A Framework for Object Preservation in Digital Repositories

Andrew Boyko, Babak Hamidzadeh, Justin Littman; Office of Strategic Initiatives, Library of Congress; Washington, DC, USA

Abstract

The term "digital repository" is used in disparate contexts, in both in a formal sense to describe a complex and complete preservation system and its policies, and in a more intuitive but less clearly defined sense, often implying simply a robust storage system, or a content management system. Lack of formalism in use of this term can impede development of formal requirements for building specific repository systems and, consequently, the widespread use of such systems. On the other hand, the complexity of a formal, system-wide view can impede understanding and implementation in many environments.

To attempt to reconcile these concerns, we present a pragmatic definition and scope for digital repositories (systems whose primary function is the long-term preservation of digital objects). Taking a "bottom-up" approach that builds abstractions on top of reliable storage systems, we establish a minimal general vocabulary for the expression of digital objects, independent of policy, in the form of a low-level content model. We show that the "primitives" supplied by this vocabulary can be used to express the particular content and metadata models for several disparate repository case studies.

Introduction

In complement with the comprehensive top-down system perspective of the OAIS model [6] for a repository system, Rosenthal [5] discusses the characteristics of a reliable storage system, taking a "bottom-up" view of repository systems that describes storage systems as the foundational layer of the repository. Proceeding in this bottom-up path from a storage system at the lowest layer, it is reasonable to define requirements for a bit-preservation system, an abstraction atop a set of robust, replicated storage systems meeting local needs. By bounding the scope of this abstracted bit layer, charging it with ensuring long-term fixity of bit-streams whose meaning is opaque to that layer, we can separate those concerns from the requirements of a system to manage the intellectual matter of digital objects (as distinct from the bits that compose the objects).

The requirements for such an object preservation system can then assume that a bit preservation system underlying it is trusted to meet the requirements of fixity and security of that repository. An object preservation system can remain conceptually unaware of the characteristics of the bit preservation system's storage layers, such as the implementation technology, replication or validation policies, or even the physical location and ownership of the storage.

Definition

We define a digital preservation system as a set of layered dependent systems, namely:

- storage systems,
- a bit preservation system, and
- an object preservation system.

These systems provide layers of technical infrastructure, atop which repository instances may be implemented. A repository instance comprises:

- named sets of deposited digital objects,
- a logical model that defines the digital objects held by the repository instance,
- preservation policies and plans,
- designated sets of users, and
- policies and guidelines for the deposit and use of the objects by each set of users.

A repository instance is a unit of curation and preservation, and not of access or presentation; an exhibit may collect and display objects from multiple repository instances.

The relationships between a preservation system, the repository instances implemented within the system, and the logical models defining those instances may be compared to the case of a database system. Given a general-purpose database engine, atop which specific databases are implemented, the particular form of the content in these databases is defined by a set of schemas, which describe the logical structure of the data. Similarly, the repository system is a general-purpose preservation tool, atop which specific instances are implemented; the forms of the digital objects managed by an instance are completely expressed by that instance's logical model.

An object preservation system may be scoped as providing a layer of abstraction over the bit-preservation system to manage and preserve complex objects over time. In this layer, the content is given identity, structure, description, and interrelationship, in order to support the needs of preservation. An object preservation system provides a language in which a logical model for a given repository instance can be expressed. As such, it may be scoped to be policy-neutral, relying on higher-level systems to provide policy and behavior.

In this bottom-up approach to scoping repository system components, the specific nature of the layers above the object preservation system has been left unstated; it is reasonable to assume that the users of the object preservation system will include both humans (curators, content users) and systems (ingestion and access systems, as well as systems implementing local repository policy and structure atop this general framework).

In this paper we present a set of basic primitives for constructing logical models that characterize and interrelate digital objects within a repository instance. We also demonstrate the applicability of the logical model primitives with two case studies.

Logical Model

The essential notion of digital objects as collected, described, and identified sets of bit-streams [1] has been enhanced by the addition of inter-object relations as primary aspects of a digital library collection [3]. Using these general concepts as background, we propose that the abstractions necessary to provide a means of expressing digital objects atop a robust bit-storage layer consist of identification and relation, and the familiar forms of common

digital object schemes can be expressed by combining these primitives. We delineate here the characteristics of a notional object preservation system beginning from this bottom-up perspective.

