Data Dictionary

for Preservation Metadata

Contents:

Acknowledgments
Introduction
The PREMIS Data Model
The PREMIS Data Dictionary version 1.0
Examples
Special Topics
Methodology
Implementation Considerations
Glossary

Final Report of the PREMIS Working Group
May 2005

Preservation Metadata: Implementation Strategies (PREMIS)
A working group jointly sponsored by OCLC and RLG

OCLC Online Computer Library Center, Inc.
6565 Frantz Road, Dublin, Ohio, 43017-3395 USA

RLG
2029 Stierlin Court, Suite 100, Mountain View, California, 94043-4684 USA

CONTENTS

Acknowledgments iv
PREMIS Web Sites and E-mail vi
Introduction vii
- Background viii
- Core preservation metadata elements and the Data Dictionary viii

1. The PREMIS Data Model 1-1
- Objects 1-2
- Intellectual Entities and Objects 1-4
- Events 1-5
- Agents 1-7
- Rights 1-7
- Relationships 1-8
- The 1:1 principle 1-9

2. The PREMIS Data Dictionary Version 1.0 2-1
- Limits to the scope of the Data Dictionary 2-3
- Object entity 2-5
- Event entity 2-74
- Agent entity 2-85
- Rights entity 2-88

3. Examples 3-1
- Example 1: Microsoft Word document complete in one file 3-2
- Example 2: ETD 3-12
- Example 3: Newspaper complex object, *Los Angeles Times* 3-34
- Example 4: Web site 3-51
- Example 5: Digital signature 3-79
- Example 6: Photograph 3-81

4. Special topics 4-1
- Format information 4-1
- Environment 4.2
- Object characteristics and composition level: the "onion" model 4-4
- Fixity, integrity, authenticity 4-5
- Digital signatures 4-6
- Non-core metadata 4-9

5. Methodology 5-1
6. Implementation considerations 6-1
- Premis conformance 6-1
- Implementation of the data model 6-2
- Storing metadata 6-3
- Supplying metadata values 6-3
- Preservation metadata for Web sites and Web pages 6-4

7. Glossary 7-1
Notes 8-1

ACKNOWLEDGMENTS

PREMIS members

Priscilla Caplan, Florida Center for Library Automation, *co-chair*
Rebecca Guenther, Library of Congress, *co-chair*
Robin Dale, *RLG liaison*
Brian Lavoie, *OCLC liaison*
George Barnum, U.S. Government Printing Office
Charles Blair, University of Chicago
Olaf Brandt, Göttingen State and University Library
Mikki Carpenter, Museum of Modern Art
Adam Farquhar, British Library
David Gewirtz, Yale University
Keith Glavash, MIT/DSpace
Andrea Goethals, Florida Center for Library Automation
Cathy Hartman, University of North Texas
Helen Hodgart, British Library
Nancy Hoebelheinrich, Stanford University
Roger Howard, J. Paul Getty Museum
Sally Hubbard, Getty Research Institute
Mela Kircher, OCLC
John Kunze, California Digital Library
Vicky McCargar, Los Angeles Times
Jerome McDonough, New York University/METS
Evan Owens, Ithaka-Electronic Archiving Initiative
Erin Rhodes, U.S. National Archives and Records Administration
Madi Solomon, Walt Disney Corporation
Angela Spinazze, ATSPIN Consulting
Stefan Strathmann, Göttingen State and University Library
Günter Waibel, RLG
Lisa Weber, U.S. National Archives and Records Administration
Robin Wendler, Harvard University
Hilde van Wijngaarden, National Library of the Netherlands
Andrew Wilson, National Archives of Australia and British Library
Deborah Woodyard-Robinson, British Library and Woodyard-Robinson Holdings Ltd.

Advisory committee

Howard Besser, University of California, Los Angeles
Liz Bishoff, OCLC
Gerard Clifton, National Library of Australia
Gail Hodge, CENDI
Steve Knight, National Library of New Zealand
Maggie Jones, Digital Preservation Coalition
Nancy McGovern, Cornell University
Cliff Morgan, John Wiley & Sons, Ltd.
John Perkins, CIMI Consortium
Richard Rinehart, University of California, Berkeley

Special thanks

These individuals contributed their expertise to help with special topics and/or comment on drafts:

Stephen Abrams, Harvard University
Reinhard Altenhöner, Die Deutsche Bibliothek
Caroline Arms, Library of Congress
Kevin Bradley, National Library of Australia
Thomas Fischer, Lower Saxony State and University Library, Göttingen
Carl Fleischhauer, Library of Congress
Mahnaz Ghaznavi, Getty Research Institute/InterPARES
Corey Harper, University of Oregon
Lori Lindberg, SLIS Cal State San Jose/InterPARES
Justin Littman, Library of Congress
Sean Martin, British Library
Quyen Nguyen, U.S. National Archives and Records Administration
Tobias Steinke, Die Deutsche Bibliothek
Robert Tansley, Hewlett-Packard
Andrew Waugh, Public Record Office, Victoria

PREMIS WEB SITES AND E-MAIL

PREMIS working group Web site: www.oclc.org/research/projects/pmwg/.

PREMIS maintenance activity Web site: www.loc.gov/standards/premis/.

Please send comments and questions to premis@loc.gov.

INTRODUCTION

In 2003 OCLC and RLG established Preservation Metadata: Implementation Strategies (PREMIS), an international working group. This report and the PREMIS Data Dictionary version 1.0 are the culmination of nearly two years of effort by PREMIS members.

The Data Dictionary defines and describes an implementable set of core preservation metadata with broad applicability to digital preservation repositories. This report is intended to put the Data Dictionary into context, explain the underlying assumptions and data model, and provide additional information about the meaning and use of semantic units defined in the Data Dictionary.

The charge of the PREMIS working group was to:

- define an implementable set of "core" preservation metadata elements, with broad applicability within the digital preservation community;
- draft a Data Dictionary to support the core preservation metadata element set;
- examine and evaluate alternative strategies for the encoding, storage, and management of preservation metadata within a digital preservation system, as well as for the exchange of preservation metadata among systems;
- conduct pilot programs for testing the group's recommendations and best practices in a variety of systems settings; and
- explore opportunities for the cooperative creation and sharing of preservation metadata.

A draft of this report and the Data Dictionary were completed in February 2005 and circulated to the PREMIS Advisory Committee and a small number of other invited reviewers. The working group received a great deal of valuable feedback from this initial review period, and spent considerable time considering each comment and making revisions. Both documents benefited immensely from this review.

With this release of the PREMIS Data Dictionary version 1.0, immediate next steps will likely focus on implementation and interoperability. Operating repositories can use PREMIS as a checklist against which to compare their own preservation metadata specifications. Repositories in development can serve as testbeds for implementing PREMIS-conformant semantics and feed their experience into future revisions of the Data Dictionary. XML bindings for the Data Dictionary are being developed to represent PREMIS-conformant metadata in the exchange of archival information packages between preservation repositories.

The working group wants to stress that the Data Dictionary is not intended to be fixed and final but to provide a starting point for improvements and enhancements based on community experience and feedback. A mechanism is being established for the ongoing maintenance and oversight of the PREMIS Data Dictionary and associated XML schemas; see www.loc.gov/standards/premis/. The PREMIS Web site at www.oclc.org/research/projects/pmwg/ should be consulted for current information about ongoing activities.

Introduction

Background

PREMIS was established to build on the earlier work of another initiative sponsored by OCLC and RLG, the Preservation Metadata Framework working group. In 2001–2002 that group outlined the types of information that should be associated with an archived digital object. Their report, *A Metadata Framework to Support the Preservation of Digital Objects* (the *Framework*), proposed a list of prototype metadata elements.[1] However, additional work was needed to make these prototype elements implementable. The PREMIS working group aimed to take the previous group's work a step further: to develop a Data Dictionary of core metadata elements to be applied to archived objects, give guidance on the implementation of that metadata element set in preservation systems, and suggest best practice for populating those elements.

As PREMIS had a practical rather than theoretical focus, members were sought from institutions known to be running or developing preservation repository systems within the cultural heritage and information industry sectors. Diverse perspectives were also sought. The working group consisted of representatives from academic and national libraries, museums, archives, government, and commercial enterprises in six different countries. In addition, PREMIS called upon an international advisory committee of experts to review progress.

To accomplish as much of the charge as possible in a reasonable timeframe, the working group divided into two subgroups. The Implementation Strategies Subgroup examined various strategies for encoding, storing, and managing preservation metadata within digital preservation systems. The Core Elements Subgroup took responsibility for selecting the core preservation metadata elements and drafting the Data Dictionary. Both subgroups conducted their work almost entirely by weekly conference calls. The Core Elements Subgroup also held two face-to-face meetings.

To find out how preservation repositories were actually implementing preservation metadata, in November 2003 the Implementation Strategies Subgroup surveyed about 70 organizations thought to be active in or interested in digital preservation. The survey provided an opportunity to explore the state of the art in digital preservation generally, and questions were drafted to elicit information about policies, governance and funding, system architecture, and preservation strategies, as well as metadata practices. The subgroup contacted 16 of 48 respondents by telephone for more in-depth interviews. In December 2004 the PREMIS working group published its report based on the survey of digital repositories, *Implementing Preservation Repositories for Digital Materials: Current Practice and Emerging Trends in the Cultural Heritage Community* (the *Implementation Survey Report*).[2]

The Implementation Strategies Subgroup had also been charged with developing pilot implementations of the PREMIS core elements. This will likely be addressed as a follow-on activity to PREMIS's work. Testing of the Data Dictionary will likely require significant effort and independent funding and will benefit from being organized as a separate activity.

Core preservation metadata elements and the Data Dictionary

The Core Elements Subgroup developed the Data Dictionary of core elements needed to support digital preservation, including implementation details such as repeatability, obligation, and

examples. The Implementation Strategies Subgroup annotated the Data Dictionary with notes about the creation and use of the metadata elements.

Both the earlier *Framework* and the PREMIS Data Dictionary build on the Open Archival Information System (OAIS) reference model (ISO14721).[3] The OAIS information model provides a conceptual foundation by providing a taxonomy of information objects and packages for archived objects and the structure of their associated metadata. The *Framework* can be viewed as an elaboration of the OAIS information model, explicated through the mapping of preservation metadata to that conceptual structure. The PREMIS work can be viewed as a translation of the *Framework* into a set of implementable semantic units in the Data Dictionary. However, it should be noted that PREMIS and OAIS use some terminology differently, as noted in the Glossary. Differences usually reflect the fact that PREMIS semantic units require more specificity than the OAIS definitions provide, which is to be expected when moving from a conceptual framework to an implementation.

Drafting the Data Dictionary required agreement on working definitions of "preservation metadata," "core," and "implementable." These working definitions provided criteria for evaluating potential semantic units.

PREMIS defines "preservation metadata" as *the information a repository uses to support the digital preservation process.* Specifically, the group looked at metadata supporting the functions of maintaining viability, renderability, understandability, authenticity, and identity in a preservation context. Preservation metadata thus spans a number of the categories typically used to differentiate types of metadata: administrative (including rights and permissions), technical, and structural. Particular attention was paid to the documentation of digital provenance (the history of an object) and to the documentation of relationships, especially relationships among different objects within the preservation repository.

The group considered a number of definitions of "core." In one view, core describes any metadata absolutely required under any circumstances. In another, core means that metadata is applicable to any type of repository implementing any type of preservation. PREMIS uses this practical definition: *things that most working preservation repositories are likely to need to know in order to support digital preservation.* The words "most" and "likely" were chosen deliberately. Core does not necessarily mean mandatory, and some semantic units were designated as optional when exceptional cases were apparent.

The idea of "implementability" also generated some discussion. Most preservation repositories will be dealing with large quantities of data. Therefore, a key factor in the implementability of preservation metadata is whether the values can be automatically supplied and automatically used by the repository. Whenever possible the group defined elements that do not require human intervention to supply or analyze. For example, coded values from an authority list are preferred over textual descriptions.

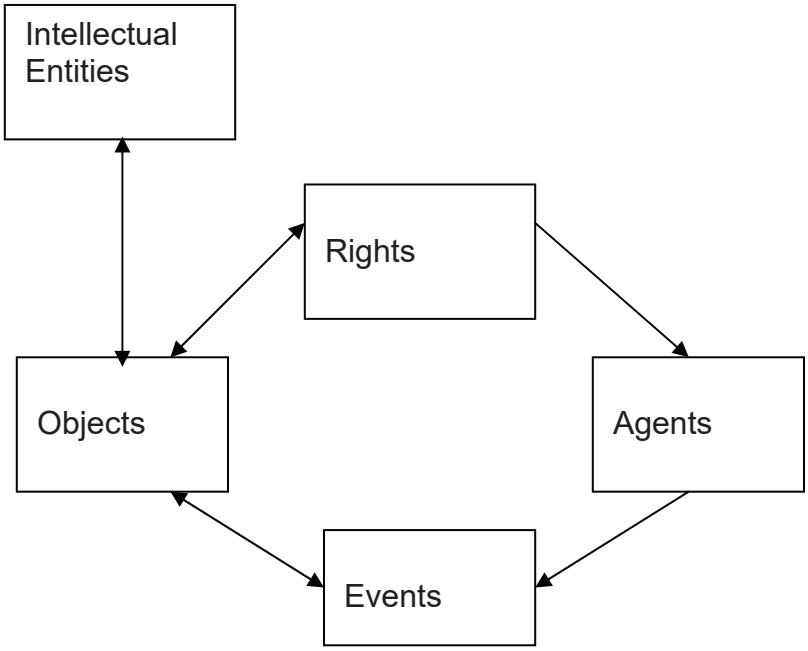
The group decided that the Data Dictionary should be wholly implementation independent. That is, the core elements define information that a repository needs to know, regardless of how, or even whether, that information is stored. For instance, for a given identifier to be usable, it is

necessary to know the identifier scheme and the namespace in which it is unique. If a particular repository uses only one type of identifier, the repository would not need to record the scheme in association with each object. The repository would, however, need to know this information and to be able to supply it when exchanging metadata with other repositories. Because of the emphasis on the need to know rather than the need to record or represent in any particular way, the group preferred to use the term "semantic unit" rather than "metadata element." The Data Dictionary names and describes semantic units, the properties of entities.

An implementable metadata scheme needs to define each semantic unit as rigorously as possible and relate it to the type of entity it describes. This was a guiding principle for the group as it compiled the Data Dictionary.

1. THE PREMIS DATA MODEL

To facilitate the logical organization of the PREMIS metadata elements, the group developed a simple model of five types of entities involved in digital preservation activities: Intellectual Entities, Objects, Events, Rights, and Agents.[4] In the data model diagram, entities are drawn as boxes while the relationships between them are drawn as lines. The direction of the arrow shows the direction of the relationship link defined in the Data Dictionary; for example, the arrow from Rights to Agents means the metadata defined for Rights includes semantic units to identify the related agent(s). A double-headed arrow means reciprocal links are defined.



An **Object**, or Digital Object, is a discrete unit of information in digital form.[5]

An **Intellectual Entity** is a coherent set of content that is reasonably described as a unit, for example, a particular book, map, photograph, or database. An Intellectual Entity can include other Intellectual Entities; for example, a Web site can include a Web page, a Web page can include a photograph. An Intellectual Entity may have one or more digital representations.

An **Event** is an action that involves at least one object or agent known to the preservation repository.

An **Agent** is a person, organization, or software program associated with preservation events in the life of an object.

Rights, or Rights Statements, are assertions of one or more rights or permissions pertaining to an object and/or agent.

A **relationship** is a statement of association between instances of entities. "Relationship" can be interpreted broadly or narrowly, and any relationship fact can be expressed in many different ways. That object A is of format B could be considered a relationship between A and B. The PREMIS model, however, treats format B as a property of object A. PREMIS reserves "relationship" for associations between two or more Object entities or between entities of different types, such as an Object and an Agent.

Semantic units are the properties of an entity (the thing being described). Semantic units have values; for example, the semantic unit *size* is a property of an Object entity. For a particular object the value of *size* might be "843200004."

In most cases a particular semantic unit is clearly a property of only one type of entity. The size of an object is clearly a property of the Object entity. In some cases a semantic unit applies equally to two or more different types of entity. For example, events have outcomes. If a migration event creates a file that has lost some important feature, the loss of that feature might be considered a sort of outcome (and so a property of the Event entity) or it might be considered an attribute of the new file (and so a property of the Object entity). When a semantic unit applies equally to different types of entities, the semantic unit is associated with only one type of entity in the Data Dictionary. The model relies upon links between the different entities to make these relationships clear. In the example above, the loss of the feature is treated as a detailed outcome of the Event, where the Event contains the identifier of the Object involved. What is important is that this association is arbitrary and is not meant to imply that a particular implementation is required.

In some cases a semantic unit is an umbrella or **container** that groups a set of related semantic units. For example, a semantic unit *identifier* groups the two semantic units *identifierType* and *identifierValue*. The grouped subunits are called **semantic components** of the semantic unit.

Objects

The Object entity has three subtypes: file, bitstream, and representation.

A **file** is a named and ordered sequence of bytes that is known by an operating system. A file can be zero or more bytes and has a file format, access permissions, and file system statistics such as size and last modification date.

A **bitstream** is contiguous or non-contiguous data within a file that has meaningful common properties for preservation purposes. A bitstream cannot be transformed into a standalone file without the addition of file structure (headers, etc.) and/or reformatting the bitstream to comply with some particular file format.

A **representation** is the set of files, including structural metadata, needed for a complete and reasonable rendition of an Intellectual Entity. For example, a journal article may be complete in one PDF file; this single file constitutes the representation. Another journal article may consist of one SGML file and two image files; these three files constitute the representation. A third article may be represented by one TIFF image for each of 12 pages plus an XML file of structural metadata showing the order of the pages; these 13 files constitute the representation.

Files, bitstreams, and filestreams

A file in the PREMIS data model is similar to the idea of a computer file in ordinary usage: a set of zero or more bytes known to an operating system. Files can be read, written, and copied. Files have names and formats.

A bitstream as defined in the PREMIS data model is a set of bits embedded within a file. This differs from common usage, where a bitstream could in theory span more than one file. A good example of a file with embedded bitstreams is a TIFF file containing two images.

According to the TIFF file format specification a TIFF file must contain a header containing some information about the file. It may then contain one or more images. In the PREMIS data model each of these images is a bitstream and can have properties such as identifiers, location, inhibitors, and detailed technical metadata (e.g., color space).

Some bitstreams have the same properties as files and some do not. The image embedded within the TIFF file clearly has properties different from the file itself. However, in another example, three TIFF files could be aggregated within a larger tar file. In this case the three TIFF files are also embedded bitstreams, but they have all the properties of TIFF files.

The PREMIS data model refines the definition of bitstream to include only an embedded bitstream that cannot be transformed into a standalone file without the addition of file structure (e.g., headers) or other reformatting to comply with some particular file format specification. Examples of these bitstreams include an image within a TIFF 6.0 file, audio data within a WAVE file, or graphics within a Microsoft Word file.

Some embedded bitstreams can be transformed into standalone files without adding any additional information, although a transformation process such as decompression, decryption, or decoding may have to be performed on the bitstream in the extraction process. Examples of these bitstreams include a TIFF within a tar file, or an encoded EPS within an XML file.

In the PREMIS data model these bitstreams are defined as “filestreams,” that is, true files embedded within larger files. Filestreams have all of the properties of files, while bitstreams do not. In the Data Dictionary, the column for “File” applies to both files and filestreams. The column for “Bitstream” applies to the subset of bitstreams that are not filestreams and that adhere to the stricter PREMIS definition of bitstream. The location (*contentLocation* in the Data Dictionary) of a file would normally be a location in storage; while the location of a filestream or bitstream would normally be the starting offset within the embedding file.

Representations

The goal of many preservation repositories is to maintain usable versions of intellectual entities over time. For an intellectual entity to be displayed, played, or otherwise made useable to a human, all of the files making up at least one version of that intellectual entity must be identified, stored, and maintained so that they can be assembled and rendered to a user at any given point. A representation is the set of files required to do this.

PREMIS chose the term "representation" to avoid the term "manifestation" as it is used in the *Functional Requirements for Bibliographic Records* (FRBR).[6] In FRBR a manifestation entity is "all the physical objects that bear the same characteristics in respect to both intellectual content and physical form." In the PREMIS model a representation is *a single digital instance of an intellectual entity held in a preservation repository*.

A preservation repository might hold more than one representation for the same intellectual entity. For example, the repository might acquire a single image (say, "Statue of a horse") as a TIFF file. At some point the repository creates a derivative JPEG2000 file from the TIFF and keeps both files. Each of these files would constitute a representation of "Statue of a horse."

In a more complicated example, "Statue of a horse" might be a part of an article consisting of that TIFF image and a file of SGML-encoded text. If the repository created a JPEG2000 version of the TIFF, it would hold two representations of the article: the TIFF and the SGML files would make up one representation, while the JPEG2000 and the SGML files would make up another representation. How those representations are stored is implementation specific. A repository might chose to store a single copy of the SGML file, which would then be shared between representations. Alternately, the repository could choose to duplicate the SGML file and store two identical copies of it. The two representations would then consist of the TIFF and SGML copy 1, and the JPEG2000 and SGML copy 2.

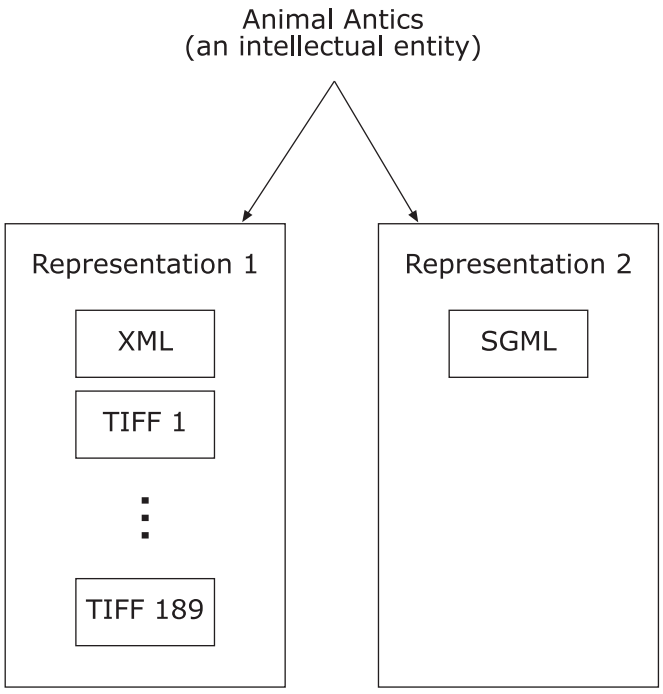
Not all preservation repositories will be concerned with representations. A repository might, for example, preserve file objects only and rely on external agents to assemble these objects into usable representations. If the repository does not manage representations, it does not need to record metadata about them.

Intellectual Entities and Objects

The relationship between Intellectual Entities and Objects can be illustrated by a couple of examples:

Example 1, *Animal Antics*: The book *Animal Antics* was published in 1902. A library digitized *Animal Antics*, creating one TIFF file for each of 189 pages. As structural metadata, it created an XML file showing how the images are assembled into a complete book. The library then performed OCR on the TIFF images, ultimately creating a single large text file that was marked up by hand in SGML. The library submitted 189 TIFF files, one XML file, and one SGML file to a preservation repository.

To the repository *Animal Antics* is an Intellectual Entity: it is a reasonable unit that can be described as a whole, with properties such as an author, a title, and a publication date. The repository has two representations, one consisting of 189 TIFF files and an XML file, and the other consisting of one SGML file. Each representation could render a complete version of *Animal Antics*, albeit with different functionalities. The repository will record metadata about two representation objects and 191 file objects.



Example 2, *Welcome to U*: *Welcome to U*, submitted to a preservation repository as an AVI (Audio Video Interleaved) file, is a 10-minute movie introducing new students to a university campus.

Welcome to U is an Intellectual Entity. The repository has one representation, which consists of a single AVI file. The repository's preservation strategy requires that it manage the audio bits of the AVI file separately from the video bits. The repository will record metadata about one representation object, one file object, and two bitstream objects.

Events

The Event entity aggregates metadata about actions. A preservation repository will record events for many reasons. Documentation of actions that modify (that is, create a new version of) a digital object is critical to maintaining digital provenance, a key element of authenticity. Actions that create new relationships or alter existing relationships are important in explaining those relationships. Even actions that alter nothing, such as validity and integrity checks on objects, can be important to record for management purposes. For billing or reporting purposes some repositories may track actions such as requests for dissemination or reports.

It is up to the repository which actions to record as Events. Some actions may be considered too trivial to record, or may be recorded in other systems (as, for example, routine file backups may be recorded in storage management systems). It is also an implementation decision whether to record events that occur before an object is ingested into the preservation repository, for example, derivation from an earlier object, or changes of custody. In theory, events following the deaccessioning of an Intellectual Entity could also be recorded. For example, a repository might

first deaccession an Intellectual Entity, then delete all file Objects associated with that entity, and record each deletion as an Event.

In the data model Objects are associated with Events in two ways. If an Object is related to a second Object through (because of) an Event, the Event identifier is recorded in the *relationship* container as the semantic component *relatedEventIdentification.* If the Object simply has an associated Event with no relationship to a second Object, the Event identifier is recorded in the container *linkingEventIdentifier*. (For more information on relationships, see page 1-8.)

For example, assume a preservation repository ingests an XML file (object A) and creates a normalized version of it (object B) by running a program (event 1). In the metadata for object B, this could be recorded in *relationship* as follows:

relationshipType = “derivation”
relationshipSubType = “derived from”
relatedObjectIdentification
    relatedObjectIdentifierType = “local”
    relatedObjectIdentifierValue = “A”
    relatedObjectSequence = “not applicable”
relatedEventIdentification
    relatedEventIdentifierType = “local”
    relatedEventIdentifierValue = “1”
    relatedEventSequence = “not applicable”

Continuing with this example, assume that after object B is created it is validated by running another program (event 2). In this case event 2 pertains only to object B, not to the relationship between B and A. The link to event 2 would be recorded as *linkingEventIdentifier*:

linkingEventIdentifierType = “local”
linkingEventIdentifierValue = “2”

A given Object can be associated in these two ways with any number of Events.

