructure function into a set of independent, but interoperable, micro-

services that embody 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, 2008]. 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. Although the individual services are narrowly scoped, the complex function needed for effective curation *emerges* from the strategic combination of atomistic services [Fisher]. The emergent micro-services underlying Merritt can be summarized as a set of useful metaphors, principles, preferences, and practices (see Table 1).

Metaphors	***Preferences***	***Practices***
• Pipeline • Lego™ bricks	• Small and simple over large and complex • Minimally sufficient over feature-laden	• Define, decompose, recurse • Approach sufficiency through a series of incrementally necessary steps
Principles		
• Granularity • Orthogonality • Parsimony • Evolution	• Configurable over the prescribed • The proven over the merely novel • Outcomes over means	• Top-down design, bottom-up implementation • Code to interfaces

Table 1 – Micro-services

The general intent behind Merritt is to provide a curation environment that is comprehensive in scope, yet flexible with regard to local policies and practices and the inevitability of disruptive technological change. To achieve these goals, the Merritt approach deprecates the centrality of the curation repository as *place* [Abrams, Cruse, and Kunze, 2009]. In Merritt terms, a repository is a logical aggregate, rather than a physical unity; a collection of useful services, not an all-encompassing, monolithic system.
Merritt services can be deployed in the environments in which it makes most sense, both technically and administratively. While UC3 will continue to use Merritt as the basis for its centrally-managed curation activities, Merritt services can also be operated in local campus environments either individually or in combination. With Merritt it is no longer necessary that digital content must be transferred to a common repository in order to receive appropriate curation care.

The range of initial Merritt services (see Figure 1) is suggested by the following curation imperatives reflective of evolving community best practice:

- Providing safety through redundancy.
- Maintaining meaning through description.
- Facilitating utility through service.
- Adding value through use.

These imperatives correspond to the four level Merritt service hierarchy, each dependent on assumed lower level functionality.