We may define an *object* in the context of an object preservation system simply as a unique, stable identifier that may either be associated with a bit-stream of fixed content, or with no content at all, in which case the object simply represents a concept to be related to other objects in the repository instance. The identifier bears no meaning to the object preservation system itself. The object preservation system maintains the identifier, and the binding between the identifier and the content, stably and for the long term, just as the bit preservation maintains the content itself reliably in the long term.

With similar minimality, a *relationship* between two objects in a particular repository instance may be defined as an expression of some directed conceptual connection, analogous to an RDF statement [4]. Given two objects *a* and *b*, a particular relationship *R(a,b)* expresses the sentence "*a R b*" (as grammatical subject, verb, and object). A relationship is not an object; it is simply a fact managed by the object preservation system, and again is assumed to be reliably recorded for the long term, presumably by the bit preservation system.

Beginning with the abilities to create and identify objects, to associate them with stable bit-streams, and to relate them to one another, we consider the task of defining a core set of relationships sufficient to express the breadth and complexity of rich digital content. Using the above syntax *R(a,b)* to depict these relationships, a reasonable set might comprise:

- Descriptive relation *D: D(a,b)* indicates that the content of object *b* describes the content of object *a*.
- Typing relation *T: T(a,b)* indicates that object *b* represents the type of object *a*.
- Versioning relation *V: V(a,b)* indicates that the content of object *b* is a newer version of the content of object *a*.
- Containment relation *C*: *C(a,b)* indicates that object *a* contains object *b*, thus supporting complex structure as well as aggregation.
- Fragment relation *F: F(a,b)* indicates that the content of object *b* is logically a fragment, or a part of the whole, of the content of object *a*.

This core set is not intended to be closed, nor can it be considered provably complete, but these relationships express particular concepts that apply broadly to content across a variety of domains. In order to extend the set of relationships to express domain-specific concepts, an object preservation system may provide a means for extending this set:

- Local relationships: an relationship *R(a,b)*, defined within a particular repository instance, indicates a fact that the object preservation system can record about objects *a* and *b*.

Given the ability to create simple objects and relate them in these specific ways, we believe we are able to build a vocabulary of primitives that can be used to represent real, existing content sets of varied structure, format, and policy, without requiring transformation, or decomposition of the content being stored into a particular normalized format. By remaining neutral to the form of the content at the bit level, and providing an overlay of intellectual structure atop the content, we increase the likelihood of adoption, and ease the implementation, of such an object preservation system in a new repository instance.

Functional Areas

To clarify the implications of the scoping of object preservation systems as described above, we describe in more detail what typical repository services can be implemented in terms of the above primitives.

Identification

By ensuring that the identifiers used by the object preservation system are opaque to it, the local identifier policy for a given repository instance can be supported, whether the identifiers bear meaning to the repository's users or not. For example, a particular repository instance may desire to assign identifiers with some semantic meaning for certain types of object, while automatically assigning identifiers to other objects, perhaps those considered descriptive or otherwise auxiliary. Such policy decisions may be expressed at this layer, but at the same time are not meaningful to it.

Object contents

The object model primitives presented here do not prescribe, or even point toward, a particular approach for structuring or storing the content in a repository instance. Rather than encouraging any particular policy, these primitives merely provide a common way to describe the internal structure of a digital collection, at whatever level of content understanding is deemed necessary for curation.

The conceptual contents of an object may be any combination of components of digital data, or other objects within the instance; thus, the definition of object intentionally encompasses both a single intellectual work and complex, nested collections and aggregations.

Descriptive relations

By generalizing description as an object-to-object relationship without obliging a particular form or structure, this model equally supports unstructured descriptive elements, such as the binding of arbitrary facts to a given object, and highly structured metadata formats, such as a PREMIS [7] document containing preservation metadata for an object. Enforcement of a particular descriptive policy or format is external to the scope of the object preservation system as we describe it here.

In this model, objects that describe others via this relationship are not inherently different than objects bearing primary content.; for example, descriptions may themselves be described. The object preservation system records the fact that a given object may represent metadata about another, without otherwise treating it differently.

These relationships may also be used to relate content to *contextual* or *behavioral* information, which might specify actions and functions that can be applied to an object, in order to understand the object and to interact with it in a useful manner.