All events have outcomes (success, failure, etc.). Some events also have outputs; for example, the execution of a program creates a new file object. The semantic units *eventOutcome* and *eventOutcomeDetail* are intended for documenting qualitative outcomes. For example, if the event is an act of format validation, the value of *eventOutcome* might be a code indicating the object is fully valid. Alternatively, it might be a code indicating the object is not fully valid, and *eventOutcomeDetail* could be used to describe all anomalies found. If the program performing the validation writes a log of warnings and error messages, a second instance of *eventOutcomeDetail* could be used to store or point to that log.

If an event creates objects that are stored in the repository, those objects should be described as entities with a complete set of applicable metadata and associated with the event by links.

Agents

Agents are clearly important but are not the focus of the Data Dictionary, which defines only a means to identify the agent and a classification of agent type (person, organization or software). While more metadata is likely to be necessary, this is left to other initiatives to define.

The data model diagram shows an arrow from the Agent entity to the Event entity, but no arrow from Agent to the Object entity. Agents influence Objects only indirectly through Events. Each Event can have one or more related Objects and one or more related Agents. Because a single Agent can perform different roles in different Events, the role of the Agent is a property of the Event entity, not of the Agent entity.

Rights

Many efforts are concerned with metadata related to intellectual property rights and permissions, from rights expression languages to the <indecs> framework. However, only a small body of work addresses rights and permissions specifically related to digital preservation. The working group surveyed the literature, reviewed the *Implementation Survey Report*, and solicited use cases.

To keep the scope of the discussion manageable, the working group agreed to concentrate on rights and permissions concerned with preservation activities, leaving aside those concerned with access and/or distribution. To further narrow the scope it was proposed that only two expressions were required: "Agent A holds this right to Object B" and "Agent A grants [the repository] this permission related to Object B."

Finally, this was narrowed to the single case, "Agent A grants this permission for Object B." It could be inferred from this that the agent held the right to grant the permission. "Permission" was defined as *an agreement between the rights holder and the repository, allowing the repository to undertake some action*.

Another issue was the appropriate level of granularity for the definitions of the various aspects of permission. For example, if a repository is allowed to make up to three backup copies, this could be expressed as a single statement of permission:

permission = make up to three backup copies

Or it could be expressed as a more granular set of terms:

act = copy
purpose = backup
condition = none
quantity limit = three
time limit = none
geographic limit = none
(and so on)

The working group decided to divide these into only three semantic units: one for the allowed act, one for the expiration date of the permission, and one for all other terms, conditions, restrictions and/or limitations. The dates of the term of the grant were separated out because dates are best represented in a structured format; other restrictions were combined largely for the sake of simplicity of implementation. Finally, a note was added to allow additional or related information to be recorded. Expressing permission for three backup copies in the final structure for the *permission* semantic unit would take this form:

act = make a copy
restriction = up to three; for the purpose of backup only
termOfGrant
    startDate = 20050101
    endDate = none
permissionNote = none

Repositories requiring more granularity are free to develop their own typologies of restrictions.

The semantic units defined in the Data Dictionary should be considered only a start, focused on a very narrow need. A great deal of work remains to be done when the community has more implementation experience in this area.

Relationships

Relationships between Objects

The group began its exploration of this topic by collecting examples from existing preservation metadata projects. It found a wide range of metadata facts expressed as relationships—for example, "is migrated from," "is keyed text of," "is thumbnail of." In some cases these relationship statements combine more than one fact (e.g., "is keyed text of" combines "is a keyed text" and "is derived from"). The group also reviewed the element refinements for the Dublin Core Relation element (IsPartOf, IsFormatOf, IsVersionOf, etc.) and concluded that most relationships among objects appear to be variants of these three basic types: structural, derivation, and dependency.

Structural relationships show relationships between parts of objects. The structural relationships between the files that constitute a representation of an Intellectual Entity are clearly essential preservation metadata. If a preservation repository can't put the pieces of a digital object back together, it hasn't preserved the object. For a simple digital object (e.g., a photograph) structural information is minimal: the file constitutes the representation. Other digital objects such as e-books and Web sites can have quite complex structural relationships.

Derivation relationships result from the replication or transformation of an Object. The intellectual content of the resulting Object is the same, but the Object's instantiation, and possibly its format, are different. When file A of format X is migrated to create file B of format Y, a derivation relationship exists between A and B.

Many digital objects are complex, and both structural and derivation information can change over time as a result of preservation activities. For example, a digitized book represented by 400 TIFF page images might after migration become four PDF files each containing 100 pages.

A structural relationship among objects can be established by an act of derivation before the objects were ingested by the repository. For example, a word-processing document could have been used to create derivative files in PDF and XML formats. If only the PDF and XML files are submitted to the preservation repository, these objects are different representations of the same Intellectual Entity with parent-child relationships to the source word-processing file. They do not have derivation relationships with each other, but do have a structural relationship as siblings (children of a common parent).

There is no one way to model all possible structural or derivation information. Rather than specify a particular approach, the group identified essential information that must be captured. The PREMIS Data Dictionary describes this in the semantic components of the semantic unit *relationship*. Structural and derivative relationships link Objects; the Objects must be identified. The type of relationship must be identified in some way (e.g., "is child of") and the relationship may be associated with an Event that created that relationship. Implementers will likely choose approaches that best suit the content to be preserved by using, for example, the METS structMap or the Dublin Core Relation refinements.

A dependency relationship exists when one object requires another to support its function, delivery, or coherence of content. An object may require a font, style sheet, DTD, schema, or other file that is not formally part of the object itself but is necessary to render it. The Data Dictionary handles dependency relationships as part of the environment information, in the semantic units *dependency* and *swDependency*. In this way requirements for hardware and software are brought together with requirements for dependent files to form a complete picture of the information or assets required for the rendering and/or understanding of the object.

Relationships between entities of different types

The data model diagram uses arrows to show relationships between entities of different types. Objects are related to Intellectual Entities, Objects are related to Events, Agents are related to Events, etc. The Data Dictionary expresses relationships as linking information by including in the information for entity A a pointer to the related entity B. Every entity in the data model has a unique identifier for use as a pointer. So, for example, the Object entity has arrows pointing to Intellectual Entities and Events. These are implemented in the Data Dictionary by the semantic units *linkingIntellectualEntityIdentifier* and *linkingEventIdentifier*.

The 1:1 principle

In digital preservation it is common practice to create new copies or versions of stored objects. For example, in forward migration file A in format X may be input to a program which outputs file B in format Y. There are two ways to think about files A and B. One might think of them as a single Object, the history of which includes the transformation from X to Y, or one could think of them as two distinct Objects with a relationship created by the transformation Event.

The 1:1 principle in metadata asserts that each description describes one and only one resource. As applied to PREMIS metadata, every Object held within the preservation repository (file, bitstream, representation) is described as a static set of bits. It is not possible to change a file (or bitstream or representation); one can only create a new file (or bitstream or representation) that is related to the source Object. In the example above, therefore, files A and B are distinct Objects with a derivative relationship between them. The Data Dictionary has a semantic unit for the creation date of an Object (*dateCreatedByApplication*) but not for the modification date of an Object, because an Object, by definition, cannot be modified.

When new objects are derived from existing objects the event that created the new object should be recorded as an Event, which will have a date/time stamp. The relationship(s) among the objects should be recorded using the *relationship* semantic unit associated with the Object entity. The semantic component *relatedEventIdentification* should be used to make the association with the Event.

2. THE PREMIS DATA DICTIONARY VERSION 1.0

The PREMIS Data Dictionary includes semantic units for Objects, Events, Agents, and Rights. The fifth entity in the model, the Intellectual Entity, is considered out of scope because it is well served by descriptive metadata. The template for each entry includes a place for notes about how to create or use the semantic unit. In some cases the group felt additional information, such as the reason for a semantic unit's definition or issues that arose in the group's deliberations, would be useful; for these details, see "Special Topics," page 4-1.

A semantic component always inherits the applicability of the containing semantic unit. That is, if the containing semantic unit specifies that it is applicable to files but not to representations, each of its semantic components is applicable to files and not to representations. Repeatability and obligation, however, may vary.

Each entry in the Data Dictionary offers these attributes of a semantic unit:

- **The name of the semantic unit:** Names were devised to be descriptive and unique within the Data Dictionary. Using these names for the exchange of metadata among preservation repositories will aid interoperability. These names need not be used internally within any individual preservation repository.

- **Semantic components:** The semantic components each have their own entries later in the Data Dictionary. A semantic unit that has semantic components does not have any value of its own. Only semantic units at the lowest level have values.

- **Definition:** The meaning of the semantic unit.

- **Rationale:** Why the semantic unit is needed, if this is not self-evident from the definition.

- **Data constraint:** How the value of the semantic unit should be encoded. Some common data constraints are:

  Container—The semantic unit is an umbrella for two or more semantic components and has no value of its own.

  None—The semantic unit can take any form of value.

  Value should be taken from a controlled vocabulary—The preservation repository should establish an authority list of values that are useful and meaningful to the repository. PREMIS does not specify what this authority list should be, and it is assumed that different repositories will use different vocabularies. In general, when data is taken from a controlled vocabulary, both a scheme (the source of the vocabulary) and a value should be recorded.

- **Object category:** Whether the unit applies to a representation, file, or bitstream Object. Semantic units that apply to files also apply to filestreams (see page 1-3).

- **Applicability:** A scope of "applicable" means it applies to that category of Object.

- **Examples:** One or more examples of values the semantic unit may take. Examples are intended to be illustrative.

  An example of an actual value is set in normal text. Text in brackets presents a description of the value rather than the value itself. For example, "SHA-1 message digest" reflects the actual value of the semantic unit, while "[SHA-1 message digest]" means the value of the semantic unit is a SHA-1 message digest such as "7c9b35da4f2ebd436f1cf88e5a39b3a257edf4a22be3c955ac49da2e2107b67a1924419563"

- **Repeatability:** A semantic unit designated as "Repeatable" can take multiple values. It does not mean that a repository must record multiple instances of the semantic unit.

- **Obligation:** Whether a value for the semantic unit is mandatory (if applicable) or optional.

  A mandatory semantic unit is something that the preservation repository needs to know, independent of how or whether the repository records it. The repository might not explicitly record a value for the semantic unit if it is known by some other means (e.g., by the repository's business rules). "Mandatory" actually means "mandatory if applicable." For example, an identifier for a bitstream is mandatory only if the repository manages data at the bitstream level. When exchanging PREMIS-conformant metadata with another repository, values for mandatory semantic units must always be provided.

  Values for optional semantic units are encouraged but not required.

  If a container unit is optional, but a semantic component within that container is mandatory, the semantic component must be supplied if and only if the container unit exists. That is, if a value for any of the optional or mandatory semantic units in the container is supplied, a value for all of the mandatory semantic unites in the container must be supplied.

- **Creation/Maintenance notes:** Notes about how the values for the semantic unit may be obtained and/or updated.

- **Usage notes**: Information about the intended use of the semantic unit, or clarification of the definition.

Limits to the scope of the Data Dictionary

Descriptive metadata: Typically, descriptive metadata is used to describe Intellectual Entities. Nearly all preservation repositories either include descriptive metadata or link to descriptive metadata located outside the repository itself. Such metadata may identify a resource by publication information such as creator and title, or may characterize its intellectual content through classification, subject terms, and so on. Descriptive metadata can be important both for discovery of archived resources and for helping decision makers during preservation planning. However, the Data Dictionary does not focus on descriptive elements for two reasons.

First, descriptive metadata is well served by existing standards. MARC, MODS, the Dublin Core Metadata Element Set, the Content Standard for Digital Geospatial Metadata, the Visual Resources Core Categories, the Encoded Archival Description (EAD), and the Digital Documentation Initiative schemas are only some of the standards that define descriptive metadata elements. The working group did not want to add another set of descriptive elements to an already crowded field. Second, descriptive metadata is often domain specific. For the purposes of preservation it is less crucial that a common set of elements describe satellite telemetry and digital Picassos than that communities of interest be able to capture and exchange information in a form that reflects their materials and interests appropriately.

Agents: PREMIS does not define the characteristics of Agents in any detail. Metadata describing people, organizations, and other entities that can act as Agents has been defined in many existing formats and standards, such as MARC, vCard, MADS, and several other schemes currently under development. As long as a preservation repository can properly identify Agents that have acted upon Objects in its care, additional Agent characteristics will be determined by local requirements; many can be modeled on existing standard metadata element sets.

Rights: PREMIS only defines characteristics of rights and permissions concerned with preservation activities, not those associated with access and/or distribution. The only case of rights covered is that of a specific agent grating a specific permission (act or restrictions) for a specific object.

Technical metadata: Technical metadata describes the physical rather than intellectual characteristics of digital objects. Detailed, format-specific technical metadata is clearly necessary for implementing most preservation strategies, but the group had neither the time nor the expertise to tackle format-specific technical metadata for various types of digital files. Therefore, it restricted the technical metadata included in the Data Dictionary to the semantic units it believed apply to objects in all formats. Further development of technical metadata is left to format experts.

Media or hardware details: The working group did not attempt to define metadata for detailed documentation of media or hardware. For example, PREMIS defines a semantic unit for identifying the medium on which an object is stored. A preservation repository will probably want to know more detailed information about the media employed. If the repository stores data on DVDs, for example, it may need to know the specific technical characteristics of the specific DVD units, such as manufacturer, dye material, and dye thickness. PREMIS leaves the definition of metadata for describing media and hardware characteristics to specialists in these areas.

Business rules: The working group made no attempt to describe the business rules of a repository, although certainly this metadata is essential for preservation within the repository. Business rules codify the application of preservation strategies and document repository policies, services, charges, and roles. Retention periods, disposition, risk assessment, permanence ratings, schedules for media refreshment, and so on are pertinent to objects but are not actual properties of Objects. A single exception was made for the level of preservation treatment to be accorded an object (*preservationLevel*) because this was felt to be critical information for any preservation repository. A more thorough treatment of business rules could be added to the data model by defining a Rules entity similar to Rights.

Object Entity

The Object entity aggregates information about a digital object held by a preservation repository and describes those characteristics relevant to preservation management.

The only mandatory semantic unit that applies to all categories of object (representation, file, and bitstream) is *objectIdentifier*.

Entity types

- Representation: A digital object instantiating or embodying an Intellectual Entity. A representation is the set of stored digital files and structural metadata needed to provide a complete and reasonable rendition of the Intellectual Entity.
- File: A named and ordered sequence of bytes that is known to an operating system.
- Bitstream: Contiguous or non-contiguous data within a file that has meaningful properties for preservation purposes.

Entity properties

- Can be associated with one or more rights statements.
- Can participate in one or more events.
- Can be related to one or more agents.

Entity semantic units

- objectIdentifier
  - objectIdentifierType
  - objectIdentifierValue
- preservationLevel
- objectCategory
- objectCharacteristics
  - compositionLevel
  - fixity
    - messageDigestAlgorithm
    - messageDigest
    - messageDigestOriginator
  - size
  - format
    - formatDesignation
      - formatName
      - formatVersion
    - formatRegistry
      - formatRegistryName
      - formatRegistryKey
      - formatRegistryRole

    - significantProperties
    - inhibitors
        - inhibitorType
        - inhibitorTarget
        - inhibitorKey
- creatingApplication
    - creatingApplicationName
    - creatingApplicationVersion
    - dateCreatedByApplication
- originalName
- storage
    - contentLocation
        - contentLocationType
        - contentLocationValue
    - storageMedium
- environment
    - environmentCharacteristic
    - environmentPurpose
    - environmentNote
    - dependency
        - dependencyName
        - dependencyIdentifier
            - dependencyIdentifierType
            - dependencyIdentifierValue
    - software
        - swName
        - swVersion
        - swType
        - swOtherInformation
        - swDependency
    - hardware
        - hwName
        - hwType
        - hwOtherInformation
- signatureInformation
    - signatureInformationEncoding
    - signer
    - signatureMethod
    - signatureValue
    - signatureValidationRules
    - signatureProperties
    - keyInformation
        - keyType
        - keyValue

    - keyVerificationInformation
- relationship
    - relationshipType
    - relationshipSubType
    - relatedObjectIdentification
        - relatedObjectIdentifierType
        - relatedObjectIdentifierValue
        - relatedObjectSequence
    - relatedEventIdentification
        - relatedEventIdentifierType
        - relatedEventIdentifierValue
        - relatedEventSequence
- linkingEventIdentifier
    - linkingEventIdentifierType
    - linkingEventIdentifierValue
- linkingIntellectualEntityIdentifier
    - linkingIntellectualEntityIdentifierType
    - linkingIntellectualEntityIdentifierValue
- linkingPermissionStatementIdentifier
    - linkingPermissionStatementIdentifierType
    - linkingPermissionStatementIdentifierValue

Semantic unit	objectIdentifier		
Semantic components	objectIdentifierType, objectIdentifierValue		
Definition	A designation used to uniquely identify the object within the preservation repository system in which it is stored.		
Rationale	Each data object held in the preservation repository must have a unique identifier to relate it to descriptive, technical, and other metadata.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Mandatory	Mandatory	Mandatory
Creation/ Maintenance notes	An identifier may be created by the repository system at the time of ingest, or it may be created or assigned outside of the repository and submitted with an object as metadata. Similarly, identifiers can be automatically or manually generated. Recommended practice is for repositories to use identifiers automatically created by the repository as the primary identifier in order to insure that identifiers are unique and usable by the repository. Externally assigned identifiers can be used as secondary identifiers in order to link an object to information held outside the repository.		
Usage notes	The objectIdentifier is mandatory if the preservation repository stores and manages objects at that level (i.e., representation, file, bitstream). Identifiers must be unique within the repository. They may be preexisting, and in use in other digital object management systems. Identifiers used to identify a class of objects (e.g., the way an ISBN identifies all books in the same edition) are not acceptable as identifiers in the context of the preservation repository, which must identify the specific object in the repository. A preservation repository needs to know both the type of object identifier and the value. If the value itself contains the identifier type (e.g., “oai:lib.uchicago.edu:1”), the identifier type does not need to be explicitly recorded. Similarly, if the repository uses only one type of identifier, the type can be assumed and does not need to be explicitly recorded.		

Semantic unit	objectIdentifierType		
Semantic components	None		
Definition	A designation of the domain within which the object identifier is unique.		
Rationale	Identifier values cannot be assumed to be unique across domains; the combination of objectIdentifierType and objectIdentifierValue should ensure uniqueness.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	DLC FCLA Digital Archive DRS	DLC FCLA Digital Archive DRS	DLC FCLA Digital Archive DRS
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	The type of the identifier may be implicit within the repository as long it is can be explicitly communicated when the digital object is disseminated outside of it.		

Semantic unit	objectIdentifierValue		
Semantic components	None		
Definition	The value of the objectIdentifier.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	0000000312	IU2440 WAC1943.56 AMNH CD269/CD269/70/10596.PCD CDS-VDEP-200211119-24879.734 1001/dig/pres/2004-024 http://nrs.harvard.edu/urn-3:FHCL.Loeb:sa1	IU2440-1 IU2440-2
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	preservationLevel		
Semantic components	None		
Definition	A value indicating the set of preservation functions expected to be applied to the object.		
Rationale	Some preservation repositories will offer multiple preservation options depending on factors such as the value or uniqueness of the material, the "preservability" of the format, the amount the customer is willing to pay, etc.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Not applicable
Examples	bit-level full 0 1 2	bit-level full 0 fully supported with future migrations	
Repeatability	Not repeatable	Not repeatable	
Obligation	Mandatory	Mandatory	
Creation/ Maintenance notes	The preservation level may be assigned by the repository or requested by the depositor and submitted as metadata.		
Usage notes	If the repository offers only a single preservation level, this value does not need to be explicitly recorded within the repository.		

Semantic unit	objectCategory		
Semantic components	None		
Definition	The category of object to which the metadata applies.		
Rationale	Preservation repositories are likely to treat different categories of objects (representations, files, and bitstreams) differently in terms of metadata and data management functions.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	representation	file	bitstream
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	Suggested values: representation, file, bitstream. A filestream should be considered a file.		

Semantic unit	objectCharacteristics		
Semantic components	compositionLevel, fixity, size, format, significantProperties, inhibitors		
Definition	Technical properties of a file or bitstream that are applicable to all or most formats.		
Rationale	Format-specific properties are outside of the scope of this Data Dictionary. However, there are some important technical properties that apply to objects of any format.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Mandatory	Mandatory
Usage notes	The semantic units included in objectCharacteristics should be treated as a set of information that pertains to a single object at a single compositionLevel. Object characteristics may be repeated when an object was created by applying two or more encodings, such as compression and encryption. In this case each repetition of objectCharacteristics would have an incrementally higher compositionLevel. When encryption is applied, the objectCharacteristics block must include an inhibitors semantic unit. A bitstream embedded within a file may have different object characteristics than the file. Where these characteristics are relevant for preservation, they should be recorded. See “Object characteristics and composition level,” page 4-4.		

Semantic unit	compositionLevel		
Semantic components	None		
Definition	An indication of whether the object is subject to one or more processes of decoding or unbundling.		
Rationale	A file or bitstream can be encoded with compression, encryption, etc., or bundled with other files or bitstreams into larger packages. Knowing the order in which these actions are taken is important if the original object or objects must be recovered.		
Data constraint	Non-negative integers		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		0 1 2	0 1 2
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Creation/ Maintenance notes	Composition level will generally be supplied by the repository, which should attempt to supply this value automatically. If the object was created by the repository, the creating routine knows the composition level and can supply this metadata. If the object is being ingested by the repository, repository programs will have to attempt to identify the composition level from the object itself or from externally supplied metadata.		
Usage notes	A file or bitstream can be subject to multiple encodings that must be decoded in reverse order (highest to lowest). For example, file A may be compressed to create file B, which is encrypted to create file C. To recreate a copy of the base file A, one would have to unencrypt file C to create file B and then uncompress file B to create file A. A compositionLevel of zero indicates that the object is a base object and not subject to further decoding, while a level of 1 or higher indicates that one or more decodings must be applied. Numbering goes lowest to highest (first encoded = 0). 0 is base object; 1-n are subsequent encodings. Use 0 as the default if there is only one compositionLevel. When multiple file objects are bundled together as filestreams within a package file object (e.g., a ZIP file), the individual filestream objects are *not* composition levels of the package file object. They should be considered separate objects, each with their own		

	composition levels. For example, two encrypted files zipped together and stored in an archive as one file object would be described as three separate objects, each with its own associated metadata. The storage location of the two inner objects would point to the ZIP file, but the ZIP file itself would have only a single composition level (of zero) whose format would be "zip." See "Object characteristics and composition level," page 4-4.

Semantic unit	fixity		
Semantic components	messageDigestAlgorithm, messageDigest, messageDigestOriginator		
Definition	Information used to verify whether an object has been altered in an undocumented or unauthorized way.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable (see usage note)	Applicable	Applicable (see usage note)
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Creation/ Maintenance notes	Automatically calculated and recorded by repository.		
Usage notes	To perform a fixity check, a message digest calculated at some earlier time is compared with a message digest calculated at a later time. If the digests are the same, the object was not altered in the interim. Recommended practice is to use two or more message digests calculated by different algorithms. The act of performing a fixity check and the date it occurred would be recorded as an Event. The result of the check would be recorded as the eventOutcome. Therefore, only the messageDigestAlgorithm and messageDigest need to be recorded as objectCharacteristics for future comparison. Representation level: It could be argued that if a representation consists of a single file, or if all the files comprised by a representation are combined (e.g., zipped) into a single file, then a fixity check could be performed on the representation. However, in both cases the fixity check is actually being performed on a file, which in this case happens to be coincidental with a representation. Bitstream level: Message digests can be computed for bitstreams although they are not as common as with files. For example, the JPX format, which is a JPEG2000 format, supports the inclusion of MD5 or SHA-1 message digests in internal metadata that was calculated on any range of bytes of the file. See “Fixity, integrity, authenticity,” page 4-5.		

Semantic unit	messageDigestAlgorithm		
Semantic components	None		
Definition	The specific algorithm used to construct the message digest for the digital object.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		MD5 Adler-32 HAVAL SHA-1 SHA-256 SHA-384 SHA-512 TIGER WHIRLPOOL	
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	messageDigest		
Semantic components	None		
Definition	The output of the message digest algorithm.		
Rationale	This must be stored so that it can be compared in future fixity checks.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		7c9b35da4f2ebd436f1cf88e5a39b3a257edf4a22be3c955ac49da2e2107b67a1924419563	
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	messageDigestOriginator		
Semantic components	None		
Definition	The agent that created the original message digest that is compared in a fixity check.		
Rationale	A preservation repository may ingest files that have had message digests calculated by the submitter; checking these ensures that the file as received is the same as the file as sent. The repository may also ingest files that do not have message digests, and so must calculate the initial value upon ingest. It can be useful to know who calculated the initial value of the message digest.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		DRS A0000978	
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Creation/ Maintenance notes	If the calculation of the initial message digest is treated by the repository as an Event, this information could be obtained from an Event record.		
Usage notes	The originator of the message digest could be represented by a string representing the agent (e.g., “DRS” referring to the archive itself) or a pointer to an agent description (e.g., “A0000987” taken here to be an agentIdentifierValue).		

Semantic unit	size		
Semantic components	None		
Definition	The size in bytes of the file or bitstream stored in the repository.		
Rationale	Size is useful for ensuring the correct number of bytes from storage have been retrieved and that an application has enough room to move or process files. It might also be used when billing for storage.		
Data constraint	Integer		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		2038937	
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Creation/ Maintenance notes	Automatically obtained by the repository.		
Usage notes	Defining this semantic unit as size in bytes makes it unnecessary to record a unit of measurement. However, for the purpose of data exchange the unit of measurement should be stated or understood by both partners.		