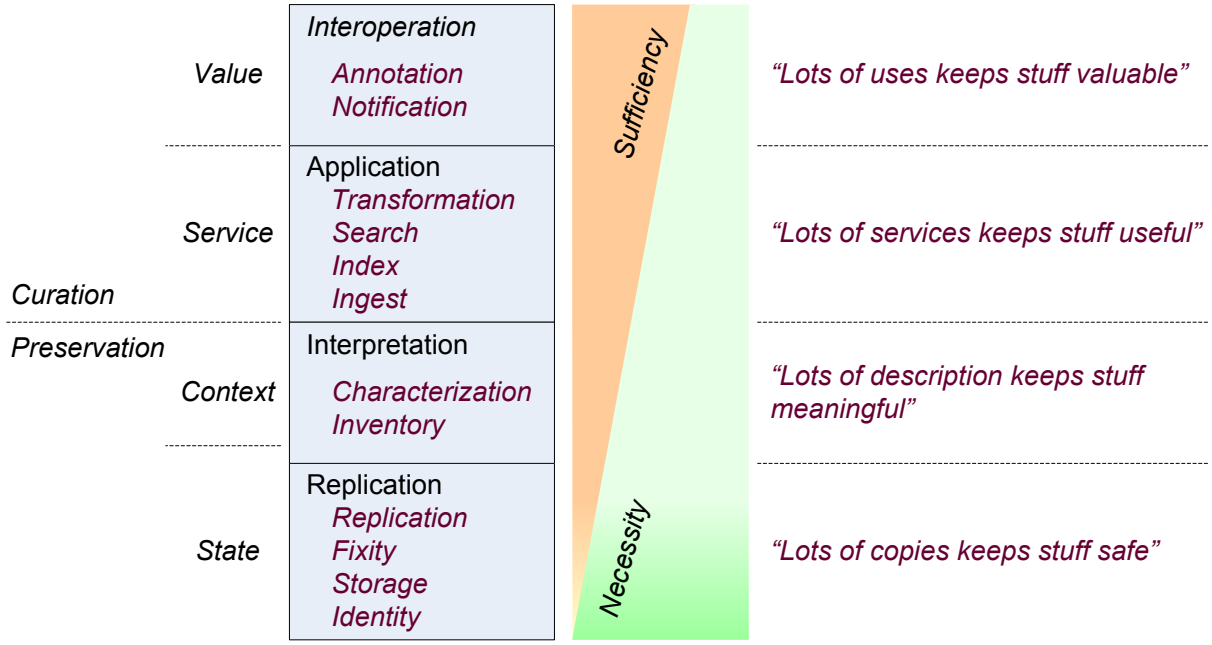

Figure 1 – Merritt micro-services

It is expected that additional, added-value services will continue to be added to Merritt repertoire over time. The application of these services extends throughout the full digital curation life cycle.

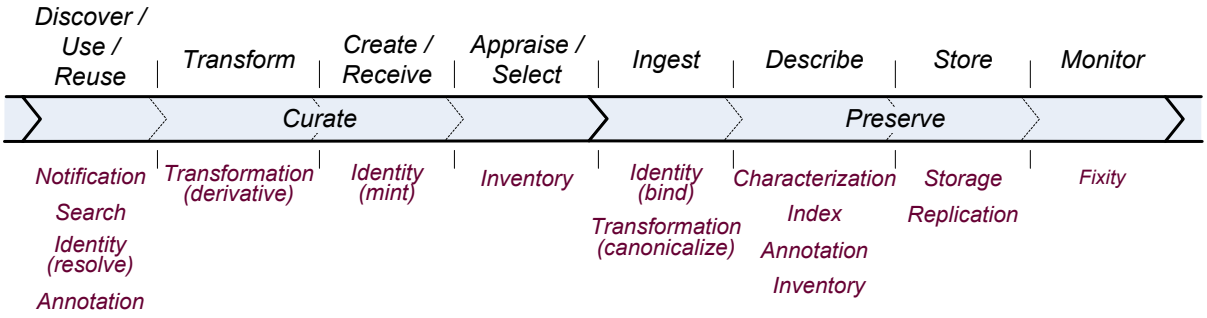

Figure 2 – Micro-service applicability throughout the curation lifecycle
Adapted from [Higgin]

4 Merritt Services

By explicit design, Merritt micro-services are granular and orthogonal. They expose their function through simple, well-defined abstract interfaces that define their public service "contract" (see Figure 3) [Liegl; O'Reill]. The design of the services conforms to the high-level UC3 mission goals:

- Providing safety through redundancy.
- Maintaining meaning through description.
- Facilitating utility through service.
- Adding value through use.