Typing

Providing the ability to relate objects to an object defining a type is analogous to description, but specifically provides a way to document the kind of multiple objects in accord with local

repository instance policy. By providing the type relation primitive, the object preservation system gains the ability to express, in essence, metadata at the level of the repository instance, describing the types and structure of the content therein, as well as allowing types themselves to be related or described as needed.

The meaning of "type" in this context is unstated, and applies with equal validity to digital formats, to metadata container formats, or merely to general intellectual categories; in this, as in other cases, the object preservation system provides a means of expression without requiring a particular policy.

Versioning

By providing versioning primitives, the change in a piece of content over time can be managed, without obliging a bit preservation system to allow updates to content. With this primitive, the historical sequence representing an object's lifespan can be expressed as required by local policy. A repository instance may thus choose to represent an object's change over time since creation as individual objects, or perhaps to record only the previous version, or to allow no change at all.

Containment

Expressing containment as an object-to-object relationship supports a range of content models, allowing the expression of a containment structure (analogous to directories or folders in a bit preservation system, but less rigidly expressive) while not precluding an instance from preferring to express containment internally to the object content itself (as in the case of a structured metadata document, or a ZIP-format archive). Expressed as a relationship, containment can thus support objects representing arbitrarily nested containers, each container in turn able to be described, typed, or versioned in the same way as any other object.

Fragments (Part of whole)

Fragment relations allow an object to refer to a portion of another object, with the assumption that the contents of the portion in question are referred to, rather than replicated entirely. An object preservation system is thus able to represent objects whose content is physically a sub-component of the content of other existing objects, without requiring objects to be disassembled to the level of the desired fragments. How to express a reference to a portion of an object (e.g. a region of an image, or a page of a document, or a portion of an audio recording) is necessarily dependent on the format of the object, and cannot be expressed solely in the primitives of the object preservation system. However, typing relations can be used to bind a fragment object to appropriate format or algorithm descriptions, so that the actual extraction of the fragment's data can be performed at dissemination time, by a system with understanding of the particular expression.

For example, a fragment object representing a particular region of a large image might be expressed, in its content, in terms of a set of coordinates within that image. For an object preservation system to be able to actually extract the relevant region of the image based on that fragment's description as coordinates would require some indication of the type of the fragment object, as well as requiring a client with specific understanding of that type to also understand how to interpret the fragment's contents in order to extract the bits representing the desired region from the original image file.

Local relations

Local relations provide a way to define, in the context of a particular repository instance, a relationship that is descriptive of a state between two objects, without being logically part of either object. This extends the expressiveness of the core set of relations to domain-specific relations, supporting fine semantic distinctions.

While, for simplicity, the relation language is limited to unidirectional binary relations, more complex relationships can be expressed as objects themselves, and related to their participating objects as required.

Case Studies

In this section, we describe the logical models for two digital collections, in order to demonstrate the generality and sufficiency of the concepts and definitions discussed above.

Prokudin-Gorskii Photo Archive

The Library of Congress's Prokudin-Gorskii photographic collection [8] contains full-color digitally composited images, created in the digital domain from scans of early 20th century glass plate negatives. This collection presents a more complex scenario than the general case of a collection of digital images, for each composite image is generated from superimposed scans of a plate containing three filtered images, corresponding to the red, green, and blue channels of the composite.

The digital representation of an image in this collection might be represented as discrete files at the bit storage layer, corresponding to the components:

- Master images for each color channel, produced by scanning the negative.
- The digitally composited full-color image.
- Derivatives (lower-resolution, compressed renditions) of the composited image, such as a thumbnail image for access.

Given these artifacts of interest, one possible use of the primitives would construct an object model containing, for a given image:

- Objects created to provide an identity for each of the image files stored in the bit preservation system (the scanned negatives, the composite, and the derivatives). In this example, we might identify these image objects as:
  $I_{red\text{-}channel\text{-}scan\text{-}master}$
  $I_{blue\text{-}channel\text{-}scan\text{-}master}$
  $I_{green\text{-}channel\text{-}scan\text{-}master}$
  $I_{digital\text{-}composite\text{-}master}$
  $I_{digital\text{-}composite\text{-}derivative\text{-}thumbnail}$
- Local relations declared to represent the derivative relationship between the composite master and the derivatives produced from it. Expressed in the syntax given earlier:
  $is\text{-}derived\text{-}from(I_{composite\text{-}master}, I_{composite\text{-}derivative\text{-}thumbnail})$
- Local relations defined to represent the relationship between the monochromatic color channel images and the composited full-color work:
  $is\text{-}composited\text{-}from(I_{composite\text{-}master}, I_{red\text{-}channel\text{-}scan\text{-}master})$
  $is\text{-}composited\text{-}from(I_{composite\text{-}master}, I_{green\text{-}channel\text{-}scan\text{-}master})$
  $is\text{-}composited\text{-}from(I_{composite\text{-}master}, I_{blue\text{-}channel\text{-}scan\text{-}master})$
- Objects bearing technical metadata for each image file, and a descriptive relation binding each metadata object to the

appropriate image object. With the technical metadata for a given composite master identified by $TM_{composite\text{-}master}$, the core descriptive relation $D(a,b)$ is used:

$D(I_{composite\text{-}master}, TM_{composite\text{-}master})$

which can be read as:

The object $TM_{composite\text{-}master}$ *describes object* $I_{composite\text{-}master}$.

- A "primary" object representing the actual physical photographic negative, with no bit-stream content of its own, and relationships between this primary object and the various objects described above that represent it partially or completely. Identifying this primary object as $I_{physical\text{-}negative}$, these relationships may be expressed as, for example:

  $is\text{-}represented\text{-}by(I_{physical\text{-}negative}, I_{composite\text{-}master})$

  How to define the particular local relationships of a repository instance is curatorial policy; the object preservation system is agnostic regarding the semantics of these relationships.

These objects and relations thus convey a reasonable approximation of the intellectual relations between the physical work and the digital representations, and are stated as a conceptual overlay atop the digital data that make up the collection.

Web Archive

The Library of Congress's collections of content harvested from the World Wide Web consist of approximately forty terabytes of material, comprising content harvested within the institution, content harvested by contractors, and donated content. This content varies widely in essentially every characteristic – content type, original source, complexity – other than having been published on the Web at some point. Each item in the collection was delivered in a Hypertext Transfer Protocol (HTTP) response, consisting of two parts, an HTTP-specific preamble (header information) and a message body.

A Web harvesting tool might, when collecting a site, store each harvested resource in a file, containing the complete HTTP message retrieved from the Web server. A mapping of this content to the object system primitives would include:

- An object providing identification for the HTTP message that was received from the Web server and stored in the bit preservation system.
- Objects representing the HTTP header preamble and the message body. A fragment relation between these objects would relate them to the object containing the HTTP message; rather than repeating the content of the message, the content of these objects would express a reference to the relevant portion of the HTTP message. Note that the language in which this reference would be expressed would depend on the kind of content, and thus would require logic external to a generic object preservation system to interpret when the fragment is to be retrieved.

These individual items, each originally addressable by its own URL when retrieved from the Web, are highly dependent on one another for actual use. The Web depends on a rich variety of expressions of interrelationships between content; without expressing these in the repository, the content is not completely usable. A Web page is actually composed of potentially dozens of individual items, including a containing HTML document and logically embedded but physically distinct images, subdocuments, and media objects. The object preservation system can express these relationships, in order to make these dependencies explicit.

The constitution of a page, a single intellectual item in the user's view but made up in actuality of many repository objects, might be expressed by defining a local relationship:

$is\text{-}embedded\text{-}in(O_{embedded\text{-}item}, O_{embedding\text{-}item})$

which expresses the logical sentence:

$O_{embedded\text{-}item}$ *is embedded in* $O_{embedding\text{-}item}$.

or, more generally,

$O_{embedding\text{-}item}$ *depends on* $O_{embedded\text{-}item}$ *in order to be usable*.

Given an HTML document, and an image (such as a logo graphic, or an advertisement), this relationship definition allows us to express the nature of their dependency. We choose these local relationships, instead of expressing this dependency as the core containment relationship, because the nature of the embedding relationship is a referential one; an HTML document may embed any other item by naming its URL, without having secured any intellectual rights to that object, nor any commitment that the named item actually exists. Thus the fact of an actual embedding relationship between two particular items in a repository is a separate piece of information from any information within the items themselves, which is precisely the reason for having distinguished relationships as inherently external to objects.