Semantic unit	format		
Semantic components	formatDesignation, formatRegistry		
Definition	Identification of the format of a file or bitstream where format is the organization of digital information according to preset specifications.		
Rationale	Many preservation activities depend on detailed knowledge about the format of the digital object. An accurate identification of format is essential. The identification provided, whether by name or pointer into a format registry, should be sufficient to associate the object with more detailed format information.		
Data constraint	Container		
Object category	**Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Creation/ Maintenance notes	The format of a file or bitstream should be ascertained by the repository on ingest. Even if this information is provided by the submitter, directly in metadata or indirectly via the file name extension, recommended practice is to independently identify the format by parsing the file when possible. If the format can not be identified at the time of ingest, it is valid to record that the format is unknown, but the repository should subsequently make an effort to identify the format, even if manual intervention is required.		
Usage notes	A bitstream embedded within a file may have different characteristics than the larger file. For example, a bitstream in LaTex format could be embedded within an SGML file, or multiple images using different colorspaces could be embedded within a TIFF file. Format must be recorded for every file. When the bitstream format can be recognized by the repository and the repository might want to treat the bitstream differently from the embedding file for preservation purposes, format can be recorded for embedded bitstreams. Either formatDesignation or formatRegistry should be recorded. Both are optional, but since format (the container) is mandatory, one of these must be used. See “Format information,” page 4-1.		

Semantic unit	formatDesignation		
Semantic components	formatName, formatVersion		
Definition	An identification of the format of the object.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Usage notes	Either formatDesignation or at least one instance of formatRegistry is required. The most specific format (or format profile) should be recorded. A repository (or formats registry) may wish to use multipart format names (e.g., "TIFF_GeoTIFF" or "WAVE_MPEG_BWF") to achieve this specificity.		

Semantic unit	formatName		
Semantic components	None		
Definition	A designation of the format of the file or bitstream.		
Data constraint	Value should be taken from a controlled vocabulary		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		Text/sgml image/tiff/geotiff Adobe PDF DES PGP base64	LaTex
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	formatVersion		
Semantic components	None		
Definition	The version of the format named in formatName.		
Rationale	Many authority lists of format names are not granular enough to indicate version, for example, MIME Media types.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		6.0 2003	
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Usage notes	If the format is versioned, formatVersion should be recorded. It can be either a numeric or chronological designation.		

Semantic unit	formatRegistry		
Semantic components	formatRegistryName, formatRegistryKey, formatRegistryRole		
Definition	Identifies and/or gives further information about the format by reference to an entry in a format registry.		
Rationale	If central format registries are available to the preservation repository, they may provide an excellent way of referencing detailed format information.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	Either formatDesignation or at least one instance of formatRegistry is required. The PREMIS working group assumed that a number of format registries will be developed and maintained to support digital preservation efforts. The proposal for a Global Digital Format Registry (GDFR), for example, would create a network-accessible registry designed to store detailed specifications on formats and profiles.		

Semantic unit	formatRegistryName		
Semantic components	None		
Definition	A designation identifying the referenced format registry.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		FRED: A format registry demonstration, release 0.07 http://tom.library.upenn.edu/cgi-bin/fred PRONOM	FRED: A format registry demonstration, release 0.07
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	This can be a formal name, internally used name, or URI.		

Semantic unit	formatRegistryKey		
Semantic components	None		
Definition	The unique key used to reference an entry for this format in a format registry.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		info:dgfr/fred/f/tiff TIFF/6.0	
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	formatRegistryRole		
Semantic components	None		
Definition	The purpose or expected use of the registry.		
Rationale	The same format may be defined in different registries for different purposes. For example, one registry may give detailed format specifications while another has profile information. If multiple registries are recorded, this semantic unit can be used to distinguish among them.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		Specification Validation profile	
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional

Semantic unit	significantProperties		
Semantic components	None		
Definition	Characteristics of a particular object subjectively determined to be important to maintain through preservation actions.		
Rationale	Objects that have the same technical properties may still differ as to the properties that should be preserved for future presentation or use.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	[for a Web page containing animation that is not considered essential] Content only.	[for a PDF with embedded links that are not considered essential] Content only.	[for a PDF with an embedded graph, where the lines' color determines the lines' meaning] Color.
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Creation/ Maintenance notes	Significant properties may pertain to all objects of a certain class; for example, the repository can decide that for all PDF files, only the content need be preserved. In other cases, for example, for media art, the significant properties may be unique to each individual object. Where values are unique, they must be supplied by the submitter or provided by the curatorial staff of the repository.		
Usage notes	Significant properties may be objective technical characteristics subjectively considered important, or subjectively determined characteristics. For example, a PDF may contain links that are not considered important and JavaScript that is considered important. Or future migrations of a TIFF image may require optimization for line clarity or for color; the option chosen would depend upon a curatorial judgment of the significant properties of the image. Listing significant properties implies the repository plans to preserve these properties in emulation or through migrations. It also implies the repository would note when preservation results in modification of significant properties. More experience with digital preservation is needed to determine the best ways of representing this information. One possible way involves the use of Object and Event information: Object A has significant properties volume and timing, which are recorded as significantProperties of A. In migrated version B, the timing is modified, which is noted in the eventOutcome of the migration Event. Only volume is listed as a significant property of B.		

Semantic unit	inhibitors		
Semantic components	inhibitorType, inhibitorTarget, inhibitorKey		
Definition	Features of the object intended to inhibit access, use, or migration.		
Rationale	Format information may indicate whether a file is encrypted, but the nature of the encryption also must be recorded, as well as the access key.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Creation/ Maintenance notes	Inhibitors are more likely to be present on an object ingested by the repository than applied by the repository itself. It is often not possible to tell that a file has been encrypted by parsing it; the file may appear to be ASCII text. Therefore, information about inhibitors should be supplied as metadata with submitted objects when possible.		
Usage notes	Some file formats allow encryption for embedded bitstreams. Some file formats such as PDF use passwords to control access to content or specific functions. Although this is actually implemented at the bitstream level, for preservation purposes it is effectively managed at the file level, that is, passwords would not be recorded for individually addressable bitstreams. For certain types of inhibitor keys, more granularity may be required. If the inhibitor key information is identical to key information in digital signatures, use those semantic units.		

Semantic unit	inhibitorType		
Semantic components	None		
Definition	The inhibitor method employed.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		DES PGP Blowfish Password protection	
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	Common inhibitors are encryption and password protection. When encryption is used the type of encryption should be specifically indicated, that is, record “DES”, not “encryption”.		

Semantic unit	inhibitorTarget		
Semantic components	None		
Definition	The content or function protected by the inhibitor.		
Data constraint	Values should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		All content Function: Play Function: Print	
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	If not supplied, assume that the target is the content of the object.		

Semantic unit	inhibitorKey		
Semantic components	None		
Definition	The decryption key or password.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		[DES decryption key]	
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Usage notes	The key should be provided if known. However, it is not advisable to actually store the inhibitorKey in plaintext in an unsecure database.		

Semantic unit	creatingApplication		
Semantic components	creatingApplicationName, creatingApplicationVersion, dateCreatedByApplication		
Definition	Information about the application that created the object.		
Rationale	Information about the creating application, including the version of the application and the date the file was created, can be useful for problem solving purposes. For example, it is not uncommon for certain versions of software to be known for causing conversion errors or introducing artifacts.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Creation/ Maintenance notes	If the object was created by the repository, assignment of creating application information should be straightforward. If the object was created outside the repository, it is possible this information could be supplied by the depositor. It might also be extracted from the file itself; the name of the creating application is often embedded within the file.		
Usage notes	This semantic unit applies to both objects created external to the repository and subsequently ingested, and to objects created by the repository, for example, through migration events. The creatingApplication container is repeatable if more than one application processed the object in turn. For example, a file could be created by Microsoft Word and later turned into a PDF using Adobe Acrobat. Details of both the Word and Acrobat applications may be recorded. However, if both files are stored in the repository, each file should be completely described as an Object entity and linked by using relationship information with a relationshipType “derivation.” It may also be repeated to record the creating application before the object was ingested as well as the creating application used as part of the ingest process. For example, an HTML file was created pre-ingest using Dreamweaver, and the Web crawler Heritrix then captured a snapshot of the files as part of the ingest. The amount of information needed for creatingApplication given here is minimal. For more granularity, semantic units using the same model as under environment may be used. Rather than having each repository record this locally, it would be preferable to have a registry of this information similar to format or environment registries.		

Semantic unit	creatingApplicationName		
Semantic components	None		
Definition	A designation for the name of the software program that created the object.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	Flash MX	MSWord	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Optional	Optional	Optional
Usage notes	The creatingApplication is the application that created the object in its current format, not the application that created the copy written to storage. For example, if a document is created by Microsoft Word and subsequently copied to archive storage by a repository's Ingest program, the creatingApplication is Word, not the Ingest program.		

Semantic unit	creatingApplicationVersion		
Semantic components	None		
Definition	The version of the software program that created the object.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		2000	1.4
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Optional	Optional	Optional

Semantic unit	dateCreatedByApplication		
Semantic components	None		
Definition	The actual or approximate date and time the object was created.		
Data constraint	Value should be formatted according to ISO 8601.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		2000-12-01 20030223151047.0	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Optional	Optional	Optional
Usage notes	Use the most precise date available. This is the date the object was created by the creating application, not the date any copy was made externally or by the repository. For example, if a file is created by Microsoft Word in 2001 and two copies are made in 2003, the dateCreatedByApplication of all three files is 2001. The date a file is written to storage can be recorded as an Event. If the object itself contains internal creation and modification dates, the modification date should be used as dateCreatedByApplication. If the application is a Web harvester capturing an object at a point of time, use for date captured.		

Semantic unit	originalName		
Semantic components	None		
Definition	The name of the object as submitted to or harvested by the repository, before any renaming by the repository.		
Rationale	The name used within the preservation repository may not be known outside of the repository. A depositor might need to request a file by its original name. Also, the repository may need to reconstruct internal links for dissemination.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Not applicable
Examples		N419.pdf	
Repeatability		Not repeatable	
Obligation		Optional	
Creation/ Maintenance notes	This value would always be supplied to the repository by the submitter or harvesting application. How much of the filepath to preserve would be up to the repository.		
Usage notes	This is the name of the file as designated in the Submission Information Package (SIP). The file may have other names in different contexts.		

Semantic unit	storage		
Semantic components	contentLocation, storageMedium		
Definition	Information about how and where a file is stored in the storage system.		
Rationale	It is necessary for a repository to associate the contentLocation with the storageMedium.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Mandatory	Mandatory
Usage notes	Normally there would be a single storage location and medium for an object, because an object in another location would be considered a different object. The storage composite should be repeated if there are two or more copies that are identical bit-wise and managed as a unit except for the medium on which they are stored. They must have a single objectIdentifier and be managed as a single object by the repository.		

Semantic unit	contentLocation		
Semantic components	contentLocationType, contentLocationValue		
Definition	Information needed to retrieve a file from the storage system, or to access a bitstream within a file.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Creation/ Maintenance notes	A preservation repository should never refer to content that it does not control. Therefore, the PREMIS working group assumed that the repository will always assign the contentLocation, probably by program.		
Usage notes	If the preservation repository uses the objectIdentifier as a handle for retrieving data, contentLocation is implicit and does not need to be recorded.		

Semantic unit	contentLocationType		
Semantic components	None		
Definition	The means of referencing the location of the content.		
Rationale	To understand the meaning of the value it is necessary to know what location scheme is used.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		URI hdl	byte offset
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	contentLocationValue		
Semantic components	None		
Definition	The reference to the location of the content.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		http://wwasearch.loc.gov/107th/20021210 7035/http://house.gov/langevin/ hdl:loc.pnp/cph.3b34188	64 [offset from start of file]
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	This could be a fully qualified path and filename, or the information used by a resolution system (e.g., a handle) or the information used by a storage management system. For a bitstream or filestream, this would probably be the reference point and offset of the starting position of the bitstream.		

Semantic unit	storageMedium		
Semantic components	None		
Definition	The physical medium on which the object is stored (e.g., magnetic tape, hard disk, CD-ROM, DVD).		
Rationale	The repository needs to know the medium on which an object is stored in order to know how and when to do media refreshment and media migration.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Not applicable
Examples		Magnetic tape Hard disk TSM	
Repeatability		Not Repeatable	
Obligation		Mandatory	
Usage notes	In some cases this can be masked from direct repository management by storage management systems but the underlying assumption is that the repository ultimately is in control and needs to manage for technological obsolescence. In some cases the value may not be the specific medium, but the system that knows the medium, e.g., Tivoli Storage Manager (TSM).		

Semantic unit	environment		
Semantic components	environmentCharacteristic, environmentPurpose, environmentNote, dependency, software, hardware		
Definition	Hardware/software combinations supporting use of the object.		
Rationale	Environment is the means by which the user renders and interacts with content. Separation of digital content from its environmental context can result in the content becoming unusable.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Creation/ Maintenance notes	This information may be omitted when the repository is doing only bit-level preservation on the object. Rather than having each repository record this locally, it would be preferable to have a registry of environment information similar to proposed registries of format information. Repositories may choose to design mechanisms for inheritance, so that if the environment required for each file within a representation is identical to the environment recorded for the representation as a whole, it is not necessary to store this information in each file. See “Environment,” page 4-2.		

Semantic unit	environmentCharacteristic		
Semantic components	None		
Definition	An assessment of the extent to which the described environment supports its purpose.		
Rationale	If multiple environments are described, this element can help to distinguish among them.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	unspecified minimum	recommended minimum	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Optional	Optional	Optional
Creation/ Maintenance notes	This value could be supplied by the submitter or by the repository. If environment software and hardware information is obtained from an environments registry, environmentCharacteristic might also be obtained from the registry. Note however that the criteria for "recommended" may be different for different repositories.		
Usage notes	Suggested values: unspecified = no attempt made to provide this value known to work = the object can be rendered in this environment minimum = the least demanding (in terms of components or resources needed) environment known to work by the repository recommended = an environment preferred for optional rendering If an environment is both "minimum" and "recommended," use "recommended." "Known to work" implies the object is supported by the described environment but the repository doesn't know if this environment is minimum or recommended.		

Semantic unit	environmentPurpose		
Semantic components	None		
Definition	The use(s) supported by the specified environment.		
Rationale	Different environments can support different uses of objects. For example, the environment needed to edit and modify a file can be quite different than the environment needed to render it.		
Data constraint	Values should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Creation/ Maintenance notes	This value would have to be supplied by the agent that provided the hardware and software environment information, which might be the submitter, the repository, or an environments registry.		
Usage notes	A starter list of suggested values: render, edit. This list may need to be expanded. Other values might indicate the ability to transform, print, and manipulate by program.		

Semantic unit	environmentNote		
Semantic components	None		
Definition	Additional information about the environment.		
Rationale	There may be a need to give a textual description of the environment for additional explanation.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		This environment assumes that the PDF will be stored locally and used with a standalone PDF reader.	
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	This note could be used to record the context of the environment information. For example, if a file can be rendered through a PC client application or through a browser with a plug-in, this note could be used to identify which situation applies. The note should not be used for a textual description of environment information recorded more rigorously elsewhere.		

Semantic unit	dependency		
Semantic components	dependencyName, dependencyIdentifier		
Definition	Information about a non-software component or associated file needed in order to use or render the representation or file, for example, a schema, a DTD, or an entity file declaration.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Creation/ Maintenance notes	Recommended practice is for a repository to archive objects on which other objects depend. These may be sent by the submitter of the primary object, or they may in some cases be automatically obtained by the repository. For example, a markup file will often contain links to other objects it requires such as DTDs or XML Schema. If it does, these objects can often be identified by the link and downloaded by the repository.		
Usage notes	This semantic unit is for additional objects that are necessary to render a file or representation, not for required software or hardware. It may also be used for a non-executable component of the object, such as a font or style sheet. For things that the software requires, see swDependency. This semantic unit does not include objects required by structural relationships, such as child content objects (e.g., figures that are part of an article), which are recorded under relationship with a relationshipType of “structural”. It is up to the repository to determine what constitutes a dependency in the context of the designated community. The objects noted may be internal or external to the preservation repository.		

Semantic unit	dependencyName		
Semantic components	None		
Definition	A designation for a component or associated file needed by the representation or file.		
Rationale	It may not be self-evident from the dependencyIdentifier what the name of the object actually is.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		Additional Element Set for Language Corpora	
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional

Semantic unit	dependencyIdentifier		
Semantic components	dependencyIdentifierType, dependencyIdentifierValue		
Definition	A unique designation used to identify a dependent resource.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	The dependencyIdentifier must be unique within the preservation repository, although it might not be globally unique.		

Semantic unit	dependencyIdentifierType		
Semantic components	None		
Definition	A designation of the domain in which the identifier of the dependent resource is unique.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		URI	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	A preservation repository needs to know both the type of object identifier and the value. When the value itself contains the identifier type (e.g., "oai:lib.uchicago.edu:1"), the identifier type does not need to be recorded explicitly. Similarly, if the repository uses only one type of identifier, the type can be assumed and does not need to be recorded explicitly.		

Semantic unit	dependencyIdentifierValue		
Semantic components	None		
Definition	The value of the dependencyIdentifier.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		http://www.tei-c.org/P4X/DTD/teicorp2.dtd	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	software		
Semantic components	swName, swVersion, swType, swOtherInformation, swDependency		
Definition	Software required to render or use the object.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Creation/ Maintenance notes	If recording this explicitly, many different software environments may apply; for example, a particular object such as a PDF file may be viewable by several versions of several applications running under several operating systems and operating system versions. Although at least one software environment should be recorded, it is not necessary to record them all and each repository will have to make its own decisions about which software environments to record. Also, what appears to the user as a single rendering program can have many dependencies, including system utilities, runtime libraries, and so on, which each might have their own dependencies in turn. As with environment, metadata may be more efficiently managed in conjunction with a format registry either internal or external to a repository. In the absence of a global mechanism, repositories may be forced to develop their own local "registries" relating format to software environment.		

Semantic unit	swName		
Semantic components	None		
Definition	Manufacturer and title of the software application.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	Sybase	Adobe Photoshop Adobe Acrobat Reader	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	Include manufacturer when this helps to identify or disambiguate the product, for example, use “Adobe Photoshop” rather than “Photoshop.”		

Semantic unit	swVersion		
Semantic components	None		
Definition	The version or versions of the software referenced in swName.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		>=2.2.0 6.0 2000	
Repeatability	Not repeatable	Repeatable	Not repeatable
Obligation	Optional	Optional	Optional
Usage notes	If there is no formal version, the date of issuance can be used.		

Semantic unit	swType		
Semantic components	None		
Definition	Class or category of software.		
Rationale	Several different layers of software can be required to support an object.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	Suggested values: renderer = application that can display/play/execute the format instance, e.g., image viewer, video player, Java virtual machine (when the format instance is a java class file) ancillary = required ancillary software, e.g., run time libraries, browser plug-ins, compression/decompression routines, utilities, operating system emulators, etc. operatingSystem = software that supports application execution, process scheduling, memory management, file systems, etc. driver = software with the primary function of communicating between hardware and the operating system or other software		

Semantic unit	swOtherInformation		
Semantic components	None		
Definition	Additional requirements or instructions related to the software referenced in swName.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		Install acroread (Adobe Acrobat) first; copy nppdf.so (the plug-in) to your Mozilla plug-ins directory, and make sure a copy of (or symlink to) acroread is in your PATH.	
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	This could be a reliable persistent identifier or URI pointing to software documentation within or outside the repository.		

Semantic unit	swDependency		
Semantic components	None		
Definition	The name and, if applicable, version of any software component needed by the software referenced in swName in the context of using this object.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		GNU gcc >= 2.7.2	
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	The value should be constructed in a way that is consistent with the construction of swName and swVersion. This semantic unit identifies the software that is needed by what is recorded in swName, for example, a Perl script that depends on a Perl module. In this case the Perl script is listed in swName, with the module in swDependency within a software container.		

Semantic unit	hardware		
Semantic components	hwName, hwType, hwOtherInformation		
Definition	Hardware components needed by the software referenced in swName or the human user of the referenced software.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Creation/ Maintenance notes	Hardware environment information can be very difficult to provide. Many different hardware environments may apply; there are a huge number of combinations of maker and type of CPU, memory, video drivers, and so on. Although at least one hardware environment should be recorded, it is not necessary to record them all and each repository will have to make its own decisions about which hardware environments to record. Because of the difficulty recording this information comprehensively, it would be optimal if central registries of environment information existed. In many cases the environment of a file object is directly associated with the format, making registry lookup by format feasible. In the absence of a global mechanism, repositories may be forced to develop their own local “registries” relating format to hwEnvironment.		

Semantic unit	hwName		
Semantic components	None		
Definition	Manufacturer, model, and version (if applicable) of the hardware.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		Intel Pentium III 1 GB DRAM Windows XP-compatible joystick	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	Include manufacturer when this helps to identify or disambiguate the product. Include version for firmware or other components where that information is pertinent.		

Semantic unit	hwType		
Semantic components	None		
Definition	Class or category of the hardware.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	Suggested values: processor, memory, input/output device, storage device.		

Semantic unit	hwOtherInformation		
Semantic components	None		
Definition	Additional requirements or instructions related to the hardware referenced in hwName.		
Rationale	For hardware, the amount of computing resource needed (such as memory, storage, processor speed, etc.) may need to be documented. In addition, more detailed instructions may be needed to install and/or operate the hardware.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	32MB minimum	32MB minimum Required RAM for Apache is unknown	
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	This could be an identifier or URI used to point to hardware documentation.		

Semantic unit	signatureInformation		
Semantic components	signatureInformationEncoding, signer, signatureMethod, signatureValue, signatureValidationRules, signatureProperties, keyInformation		
Definition	Information needed to use a digital signature to authenticate the signer of an object and/or the information contained in the object.		
Rationale	A repository may have a policy of generating digital signatures for files on ingest, or may have a need to store and later validate incoming digital signatures.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	Several of the semantic components of signatureInformation are taken from the W3C's *XML-Signature Syntax and Processing*; see www.w3.org/TR/2002/REC-xmldsig-core-20020212/ for more information on the structure and application of these semantic units. (See also the discussion of digital signatures, page 4-6.)		

Semantic unit	signatureInformationEncoding		
Semantic components	None		
Definition	The encoding used for the values of signatureValue, keyInformation, certificateInformation.		
Rationale	These values cannot be interpreted correctly if the encoding is unknown.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		Base64 Ds:CrytoBinary	
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	signer		
Semantic components	None		
Definition	The individual, institution, or authority responsible for generating the signature.		
Rationale	The signer might also be carried in the keyInformation, but it can be accessed more conveniently if recorded here.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Usage notes	If the signer is an Agent known to the repository, an agentIdentifier can be used here.		

Semantic unit	signatureMethod		
Semantic components	None		
Definition	A designation for the encryption and hash algorithms used for signature generation.		
Rationale	The same algorithms must be used for signature validation.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		DSA-SHA1 RSA-SHA1	
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	Recommended practice is to encode the encryption algorithm first, followed by a hyphen, followed by the hash (message digest) algorithm.		

Semantic unit	signatureValue		
Semantic components	None		
Definition	The digital signature; a value generated from the application of a private key to a message digest.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		juS5RhJ884qoFR 8flVXd/rbrSDVGn 40CapgB7qeQiT +rr0NekEQ6BHh UA8dT3+BCTBU QI0dBjlml9lwzEN XvS83zRECjzXb MRTUtVZiPZG2p qKPnL2YU3A964 5UCjTXU+jgFum v7k78hieAGDzNc i+PQ9KRmm//icT 7JaYztgt4=	
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	signatureValidationRules		
Semantic components	None		
Definition	The operation to be performed as part of signature validation.		
Rationale	The repository should not assume that the procedure for validating any particular key will be known many years in the future without documentation.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability		Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	This may include the canonicalization method used before calculating the message digest, if the object was normalized before signing. This value could also be a pointer to archive documentation.		

Semantic unit	signatureProperties		
Semantic components	None		
Definition	Additional information about the generation of the signature.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	This may include the date/time of signature generation, the serial number of the cryptographic hardware used, or other information related to the generation of the signature. Repositories will likely want to define a suitably granular structure to signatureProperties.		

Semantic unit	keyInformation		
Semantic components	keyType, keyValue, keyVerificationInformation		
Definition	Information about the signer's public key needed to validate the digital signature.		
Rationale	To validate a digital signature for an object, one first recalculates the message digest for the object, and then uses the public key of the signer to verify that the value of the signature (signatureValue) is correct. The repository must therefore have the public key value and some assurance that it truly belongs to the signer.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional

Semantic unit	keyType		
Semantic components	None		
Definition	The type of key, denoted by the algorithm used to generate the key.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Examples		DSA RSA PGP SPKI	
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	keyValue		
Semantic components	None		
Definition	The value of the signer's public key.		
Rationale	The signer's public key might be included in the signer's X509 certificate, if this is recorded under keyVerificationInformation. However, since the key itself is necessary, it is useful to isolate it as a separate and required semantic unit.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	Different types of key will have different structures and parameters. Recommended practice is to represent key values following the W3C's *XML-Signature Syntax and Processing* (www.w3.org/TR/2002/REC-xmldsig-core-20020212/).		

Semantic unit	keyVerificationInformation		
Semantic components	None		
Definition	Additional information needed to verify the signer's public key used to validate the digital signature.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Usage notes	This may include a certificate or certificate chain, and/or a revocation list. Repositories will likely want to define a suitably granular structure to keyVerificationInformation.		

Semantic unit	relationship		
Semantic components	relationshipType, relationshipSubType, relatedObjectIdentification, relatedEventIdentification		
Definition	Information about a relationship between this object and one or more other objects.		
Rationale	A preservation repository must know how to assemble complex objects from component parts (structural relationships) and rigorously track digital provenance (derivation relationships). Documentation about relationships between different objects is crucial to these purposes.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	Most preservation repositories will want to record all relevant relationships. Many formats for representing structural information may be used instead of the semantic units specified here. This information must be known, and some implementations may know it by using other structures. Structural relationships at the file level are necessary to reconstruct a representation in order to ascertain that the representation is renderable. A record of structural relationships at the representation level may be necessary to render the representation. Structural relationships at the bitstream level can relate bitstreams within a file. Derivative relationships at the file and representation level are important for documenting digital provenance.		