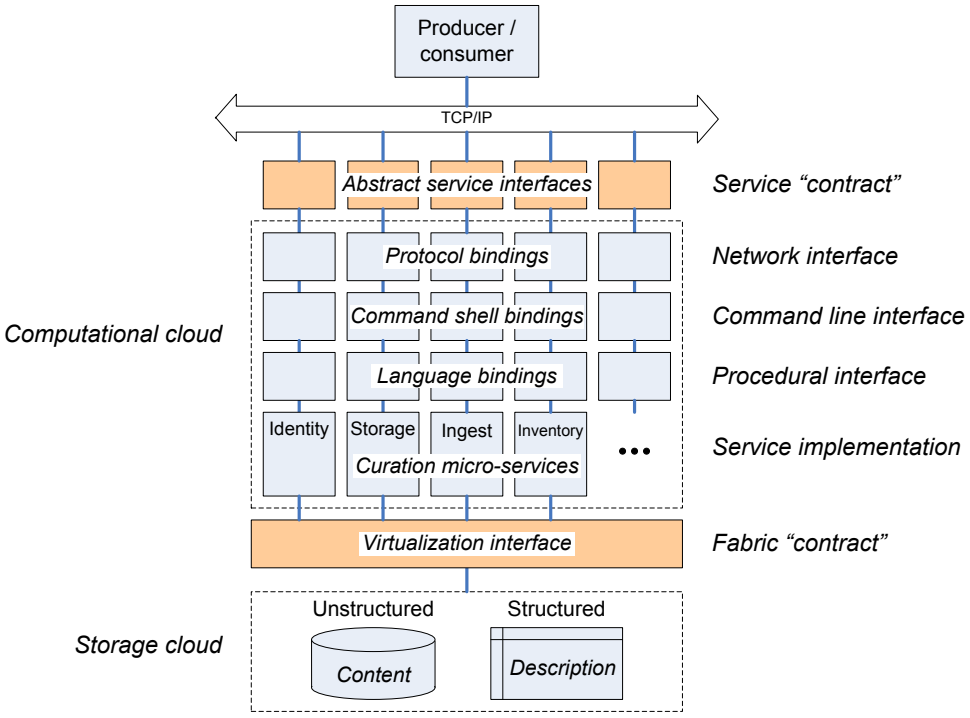

Figure 3 – Teak service stack

and reflects the following high-level design goals:

- Open source deliverables.
- Granularity and orthogonality (both *across* and *within* components).
- Persistent interfaces, evolving implementations.
- Complexity by composition, not addition.
- Flexible configuration, but meaningful default behavior ("Principle of least surprise").
- Strategic re-use ("Principle of least effort").

All service interfaces support uniform methods for requesting help information about the use of the service, for retrieving global service state information, and for submitting questions, comments, and error reports to the service providers (not explicitly documented in the following service descriptions). Similarly, all interface methods accept user agent *credentials* to authorize the requested operation, and accept user-supplied *hints* for the preferred form of the response (see Table 2). User agent identity and role management and authorization should be consistent with the evolving common UC infrastructure.

Service request		
Method	Mandatory	Interface method
Arguments	Optional	Method arguments
Credentials	Optional	User agent credentials
Hint	Optional	Requested output form

Table 2 – Common method request format

Interaction with Merritt services can occur in several modalities:

- *Procedural APIs*, with various language bindings.
- *Command line APIs* supported in various operating system command shells.
- *Web APIs*, composed of:
    - *Thin client GUIs*, supported in various browser platforms.
    - *RESTful service interfaces* [Fielding].

The preferred language bindings for Teak procedural APIs are Java and Perl. The preferred Java platform for Teak RESTful APIs is Jersey (the reference implementation of JSR 311, *JAX-RS – Java API for RESTful web services*) running in a Jetty or Tomcat container.

Service methods are categorized with respect to two important transactional properties:

- *Safety*. A method is safe if it does not implicate side effects – changes to persistent state – on the server.
- *Idempotency*. A method is idempotent if multiple invocations are functionally indistinguishable from a single invocation.

Since idempotency is a necessary consequence of safety, all methods will fall into one of three categories:

- Idempotent / safe.
- Idempotent / unsafe.
- Non-idempotent / unsafe.

4.1 Identity service

The Identity service manages persistent identifiers that can be associated with digital content. The service is based on two conceptual entities:

- *Minter*. A minter generates a bounded or unbounded deterministic sequence of identifier strings conforming to a specific pattern or *namespace*.
- *Binder/resolver*. A binder maintains the association between identifier strings and named, typed values. One important value type is "`goto`", which defines an actionable URL. The process of dereferencing the "`goto`" URL is known as identifier *resolution*.

The Identity service supports the following methods:

- *Mint identifier*(*s*).
- *Set binding*.
- *Get binding*(*s*).
- *Resolve identifier*.
- *Get minter state*.
- *Get binder/resolver state*.

NOTE In addition to the methods explicitly noted in this and subsequent sections, all Teak services support a *Help* method and a *Get global service state* method.

4.2 Storage service

The Storage service manages the versioned, unstructured, opaque storage of the file components of digital objects. The service is based on four conceptual entities:

- *Node*. A storage node is a named instance of specific storage technology and policy.
- *Object*. A digital object is the digital representation of a coherent unitary intellectual or aesthetic work.
- *Version*. A version is a discrete state of an object at a point in time.
- *File*. A file is formatted byte stream.

The Storage service supports the following methods:

- *Add version*.
- *Get file*.
- *Get version*.
- *Get object*.
- *Delete version*.
- *Delete object*.
- *Get node state*.
- *Get object state*.
- *Get version state*.
- *Get file state*.