Perhaps even more essential to the nature of the Web is a linking relationship between web items; links provide the Web with its entire form. While the actual expression of a link from one document to another is similar to embedding (one object references the name of another), the meaning is different, and so we define a second local relationship, $links\text{-}to(O_{source\text{-}item}, O_{destination\text{-}item})$, which expresses the sentence:

$O_{source\text{-}item}$ *contains a link to* $O_{destination\text{-}item}$.

As with embedding, the information contained in a particular *links-to* relationship expressed between two items in the repository is distinct from the mere fact that a particular object contains a link; it indicates something specific about the items of our collection, rather than what the producer wished to express.

As an additional complication, most large-scale Web archiving aggregates collected content in platform-neutral bulk container files, typically in the ARC format [1]. In this approach, there are no individual files for each HTTP message; messages are compressed and concatenated in the container. The object preservation system can continue to represent Web archive content stored in this form at the bit preservation level, without requiring the unpacking of these containers. The only necessary modification to the conceptual object model described here is to represent the individual HTTP messages not as objects directly identifying a file in the bit preservation layer, but rather as fragments, related to the object representing the ARC container. The object representing the message would then refer to the location of the message within that container, rather than holding the content directly.

Conclusion

By treating a repository system as a stack of layered services, a simple object preservation layer can be defined that provides:

- Independence from storage hardware and policies, considering them a distinct layer "below" in the stack.

- A logical model, based on simple objects and a small set of relations, which provides sufficient expressiveness to neutrally support disparate content models and policies.
- Independence from content policy and meaning, considering that a distinct layer "above" in the stack.

This paper discusses a logical model of this object preservation system, and its primitives, in more detail. It also provides examples that demonstrate how the model and its primitives can be applied to several existing digital collections. Based on this conceptual model, we are developing requirements for a system to provide this object preservation function, in parallel with work on bit preservation policy, and on the content policies for repository instances implemented in terms of this object preservation system.

References

[1] M. Burner and B. Kahle, "Arc File Format", http://www.archive.org/web/researcher/ArcFileFormat.php, 1996

[2] R. Kahn and R. Wilensky, *"A Framework for Distributed Digital Object Services"*, May 1995, http://www.cnri.reston.va.us/k-w.html

[3] C. Lagoze *et al*, *"What Is a Digital Library Anymore, Anyway? Beyond Search and Access in the NSDL"*, D-Lib Magazine, November 2005

[4] O. Lassila and R.R. Swick. Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation, W3C, February 1999.

[5] D.S.H. Rosenthal *et al*, "Requirements for Digital Preservation Systems", D-Lib Magazine, November 2005

[6] *Reference Model for an Open Archival Information System (OAIS).* Blue Book. Issue 1. January 2002

[7] Preservation Metadata Maintenance Activity (PREMIS), http://www.loc.gov/standards/premis/, 2006

[8] "The Empire That Was Russia: The Prokudin Gorskii Photographic Record Recreated", http://www.loc.gov/exhibits/empire/, 2003

Author Biographies

Andy Boyko is a digital media project coordinator at the Library of Congress, working on digital repositories. His work encompasses general repository and storage systems, and specific content repositories including electronic journals and Web archives. He received a B.S. in computer engineering from Virginia Polytechnic Institute and State University in 1994.

Babak Hamidzadeh received his Ph.D. degree in Computer Science and Engineering from the University of Minnesota in 1993. From 1993 to 1996 he was an Assistant Professor of Computer Science and Computer Engineering at the Hong Kong University of Science and Technology. He then joined the Electrical and Computer Engineering Department of the University of British Columbia and became an Associate Professor in that department in 2000. From 2002 to 2004, he was Senior Manager of Information Management and Collaborative Technologies Research at the Mathematics and Computing Technology Center of the Boeing Company. Presently, he is Chief Architect and Senior Advisor on Technology in the Office of Strategic Initiatives of the Library of Congress.

Justin Littman is a digital media project coordinator at the Library of Congress, working on digital repositories. His work focuses on digital object representation and validation, including contributions to various projects such as the National Digital Newspaper Project. He received a B.A. in philosophy and economics from Amherst College in 1995 and an M.L.I.S. from the University of Denver in 2002.