Semantic unit	relationshipType		
Semantic components	None		
Definition	A high-level categorization of the nature of the relationship.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	Suggested values: structural = a relationship between parts of an object derivation = a relationship where one object is the result of a transformation performed on the related object A repository may find it necessary to define additional relationship types.		

Semantic unit	relationshipSubType		
Semantic components	None		
Definition	A specific characterization of the nature of the relationship documented in relationshipType.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	Suggested values: is child of = the object is directly subordinate in a hierarchy to the related object (Note that this is semantically equivalent to "Has parent," which may be preferred by some implementations. is parent of = the object is directly superior in a hierarchy to the related object (Note that this is semantically equivalent to "Has child," which may be preferred by some implementations. has sibling = the object shares a common parent with the related object is part of = the object is contained by the related object has part = the object contains the related object source of = the related object is a version of this object created by a transformation has root = for a representation only, the related object is the file that must be processed first in order to render the representation A repository may find it necessary to define more or less granular relationships. For derivation relationships, note that the precise relationship may be indicated by the type of the related event. For relationships between files and representations, use "has part" for the relationship of a representation to a file. Use "is part of" for the relationship of the file to the representation. The relationship "has root" is applicable only to the representation, because it implies that a compound object (i.e., one made up of multiple files) requires that one file be picked up first as its root to render it. In the metadata for the representation, "has root" identifies that particular file.		

Semantic unit	relatedObjectIdentification		
Semantic components	relatedObjectIdentifierType, relatedObjectIdentifierValue, relatedObjectSequence		
Definition	The identifier and sequential context of the related resource.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	The related object may or may not be held within the preservation repository. Recommended practice is that objects reside within the repository unless there is a good reason to reference an object outside. Internal and external references should be clear.		

Semantic unit	relatedObjectIdentifierType		
Semantic components	None		
Definition	A designation of the domain within which the identifier is unique.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	[see examples for objectIdentifierType]	[see examples for objectIdentifierType]	[see examples for objectIdentifierType]
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	If the related object is held within the preservation repository, this should be the value of that object's objectIdentifierType.		

Semantic unit	relatedObjectIdentifierValue		
Semantic components	None		
Definition	The value of the related object identifier.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	[see examples for objectIdentifierValue]	[see examples for objectIdentifierValue]	[see examples for objectIdentifierValue]
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	If the related object is held within the preservation repository, this should be the value of that object's objectIdentifierValue.		

Semantic unit	relatedObjectSequence		
Semantic components	None		
Definition	The order of the related object relative to other objects with the same type of relationship.		
Rationale	This semantic unit is particularly useful for structural relationships. In order to reconstruct a representation, it may be necessary to know the order of components with sibling or part-whole relationships. For example, to render a page-image book, it is necessary to know the order of files representing pages.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		1 2 3	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	This semantic unit could be implemented in several ways. It might be recorded explicitly in metadata as a sequence number or as a pointer. It might be implicit in some other ordering of objects, for example, incrementing identifier values. The value of relationshipSubType might imply the sequence (e.g., “is preceding sibling,” “is following sibling”). There is no requirement that sequence numbers must be unique or sequential. Some related objects have no inherent sequence, for example, unordered Web pages making up a Web site. In this case all related objects can be given the “dummy” sequence number zero.		

Semantic unit	relatedEventIdentification		
Semantic components	relatedEventIdentifierType, relatedEventIdentifierValue, relatedEventSequence		
Definition	The identifier and contextual sequence of an event associated with the relationship.		
Rationale	An object may be related to another object because of an event, for example, migration.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	relatedEventIdentifierType		
Semantic components	None		
Definition	The eventIdentifierType of the related event.		
Data constraint	Must be an existing eventIdentifierType value.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	[see examples for eventIdentifierType]	[see examples for eventIdentifierType]	[see examples for eventIdentifierType]
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	For most preservation repositories, the eventIdentifierType will simply be their own internal numbering system. It can be implicit within the system and provided explicitly only if the data is exported.		

Semantic unit	relatedEventIdentifierValue		
Semantic components	None		
Definition	The eventIdentifierValue of the related event.		
Data constraint	Must be an existing eventIdentifierValue value.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	[see examples for eventIdentifierValue]	[see examples for eventIdentifierValue]	[see examples for eventIdentifierValue]
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	relatedEventSequence		
Semantic components	None		
Definition	The order of the related event.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		1 2 3	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Optional	Optional	Optional
Usage notes	The sequence of a related event can be inferred from the eventDateTime associated with the related event.		

Semantic unit	linkingEventIdentifier		
Semantic components	linkingEventIdentifierType, linkingEventIdentifierValue		
Definition	The eventIdentifier of an event associated with the object.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	Use to link to events that are not associated with relationships between objects, such as format validation, virus checking, etc.		

Semantic unit	linkingEventIdentifierType		
Semantic components	None		
Definition	The eventIdentifierType value of the related event.		
Data constraint	Must be an existing eventIdentifierType value.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	[see examples for eventIdentifierType]	[see examples for eventIdentifierType]	[see examples for eventIdentifierType]
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	For most preservation repositories, the eventIdentifierType will simply be their own internal numbering system. It can be implicit within the system and provided explicitly only if the data is exported.		

Semantic unit	linkingEventIdentifierValue		
Semantic components	None		
Definition	The eventIdentifierValue value of the related event.		
Data constraint	Must be an existing eventIdentifierValue value.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	[see examples for eventIdentifierValue]	[see examples for eventIdentifierValue]	[see examples for eventIdentifierValue]
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	linkingIntellectualEntityIdentifier		
Semantic components	linkingIntellectualEntityIdentifierType, linkingIntellectualEntityIdentifierValue		
Definition	An identifier for an Intellectual Entity associated with the object.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	Use to link to an Intellectual Entity that is related to the object. This may be a link to descriptive metadata that describes the Intellectual Entity or some other surrogate for it that can be referenced. This link will likely be to an identifier of an object that is at a higher conceptual level than the object for which the metadata is provided, for example, to a collection or parent object.		

Semantic unit	linkingIntellectualEntityIdentifierType		
Semantic components	None		
Definition	A designation of the domain within which the linking intellectual entity identifier is unique.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		URI LCCN	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	linkingIntellectualEntityIdentifierValue		
Semantic components	None		
Definition	The value of the linkingIntellectualEntityIdentifier.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples	hdl:loc.natlib/mrva0002.0495 info:lccn/19018302		
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	linkingPermissionStatementIdentifier		
Semantic components	linkingPermissionStatementIdentifierType, linkingPermissionStatementIdentifierValue		
Definition	An identifier for a permission statement associated with the object.		
Rationale	A repository may choose to link from a permission statement to an object or from an object to a permission statement or both.		
Data constraint	Container		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional

Semantic unit	linkingPermissionStatementIdentifierType		
Semantic components	None		
Definition	A designation of the domain within which the linkingPermission StatementIdentifier is unique.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Examples		URI LCCN	
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	linkingPermissionStatementIdentifierValue		
Semantic components	None		
Definition	The value of the linkingPermissionStatementIdentifier.		
Data constraint	None		
Object category	Representation	File	Bitstream
Applicability	Applicable	Applicable	Applicable
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Event Entity

The Event entity aggregates information about an action that involves one or more Object entities. Metadata about an Event would normally be recorded and stored separately from the digital object.

Whether or not a preservation repository records an Event depends upon the importance of the event. Actions that modify objects should always be recorded. Other actions such as copying an object for backup purposes may be recorded in system logs or an audit trail but not necessarily in an Event entity.

Mandatory semantic units are: *eventIdentifier*, *eventType*, *eventDateTime*.

Entity properties

- Must be related to one or more objects.
- Can be related to one or more agents.

Entity semantic units

- eventIdentifier
  - eventIdentifierType
  - eventIdentifierValue
- eventType
- eventDateTime
- eventDetail
- eventOutcomeInformation
  - eventOutcome
  - eventOutcomeDetail
- linkingAgentIdentifier
  - linkingAgentIdentifierType
  - linkingAgentIdentifierValue
  - linkingAgentRole
- linkingObjectIdentifier
  - linkingObjectIdentifierType
  - linkingObjectIdentifierValue

Semantic unit	eventIdentifier
Semantic components	eventIdentifierType, eventIdentifierValue
Definition	A designation used to uniquely identify the event within the preservation repository system.
Rationale	Each event recorded by the preservation archive must have a unique identifier to allow it to be related to objects, agents, and other events.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Mandatory
Creation/ Maintenance notes	The eventIdentifier is likely to be system generated. There is no global scheme or standard for event identifiers.

Semantic unit	eventIdentifierType
Semantic components	None
Definition	A designation of the domain within which the event identifier is unique.
Data constraint	None
Examples	FDA Stanford Repository Event ID UUID
Repeatability	Not repeatable
Obligation	Mandatory
Creation/ Maintenance notes	For most preservation repositories, the eventIdentifierType will be their own internal numbering system. It can be implicit within the system and provided explicitly only if the data is exported.

Semantic unit	eventIdentifierValue
Semantic components	None
Definition	The value of the eventIdentifier.
Data constraint	None
Examples	[a binary integer] E-2004-11-13-000119 58f202ac-22cf-11d1-b12d-002035b29092
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	eventType
Semantic components	None
Definition	A categorization of the nature of the event.
Rationale	Categorizing events will aid the preservation repository in machine processing of event information, particularly in reporting.
Data constraint	Value should be taken from a controlled vocabulary.
Examples	E77 [a code used within a repository for a particular event type] Ingest
Repeatability	Not repeatable
Obligation	Mandatory
Usage notes	Each repository should define its own controlled vocabulary of eventType values. A suggested starter list for consideration (see also the Glossary for more detailed definitions): capture = the process whereby a repository actively obtains an object compression = the process of coding data to save storage space or transmission time deaccession = the process of removing an object from the inventory of a repository decompression = the process of reversing the effects of compression decryption = the process of converting encrypted data to plaintext deletion = the process of removing an object from repository storage digital signature validation = the process of determining that a decrypted digital signature matches an expected value dissemination = the process of retrieving an object from repository storage and making it available to users fixity check = the process of verifying that an object has not been changed in a given period ingestion = the process of adding objects to a preservation repository message digest calculation = the process by which a message digest ("hash") is created migration = a transformation of an object creating a version in a more contemporary format normalization = a transformation of an object creating a version more conducive to preservation replication = the process of creating a copy of an object that is, bit-

	wise, identical to the original validation = the process of comparing an object with a standard and noting compliance or exceptions virus check = the process of scanning a file for malicious programs The level of specificity in recording the type of event (e.g., whether the eventType indicates a transformation, a migration or a particular method of migration) is implementation specific and will depend upon how reporting and processing is done. Recommended practice is to record detailed information about the event itself in eventDetail rather than using a very granular value for eventType.

Semantic unit	**eventDateTime**
Semantic components	None
Definition	The single date and time, or date and time range, at or during which the event occurred.
Data constraint	Any date/time convention may be used, as long as it is consistent and can be translated into ISO 8601 for export if necessary.
Examples	20050704T071530-0500 [July 4, 2005 at 7:15:30 a.m. EST] 2006-07-16T19:20:30+01:00 20050705T0715-0500/20050705T0720-0500 [from 7:15 a.m. EST to 7:20 a.m. EST on July 4, 2005] 2004-03-17 [March 17, 2004, only the date is known]
Repeatability	Not repeatable
Obligation	Mandatory
Usage notes	Recommended practice is to record the most specific time possible and to designate the time zone.

Semantic unit	eventDetail
Semantic components	None
Definition	Additional information about the event.
Data constraint	None
Examples	Object permanently withdrawn by request of Caroline Hunt. Program="MIGJP2JP2K";version="2.2"
Repeatability	Not repeatable
Obligation	Optional
Usage notes	eventDetail is not intended to be processed by machine. It may record any information about an event and/or point to information stored elsewhere.

Semantic unit	eventOutcomeInformation
Semantic components	eventOutcome, eventOutcomeDetail
Definition	Information about the outcome of an event.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	A repository may wish to supplement a coded eventOutcome value with additional information in eventOutcomeDetail. Since events may have more than one outcome, the container is repeatable.

Semantic unit	eventOutcome
Semantic components	None
Definition	A categorization of the overall result of the event in terms of success, partial success, or failure.
Rationale	A coded way of representing the outcome of an event may be useful for machine processing and reporting. If, for example, a fixity check fails, the event record provides both an actionable and a permanent record.
Data constraint	Value should be taken from a controlled vocabulary.
Examples	00 [a code meaning “action successfully completed”] CV-01 [a code meaning “checksum validated”]
Repeatability	Not repeatable
Obligation	Optional
Usage notes	Recommended practice is to use controlled vocabulary that a system can act upon automatically. More detail about the outcome may be recorded in eventOutcomeDetail. Recommended practice is to define events with sufficient granularity that each event has a single outcome.

Semantic unit	eventOutcomeDetail
Semantic components	None
Definition	A non-coded detailed description of the result or product of the event.
Rationale	An event outcome may be sufficiently complex that a coded description is not adequate to document it.
Data constraint	None
Examples	LZW compressed file Non-standard tags found in header
Repeatability	Not repeatable
Obligation	Optional
Usage notes	This may be used to record all error and warning messages issued by a program involved in the event or to record a pointer to an error log. If the event was a validity check (e.g., profile conformance) any anomalies or quirks discovered would be recorded here.

Semantic unit	linkingAgentIdentifier
Semantic components	linkingAgentIdentifierType, linkingAgentIdentifierValue, linkingAgentRole
Definition	Information about an agent associated with an event.
Rationale	Digital provenance requires often that relationships between agents and events are documented.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	Recommended practice is to record the agent if possible.

Semantic unit	linkingAgentIdentifierType
Semantic components	None
Definition	A designation of the domain in which the linking agent identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary.
Examples	[see examples for agentIdentifierType]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	linkingAgentIdentifierValue
Semantic components	None
Definition	The value of the linking agent identifier.
Data constraint	None
Examples	[see examples for agentIdentifierValue]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	linkingAgentRole
Semantic components	None
Definition	The role of the agent in relation to this event.
Rationale	Events can have more than one agent associated with them. The role of each agent may need to be documented.
Data constraint	Values should be taken from a controlled vocabulary.
Examples	Authorizer Implementer Validator Executing program
Repeatability	Repeatable
Obligation	Optional

Semantic unit	linkingObjectIdentifier
Semantic components	linkingObjectIdentifierType, linkingObjectIdentifierValue
Definition	Information about an object associated with an event.
Rationale	Digital provenance often requires that relationships between objects and events are documented.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional

Semantic unit	linkingObjectIdentifierType
Semantic components	None
Definition	A designation of the domain in which the linking object identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary.
Examples	[see examples for objectIdentifierType]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	linkingObjectIdentifierValue
Semantic components	None
Definition	The value of the linking object identifier.
Data constraint	None
Examples	[see examples for objectIdentifierValue]
Repeatability	Not repeatable
Obligation	Mandatory

Agent Entity

The Agent entity aggregates information about attributes or characteristics of agents (persons, organizations, or software) associated with rights management and preservation events in the life of a data object. Agent information serves to identify an agent unambiguously from all other Agent entities.

The only mandatory semantic unit is agentIdentifier.

Entity properties

- May hold or grant one or more rights.
- May carry out, authorize, or compel one or more events.
- May create or act upon one or more objects.

Entity semantic units

- agentIdentifier
  - agentIdentifierType
  - agentIdentifierValue
- agentName
- agentType

Semantic unit	agentIdentifier
Semantic components	agentIdentifierType, agentIdentifierValue
Definition	The designation used to uniquely identify the agent within a preservation repository system.
Rationale	Each agent associated with the preservation repository must have a unique identifier to allow it to be related to events and permission statements.
Data constraint	Container
Repeatability	Repeatable
Obligation	Mandatory

Semantic unit	agentIdentifierType
Semantic components	None
Definition	A designation of the domain in which the agent identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary.
Examples	LCNAF SAN MARC Organization Codes URI
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	agentIdentifierValue
Semantic components	None
Definition	The value of the agentIdentifier.
Data constraint	None
Examples	92-79971 Owens, Erik C. 234-5676 MH-CS info:lccn/n78890351
Repeatability	Not repeatable
Obligation	Mandatory
Usage notes	May be a unique key or a controlled textual form of name.

Semantic unit	agentName
Semantic components	None
Definition	A text string which could be used in addition to agentIdentifer to identify an agent.
Rationale	This semantic unit provides a more reader-friendly version of the agent identified by the agentIdentifier.
Data constraint	None
Examples	Erik Owens Woodyard Pc
Repeatability	Repeatable
Obligation	Optional
Usage notes	The value is not necessarily unique.

Semantic unit	agentType
Semantic components	None
Definition	A high-level characterization of the type of agent.
Data constraint	Value should be taken from a controlled vocabulary.
Repeatability	Not repeatable
Obligation	Optional
Usage notes	Suggested values: person organization software

Rights Entity

For the purpose of the PREMIS Data Dictionary, statements of rights and permissions are taken to be constructs that can be described as the Rights entity. Rights are entitlements allowed to agents by copyright or other intellectual property law. Permissions are powers or privileges granted by agreement between a rightsholder and another party or parties.

A repository might wish to record a variety of rights information including abstract rights statements and statements of permissions that apply to external agents and to objects not held within the repository. The minimum core rights information that a preservation repository must know, however, is what permissions have been granted to the repository itself to carry out actions related to objects within the repository.

If the repository records rights information, the *permissionStatementIdentifier*, *linkingObject*, and *permissionGranted* are mandatory.

Entity properties

- Must be related to one or more objects.
- Must be related to one or more agents.

Entity semantic units

- permissionStatement
  - permissionStatementIdentifier
    - permissionStatementIdentifierType
    - permissionStatementIdentifierValue
  - linkingObject
  - grantingAgent
  - grantingAgreement
    - grantingAgreementIdentification
    - grantingAgreementInformation
  - permissionGranted
    - act
    - restriction
    - termOfGrant
      - startDate
      - endDate
    - permissionNote

Semantic unit	permissionStatement
Semantic components	permissionStatementIdentifier, linkingObject, grantingAgent, grantingAgreement, permissionGranted
Definition	An agreement with a rightsholder that allows a repository to take action(s) related to objects in the repository.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	If repository wants to control what actions can be taken on an object-by-object basis, it will want to record these. Some archives might have an institution-wide policy. The unit is optional because institutions may have other means to give a permission statement.

Semantic unit	permissionStatementIdentifier
Semantic components	permissionStatementIdentifierType, permissionStatementIdentifierValue
Definition	A designation used to identify the permission statement uniquely within the preservation repository system.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	permissionStatementIdentifierType
Semantic components	None
Definition	A designation of the domain within which the permission identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary.
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	permissionStatementIdentifierValue
Semantic components	None
Definition	The value of the permissionStatementIdentifier.
Data constraint	None
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	linkingObject
Semantic components	None
Definition	An identifying designation for the object or objects to which the permission pertains.
Data constraint	None
Examples	iu2440 application/pdf all 0000000312
Repeatability	Repeatable
Obligation	Mandatory
Usage notes	This could be the objectIdentifierValue of a specific object, or an identifying designation for a class of objects, such as all objects of a particular type, or owned by a particular agent. The linking object may be a representation, file, or bitstream.

Semantic unit	grantingAgent
Semantic components	None
Definition	An identifying designation for the agent or agents granting the permission.
Data constraint	None
Repeatability	Repeatable
Obligation	Optional
Usage notes	If the agent granting the permission is described as an entity within the repository system, this designation should be the agentIdentifier of the agent.

Semantic unit	grantingAgreement
Semantic components	grantingAgreementIdentification, grantingAgreementInformation
Definition	The agreement by which the permission was granted.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional
Usage notes	This semantic unit is intended to refer to a document recording the granting of permission. For some repositories this may be a formal signed contract with a customer. In other cases this may be e-mail or other informal communication. If the granting agreement is verbal, this could point to a memo by the repository documenting the verbal agreement.

Semantic unit	grantingAgreementIdentification
Semantic components	None
Definition	An identifying designation for an agreement by which the permission was granted.
Data constraint	None
Repeatability	Not repeatable
Obligation	Optional
Usage notes	This semantic unit should be the means by which the repository uniquely identifies the granting agreement. It may be a formal identifier with type and value or a more informal designation.

Semantic unit	grantingAgreementInformation
Semantic components	None
Definition	Text describing the agreement by which the permission was granted.
Data constraint	None
Repeatability	Not repeatable
Obligation	Optional
Usage notes	This could contain the actual text of the agreement, a paraphrase, or other information describing the agreement or its content.

Semantic unit	permissionGranted
Semantic components	act, restriction, termOfGrant, permissionNote
Definition	The action(s) that the grantingAgency has allowed the repository.
Data constraint	Container
Repeatability	Repeatable
Obligation	Mandatory

Semantic unit	act
Semantic components	None
Definition	The action the preservation repository is allowed to take.
Data constraint	Value should be taken from a controlled vocabulary.
Repeatability	Not repeatable
Obligation	Mandatory
Usage notes	Suggested values: replicate = make an exact copy migrate = make a copy identical in content in a different file format modify = make a version different in content use = read without copying or modifying (e.g., to validate a file or run a program) disseminate = create a DIP for use outside of the preservation repository delete = remove from the repository It is up to the preservation repository to decide how granular the controlled vocabulary should be. It may be useful to employ the same controlled values that the repository uses for eventType.

Semantic unit	restriction
Semantic components	None
Definition	A condition or limitation on the act.
Data constraint	None
Examples	No more than three Allowed only after one year of archival retention has elapsed Rightsholder must be notified after completion of act
Repeatability	Repeatable
Obligation	Optional

Semantic unit	termOfGrant
Semantic components	startDate, endDate
Definition	The time period for the permissions granted.
Rationale	The permission to preserve may be time bounded.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	startDate
Semantic components	None
Definition	The beginning date of the permission granted.
Data constraint	Value should be formatted according to ISO 8601.
Examples	2006-01-02 20050723
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	endDate
Semantic components	None
Definition	The ending date of the permission granted.
Data constraint	Value should be formatted according to ISO 8601.
Examples	2010-01-02 20120723
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	permissionNote
Semantic components	None
Definition	Additional information about the permissions.
Rationale	A textual description of the permissions may be needed for additional explanation.
Data constraint	None
Repeatability	Repeatable
Obligation	Optional
Usage notes	This semantic unit may include a statement about risk assessment, for example, when a repository is not certain about what permissions have been granted.

This page intentionally blank.

3. EXAMPLES

These examples illustrate both simple and complex applications of PREMIS-conformant metadata. They are as real as possible—the objects being described do exist and the values for the semantic units are correct and plausible. In some cases the objects exist in a working preservation repository, and metadata used in the repository application were mapped to PREMIS semantic units.

Example 1 is a document that is complete in a single word-processing file, ingested into a repository that manages both representations and files. The values of PREMIS semantic units were supplied by mapping metadata actually recorded in the OCLC Digital Archive.

Example 2 is slightly more complex, describing a dissertation consisting of one PDF file and one MP3 file. Because MP3 is not one of the repository's preferred formats, it created a WAVE file from the MP3.

In Example 3, the archived object is a tar file that contains within it a file produced by the application QuarkXPress (file extension .qxp), which in turn contains (links to) an Encapsulated PostScript (EPS) file. The source file for the EPS file is a Macromedia FreeHand MX file also included in the tar file.

Example 4 shows one way of representing a harvested Web site, where the entire snapshot is considered a representation and each captured page is considered an equal part of the whole.

Example 5 shows only the semantic units pertaining to digital signatures, to illustrate how this information might be recorded. In this example, the values are formatted plausibly but are not necessarily real.

Example 6 shows PREMIS metadata for two files: a TIFF image and an XML file containing descriptive metadata for the image. Both files are stored in the preservation repository.

For additional examples, see the PREMIS maintenance activity Web site at www.loc.gov/standards/premis/.

Example 1: Microsoft Word Document Complete in One File

This example shows a mapping of metadata elements and values used by the OCLC Digital Archive to PREMIS. It does not represent all possible elements used in the Digital Archive, nor does it represent how the elements are stored in the Digital Archive. For a list of all OCLC Digital Archive elements see www.oclc.org/support/documentation/digitalarchive/da_metadata_elements/.

Metadata in the OCLC Digital Archive comes from a variety of sources. Some is ingested into the Archive with the object. Some is created by the Archive as a result of internal processes. For example, *formatVersion* is supplied for some format types by the Archive using JHOVE 1.0 software. Some values are not stored in the Digital Archive but can be supplied on output. These are noted as "Implicit" below. There is no Rights information because OCLC stores the Terms and Conditions outside the Digital Archive.