4.3 Fixity service

The Fixity service verifies the bit-level integrity of files. The service is based on two conceptual entities:

- *File*. A file is a byte stream.
- *Digest*. A digest is a value resulting from the application of a mathematical algorithm to a file. The service supports a number of digest algorithms with useful cryptographic properties, including:
    - Adler-32, CRC-32, MD2, MD5, SHA-1, SHA-256, SHA-384, SHA-512.

The service supports the following methods:

- *Verify*.
- *Get file state*.

4.4 Replication service

The Replication service manages the synchronization of content replicas in diverse instances of the Storage service. The service is based on two conceptual entities:

- *Object*. A digital object is the digital representation of a coherent unitary intellectual or aesthetic work.
- *Rule*. A rule is a policy-based assertion of desired replication status.

The service supports the following methods:

- *Set rule*.
- *Replicate*.
- *Get object state*.

4.5 Inventory service

The Inventory service manages structured, typed information that is syntactically, semantically, and pragmatically descriptive of curated digital objects.

NOTE The term "descriptive" is used here in its general sense, that is, information *about* something, and is not intended to specify strictly intellectual as opposed to technical, structural, administrative, etc. information

The service is based on the conceptual entities:

- *Collection*. A collection is an arbitrary administrative aggregation of digital objects.
- *Object*. A digital object is the digital representation of a coherent unitary intellectual or aesthetic work
- *Version*. A version is a discrete state of an object at a point in time
- *File*. A file is formatted byte stream.

The service supports the following methods:

- *Get collection state*.
- *Get object state*.
- *Get version state*.
- *Get file state*.

4.6 Characterization service

The Characterization service provides a mechanism for the automated examination of digital objects to determine their significant properties and implications of those properties. The characterization process has four aspects [Abrams Owens, and Cramer, 2008]:

- *Identification*. The determination of the *presumptive* format of a digital object on the basis of suggestive extrinsic hints and intrinsic internal and external signatures.
- *Validation*. The determination of the *conformance* of an object to the normative requirements of its format's specification.
- *Feature extraction*. The reporting of intrinsic properties of a digital object that can be used as a *surrogate* for the object itself for purposes of curation analysis and decision making.
- *Assessment*. The determination of the extrinsic *acceptability* of a digital object for a specific purpose on the basis of local policy rules.

The service is based on three conceptual entities:

- *Source unit*. A source unit is the unit of examination and reporting, and may be aggregate, such as a directory, set of files, or a bundling container (e.g. tar or zip), or unitary, such as a file or an encapsulated byte stream.
- *Reportable*. A reportable is a named container of a set of properties.
- *Property*. A property is a named, type value.

The service supports the following methods:

- *Characterize*.
- *Get source unit state*.
- *Get reportable*.
- *Get property*.

4.7 Ingest service

The Ingest service provides a mechanism to accept digital content into a managed curation environment. The service is based on four conceptual entities:

- *Batch*. A batch is an arbitrary collection of jobs submitted for ingest, and that *succeeds or fails as an atomic unit*.
- *Job*. A job is a set of files representing a new discrete state of a digital object.
- *Object*. A digital object is the digital representation of a coherent unitary intellectual or aesthetic work
- *File*. A file is a formatted byte stream.

The service supports the following methods:

- *Submit*.
- *Get batch state*.
- *Get job state*.

4.8 Index service

The Index service manages searchable indexes of object content and descriptive information. The service supports the following methods:

- *Index*.

4.9 Search service

The Search service provides content discovery and delivery through index-based search and browse of object content and descriptive information. The service supports the following methods:

- *Search*.
- *Browse*.

4.10 Transformation service

The Transformation service provides a mechanism to transcode object representations from existing forms to desired forms. The service is based on three conceptual entities:

- *Object*. A digital object is the digital representation of a coherent unitary intellectual or aesthetic work
- *File*. A file is formatted byte stream.
- *Format*. A format is a set of syntactic rules governing the mapping from an abstract information model to a serialized byte stream.