Example 1, Object 1: the representation				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			OCLC Object Identifier [implicit in OCLC]
objectIdentifier	objectIdentifierValue			0000009605
preservationLevel				store
objectCategory				representation
objectCharacteristics	compositionLevel			n/a
objectCharacteristics	fixity	messageDigestAlgorithm		n/a
objectCharacteristics	fixity	messageDigest		n/a
objectCharacteristics	fixity	messageDigestOriginator		n/a
objectCharacteristics	size			n/a
objectCharacteristics	format	formatDesignation	formatName	n/a
objectCharacteristics	format	formatDesignation	formatVersion	n/a
objectCharacteristics	format	formatRegistry	formatRegistryName	n/a
objectCharacteristics	format	formatRegistry	formatRegistryKey	n/a
objectCharacteristics	format	formatRegistry	formatRegistryRole	n/a
objectCharacteristics	significantProperties			n/a
objectCharacteristics	inhibitors	inhibitorType		n/a
objectCharacteristics	inhibitors	inhibitorTarget		n/a
objectCharacteristics	inhibitors	inhibitorKey		n/a

Example 1, Object 1: the representation				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
creatingApplication	creatingApplication Name			
creatingApplication	creatingApplication Version			
creatingApplication	dateCreatedBy Application			
originalName				http://www.archivesdept.org/annualreports2000.doc
storage	contentLocation	contentLocationType		Open URL [implicit in OCLC]
storage	contentLocation	contentLocationValue		http://digitalarchive.oclc.org/request?pid%3Dobjid%3A0000003174
storage	storageMedium			hard disk [implicit in OCLC]
environment	environment Characteristic			current [equivalent to “known to work”]
environment	environmentPurpose			render [implicit in OCLC]
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		MSWindows
environment	software	swVersion		2000
environment	software	swType		operating system
environment	software	swOtherInformation		
environment	software	swDependency		
environment	software	swName		MSWord
environment	software	swVersion		2000
environment	software	swType		application
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		Celeron

3. Examples

Example 1, Object 1: the representation				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
environment	hardware	hwType		processor
environment	hardware	hwOtherInformation		512 MB RAM
signatureInformation	signatureInformation Encoding			n/a
signatureInformation	signatureMethod			n/a
signatureInformation	signatureValue			n/a
signatureInformation	signatureValidation Rules			n/a
signatureInformation	signatureProperties			n/a
signatureInformation	keyInformation	keyType		n/a
signatureInformation	keyInformation	keyValue		n/a
signatureInformation	keyInformation	keyVerificationInformation		n/a
relationship	relationshipType			structural
relationship	relationshipSubType			has part
relationship	relatedObject Identification	relatedObjectIdentifierType		OCLC Object Identifier [implicit in OCLC]
relationship	relatedObject Identification	relatedObjectIdentifierValue		0000210958
relationship	relatedObject Identification	relatedObjectSequence		1
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			OCLC Event Identifier [implicit in OCLC]
linkingEventIdentifier	linkingEventIdentifier Value			00000033393
linkingEventIdentifier	linkingEventIdentifier Type			OCLC Event Identifier [implicit in OCLC]

Example 1, Object 1: the representation				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingEventIdentifier	linkingEventIdentifier Value			00000034287
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			OCLC Bibliographic Identifier [implicit in OCLC]
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			12345678
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 1, Object 2: the file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			OCLC Object Identifier [implicit in OCLC]
objectIdentifier	objectIdentifierValue			0000210958
preservationLevel				Store
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		Adler-32
objectCharacteristics	fixity	messageDigest		7c9b35da
objectCharacteristics	fixity	messageDigestOriginator		OCLC [implicit in OCLC]
objectCharacteristics	size			230400
objectCharacteristics	format	formatDesignation	formatName	MSWORD
objectCharacteristics	format	formatDesignation	formatVersion	2000
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	

3. Examples

Example 1, Object 2: the file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectCharacteristics	significantProperties			
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			Microsoft Word
creatingApplication	creatingApplication Version			2000
creatingApplication	dateCreatedBy Application			unknown
originalName				
storage	contentLocation	contentLocationType		content management foldering system [implicit in OCLC]
storage	contentLocation	contentLocationValue		/0/0000009605/file.doc
storage	storageMedium			hard disk [implicit in OCLC]
environment	environment Characteristic			
environment	environmentPurpose			
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		
environment	software	swVersion		
environment	software	swType		
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		
environment	hardware	hwType		
environment	hardware	hwOtherInformation		

Example 1, Object 2: the file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
signatureInformation	signatureInformation Encoding			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	keyVerificationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			is part of
relationship	relatedObject Identification	relatedObjectIdentifierType		OCLC Object Identifier [implicit in OCLC]
relationship	relatedObject Identification	relatedObjectIdentifierValue		0000009605
relationship	relatedObject Identification	relatedObjectSequence		1
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			OCLC Event Identifier [implicit in OCLC]
linkingEventIdentifier	linkingEventIdentifier Value			00000034000
linkingEventIdentifier	linkingEventIdentifier Type			OCLC Event Identifier [implicit in OCLC]
linkingEventIdentifier	linkingEventIdentifier Value			00000034200
linkingEventIdentifier	linkingEventIdentifier Type			OCLC Event Identifier [implicit in OCLC]

3. Examples

Example 1, Object 2: the file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingEventIdentifier	linkingEventIdentifier Value			00000033893
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 1, Event 1		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	OCLC Event Identifier [implicit in OCLC]
eventIdentifier	eventIdentifierValue	00000033393
eventType		Ingest
eventDateTime		2004-12-05 07:00:41.0
eventDetail		
eventOutcome Information	eventOutcome	Status_Success
eventOutcome Information	eventOutcomeDetail	[Contains database ID for XML-encoded data elements composing the ingest report: object ID, metadata record ID, files composing the object, fixity, and virus details.]
linkingAgentIdentifier	linkingAgentIdentifierType	OCLC Institution Number [implicit in OCLC]
linkingAgentIdentifier	linkingAgentIdentifierValue	28765
linkingAgentIdentifier	linkingAgentRole	Implementer [implicit in OCLC]
linkingObjectIdentifier	linkingObjectIdentifierType	
linkingObjectIdentifier	linkingObjectIdentifierValue	

Example 1, Event 2		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	OCLC Event Identifier [implicit in OCLC]
eventIdentifier	eventIdentifierValue	00000034000
eventType		Fixity check
eventDateTime		2004-12-05 07:00:55.0
eventDetail		
eventOutcome Information	eventOutcome	Status_Success
eventOutcome Information	eventOutcomeDetail	[Contains XML-encoded Date, object ID, any errors found.]
linkingAgentIdentifier	linkingAgentIdentifierType	OCLC Institution Number [implicit in OCLC]
linkingAgentIdentifier	linkingAgentIdentifierValue	0 [implicit in OCLC]
linkingAgentIdentifier	linkingAgentRole	Implementer [implicit in OCLC]
linkingObjectIdentifier	linkingObjectIdentifierType	
linkingObjectIdentifier	linkingObjectIdentifierValue	

Example 1, Event 3		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	OCLC Event Identifier [implicit in OCLC]
eventIdentifier	eventIdentifierValue	00000034200
eventType		Virus check
eventDateTime		2004-12-05 07:00:56.0
eventDetail		n/a
eventOutcome Information	eventOutcome	Status_Success
eventOutcome Information	eventOutcomeDetail	[Contains XML-encoded Date, object ID, any errors found.]
linkingAgentIdentifier	linkingAgentIdentifierType	OCLC Institution Number [implicit in OCLC]
linkingAgentIdentifier	linkingAgentIdentifierValue	0 [implicit in OCLC]
linkingAgentIdentifier	linkingAgentRole	Implementer [implicit in OCLC]

Example 1, Event 3		
EVENT		
semantic unit	**semantic unit**	**Value**
linkingObjectIdentifier	linkingObjectIdentifierType	
linkingObjectIdentifier	linkingObjectIdentifierValue	

Example 1, Event 4		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	OCLC Event Identifier [implicit in OCLC]
eventIdentifier	eventIdentifierValue	00000034287
eventType		Object validation
eventDateTime		2004-12-05 07:00:60.0
eventDetail		
eventOutcome Information	eventOutcome	Status_Success
eventOutcome Information	eventOutcomeDetail	[Contains XML-encoded Date, object ID, any errors found.]
linkingAgentIdentifier	linkingAgentIdentifierType	OCLC Institution Number [implicit in OCLC]
linkingAgentIdentifier	linkingAgentIdentifierValue	0 [implicit in OCLC]
linkingAgentIdentifier	linkingAgentRole	Implementer [implicit in OCLC]
linkingObjectIdentifier	linkingObjectIdentifierType	
linkingObjectIdentifier	linkingObjectIdentifierValue	

Example 1, Event 5		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	OCLC Event Identifier [implicit in OCLC]
eventIdentifier	eventIdentifierValue	00000033893
eventType		Annotation validation
eventDateTime		2004-12-05 07:01:02.0

eventDetail		
eventOutcome Information	eventOutcome	Status_Complete
eventOutcome Information	eventOutcomeDetail	[Contains XML-encoded Date, object ID, any errors found.]
linkingAgentIdentifier	linkingAgentIdentifierType	OCLC Institution Number [implicit in OCLC]
linkingAgentIdentifier	linkingAgentIdentifierValue	0 [implicit in OCLC]
linkingAgentIdentifier	linkingAgentRole	Implementer [implicit in OCLC]
linkingObjectIdentifier	linkingObjectIdentifierType	
linkingObjectIdentifier	linkingObjectIdentifierValue	

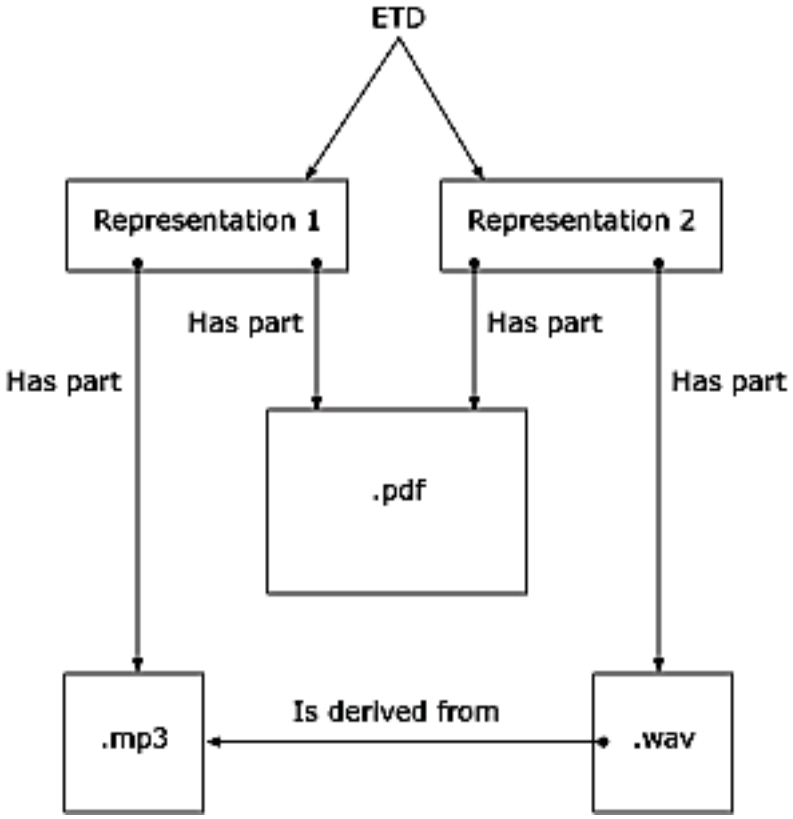
Example 1, Agent 1		
AGENT		
semantic unit	**semantic unit**	**Value**
agentIdentifier	agentIdentifierType	OCLC Institution Number [implicit in OCLC]
agentIdentifier	agentIdentifierValue	0 [implicit in OCLC]
agentName		OCLC [implicit in OCLC]
agentType		Organization [implicit in OCLC]

Example 1, Agent 2		
AGENT		
semantic unit	**semantic unit**	**Value**
agentIdentifier	agentIdentifierType	OCLC Institution Number [implicit in OCLC]
agentIdentifier	agentIdentifierValue	28765
agentName		Connecticut State Library [implicit in OCLC]
agentType		Organization [implicit in OCLC]

Example 2: ETD

This example is an electronic dissertation (ETD) stored in a repository called the Florida Digital Archive (FDA). The ETD consists of two files: a PDF and an MP3, which is actually an appendix to the dissertation. Because MP3 is not a format "preferred" by the repository, the FDA creates a WAVE file from the MP3.

On ingest, the FDA will create three Object descriptions: one for the representation (the dissertation as a whole), one for the PDF, and one for the MP3. When the WAVE file is created, the FDA creates an Object description for the new WAVE file and an object description for the new representation, which consists of the original PDF and the WAVE.



Example 2, Object 1: the original representation				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			FDA-R
objectIdentifier	objectIdentifierValue			R-2005-000217
preservationLevel				Mixed
objectCategory				representation
objectCharacteristics	compositionLevel			n/a
objectCharacteristics	fixity	messageDigestAlgorithm		n/a
objectCharacteristics	fixity	messageDigest		n/a
objectCharacteristics	fixity	messageDigestOriginator		n/a
objectCharacteristics	size			n/a
objectCharacteristics	format	formatDesignation	formatName	n/a
objectCharacteristics	format	formatDesignation	formatVersion	n/a
objectCharacteristics	format	formatRegistry	formatRegistryName	n/a
objectCharacteristics	format	formatRegistry	formatRegistryKey	n/a
objectCharacteristics	format	formatRegistry	formatRegistryRole	n/a
objectCharacteristics	significantProperties			n/a
objectCharacteristics	inhibitors	inhibitorType		n/a
objectCharacteristics	inhibitors	inhibitorTarget		n/a
objectCharacteristics	inhibitors	inhibitorKey		n/a
creatingApplication	creatingApplication Name			
creatingApplication	creatingApplication Version			
creatingApplication	dateCreatedBy Application			
originalName				n/a
storage	contentLocation	contentLocationType		n/a
storage	contentLocation	contentLocationValue		n/a
storage	storageMedium			n/a
environment	environment Characteristic			known to work

Example 2, Object 1: the original representation				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Mozilla Firefox
environment	software	swVersion		1.0
environment	software	swType		renderer
environment	software	swOtherInformation		requires swDependencies as plug-ins
environment	software	swDependency		Adobe Acrobat Reader 7.0
environment	software	swDependency		RealPlayer 10
environment	software	swName		Windows NT
environment	software	swVersion		5.0
environment	software	swType		operatingSystem
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		Intel Pentium II
environment	hardware	hwType		processor
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			n/a
signatureInformation	signer			n/a
signatureInformation	signatureMethod			n/a
signatureInformation	signatureValue			n/a
signatureInformation	signatureValidation Rules			n/a
signatureInformation	signatureProperties			n/a
signatureInformation	keyInformation	keyType		n/a
signatureInformation	keyInformation	keyValue		n/a

Example 2, Object 1: the original representation				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
signatureInformation	keyInformation	keyVerificationInformation		n/a
relationship	relationshipType			structural
relationship	relationshipSubType			Has part
relationship	relatedObject Identification	relatedObjectIdentifierType		FDA-DF
relationship	relatedObject Identification	relatedObjectIdentifierValue		DF-2005-001002 [the identifier of the stored PDF file]
relationship	relatedObject Identification	relatedObjectSequence		
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
relationship	relationshipType			structural
relationship	relationshipSubType			Has part
relationship	relatedObject Identification	relatedObjectIdentifierType		FDA-DF
relationship	relatedObject Identification	relatedObjectIdentifierValue		DF-2005-001003 [the identifier of the stored MP3 file]
relationship	relatedObject Identification	relatedObjectSequence		
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			FDA-E
linkingEventIdentifier	linkingEventIdentifier Value			E-2005-863740 [the identifier of the Ingest event]

Example 2, Object 1: the original representation				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			URL
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			www.fcla.edu/catalog/bibkey/AAA1234 [the URL of the MARC record describing the dissertation]
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 2, Object 2: the PDF file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			FDA-DF
objectIdentifier	objectIdentifierValue			DF-2005-001002
preservationLevel				Full
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		SHA-1
objectCharacteristics	fixity	messageDigest		7c9b35da4f2ebd436f1cf88e5a39b3a257edf4a22be3c955ac49da2e2107b67a1924419563
objectCharacteristics	fixity	messageDigestOriginator		submitter
objectCharacteristics	size			1132321
objectCharacteristics	format	formatDesignation	formatName	PDF
objectCharacteristics	format	formatDesignation	formatVersion	1.4
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	

Example 2, Object 2: the PDF file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			Adobe Acrobat
creatingApplication	creatingApplication Version			5.0
creatingApplication	dateCreatedBy Application			2004
originalName				main.pdf
storage	contentLocation	contentLocationType		FDA
storage	contentLocation	contentLocationValue		fda/prod/data/out/classa/ DF-2005-001002
storage	storageMedium			3590 [a kind of tape unit]
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Adobe Acrobat Reader
environment	software	swVersion		6.1
environment	software	swType		renderer
environment	software	swOtherInformation		
environment	software	swDependency		Windows NT
environment	software	swDependency		Mozilla Firefox 1.0
environment	hardware	hwName		Intel Pentium II

Example 2, Object 2: the PDF file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
environment	hardware	hwType		processor
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signer			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	keyverificationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			Has sibling
relationship	relatedObject Identification	relatedObjectIdentifierType		FDA-DF
relationship	relatedObject Identification	relatedObjectIdentifierValue		DF-2005-001003 [the identifier of the MP3 appendix]
relationship	relatedObject Identification	relatedObjectSequence		
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			FDA-E
linkingEventIdentifier	linkingEventIdentifier Value			E-2005-863721 [the identifier of the Ingest event]
linkingEventIdentifier	linkingEventIdentifier Type			FDA-E

Example 2, Object 2: the PDF file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingEventIdentifier	linkingEventIdentifier Value			E-2005-863722 [the identifier of a format validation event]
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 2, Object 3: the MP3 file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			FDA-DF
objectIdentifier	objectIdentifierValue			DF-2005-001003
preservationLevel				Bit [repository will do only bit-level preservation on this “nonpreferred” format]
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		SHA-1
objectCharacteristics	fixity	messageDigest		[a message digest]
objectCharacteristics	fixity	messageDigestOriginator		submitter
objectCharacteristics	size			505059321
objectCharacteristics	format	formatDesignation	formatName	MP3
objectCharacteristics	format	formatDesignation	formatVersion	
objectCharacteristics	format	formatRegistry	formatRegistryName	

Example 2, Object 3: the MP3 file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			unknown
creatingApplication	creatingApplication Version			unknown
creatingApplication	dateCreatedBy Application			2004
originalName				appendix.mp3
storage	contentLocation	contentLocationType		FDA
storage	contentLocation	contentLocationValue		fda/prod/data/out/classa/ DF-2005-001003
storage	storageMedium			3590 [a kind of tape unit]
environment	environment Characteristic			[Because the FDA is not doing full preservation on this file, it is not recording environment information.]
environment	environmentPurpose			
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		
environment	software	swVersion		
environment	software	swType		
environment	software	swOtherInformation		

Example 2, Object 3: the MP3 file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
environment	software	swDependency		
environment	software	swDependency		
environment	hardware	hwName		
environment	hardware	hwType		
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signer			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	keyVerificationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			Has sibling
relationship	relatedObject Identification	relatedObjectIdentifierType		FDA-DF
relationship	relatedObject Identification	relatedObjectIdentifierValue		DF-2005-001002 [the identifier of the PDF file]
relationship	relatedObject Identification	relatedObjectSequence		
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			FDA-E

Example 2, Object 3: the MP3 file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingEventIdentifier	linkingEventIdentifier Value			E-2005-863722
linkingEventIdentifier	linkingEventIdentifier Type			FDA-E
linkingEventIdentifier	linkingEventIdentifier Value			E-2005-863723
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 2, Object 4: the WAVE file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			FDA-DF
objectIdentifier	objectIdentifierValue			DF-2005-001013
preservationLevel				Full
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		SHA-1
objectCharacteristics	fixity	messageDigest		[a message digest]
objectCharacteristics	fixity	messageDigestOriginator		FDA
objectCharacteristics	size			529885999
objectCharacteristics	format	formatDesignation	formatName	WAVE
objectCharacteristics	format	formatDesignation	formatVersion	

Example 2, Object 4: the WAVE file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			MP32WAV
creatingApplication	creatingApplication Version			1.1
creatingApplication	dateCreatedBy Application			20050704T071532-0500
originalName				
storage	contentLocation	contentLocationType		FDA
storage	contentLocation	contentLocationValue		fda/prod/data/out/classa/ DF-2005-001013
storage	storageMedium			3590 [a kind of tape unit]
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		RealPlayer
environment	software	swVersion		10
environment	software	swType		renderer
environment	software	swOtherInformation		RealPlayer10-5GOLD.exe
environment	software	swDependency		

Example 2, Object 4: the WAVE file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
environment	software	swDependency		
environment	software	swName		Windows NT
environment	software	swVersion		
environment	software	swType		operating system
environment	software	swOtherInformation		
environment	software	swDependency		
environment	software	swDependency		
environment	hardware	hwName		
environment	hardware	hwType		
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signer			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	keyVerificationInformation		
relationship	relationshipType			derivative
relationship	relationshipSubType			Derived from
relationship	relatedObject Identification	relatedObjectIdentifierType		FDA-DF
relationship	relatedObject Identification	relatedObjectIdentifierValue		DF-2005-001003
relationship	relatedObject Identification	relatedObjectSequence		
relationship	relatedEvent Identification	relatedEventIdentifierType		FDA-E

Example 2, Object 4: the WAVE file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
relationship	relatedEvent Identification	relatedEventIdentifierValue		E-2005-9963733 [the key of the derivation event]
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			
linkingEventIdentifier	linkingEventIdentifier Value			
linkingEventIdentifier	linkingEventIdentifier Type			
linkingEventIdentifier	linkingEventIdentifier Value			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 2, Object 5: the new representation				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			FDA-R
objectIdentifier	objectIdentifierValue			R-2005-0002331
preservationLevel				Full
objectCategory				representation
objectCharacteristics	compositionLevel			n/a
objectCharacteristics	fixity	messageDigestAlgorithm		n/a

Example 2, Object 5: the new representation				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
objectCharacteristics	fixity	messageDigest		n/a
objectCharacteristics	fixity	messageDigestOriginator		n/a
objectCharacteristics	size			n/a
objectCharacteristics	format	formatDesignation	formatName	n/a
objectCharacteristics	format	formatDesignation	formatVersion	n/a
objectCharacteristics	format	formatRegistry	formatRegistryName	n/a
objectCharacteristics	format	formatRegistry	formatRegistryKey	n/a
objectCharacteristics	format	formatRegistry	formatRegistryRole	n/a
objectCharacteristics	significantProperties			n/a
objectCharacteristics	inhibitors	inhibitorType		n/a
objectCharacteristics	inhibitors	inhibitorTarget		n/a
objectCharacteristics	inhibitors	inhibitorKey		n/a
creatingApplication	creatingApplication Name			
creatingApplication	creatingApplication Version			
creatingApplication	dateCreatedBy Application			
originalName				n/a
storage	contentLocation	contentLocationType		n/a
storage	contentLocation	contentLocationType		n/a
storage	storageMedium			n/a
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	

Example 2, Object 5: the new representation				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
environment	software	swName		Mozilla Firefox
environment	software	swVersion		1.0
environment	software	swType		renderer
environment	software	swOtherInformation		requires swDependencies as plug-ins
environment	software	swDependency		Adobe Acrobat Reader 7.0
environment	software	swDependency		RealPlayer 10
environment	software	swName		Windows NT
environment	software	swVersion		5.0
environment	software	swType		operatingSystem
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		Intel Pentium II
environment	hardware	hwType		processor
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			n/a
signatureInformation	signatureMethod			n/a
signatureInformation	signatureValidation Rules			n/a
signatureInformation	signatureProperties			n/a
signatureInformation	signatureValue			n/a
signatureInformation	signatureValidation Rules			n/a
signatureInformation	keyInformation	keyType		n/a
signatureInformation	keyInformation	keyValue		n/a
signatureInformation	keyInformation	keyVerificationInformation		n/a
relationship	relationshipType			structural
relationship	relationshipSubType			Has part
relationship	relatedObject Identification	relatedObjectIdentifierType		FDA-DF

Example 2, Object 5: the new representation				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
relationship	relatedObject Identification	relatedObjectIdentifierValue		DF-2005-001002
relationship	relatedObject Identification	relatedObjectSequence		1
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
relationship	relationshipType			structural
relationship	relationshipSubType			Has part
relationship	relatedObject Identification	relatedObjectIdentifierType		FDA-DF
relationship	relatedObject Identification	relatedObjectIdentifierValue		DF-2005-001013
relationship	relatedObject Identification	relatedObjectSequence		2
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			
linkingEventIdentifier	linkingEventIdentifier Value			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			URL
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			www.fcla.edu/catalog/bibkey/AAA1234
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			

Example 2, Object 5: the new representation				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 2, Event 1		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	FDA-E
eventIdentifier	eventIdentifierValue	E-2005-863721
eventType		validation
eventDateTime		20050704T071530-0500
eventDetail		
eventOutcome Information	eventOutcome	00 [code meaning “successfully completed”]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	
linkingObjectIdentifier	linkingObjectIdentifierType	FDA-DF
linkingObjectIdentifier	linkingObjectIdentifierValue	DF-2005-001002

Example 2, Event 2		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	FDA-E
eventIdentifier	eventIdentifierValue	E-2005-863722
eventType		ingestion
eventDateTime		20050704T071531-0500

Example 2, Event 2		
EVENT		
semantic unit	**semantic unit**	**Value**
eventDetail		
eventOutcome Information	eventOutcome	00 [code meaning "successfully completed"]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	
linkingObjectIdentifier	linkingObjectIdentifierType	FDA-DF
linkingObjectIdentifier	linkingObjectIdentifierValue	DF-2005-001002

Example 2, Event 3		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	FDA-E
eventIdentifier	eventIdentifierValue	E-2005-863723
eventType		validation
eventDateTime		20050704T071532-0500
eventDetail		
eventOutcome Information	eventOutcome	00 [code meaning "successfully completed"]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	
linkingObjectIdentifier	linkingObjectIdentifierType	FDA-DF
linkingObjectIdentifier	linkingObjectIdentifierValue	DF-2005-001003

Example 2, Event 4		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	FDA-E
eventIdentifier	eventIdentifierValue	E-2005-863724
eventType		ingestion
eventDateTime		20050704T071533-0500
eventDetail		
eventOutcome Information	eventOutcome	00 [code meaning "successfully completed"]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	
linkingObjectIdentifier	linkingObjectIdentifierType	FDA-DF
linkingObjectIdentifier	linkingObjectIdentifierValue	DF-2005-001003

Example 2, Event 5		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	FDA-E
eventIdentifier	eventIdentifierValue	E-2005-863740
eventType		ingestion
eventDateTime		20050704T071653-0500
eventDetail		
eventOutcome Information	eventOutcome	00 [code meaning "successfully completed"]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	

Example 2, Event 5		
EVENT		
semantic unit	**semantic unit**	**Value**
linkingObjectIdentifier	linkingObjectIdentifierType	FDA-R
linkingObjectIdentifier	linkingObjectIdentifierValue	R-2005-000217

Example 2, Event 6		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	FDA-E
eventIdentifier	eventIdentifierValue	E-2005-9963733
eventType		migration
eventDateTime		20050705T077655-0500
eventDetail		
eventOutcome Information	eventOutcome	00 [code meaning “successfully completed”]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	
linkingObjectIdentifier	linkingObjectIdentifierType	FDA-DF
linkingObjectIdentifier	linkingObjectIdentifierValue	DF-2005-001013

Example 2, Agent 1		
AGENT		
semantic unit	**semantic unit**	**Value**
agentIdentifier	agentIdentifierType	FDA-A
agentIdentifier	agentIdentifierValue	A-554
agentName		University of Florida, University Library
agentType		organization

Example 2, Rights 1			
RIGHTS			
semantic unit	**semantic unit**	**semantic unit**	**Value**
permissionStatement	permissionStatementIdentifier	permissionStatement IdentifierType	FDA-PS
permissionStatement	permissionStatementIdentifier	permissionStatement IdentifierValue	PS-4
permissionStatement	linkingObject		R-2005-000217
permissionStatement	grantingAgent		A-554
permissionStatement	grantingAgreement	grantingAgreementIdentification	UF-1 [a local designation for this agreement with the University of Florida]
permissionStatement	grantingAgreement	grantingAgreementInformation	
permissionStatement	permissionGranted	act	all necessary
permissionStatement	permissionGranted	restriction	none
permissionStatement	permissionGranted	termOfGrant/startDate	20050101
permissionStatement	permissionGranted	termOfGrant/endDate	9999
permissionStatement	permissionGranted	permissionNote	

Example 3: Newspaper Complex Object, *Los Angeles Times*

This is a compound object created in QuarkXPress 3.1, a vector program widely used by newspapers for creating pages and information graphics. Rendered within the Quark file is a visual element created in a second vector program, Macromedia FreeHand MX 7.0 and exported as an Encapsulated PostScript (EPS) file for placement in Quark.

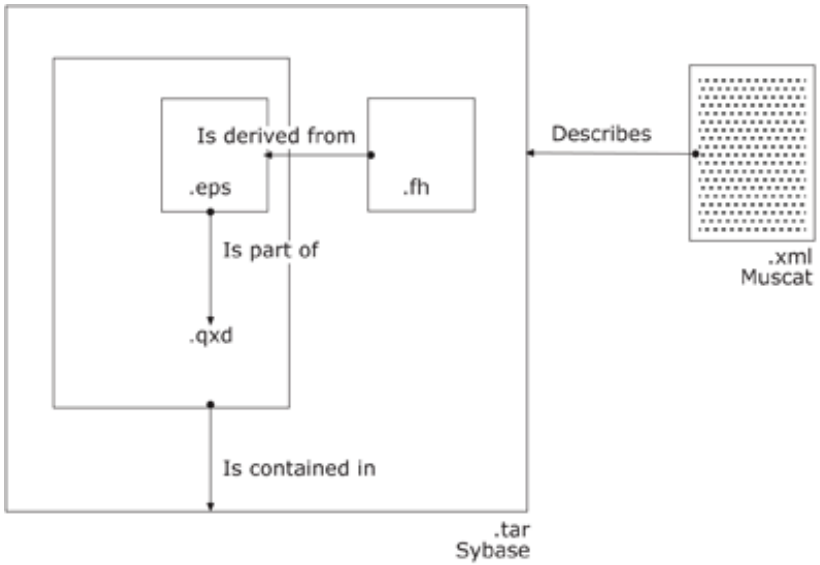
The Submission Information Package comprises a Quark file, the EPS file, and the native file used to create the EPS. At ingest, a UNIX tar file is created for the three objects (a lot of other events happen related to extracting metadata and PostScript, indexing text, and creation of thumbnail images supporting search and retrieval, but they aren't addressed here). The DIP comprises use copies of the original three objects.

In this example, these are treated as four separate file objects: the tar file (.tar), the Quark file (.qxp), the Encapsulated PostScript (.eps) and the native FreeHand file (.fh) from which the EPS was derived. The repository does not manage data at the representation level, but the tar file embeds all the filestreams that constitute the representation.

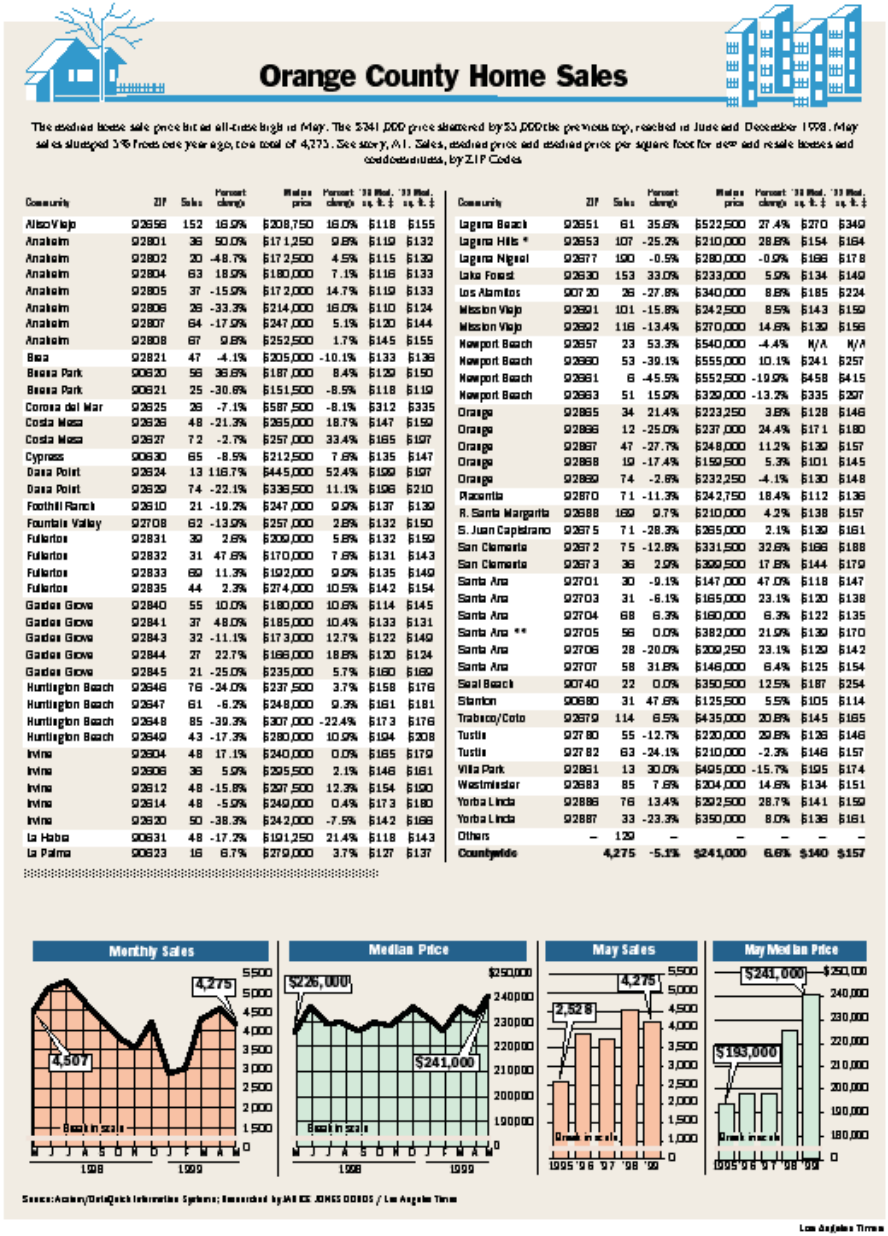
The tar file sits in a Sybase database called Mediasphere, and the metadata resides as a text file in a separate, proprietary database called Muscat. The tar file and metafile are linked by their common unique identifier assigned to the SIP. Metadata is structured in the Muscat metafile according to the news media standard known as IPTC.

All four formats appear in the format registry PRONOM, but there is a dearth of information about .qxd and .fh in particular. Some of the reasons for this are discussed in "Survey and Assessment of Sources of Information on File Formats and Software Documentation: Final Report," by the Representation and Rendering Project at the University of Leeds, www.jisc.ac.uk/uploaded_documents/FileFormatsreport.pdf.

About the repository: The Editorial Art Database contains about 26,000 similar complex objects dating as early as 1991. The contents of the SIP are based on a business case requiring retention of native files alongside derivative files and other files necessary for dissemination. This is subject to change in future iterations of the archive because of the difficulty of preserving proprietary vector formats. It is not an OAIS-compliant archive.



Above: diagram of relationships among files
Right: the rendered object



Orange County Home Sales

The resale home sale price hit an all-time high in May. The $341,000 price shattered by $3,000 the previous top, reached in June and December 1998. May sales slumped 3% from one year ago, to a total of 4,275. See story, M1. Sales, resale price and resale price per square foot for new and resale homes and condominiums, by ZIP Codes

Community	ZIP	Sales	Percent change	Median price	Percent change	'99 Med. sq. ft. ‡	'98 Med. sq. ft. ‡
Aliso Viejo	92656	152	16.9%	$208,750	16.0%	$118	$155
Anaheim	92801	36	50.0%	$171,250	9.8%	$119	$132
Anaheim	92802	20	-48.7%	$172,500	4.5%	$115	$139
Anaheim	92804	63	18.9%	$180,000	7.1%	$116	$133
Anaheim	92805	37	-15.9%	$172,000	14.7%	$119	$133
Anaheim	92806	26	-33.3%	$214,000	16.0%	$110	$124
Anaheim	92807	64	-17.9%	$247,000	5.1%	$120	$144
Anaheim	92808	67	9.8%	$252,500	1.7%	$145	$155
Aliso	92821	47	-4.1%	$205,000	-10.1%	$133	$136
Buena Park	90620	56	36.6%	$187,000	8.4%	$129	$150
Buena Park	90621	25	-30.6%	$151,500	-8.5%	$118	$119
Corona del Mar	92625	26	-7.1%	$587,500	-8.1%	$312	$335
Costa Mesa	92626	48	-21.3%	$265,000	18.7%	$147	$159
Costa Mesa	92627	72	-2.7%	$257,000	33.4%	$165	$197
Cypress	90630	65	-8.5%	$212,500	7.6%	$135	$147
Dana Point	92624	13	116.7%	$445,000	52.4%	$199	$197
Dana Point	92629	74	-22.1%	$336,500	11.1%	$196	$210
Foothill Ranch	92610	21	-19.2%	$247,000	9.9%	$137	$139
Fountain Valley	92708	62	-13.9%	$257,000	2.8%	$132	$150
Fullerton	92831	30	2.6%	$209,000	5.8%	$132	$150
Fullerton	92832	31	47.6%	$170,000	7.6%	$131	$143
Fullerton	92833	69	11.3%	$192,000	9.9%	$135	$149
Fullerton	92835	44	2.3%	$274,000	10.5%	$142	$154
Garden Grove	92840	55	10.0%	$180,000	10.6%	$114	$145
Garden Grove	92841	37	48.0%	$185,000	10.4%	$133	$131
Garden Grove	92843	32	-11.1%	$173,000	12.7%	$122	$149
Garden Grove	92844	27	22.7%	$188,000	18.8%	$120	$124
Garden Grove	92845	21	-25.0%	$235,000	5.7%	$160	$169
Huntington Beach	92646	76	-24.0%	$237,500	3.7%	$158	$176
Huntington Beach	92647	61	-6.2%	$248,000	9.3%	$161	$181
Huntington Beach	92648	85	-39.3%	$307,000	-22.4%	$173	$176
Huntington Beach	92649	43	-17.3%	$280,000	10.9%	$194	$208
Irvine	92604	48	17.1%	$240,000	0.0%	$165	$179
Irvine	92606	36	5.9%	$295,500	2.1%	$146	$161
Irvine	92612	48	-15.8%	$297,500	12.3%	$154	$190
Irvine	92614	48	-5.9%	$249,000	0.4%	$173	$180
Irvine	92620	50	-38.3%	$242,000	-7.5%	$142	$166
La Habra	90631	48	-17.2%	$191,250	21.4%	$118	$143
La Palma	90623	16	6.7%	$279,000	3.7%	$127	$137

Community	ZIP	Sales	Percent change	Median price	Percent change	'99 Med. sq. ft. ‡	'98 Med. sq. ft. ‡
Laguna Beach	92651	61	35.6%	$522,500	27.4%	$270	$349
Laguna Hills *	92653	107	-25.2%	$210,000	28.8%	$154	$184
Laguna Niguel	92677	190	-0.5%	$280,000	-0.9%	$166	$178
Lake Forest	92630	153	33.0%	$233,000	5.9%	$134	$149
Los Alamitos	90720	26	-27.8%	$340,000	8.8%	$185	$224
Mission Viejo	92691	101	-15.8%	$242,500	8.5%	$143	$159
Mission Viejo	92692	116	-13.4%	$270,000	14.6%	$130	$156
Newport Beach	92657	23	53.3%	$540,000	-4.4%	N/A	N/A
Newport Beach	92660	53	-39.1%	$555,000	10.1%	$241	$257
Newport Beach	92661	6	-45.5%	$552,500	-19.9%	$458	$415
Newport Beach	92663	51	15.9%	$320,000	-13.2%	$335	$297
Orange	92865	34	21.4%	$223,250	3.8%	$128	$146
Orange	92866	12	-25.0%	$237,000	24.4%	$171	$180
Orange	92867	47	-27.7%	$248,000	11.2%	$130	$157
Orange	92868	19	-17.4%	$159,500	5.3%	$101	$145
Orange	92869	74	-2.6%	$232,250	-4.1%	$130	$148
Placentia	92870	71	-11.3%	$242,750	18.4%	$112	$136
R. Santa Margarita	92688	169	9.7%	$210,000	4.2%	$138	$157
S. Juan Capistrano	92675	71	-28.3%	$265,000	2.1%	$130	$161
San Clemente	92672	75	-12.8%	$331,500	32.6%	$166	$188
San Clemente	92673	36	2.9%	$309,500	17.8%	$144	$179
Santa Ana	92701	30	-9.1%	$147,000	47.0%	$118	$147
Santa Ana	92703	31	-6.1%	$165,000	23.1%	$120	$138
Santa Ana	92704	68	6.3%	$160,000	6.3%	$122	$135
Santa Ana **	92705	56	0.0%	$382,000	21.9%	$130	$170
Santa Ana	92706	28	-20.0%	$209,250	23.1%	$120	$142
Santa Ana	92707	58	31.8%	$146,000	6.4%	$125	$154
Seal Beach	90740	22	0.0%	$350,500	12.5%	$187	$254
Stanton	90680	31	47.6%	$125,500	5.5%	$105	$114
Trabuco/Coto	92679	114	6.5%	$435,000	20.8%	$145	$165
Tustin	92780	55	-12.7%	$220,000	29.8%	$126	$146
Tustin	92782	63	-24.1%	$210,000	-2.3%	$146	$157
Villa Park	92861	13	30.0%	$495,000	-15.7%	$195	$174
Westminster	92683	85	7.6%	$204,000	14.6%	$134	$151
Yorba Linda	92886	76	13.4%	$292,500	28.7%	$141	$159
Yorba Linda	92887	33	-23.3%	$350,000	8.0%	$136	$161
Others	–	129	–	–	–	–	–
Countywide		**4,275**	**-5.1%**	**$241,000**	**6.6%**	**$140**	**$157**

Source: Acxiom/DataQuick Information Systems; Researched by MARK JONES DAVIS / Los Angeles Times

Los Angeles Times

Example 3, Object 1: the tar file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			LATMD [a code indicating an ID assigned by the Los Angeles Times Mediasphere Editorial Art Database]
objectIdentifier	objectIdentifierValue			FD3OLGGY
preservationLevel				full
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			
objectCharacteristics	format	formatDesignation	formatName	.tar
objectCharacteristics	format	formatDesignation	formatVersion	
objectCharacteristics	format	formatRegistry	formatRegistryName	PRONOM
objectCharacteristics	format	formatRegistry	formatRegistryKey	tar
objectCharacteristics	format	formatRegistry	formatRegistryRole	Basic
objectCharacteristics	significantProperties			
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			
creatingApplication	creatingApplication Version			
creatingApplication	dateCreatedBy Application			
originalName				
storage	contentLocationType			Pathname
storage	contentLocationValue			http://dali/stores/store11/FD3OLGGY

Example 3, Object 1: the tar file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
storage	storageMedium			server
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		Muscat metafile
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	Muscat
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	cat/tmp/FD3OLGGY.rec[5702]
environment	software	swName		Mediasphere
environment	software	swVersion		1.4b9
environment	software	swType		renderer
environment	software	swOtherInformation		
environment	software	swDependency		
environment	software	swName		Unix
environment	software	swVersion		
environment	software	swType		operating system
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		
environment	hardware	hwType		
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			None
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		

Example 3, Object 1: the tar file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			has part
relationship	relatedObject Identification	relatedObjectIdentifierType		LATMD
relationship	relatedObject Identification	relatedObjectIdentifierValue		FD3OLGGY/gr-zip 6/10
relationship	relatedObject Identification	relatedObjectSequence		1
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
relationship	relationshipType			structural
relationship	relationshipSubType			has part
relationship	relatedObject Identification	relatedObjectIdentifierType		LATMD
relationship	relatedObject Identification	relatedObjectIdentifierValue		FD3OLGGY/FI.0610.homesales.eps
relationship	relatedObject Identification	relatedObjectSequence		2
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
relationship	relationshipType			structural
relationship	relationshipSubType			has part
relationship	relatedObject Identification	relatedObjectIdentifierType		LATMD

Example 3, Object 1: the tar file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
relationship	relatedObject Identification	relatedObjectIdentifierValue		FD3OLGGY/Home sales/charts May.99
relationship	relatedObject Identification	relatedObjectSequence		3
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			LATMD
linkingEventIdentifier	linkingEventIdentifier Value			54780
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			TimesOnline
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			LA99/000052372
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 3, Object 2: the QuarkXPress file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			LATMD
objectIdentifier	objectIdentifierValue			FD3OLGGY/gr-zip 6/10
preservationLevel				Bit level; future migration not assured
objectCategory				file

Example 3, Object 2: the QuarkXPress file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		None
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			
objectCharacteristics	format	formatDesignation	formatName	.qxd
objectCharacteristics	format	formatDesignation	formatVersion	
objectCharacteristics	format	formatRegistry	formatRegistryName	PRONOM
objectCharacteristics	format	formatRegistry	formatRegistryKey	qxd
objectCharacteristics	format	formatRegistry	formatRegistryRole	Basic
objectCharacteristics	significantProperties			
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			Quark Xpress
creatingApplication	creatingApplication Version			3.1
creatingApplication	dateCreatedBy Application			19990609
originalName				
storage	contentLocationType			offset
storage	contentLocationValue			FD3OLGGY+256
storage	storageMedium			server
environment	environment Characteristic			known to work
environment	environmentPurpose			edit
environment	environmentPurpose			manipulate
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier	

Example 3, Object 2: the QuarkXPress file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
			Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Quark Xpress
environment	software	swVersion		3.1
environment	software	swType		creator
environment	software	swOtherInformation		
environment	software	swDependency		
environment	software	swName		MAC 0S
environment	software	swVersion		9.5.1
environment	software	swType		operating system
environment	software	swOtherInformation		9.5.1 or lower required
environment	software	swDependency		
environment	hardware	hwName		Apple PowerMac 8600
environment	hardware	hwType		cpu
environment	hardware	hwOtherInformation		monitor resolution 1172 x 870
signatureInformation	signatureInformation Encoding			None
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			has part
relationship	relatedObject Identification	relatedObjectIdentifierType		LATMD
relationship	relatedObject	relatedObjectIdentifierValue		FD3OLGGY/FI.0610.homesales.eps

Example 3, Object 2: the QuarkXPress file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
	Identification			
relationship	relatedObject Identification	relatedObjectSequence		1
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			
linkingEventIdentifier	linkingEventIdentifier Value			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 3, Object 3: the EPS file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
objectIdentifier	objectIdentifierType			LATMD
objectIdentifier	objectIdentifierValue			FD3OLGGY/FI.0610.homesales.eps
preservationLevel				Bit level; future migration not assured
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics				file

Example 3, Object 3: the EPS file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectCharacteristics	fixity	messageDigestAlgorithm		
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			
objectCharacteristics	format	formatDesignation	formatName	.eps
objectCharacteristics	format	formatDesignation	formatVersion	2.0
objectCharacteristics	format	formatRegistry	formatRegistryName	PRONOM
objectCharacteristics	format	formatRegistry	formatRegistryKey	eps
objectCharacteristics	format	formatRegistry	formatRegistryRole	Basic
objectCharacteristics	significantProperties			Editable eps
objectCharacteristics	inhibitors	inhibitorType		None
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			Macromedia Freehand
creatingApplication	creatingApplication Version			7.0
creatingApplication	dateCreatedBy Application			19990609
originalName				
storage	contentLocationType			offset
storage	contentLocationValue			FD3OLGGY+8755
storage	storageMedium			server
environment	environmentPurpose			render
environment	environment Characteristic			known to work
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier	

Example 3, Object 3: the EPS file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
			Value	
environment	software	swName		Macromedia Freehand
environment	software	swVersion		7.0
environment	software	swType		renderer
environment	software	swOtherInformation		
environment	software	swDependency		
environment	software	swName		Mac OS
environment	software	swVersion		9.5.1
environment	software	swType		operating system
environment	software	swOtherInformation		9.5.1 or lower required
environment	software	swDependency		
environment	hardware	hwName		Apple PowerMac 8600
environment	hardware	hwType		cpu
environment	hardware	hwOtherInformation		Monitor resolution 1172 x 870
signatureInformation	signatureInformation Encoding			None
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			is part of
relationship	relatedObject Identification	relatedObjectIdentifierType		LATMD
relationship	relatedObject Identification	relatedObjectIdentifierValue		FD3OLGGY
relationship	relatedObject	relatedObjectSequence		0

Example 3, Object 3: the EPS file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
	Identification			
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
relationship	relationshipType			derivative
relationship	relationshipSubType			is derived from
relationship	relatedObject Identification	relatedObjectIdentifierType		FD3OLGGY
relationship	relatedObject Identification	relatedObjectIdentifierValue		Home sales/Charts May.99
relationship	relatedObject Identification	relatedObjectSequence		1
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			
linkingEventIdentifier	linkingEventIdentifier Value			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 3, Object 4: the Macromedia FreeHand file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			LATMD
objectIdentifier	objectIdentifierValue			FD3OLGGY/Home sales/Charts May.99
preservationLevel				Bit level; future migration not assured
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		None
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			
objectCharacteristics	format	formatDesignation	formatName	.fh
objectCharacteristics	format	formatDesignation	formatVersion	
objectCharacteristics	format	formatRegistry	formatRegistryName	PRONOM
objectCharacteristics	format	formatRegistry	formatRegistryKey	fh
objectCharacteristics	format	formatRegistry	formatRegistryRole	Basic
objectCharacteristics	significantProperties			
objectCharacteristics	inhibitors	inhibitorType		None
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			Macromedia Freehand
creatingApplication	creatingApplication Version			7.0
creatingApplication	dateCreatedBy Application			19990609
originalName				
storage	contentLocationType			offset
storage	contentLocationValue			FD3OLGGY+14243
storage	storageMedium			server

Example 3, Object 4: the Macromedia FreeHand file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
environment	environment Characteristic			known to work
environment	environmentPurpose			edit
environment	environmentNote			Business case for retention of this object
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Macromedia Freehand
environment	software	swVersion		7.0
environment	software	swType		Creator
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		Apple PowerMac 8600
environment	hardware	hwType		Desktop computer
environment	hardware	hwOtherInformation		Monitor resolution 1172 x 870
signatureInformation	signatureInformation Encoding			None
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			Is part of
relationship	relatedObject Identification	relatedObjectIdentifierType		LATMD

Example 3, Object 4: the Macromedia FreeHand file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
relationship	relatedObject Identification	relatedObjectIdentifierValue		FD3OLGGY
relationship	relatedObject Identification	relatedObjectSequence		0
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
relationship	relationshipType			derivative
relationship	relationshipSubType			Is source of
relationship	relatedObject Identification	relatedObjectIdentifierType		LATMD
relationship	relatedObject Identification	relatedObjectIdentifierValue		FD3OLGGY/FI.0610.homesales.eps
relationship	relatedObject Identification	relatedObjectSequence		0
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			
linkingEventIdentifier	linkingEventIdentifier Value			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			

Example 3, Object 4: the Macromedia FreeHand file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 3, Event 1		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	LATMD
eventIdentifier	eventIdentifierValue	54780
eventType		Ingest
eventDateTime		199906101330
eventDetail		
eventOutcome Information	eventOutcome	1 [ingestion successful]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	
linkingObjectIdentifier	linkingObjectIdentifierType	
linkingObjectIdentifier	linkingObjectIdentifierValue	

Example 3, Agent 1		
AGENT		
semantic unit	**semantic unit**	**Value**
agentIdentifier	agentIdentifierType	Mediasphere
agentIdentifier	agentIdentifierValue	pjohnson
agentName		Peter Johnson
agentType		Person

Example 3, Rights 1			
RIGHTS			
semantic unit	**semantic unit**	**semantic unit**	**Value**
permissionStatement	permissionStatementIdentifier	permissionStatementIdentifierType	Los Angeles Times rights declaration
permissionStatement	permissionStatementIdentifier	permissionStatementIdentifierValue	1999-14
permissionStatement	linkingObject		LATMD:FD3OLGGY/gr-zip 6/10
permissionStatement	grantingAgent		Los Angeles Times LLC
permissionStatement	grantingAgreement	grantingAgreementIdentification	
permissionStatement	grantingAgreement	grantingAgreementInformation	
permissionStatement	permissionGranted	act	Disseminate
permissionStatement	permissionGranted	restriction	Tribune Co. properties only without prior approval by Los Angeles Times LLC
permissionStatement	permissionGranted	termOfGrant/startDate	1999
permissionStatement	permissionGranted	termOf Grant/endDate	9999
permissionStatement	permissionGranted	permissionNote	

Example 4: Web Site

This example is a single snapshot of a Web site harvested by HTTrack on 2003-04-17 and stored on a UNIX file system. Subsequent snapshots of the same Web site exist in the repository. The relationships among the archived snapshots are documented by a METS Aggregator object that has descriptive metadata for the Web site and represents the intellectual entity.