The service supports the following methods:

- *Transform*.

4.11 Notification service

The Notification service provides a mechanism to notify user communities that new digital content is being managed and is available for use. The service supports the following methods:

- *Subscribe*.

4.12 Annotation service

The Annotation service provides a mechanism by which the consumers of managed digital content can enrich that content through additional description, association, or related content. The service supports the following methods:

- *Annotate*.
- *Submit*.

5 Conclusion

The 12 Merritt micro-services build from a base of minimal necessity and extend towards increasing sufficiency (see Figure 8). Taken together, they provide the baseline function needed to meet UC3 obligations for effective, efficient, and sustainable curation services. The services can be aggregated into four service levels – *protection*, *interpretation*, *application*, and *interoperation*, respectively providing maintenance of content *state* and *context*, and provision of content *service* and *value* – which suggests a pithy summarization of UC3 intentions in terms of four rather simple aphorisms:

- Lots of copies keeps stuff safe.
- Lots of description keeps stuff meaningful.
- Lots of services keeps stuff useful.
- Lots of uses keeps stuff valuable.

Rendundancy ("lots of copies") is the key principle for ensuring the safety and accessibility of curated assets and the availability of services built around those assets; abundant characterization ("lots of description") is the key principle for exposing content to user communities in useful contexts; responsiveness to the needs of users ("lots of services") is the key principle for ensuring the widespread integration of curated assets into the research, teaching, and learning activities of the University; and the multiplier effect of the creative use and re-use of curated assets ("lots of uses") is the key principle for enriching that discourse.

Work on the micro-services will progress in a series of incremental development waves, with siginficant milestones at the completion of the second, fourth, and sixth waves (see Table 10).

First wave	*Second wave* ✓	*Third wave*	*Fourth wave* ✓	*Fifth wave*	*Sixth wave* ✓
Identity	Inventory	Index	Search	Notification	Annotation
Storage	Ingest	Fixity	Replication	Characterization	Transformation
Object and collection modeling			Authentication and authorization		
Policy and business model development					

Table 10 – Micro-service development plan

NOTE While this table presents a *logival* view of high-level priorities and dependencies, it should not be interpreted literally as a *schedule*. The 12 micro-services vary widely in scope and complexity and this variance will be reflected in the time necessary for individual service implementation. Additionally, parallelism and pipelining beyond what is represented above may be possible subject to the overall schedule and availability of appropriate resources.

References

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

[Abrams] Stephen Abrams, Evan Owens, and Tom Cramer, "'What? So what?': The next-generation JHOVE2 architecture for format-aware characterization," *The Fifth International*

Conference on Preservation of Digital Objects, London, September 29-30, 2008.

[Abrams] Stephen Abrams, Patricia Cruse, and John Kunze, "Preservation is not a place," *International Journal of Digital Preservation* 4:1 (2009): 22-33 <http://www.ijdc.net/index.php/ijdc/article/viewFile/99/74>.

[Foundations] UC3, *UC3 Curation Foundations*, 2010.

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

[Fielding] Roy Fielding and Richard Taylor, "Principled design of the modern web architecture," *ACM Transactions on Internet Technology* 2:2 (May 2002): 115-150 <doi:10.1145/514183.514185>.

[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>.

[Higgins] Sarah Higgins, "The DCC curation lifecycle model," *International Journal of Digital Curation* 1:3 (2008): 134-140 <http://www.ijdc.net/index.php/ijdc/article/view/69/48>.

[Liegl] Philipp Liegl, "The strategic impact of service oriented architectures," *Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS '07)*, Tucson, March 25-29, 2007.

[O'Reilly] Tim O'Reilly, *Web 2.0: Design Patterns and Business Models for the Next Generation of Software*, March 30, 2005 <http://oreilly.com/web2/archive/what-is-web-20.html>.

[Rusbridge] Chris Rusbridge, "'Digital preservation' term considered harmful?" *Digital Curation Blog*, July 29, 2008 <http://digitalcuration.blogspot.com/2008/07/digital-preservation-term-considered.html>.