This Web site belongs to the Women's Division of the All Progressives Grand Alliance Party in Nigeria. It consists of a Flash-enhanced splash page which functions as an entry to a non-Flash home page. This is a site containing eight HTML pages and two linked Microsoft Word documents. For brevity, the example shows metadata for only two of the HTML pages and a few linked images and Flash files.

All files that are necessary to render the page are handled as equivalent in terms of hierarchy with one file designated as the root of the others in the metadata for the representation. The representation of the Web site snapshot as a whole includes a number of files: the entry page with its domain URL as its identifier and all related objects with relationship "has part." The files that make up the Web site each have metadata with an "is part of" relationship to the Web site. Although there are reciprocal relationships in this example, a repository might choose to explicitly encode the relationship one way or the other, i.e. either in the metadata for the representation with "has part" for each file object or in each file object with "is part of" for the representation.

There are several other ways a Web site could be modeled. For example, there could be a representation object for each page included in the site in addition to a representation for the Web site as a whole, with all files having an "is part of" relationship to the representation and one file designated as root object. For more information and for diagrams illustrating these options, see "Preservation metadata for Web sites and Web pages," page 6-4.

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			URI
	objectIdentifierValue			http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/ [note that a persistent identifier would be preferable]
preservationLevel				General look and feel
objectCategory				representation
objectCharacteristics	compositionLevel			n/a
	fixity	messageDigestAlgorithm		n/a
		messageDigest		n/a
		messageDigestOriginator		n/a
	Size			n/a
	Format	formatDesignation	formatName	n/a
			formatVersion	n/a
		formatRegistry	formatRegistryName	n/a
			formatRegistryKey	n/a
			formatRegistryRole	n/a
	significantProperties			n/a
	inhibitors	inhibitorType		n/a
		inhibitorTarget		n/a
		inhibitorKey		n/a
creatingApplication	creatingApplication Name			
	creatingApplication Version			
	dateCreatedBy Application			
originalName				
storage	contentLocation	contentLocationType		n/a

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
	contentLocation	contentLocationValue		n/a
	storageMedium			n/a
environment	environment Characteristic			known to Work
	environmentPurpose			render
	environmentNote			
	dependency	dependencyName		
		dependencyIdentifier	dependencyIdentifier Type	
			dependencyIdentifier Value	
	software	swName		Mozilla Browser
		swVersion		1.x.x
		swType		renderer
		swOtherInformation		
		swDependency		Javascript Plugin
		swDependency		Shockwave-Flash Plugin
		swDependency		MS Word Reader Plugin
	software	swName		Mac OS X
		swVersion		10.2.3
		swType		Operating System
		swOtherInformation		
		swDependency		
	software	swName		Sun Java Virtual Machine
		swVersion		1.4
		swType		ancillary
		swOtherInformation		
		swDependency		
	hardware	hwName		Intel Pentium II
		hwType		processor
		hwOtherInformation		

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
signatureInformation	signatureInformationEncoding			n/a
	signatureMethod			n/a
	signatureValue			n/a
	signatureValidation Rules			n/a
	signatureProperties			n/a
	keyInformation	keyType		n/a
		keyValue		n/a
	validationInformation			n/a
relationship	relationshipType			structural
	relationshipSubType			has root
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/index.html
		relatedObjectSequence		
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/notjust.swf
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/enterarrow.gif

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/apgawnew.swf
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/home.htm
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/welcome.gif
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/allprogressives.swf
relationship	relationshipType			structural
	relationshipSubType			has part

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/apgawomenb.jpg
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/sidelink.gif
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/quote.gif
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/apganews.gif
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/links.gif
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/contact.gif
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/ojeozi1.jpg
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/newsarchives.htm
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/officers.htm

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/officers.jpg
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/calender.htm
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metestest/20030417/apgawomen/www.apgawomen.org/calender.jpg
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/projects.htm
relationship	relationshipType			structural
	relationshipSubType			has part

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/projects.jpg
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/membership.htm
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/membership.jpg
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/about%20us.htm
relationship	relationshipType			structural
	relationshipSubType			has part
	relatedObject Identification	relatedObjectIdentifierType		URI

Example 4, Object 1: the entire Web site (representation level)				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
		relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/about%20us.jpg
linkingEventIdentifier	linkingEventIdentifierType			
	linkingEventIdentifierValue			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			URI
	linkingIntellectual EntityIdentifierValue			http://dlib.nyu.edu:8083/xmldev/apgawomen-root.xml [the METS aggregator document]
linkingPermission StatementIdentifier	linkingPermissionStatementIdentifierType			
linkingPermission StatementIdentifier	linkingPermissionStatementIdentifierValue			

Example 4, Object 2: an HTML file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			URI
objectIdentifier	objectIdentifierValue			http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/index.html
preservationLevel				full
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			1656
objectCharacteristics	format	formatDesignation	formatName	HTML

Example 4, Object 2: an HTML file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectCharacteristics	format	formatDesignation	formatVersion	unknown
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			Two embedded Flash files and a hard link wrapped around an arrow image for navigation to the non-Flash homepage
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			unknown
creatingApplication	creatingApplication Version			unknown
creatingApplication	dateCreatedBy Application			20030101
originalName				
storage	contentLocation	contentLocationType		URI
storage	contentLocation	contentLocationValue		http://dlibdev.nyu.edu/webarchive/0417/apgawomen/www.apgawomen.org/index.htm
storage	storageMedium			Hard disk
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	

Example 4, Object 2: an HTML file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
environment	software	swName		Mozilla Browser
environment	software	swVersion		1.x.x
environment	software	swType		renderer
environment	software	swOtherInformation		
environment	software	swDependency		Shockwave-Flash Plugin
environment	software	swDependency		Javascript Plugin
environment	software	swName		Sun Java Virtual Machine
environment	software	swVersion		1.4
environment	software	swType		ancillary
environment	software	swOtherInformation		
environment	software	swDependency		
environment	software	swName		Mac OS X
environment	software	swVersion		10.2.3
environment	software	swType		operating system
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		Intel Pentium II
environment	hardware	hwType		processor
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
relationship	relationshipType			structural
relationship	relationshipSubType			is part of

Example 4, Object 2: an HTML file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
relationship	relatedObject Identification	relatedObjectIdentifierType		URI
relationship	relatedObject Identification	relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/ [the representation]
relationship	relatedObject Identification	relatedObjectSequence		0
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermision StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermision StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 4, Event 1		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	NYU-A
eventIdentifier	eventIdentifierValue	WCR-2004-12345
eventType	eventType	ingestion
eventDateTime	eventDateTime	20040528T070531-0500
eventDetail	eventDetail	
eventOutcome Information	eventOutcome	SC [code meaning “successfully completed”]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	

Example 4, Event 1		
EVENT		
semantic unit	semantic unit	Value
linkingAgentIdentifier	linkingAgentIdentifierRole	
linkingObjectIdentifier	linkingObjectIdentifierType	
linkingObjectIdentifier	linkingObjectIdentifierValue	

Example 4, Object 3: a Flash file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
objectIdentifier	objectIdentifierType			URI
objectIdentifier	objectIdentifierValue			http://dlibdev.nyu.edu/webarchive/0417/apgawomen/www.apgawomen.org/notjust.swf
preservationLevel				full
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			21688320
objectCharacteristics	format	formatDesignation	formatName	flash
objectCharacteristics	format	formatDesignation	formatVersion	unknown
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			One of two embedded Flash files for splash page
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		

Example 4, Object 3: a Flash file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
creatingApplication	creatingApplication Name			MIX-FX
creatingApplication	creatingApplication Version			unknown
creatingApplication	dateCreatedBy Application			20030101
originalName				
storage	contentLocation	contentLocationType		URI
storage	contentLocation	contentLocationValue		http://dlibdev.nyu.edu/webarchive/0417/apgawomen/www.apgawomen.org/notjust.swf
storage	storageMedium			Hard disk
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Mozilla Browser
environment	software	swVersion		1.x.x
environment	software	swType		renderer
environment	software	swOtherInformation		
environment	software	swDependency		Shockwave-Flash Plugin
environment	software	swName		Mac OS X
environment	software	swVersion		10.2.3
environment	software	swType		operating system
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		Intel Pentium II

Example 4, Object 3: a Flash file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
environment	hardware	hwType		processor
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	keyInformation	keyType		
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			is part of
relationship	relatedObject Identification	relatedObjectIdentifierType		URI
relationship	relatedObject Identification	relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/
relationship	relatedObject Identification	relatedObjectSequence		0
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			NYA-A
linkingEventIdentifier	linkingEventIdentifier Value			WCR-2004-12346
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			

Example 4, Object 3: a Flash file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 4, Event 2		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	NYU-A
eventIdentifier	eventIdentifierValue	WCR-2004-12346
eventType		ingestion
eventDateTime		20040528T070531-0500
eventOutcome Information	eventOutcome	SC [code meaning “successfully completed”]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	

[Metadata for other Flash files left out for brevity.]

Example 4, Object 4: a GIF file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			URI
objectIdentifier	objectIdentifierValue			http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/enterarrow.gif
preservationLevel				full
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			1724
objectCharacteristics	format	formatDesignation	formatName	GIF
objectCharacteristics	format	formatDesignation	formatVersion	unknown
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			Necessary icon for navigation to non-Flash home page
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			unknown
creatingApplication	creatingApplication Version			unknown
creatingApplication	dateCreatedBy Application			20030101
originalName				
storage	contentLocation	contentLocationType		URI

Example 4, Object 4: a GIF file				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
storage	contentLocation	contentLocationValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/enterarrow.gif
storage	storageMedium			Hard disk
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Mozilla Browser
environment	software	swVersion		1.x.x
environment	software	swType		renderer
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		Intel Pentium II
environment	hardware	hwType		processor
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	keyInformation	keyType		
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		

Example 4, Object 4: a GIF file				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
relationship	relationshipType			structural
relationship	relationshipSubType			is part of
relationship	relatedObject Identification	relatedObjectIdentifierType		URI
relationship	relatedObject Identification	relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/20030417/apgawomen/www.apgawomen.org/
relationship	relatedObject Identification	relatedObjectSequence		0
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			NYA-A
linkingEventIdentifier	linkingEventIdentifier Value			WCR-2004-12389
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 4, Event 3		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	NYU-A
eventIdentifier	eventIdentifierValue	WCR-2004-12389
eventType		ingestion
eventDateTime		20040528T090231-0500
eventDetail		
eventOutcome Information	eventOutcome	SC [code meaning "successfully completed"]
eventOutcome Information	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	
linkingAgentIdentifier	linkingAgentIdentifierValue	
linkingAgentIdentifier	linkingAgentIdentifierRole	

Example 4, Object 5: the HTML home page				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			URI
objectIdentifier	objectIdentifierValue			http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/home.htm
preservationLevel				full
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			15606
objectCharacteristics	format	formatDesignation	formatName	HTML

Example 4, Object 5: the HTML home page				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectCharacteristics	format	formatDesignation	formatVersion	unknown
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			cgi-enabled poll can not be activated
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			unknown
creatingApplication	creatingApplication Version			
creatingApplication	dateCreatedBy Application			20030101
originalName				
storage	contentLocation	contentLocationType		URI
storage	contentLocation	contentLocationValue		http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/home.htm
storage	storageMedium			Hard disk
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Mozilla Browser
environment	software	swVersion		1.x.x
environment	software	swType		renderer

Example 4, Object 5: the HTML home page				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		Intel Pentium II
environment	hardware	hwType		processor
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		
relationship	relationshipType			structural
relationship	relationshipSubType			is part of
relationship	relatedObject Identification	relatedObjectIdentifierType		URI
relationship	relatedObject Identification	relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/
relationship	relatedObject Identification	relatedObjectSequence		0
relationship	relatedEvent Identification	relatedEventIdentifierType		NYU-A
relationship	relatedEvent Identification	relatedEventIdentifierValue		WCR-2004-12346
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			NYU-A

Example 4, Object 5: the HTML home page				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingEventIdentifier	linkingEventIdentifier Value			WCR-2004-12346
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 4, Event 4		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	NYU-A
eventIdentifier	eventIdentifierValue	WCR-2004-12346
eventType		ingestion
eventDateTime		20040529T060231-0500
eventDetail		
eventOutcome Information	eventOutcome	SC [code meaning “successfully completed”]
eventOutcome Information	eventOutcomeDetail	
relatedAgentIdentifier	relatedAgentIdentifierType	
relatedAgentIdentifier	relatedAgentIdentifierValue	
relatedAgentIdentifier	relatedAgentIdentifierRole	

Example 4, Object 6: an HTML file linked to from home page				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			URI
objectIdentifier	objectIdentifierValue			http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/ about%20us.htm
preservationLevel				full
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		
objectCharacteristics	fixity	messageDigest		
objectCharacteristics	fixity	messageDigestOriginator		
objectCharacteristics	size			1543
objectCharacteristics	format	formatDesignation	formatName	HTML
objectCharacteristics	format	formatDesignation	formatVersion	unknown
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			cgi-enabled poll can not be activated
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			unknown
creatingApplication	creatingApplication Version			unknown
creatingApplication	dateCreatedBy Application			20030101
originalName				
storage	contentLocation	contentLocationType		URI

Example 4, Object 6: an HTML file linked to from home page				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
storage	contentLocation	contentLocationValue		http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/ about%20us.htm
storage	storageMedium			Hard disk
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Mozilla Browser
environment	software	swVersion		1.x.x
environment	software	swType		renderer
environment	software	swOtherInformation		
environment	software	swDependency		
environment	hardware	hwName		
environment	hardware	hwType		
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		

Example 4, Object 6: an HTML file linked to from home page				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
relationship	relationshipType			structural
relationship	relationshipSubType			Is part of
relationship	relatedObject Identification	relatedObjectIdentifierType		URI
relationship	relatedObject Identification	relatedObjectIdentifierValue		http://dlibdev.nyu.edu/webarchive/metstest/apgawomen/20030417/www.apgawomen.org/ [the representation]
relationship	relatedObject Identification	relatedObjectSequence		0
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			NYU-A
linkingEventIdentifier	linkingEventIdentifier Value			WCR-2004-16233
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Value			

Example 4, Event 5		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	NYU-A
eventIdentifier	eventIdentifierValue	WCR-2004-16233
eventType		ingestion
eventDateTime		20040529T060231-0500
eventDetail		
eventOutcome Information	eventOutcome	SC [code meaning "successfully completed"]
eventOutcome Information	eventOutcomeDetail	
relatedAgentIdentifier	relatedAgentIdentifierType	
relatedAgentIdentifier	relatedAgentIdentifierValue	
relatedAgentIdentifier	relatedAgentIdentifierRole	

Other files detailed in the metadata for the representation with relationship "has part" would also have separate metadata.

Example 5: Digital Signature

signatureInformation	signatureInformation Encoding		base64
signatureInformation	signer		Florida Digital Archive
signatureInformation	signatureMethod		RSA-SHA1
signatureInformation	signatureValue		MC0CFFrVLtRlkMc3Daon4BqqnkhCOTFEALE =
signatureInformation	signatureValidation Rules		T1=C1
signatureInformation	signatureProperties		2003-03-19T12:25:14-05:00
signatureInformation	keyInformation	keyType	x509v3-sign-rsa2
signatureInformation	keyInformation	keyValue	<DSAKeyValue> <P> imup6lmki4rAmUstKb/xdBRMWNtQ+pDN97ZnLA9X3lKbkEHtYFyj Q3uActgVSJ75iVRuKxz4Cb5RzVm25EaKmKq8rif1MtBIi6jjDJxmIdN aEKG9zVTf9giJx1N9I0t3oh1fAVZDSrzKzJGQ2WvDfIfFHdJMtB3C0 VKGmLZR7Xk= </P> <Q> xDve3j7sEnh4rIzM5gK+5/gxxFU= </Q> <G> NLugAf6IZJxo3BCOi5yrGEVwtlEzXcnndXhd0Tz38CnQKc4SEupm4P yP5TmLvK64TDfOD7sno/W5oI1KZdimfW2c4r/6waNzZSvicMOWhL YY621Nn6njBc8VNwoxWpzCXhKm70b8+D4YZMn/eU5DN8dvhTv/b NK21FfJqjp033U= </G> <Y>W7dOmH/vWqocVCiqaxj6soxVXfR8XpMdY2Zv4Amjr3n81geyO Lb6IZ+l7MUbdp8529DQzuoVTthVpB9X4JKCprZIzifOTM1PFflTBzjx 7egJwJWAIVdWyiIPjke6Va+wuV2n4Rl/cgCvrXK5cTov5C/Bpaf6o+qrr DGFBLLZTF4= </Y> </DSAKeyValue>

signatureInformation	keyInformation	keyVerificationInformation	MIICMzCCAaCgAwIBAgIQYCEyPdEvhYNJPDnlIUVUVjAJBgUrDgMCHQUAME0xEzARBgNVBAMTCnByaXNjaWxsYTExDDAKBgNVBAcTA0VGUzEoMCYGA1UECxMfRUZTIEZpbGUgRW5jcnlwdGlvbiBDZXJ0aWZpY2F0ZTAgFw0wMzAzMTkxNzI1MTRaGA8yMTAzMDIyMzE3MjUxNFowTTETMBEGA1UEAxMKcHJpc2NpbGxhMTEMMAoGA1UEBxMDRUZTMSgwJgYDVQQLEx9FRlMgRmlsZSBFbmNyeXB0aW9uIENlcnRpZmljYXRlMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQD3f4ybaySN36xq7PEHgrVY9plRiUqqJZDGH8aoYwDcdMaV+6kJeXS/J1HMtH3FDW2ImCI6X/z2DIvQoiQ8fEUcQx19QK/ACWabAolPZn0RkDXLkg7ZOrX3zg7AV9+oAtg5oAILgQ8Vb0BGeFpEJ5xBQjh1WhfGJz7unW7ea3zfMQIDAQABoxowGDAWBgNVHSUEDzANBgsrBgEEAYI3CgMEATAJBgUrDgMCHQUAA4GBABUYv/SXRCGnSyqMClcyMScTY2jxmAMFn+OvrYYxArcO0rqP0GXg035Yawn1l9QKXrKYzK/CNLvCygM7QeWGxAfrQgU1CK3nZd/fUm0diHqwHhVJiP/6xBCL0GYzPJ8zNJMpzUlA38jF9Oz87ynWoaNClcItLQtmSp4kpjGS3KVh

Notes:

- *signer*: The preservation repository chose to use the company name of the signer.
- *signatureInformation*: Made up for this example. The repository has a shorthand for representing successive transformation on content before hashing. The example "T1=C1" should be take to mean the first transformation is by canonicalization algorithm 1, which is presumably documented elsewhere. Second and subsequent tranformations could be indicated with T2= etc.
- *signatureProperties*: The repository records only the date/time of signing.
- *keyValue*: The repository uses an XML-structured value for DSA keys. The same structure could as easily have been represented by defining local subcomponents of the semantic unit keyValue.
- *keyVerificationInformation*: The base64 representation of an X509 certificate. The certificate's reformatted content in XML is:

```
<Certificate>
<version>v3</version>
<serialnumber>6021 323D D12F 8583 493C 39E5 2145 5456</serialnumber>
<signatureAlgorith>sha1RSA</signatureAlgorithm>
<issuer ou="Florida Digital Archive" c="us" cn="Chris Franco"/>
<validFrom>Wednesday, March 19, 2003 12:25:14 PM</validFrom>
<validTo>Friday, February 23, 2103 12:25:14 PM</validTo>
<subject ou="Florida Digital Archive" c="us" cn="Chris Franco"/>
<publicKey>3081 8902 8181 00F7 7F8C 9B6B 248D DFAC 6AEC F107 82B5 58F6 9951 894A AA25 90C6 1FC6 A863 00DC 74C6 95FB A909 7974 BF27
51CC B47D C50D 6D88 9822 3A5F FCF6 0C8B D0A2 243C 7C45 1C43 1D7D 40AF C009 669B 0289 4F66 7D11 9035 CB92 0ED9 3AB5 F7CE 0EC0 57DF
A802 D839 A002 0B81 0F15 6F40 4678 5A44 279C 4142 3875 5A17 C627 3EEE 9D6E DE6B 7CDF 3102 0301 0001</publicKey>
<thumbprintAlgorithm>sha1</thumbprintAlgorithm>
<thumbprint>DA25 71B8 DE03 ADDB A321 8B4C 1EC2 CACC 368A 15B3</thumbprint>
</certificate>
```

Example 6: Photograph

In this hypothetical example, the repository is archiving one image (an archival master in TIFF format) and one XML file (descriptive metadata for the image) from a Getty Research Institute Digitized Library Collection. Both the image and the metadata file are treated as independent file objects.

Example 6, Object 1: the TIFF image				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
objectIdentifier	objectIdentifierType			handle
	objectIdentifierValue			hdl:jpgt.gri/grl_2001pr3-f10-1.tif
preservationLevel				Fully supported with future migrations
objectCategory				file
objectCharacteristics	compositionLevel			0
	fixity	messageDigestAlgorithm		MD5
	fixity	messageDigest		aa68a8e56165bdafc9693f4a7273ab86
	fixity	messageDigestOriginator		GDAM [Getty Digital Asset Management System]
	size			45362176
	format	formatDesignation	formatName	TIFF
	format	formatDesignation	formatVersion	6.0
	format	formatRegistry	formatRegistryName	
	format	formatRegistry	formatRegistryKey	
	format	formatRegistry	formatRegistryRole	
	significantProperties			Color Accuracy (Adobe RGB 1998)
	inhibitors	inhibitorType		
	inhibitors	inhibitorTarget		
	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			ScanXact
	creatingApplication Version			II

Example 6, Object 1: the TIFF image				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
	DateCreatedBy Application			2002-02-11
	creatingApplication Name			Adobe Photoshop
	creatingApplication Version			6.0.1
	DateCreatedBy Application			2002-02-11
originalName				2001PR3_10
storage	contentLocation	contentLocationType		filename
	contentLocation	contentLocationValue		grl_2001pr3-f10-1.tif
	storageMedium			disk
storage	contentLocation	contentLocationType		physical location
	contentLocation	contentLocationValue		grl_2001pr3-f10-1, CD 1, Box 1, 2001pr3, Vault 4
	storageMedium			CD-ROM
environment	environment Characteristic			known to work
	environmentPurpose			Search, render, transform, attach metadata
	environmentNote			
	dependency	dependencyName		
	dependency	dependencyIdentifier	dependencyIdentifier Type	
	dependency	dependencyIdentifier	dependencyIdentifier Value	
	software	swName		Artesia TEAMS
	software	swVersion		5.1
	software	swType		Digital Asset Management System
	software	swOtherInformation		
	software	swDependency		ImageAlchemy
	software	swDependency		Oracle 9i

Example 6, Object 1: the TIFF image				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
	software	swName		Sun Solaris
	software	swVersion		9
	software	swType		Operating system
	software	swOtherInformation		
	software	swDependency		
	hardware	hwName		SunFire
	hardware	hwType		cpu
	hardware	hwOtherInformation		V series
signatureInformation	signatureInformation Encoding			None
	signatureMethod			
	signatureValue			
	signatureValidation Rules			
	signatureProperties			
	keyInformation	keyType		
	keyInformation	keyValue		
	keyInformation	validationInformation		
relationship	relationshipType			local
	relationshipSubType			has metadata
	relatedObject Identification	relatedObjectIdentifierType		handle
	relatedObject Identification	relatedObjectIdentifierValue		hdl:jpgt.gri/grl_2001pr3-f10-1.xml
	relatedObject Identification	relatedObjectSequence		1
	relatedEvent Identification	relatedEventIdentifierType		
	relatedEvent Identification	relatedEventIdentifierValue		
	relatedEvent Identification	relatedEventSequence		

Example 6, Object 1: the TIFF image				
OBJECT				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
linkingEventIdentifier	linkingEventIdentifier Type			GDAM-E
	linkingEventIdentifier Value			Ingest Object 20050209 2:12:05 PM
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			handle
	linkingIntellectual EntityIdentifierValue			hdl:jpgt.gri/grl_2001pr3-f10-1.xml
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			GRMS [Getty Research Institute Resource Management System]
	linkingPermission StatementIdentifier Value			2001.pr.3.f10.1

Example 6, Event 1		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	GDAM-E
	eventIdentifierValue	Ingest Object 20050209 2:12:05 PM
eventType		Ingestion
eventDateTime		20050209T141205-0500
eventDetail		
eventOutcome Information	eventOutcome	Successful
	eventOutcomeDetail	
linkingAgentIdentifier	linkingAgentIdentifierType	GDAM-A
	linkingAgentIdentifierValue	shubbard
	linkingAgentRole	Implementor
	linkingObjectIdentifierType	handle
	linkingObjectIdentifierValue	hdl:jpgt.gri/grl_2001pr3-f10-1.tif

Example 6, Agent 1		
AGENT		
semantic unit	**semantic unit**	**Value**
agentIdentifier	agentIdentifierType	GDAM-A
	agentIdentifierValue	shubbard
agentName		Sally Hubbard
agentType		Person

Example 6, Rights 1				
RIGHTS				
semantic unit	**semantic unit**	**semantic unit**	**semantic unit**	**Value**
permissionStatement	permissionStatementIdentifier	permissionStatement IdentifierType		GDAM-R
	permissionStatementIdentifier	permissionStatement IdentifierValue		2001.pr.3.f10.1-1
	linkingObject			hdl:jpgt.gri/grl_2001pr3-f10-1.tif
	grantingAgent			Public Domain
	grantingAgreement	grantingAgreement Identification		
	grantingAgreement	grantingAgreement Information		
	permissionGranted	act		Replicate, Migrate, Modify, Use, Disseminate
	permissionGranted	restriction		None
	permissionGranted	termOfGrant	startDate	0000
	permissionGranted	termOfGrant	endDate	9999
	permissionGranted	permissionNote		

Example 6, Object 2: the XML metadata				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
objectIdentifier	objectIdentifierType			handle
objectIdentifier	objectIdentifierValue			hdl:jpgt.gri/grl_2001pr3-f10-1.xml
preservationLevel				Fully supported with future migrations
objectCategory				file
objectCharacteristics	compositionLevel			0
objectCharacteristics	fixity	messageDigestAlgorithm		MD5
objectCharacteristics	fixity	messageDigest		6df23dc03f9b54cc38a0fc1483df6e21
objectCharacteristics	fixity	messageDigestOriginator		GDAM
objectCharacteristics	size			7457
objectCharacteristics	format	formatDesignation	formatName	XML
objectCharacteristics	format	formatDesignation	formatVersion	1.0
objectCharacteristics	format	formatRegistry	formatRegistryName	
objectCharacteristics	format	formatRegistry	formatRegistryKey	
objectCharacteristics	format	formatRegistry	formatRegistryRole	
objectCharacteristics	significantProperties			UTF-8
objectCharacteristics	inhibitors	inhibitorType		
objectCharacteristics	inhibitors	inhibitorTarget		
objectCharacteristics	inhibitors	inhibitorKey		
creatingApplication	creatingApplication Name			BlastRadius XMetal
creatingApplication	creatingApplication Version			3.1
creatingApplication	DateCreatedBy Application			20021119
originalName				grl_2001pr3-f10-1.xml
storage	contentLocation	contentLocationType		filename
storage	contentLocation	contentLocationValue		grl_2001pr3-f10-1.xml
storage	storageMedium			Hard Disk

Example 6, Object 2: the XML metadata				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
environment	environment Characteristic			known to work
environment	environmentPurpose			render
environment	environmentNote			
environment	dependency	dependencyName		GettyVRA DTD 4.0
environment	dependency	dependencyIdentifier	dependencyIdentifier Type	
environment	dependency	dependencyIdentifier	dependencyIdentifier Value	
environment	software	swName		Microsoft Internet Explorer
environment	software	swVersion		6.0 SP1
environment	software	swType		renderer
environment	software	swOtherInformation		
environment	software	swDependency		Xalan
environment	hardware	hwName		
environment	hardware	hwType		
environment	hardware	hwOtherInformation		
signatureInformation	signatureInformation Encoding			
signatureInformation	signatureMethod			
signatureInformation	signatureValue			
signatureInformation	signatureValidation Rules			
signatureInformation	signatureProperties			
signatureInformation	keyInformation	keyType		
signatureInformation	keyInformation	keyValue		
signatureInformation	keyInformation	validationInformation		
relationship	relationshipType			local
relationship	relationshipSubType			Is metadata for

Example 6, Object 2: the XML metadata				
OBJECT				
semantic unit	semantic unit	semantic unit	semantic unit	Value
relationship	relatedObject Identification	relatedObjectIdentifierType		handle
relationship	relatedObject Identification	relatedObjectIdentifierValue		hdl:jpgt.gri/grl_2001pr3-f10-1.tif
relationship	relatedObject Identification	relatedObjectSequence		1
relationship	relatedEvent Identification	relatedEventIdentifierType		
relationship	relatedEvent Identification	relatedEventIdentifierValue		
relationship	relatedEvent Identification	relatedEventSequence		
linkingEventIdentifier	linkingEventIdentifier Type			GDAM-E
linkingEventIdentifier	linkingEventIdentifier Value			Update Object Descriptive Metadata 20050216 4:32:05 PM
linkingIntellectual EntityIdentifier	linkingIntellectual EntityIdentifierType			
	linkingIntellectual EntityIdentifierValue			
linkingPermission StatementIdentifier	linkingPermission StatementIdentifier Type			GDAM-R
	linkingPermission StatementIdentifier Value			2001.pr.3.f10.1-2

Example 6, Event 2		
EVENT		
semantic unit	**semantic unit**	**Value**
eventIdentifier	eventIdentifierType	GDAM-E
eventIdentifier	eventIdentifierValue	Update Object Descriptive Metadata 20050216 4:32:05 PM
eventType		Modification of display
eventDateTime		20050216T163205-0500
eventDetail		Ran XLST TRNFM2.1
eventOutcome Information	eventOutcome	Successful
eventOutcome Information	eventOutcomeDetail	Revise concatenation
linkingAgentIdentifier	linkingAgentIdentifierType	GDAM-A
linkingAgentIdentifier	linkingAgentIdentifierValue	KBoughida
linkingAgentIdentifier	linkingAgentRole	executor
linkingObjectIdentifier	linkingObjectIdentifierType	handle
linkingObjectIdentifier	linkingObjectIdentifierValue	hdl:jpgt.gri/grl_2001pr3-f10-1.xml

Example 6, Agent 2		
AGENT		
semantic unit	**semantic unit**	**Value**
agentIdentifier	agentIdentifierType	GDAM-A
agentIdentifier	agentIdentifierValue	KBoughida
agentName		Karim Boughida
agentType		Person

Example 6, Rights 2				
RIGHTS				
semantic unit	semantic unit	semantic unit	semantic unit	Value
permissionStatement	permissionStatementIdentifier	permissionStatement IdentifierType		GDAM-R
	permissionStatementIdentifier	permissionStatement IdentifierValue		2001.pr.3.f10.1-2
	linkingObject			hdl:jpgt.gri/grl_2001pr3-f10-1.xml
	grantingAgent			Getty Research Library, Special Collections Cataloging
	grantingAgreement	grantingAgreement Identification		
	grantingAgreement	grantingAgreement Information		
	permissionGranted	act		Replicate, Migrate, Modify, Use, Disseminate
	permissionGranted	restriction		Core descriptive metadata not to be changed without referral to Special Collections Cataloging
	permissionGranted	termOfGrant	startDate	0000
	permissionGranted	termOfGrant	endDate	9999
	permissionGranted	permissionNote		

4. SPECIAL TOPICS

As it compiled the Data Dictionary, the PREMIS working group felt several topics were important but too detailed for the Data Dictionary itself. The discussion here provides background information about semantic units and illustrates the thinking of the working group.

Format information

The working group discussed format at length, finding a need to come to agreement on some fundamental questions before specific semantic units could be defined. These issues included:

- What is a format?
- What types of objects have format?
- How does one identify a format?
- Is there a difference between a format and a profile?

The concept of format seems almost intuitive, but given the importance of format information to digital preservation the group wanted to be very specific about its meaning. In discussion the defining feature of a format emerged as the fact that a format has to correspond to some formal or informal specification; it cannot be a random or undocumented layout of bits. The definition in the Wikipedia, "a particular way to encode information for storage in a computer file," does not seem to emphasize this feature sufficiently.[7] The group drafted its own definition: *a specific, preestablished structure for the organization of a digital file or bitstream*.

Format is obviously a property of files, but it can also apply to bitstreams. For example, an image bitstream within a TIFF file may have a format that is defined within the TIFF file format specification. For this reason PREMIS avoids the term "file format" for the more generic "format."

A preservation repository must record format information as specifically as possible. Ideally, formats would be identified by a direct link to the full format specification. In real implementations an indirect link such as a code or string that can in turn be associated with the full format specification is more practical. The group saw format name as a somewhat arbitrary designation that could be used as this indirect link. However, two complications arose when the group attempted to define the semantic unit(s) to be used as this link.

First, format designations in common use, such as MIME types and filetype extensions, are not granular enough to be used in this way without the addition of version information. There was some discussion of whether the semantic unit defined for format name should include both format and version (e.g., "TIFF 6.0") or whether two semantic units should be defined, one for name and one for version. To allow existing authority lists such as MIME type to be used the group decided on two semantic units. In the Data Dictionary *formatDesignation* has two components: *formatName* and *formatVersion*.

Second, centrally maintained format registries are expected to be the best way to get detailed format information in the future.[8] In the PREMIS model the format name provides an indirect link to the format specification. In the registry environment not one but two things must be

known: what registry is being used, and what identifies the specification within the registry. The group discussed whether to combine all format identification into a single set of semantic units, or define different containers for registry and non-registry environments. A good argument for a single set is that a repository that uses its own authority list of format names to associate digital objects with specifications is, in essence, maintaining its own format registry, where the identification of the registry itself is simply assumed. However, with major format registries still under development the group was reluctant to make assumptions about what would be needed to use them. Ultimately, two containers were defined: *formatDesignation* and *formatRegistry*. In case different registries might emerge to provide different types of information, *formatRegistry* was made repeatable.

It is not uncommon for particular implementations of formats to be specified, often called profiles. For example, GeoTIFF (for geographic images), TIFF/EP (for digital cameras), and TIFF/IT (for prepress images) are compatible with the TIFF specification, but narrow it by requiring certain options, or extend it by adding tags. Because of this it is possible for a file to have more than one format, for example, both TIFF and GeoTIFF. The group discussed various options to accommodate this, such as making the format designation repeatable or defining format profile as a separate semantic unit. Instead the decision was to recommend recording the most specific format designation that applies. A repository (or formats registry) may use multipart format names (e.g., "TIFF_GeoTIFF" or "WAVE_MPEG_BWF") to achieve this specificity.

The group recognized that the most specific designation is a matter of opinion and will be implementation specific. For example, for a METS document (that is, an XML instance conforming to the METS schema) one repository may consider XML to be the most specific format, while another may consider METS to be the most specific format.

Environment

Digital materials are distinctly different from analog materials because a complex technical environment is interposed between user and content. Application software, operating systems, computing resources, and even network connectivity allow the user to render and interact with the content. Separating digital content from its environmental context can make the content unusable. Therefore, careful documentation of the technical environment associated with an archived digital object can be an essential component of preservation metadata.

Since digital environments are made up of components that can be broken down into smaller and smaller components, their descriptions can easily become extremely complex. It is also possible that these descriptions will tend to be the same for entire classes of digital objects, for example, for all files of a particular format. Both of these factors suggest that the most efficient model for collecting and maintaining environment metadata is a centralized registry. While the development of the PREMIS *environment* container did not presuppose the existence of such a registry, it might best be interpreted as a template for the types of information an environment registry might maintain, rather than what a repository is likely to record locally.

The semantic units associated with the *environment* container represent the PREMIS working group's recommendation of what a repository needs to know about an archived object's

environment. How this information is known—through a central registry, through locally recorded metadata, or both—is an implementation issue that must be resolved by the repository.

The working group decided to limit its scope to environment metadata associated with objects currently in the repository. Strategies for recording changes to the environment over time is an implementation issue and therefore beyond the scope of the Data Dictionary.

Sometimes multiple environments support a single digital format. The Data Dictionary acknowledges this possibility by making the *environment* container repeatable, but this is in no way intended to suggest that a repository should attempt to account for every possible software/hardware combination compatible with a particular archived object. Documented environments should, however, include the semantic unit *environmentCharacteristic*, populated by an appropriate value such as "minimum," "recommended," "known to work," etc. The working group generally agreed that at least a "minimum" environment should be specified. Specification of an environment that is "known to work" may be necessary in cases where it is important to preserve certain significant properties of the object—aspects of the object's original look, feel, and functionality. In these circumstances, it is useful to document an environment that is known to render these attributes faithfully.

The working group considered whether environment metadata can usefully apply to representations, files, and bitstreams. Although in most cases it does not apply to bitstreams, since software operates on known file formats, or in the case of compound objects, on aggregations of known file formats, it could have apply to bitstreams in some situations. For instance, it is possible for a single AVI file to be used as the common container for video streams each requiring the use of specialized rendering software. In an AVI file encapsulating heterogeneous bitstreams, each of the bitstreams may require a substantially unique preservation workflow. Setting the environment at the bitstream level maintains the important association that a particular bitstream requires a particular environment. If the environment were set at the file level, this association would be lost, complicating preservation efforts that require the disaggregation of the file.

However, in other cases a file format may contain two or more discrete bitstreams with wholly different semantics, but software designed to support the format may be able to correctly interpret and/or render any bitstream appearing within the file. For example, a TIFF viewer rendering an image knows to skip past the header information (a bitstream within the file) to reach the image data (a second bitstream within the file). It is not always necessary to detail separate environment information for each of these bitstreams if they are both handled by any rendering application compatible with the TIFF format specification.

Note that environment metadata may differ at the representation and file levels for a particular Object. For example, a browser is appropriate for rendering a multimedia Web page consisting of text, static images, animation, and sound components, but each component rendered separately would require different environments than the one for the compound object as a whole.

The working group decided not to recommend supplying separate environment information for both the preservation and the dissemination versions of an Object (where the dissemination

version is the version made available to users in a Dissemination Information Package or DIP). If dissemination versions are stored by the repository separately from preservation masters, these are stored objects and can be described by all metadata applicable to Object entities. If dissemination versions are generated "on the fly" from stored preservation masters, the environment to support them is not strictly a preservation issue. While environment information for dissemination versions may in some cases be useful, it is not core in the sense of being necessary to support the preservation process. (See also the discussion of dissemination format, page 4-10.)

Another point of discussion was whether the mechanism(s) by which archived objects are delivered from the repository to the user (i.e., over a network, on CD, on DVD, etc.) should be part of the environment metadata. The argument in favor of this is that the rendering environment must support the requirements implied by the delivery mechanism—if content is delivered on CD-ROM, the rendering environment must include a CD-ROM drive. However, the group decided that knowledge of the delivery mechanism was not essential to support the preservation process and therefore not core. Moreover, the usefulness of a delivery mechanism description will likely vary from repository to repository, depending on local dissemination policies.

Despite the critical importance of environment metadata for ensuring that digital materials remain accessible and usable over the long term, the working group reluctantly decided to make the entire *environment* container optional. The group could not assert categorically that every preservation strategy that exists or might be developed would require a knowledge of environment information. However, the fact that the *environment* container is currently optional does not indicate that the working group considers this metadata unimportant. Well-documented environments for access and use are an essential component of most digital preservation strategies. Much work remains to be done, however, to establish practical mechanisms for collecting, storing, and updating this metadata.

Object characteristics and composition level: the "onion" model

When an object is compressed or encrypted, the format of the object is determined by the compression or encryption scheme. At the same time, the object has an underlying format that is different. Objects such as these pose the problem of how to describe complex layers of encodings and encryptions so that they can be reversed correctly. The group arrived at the metaphor of an onion: a digital object can be wrapped in layers of encodings that need to be "peeled off" in a particular sequence. The onion model is implemented by treating each layer as a "composition level," and organizing metadata into sets of values pertaining to each layer.

The simplest example is a single file with no encoding or encryption. In this case there would be one instance of the semantic unit *objectCharacteristics* with *compositionLevel* value of 0 (zero). The object characteristics of a simple PDF, for example, might include a message digest, a size of 500,000 bytes, a format of PDF 1.2, inhibitors such as no printing allowed, and creating application of Adobe Acrobat. If a compressed version of that PDF file were created using the UNIX gzip utility and stored in the repository, the compressed file would be described with two *objectCharacteristics* blocks. The first, with *compositionLevel* zero, would be the same as for the simple PDF, and the second with *compositionLevel* 1, would record another message digest, a

smaller size, and a format of gzip. This could continue for as many layers as necessary to describe the object completely.

To extract the content object, one works backwards through the composition levels from highest to lowest, using an application appropriate to the format of the layer. In the example above, to get to the PDF one applies a tool that understands the gzip format. Having un-gzipped the content, it can be compared to the size and fixity information previously stored to determine that the correct object has been extracted. (In practice, some of the encodings have checking mechanisms built in.)

Note that this model assumes that the object is being stored with the composition layers preserved. If the archive has already removed the layers and is storing the base object, the information about the removal of the layers is Event data rather than composition data. That is, if a decompressed version of object A is created and called object B, A is related to B by a derivation relationship (*sourceOf*) with a related decompression event.

Bitstreams and filestreams are not composition layers. If an archive chooses to manage bitstream or filestream objects, they are separate objects whose storage location is at an offset inside a file, which is itself a separate object with characteristics and metadata and its own storage location. Each of these may have composition layers including encryption and encodings. The level-zero composition layer of the file would be the file without encryption or encoding; that a bitstream inside that file is a managed object is a separate issue (and object) distinct from the layers of encodings of the file.

Formats such as tar and ZIP that can bring together ("package") several files into one file present a related but not identical problem. If the package consists of only one object, one could treat the package as yet another composition layer; for example, a file that is encrypted, then zipped would have three composition levels. If the package contains more than one file, however, it should be treated as a separate object that provides the storage location for the contained objects so that there can be distinct metadata records for each of the contained objects. For example, a ZIP file containing two PDF files should be treated as three objects: the ZIP file with a base composition format of ZIP, and two other objects whose storage location is inside the ZIP file. As with bitstreams, the objects inside the ZIP file object are logically distinct from the containing object. They each may have completely different sets of metadata and indeed may have additional composition layers as well. One could imagine an encrypted ZIP file containing two files that are themselves each separately encrypted. There would then be three objects, each with two composition levels.

Fixity, integrity, authenticity

In the process of defining core elements for preservation the working group gave considerable attention to the concepts of fixity, integrity, and authenticity of digital objects. Objects that lack these features are of little value to repositories that have the mission to protect evidentiary value or indeed to preserve the cultural memory.

In the PREMIS Data Dictionary the information needed to verify fixity (that an object is unchanged since some earlier point in time) is described by a set of semantic components under

the semantic unit *objectCharacteristics*. Running a fixity check program on an object to detect unauthorized changes to it is detailed as an Event. In the analog world acts of publication and production serve to fix an object in time. In the digital domain hash algorithms that create a message digest can be used to implement a fixity check for an object. If the message digest created by an algorithm at one point is identical to the message digest created by the same algorithm at a later point, this indicates the object did not change during the interim. In fact, recommended practice is to create and test at least two message digests using two different algorithms to be certain that an object is fixed.

While this procedure can indicate with some confidence that an object has not changed over time, it does not address the object's integrity or authenticity. In the PREMIS model, verifying the integrity of an object is considered an Event. Format identification and validation are key indicators of the integrity of a file. Software technology such as JHOVE can verify that a format is what its file extension claims as well as determine the level of compliance to a particular format specification.[9] The integrity of a representation may have to be verified by special programs that understand the structure of the representation. If the representation includes structural metadata, the structural metadata can be used to test that all files are present and appropriately named.

The authenticity of a digital object is the quality of being what it purports to be. As the Digital Preservation Coalition (DPC) explains, "In the case of electronic records, [authenticity] refers to the trustworthiness of the electronic record as a record…Confidence in the authenticity of digital materials over time is particularly crucial owing to the ease with which alterations can be made."[10]

Authentication, or the demonstration of authenticity, is multifaceted, and includes both technical and procedural aspects. Technical approaches may include the maintenance of detailed documentation of digital provenance (the history of the object), the preservation of a version of the object that is, bit-wise, identical to the content as submitted, and the use of digital signatures. PREMIS metadata supports the documentation of provenance by defining semantic units associated with events and allowing linking between Object entities and Event entities. Fixity can be tested against stored message digest information and the testing itself recorded as an event. Digital signatures are discussed next.

Digital signatures

Preservation repositories use digital signatures in three main ways:

- For submission to the repository, an agent (author or submitter) might sign an object to assert that it truly is the author or submitter.
- For dissemination from the repository, the repository may sign an object to assert that it truly is the source of the dissemination.
- For archival storage, a repository may sign an object so that it will be possible to confirm the origin and integrity of the data.

The first and second usages are common today as digital signatures are used in the transmission of business documents and other data. Typically, validation takes place shortly after signing and

there is no need to preserve the signature itself over time. In the first case the repository may record the act of validation as an Event, and save related information needed to demonstrate provenance in the event detail. In the second case the repository might also record the signing as an Event but the use of the signature is the responsibility of the receiver. Only in the third case, where digital signatures are used by the repository as a tool to confirm the authenticity of its stored digital objects over time, must the signature itself and the information needed to validate the signature be preserved.

Just as with a pen-and-ink signature or seal, reliable digital signatures require that:

- The process of producing a signature, such as a person's physical signature, is considered to be unique and uncopyable.
- The signature is related to the content of the document that was signed.
- The signature can be recognized by others to be the signature of the person or entity that produced it.

To create a digital signature, first a secure hash algorithm (SHA) is applied to content (a file or bitstream) and used to produce a short message digest from that content. The message digest is then encrypted using asymmetric cryptography. Asymmetric cryptography is based on using a pair of keys: a private key to encrypt and a public key to decrypt. The private key must be held secretly and securely by the signer, ideally in secure hardware. This accomplishes the goal of a unique and uncopyable signature. Since the message digest that is encrypted is tied directly to the content this also accomplishes the goal of relating the signature to the content. The signature can be verified by decrypting the signature with the signer's public key and comparing the now-decrypted digest with a new digest produced by the same algorithm from the same content. If the content had been changed, the comparison would fail.

The goal of connecting the signature to the signer is based on establishing trust. For example, agent A ought to trust a signature by agent B if a third party trusted by A asserts that the signature is truly B's. This principle governs notarization of written signatures. The same approach is used in digital signatures, where a trusted third party certifies that a particular key is indeed the public key of the signer. This extends to a chain of trust, whereby the trusted body trusts an intermediary which in turn certifies the signer's public key. This process is typically, but not necessarily, implemented using X.509 certificates, or certificate chains.

This is important for preservation, because the standard current mechanisms for establishing trust in a certificate relies on a set of services that are not likely to be available for the long term. For preservation widely sharing and safely storing the public key as a formal document may be a more suitable approach. For example, a university might regularly publish its public key in its annual report and make it available on its Web site.

Digital signature metadata

For a preservation repository to later validate a digital signature the repository will need to store:

- The digital signature itself.
- The name of the hash algorithm and encryption algorithm used to produce the digital signature.

- The parameters associated with these algorithms.
- The chain of certificates needed to validate the signature (if a certificate model is used to relate the signer and the signer's public key).

It is recommended that a repository also store the definitions of the algorithms and relevant standards (e.g., for encoding the keys) so that these methods could be reimplemented if necessary.

The W3C's *XML-Signature Syntax and Processing* (*XML Signatures*) is a de facto standard for encoding digital signatures that provides a clear functional model for them.[11] PREMIS adopted the names and structure of semantic units from that specification where applicable. However, *XML Signatures* is both too generalized and too specific to be applied directly in this context. It is too generalized because it allows multiple data objects (files and/or bitstreams in the PREMIS model) to be signed together, while in the PREMIS model a digital signature is a property of a single object. It is too specific because it prescribes a particular encoding and validation methodology that is not universally applicable.

The Data Dictionary defines the following structure:

signatureInformation
    signatureInformationEncoding
    signer
    signatureMethod
    signatureValue
    signatureValidationRules
    signatureProperties
    keyInformation

The digital signature itself is the *signatureValue*. The hash and encryption algorithms used are recorded in *signatureMethod*; for example, "DSA-SHA1" would indicate the encryption algorithm is DSA and the hash algorithm is SHA1. The parameters associated with these algorithms are recorded in *keyInformation,* and if X.509 certificates are used to validate the signature they are also placed in *keyInformation*. Information about the generation of the signature, such as date and time, is stored in *signatureProperties.*

The semantic units discussed above have analogs in the *XML Signatures*. Three semantic units were added: *signatureInformationEncoding, signer,* and *signatureValidationRules.* The semantic unit *signatureInformationEncoding* indicates the encoding of the values of the subsequent semantic units; this is not included in *XML Signatures* because that document mandates a particular encoding, which cannot be assumed in a broader context. The name of the signer can be extracted from the signer's certificate, but isolating this in *signer* makes it easier to access. Documentation of the process to be used in validating the signature is stored or pointed to in *signatureValidationRules.*

Non-core metadata

The working group decided not to include some metadata concepts in the Data Dictionary. Unless otherwise noted this does not imply that these semantic units are not necessary or important in other contexts. For specific implementations there may be legitimate reasons to record this information in some form.

Aggregation: Aggregation means the embedding of objects into a larger object (rather than a collection of discrete objects). The property of being an aggregate can be inferred from the presence of multiple files and/or bitstreams, which will be documented in *objectCharacteristics*. That semantic unit makes no distinction between an aggregation that is ingested and an aggregation that is created by the preservation repository for storage or other purposes; however, this distinction was not felt to be core.

Quirks and anomalies: The *Framework* defines "quirks" as "any loss in functionality or change in the look and feel of the Content Data Object resulting from the preservation processes and procedures implemented by the archive." The working group used "anomalies" to describe aspects of an object that do not meet the specification for the object. The discussions of quirks and anomalies centered on whether they should be defined as the outcomes of Events or classified as properties of Objects.

The argument for treating these as outcomes of events is that quirks by definition result from an event, and anomalies are discovered through the event of validation. If treated this way, an anomaly would be recorded as part of the description of a validation event; the semantic unit *eventOutcome* would indicate problems, and the semantic unit *eventOutcomeDetail* would record the known anomalies.

An argument for treating quirks and anomalies as properties of an object is that this appears to elevate them in importance and gives them a direct as opposed to indirect association with the object.

The decision is arbitrary. The Data Dictionary treats quirks and anomalies as outcomes of events, recorded in *eventOutcomeDetail*.

Byte order: Byte order determines whether numbers of more than eight bits are stored from most to least significant ("big-endian") or from least to most significant ("little-endian"). Byte order is hardware dependent and can cause problems when data is shared between different types of computers. However, it does not pertain to all formats. For example, it is irrelevant for encodings such as ASCII, where one byte equals one character, and UTF-8, which is byte-order independent. The working group decided that byte order might better be treated as format-specific technical metadata, and noted that NISO/AIIM Z39.87 (Technical metadata for digital still images) includes byte order as technical metadata for images.[12]

Character encoding: This element is important, but it is format-specific technical metadata, useful only for text files and files that can include text.

Dissemination format: A great deal of discussion centered on whether dissemination format was in scope. The working group concluded that the "preservation format" is the object of preservation activity, which may or may not be the same as the dissemination format. Whether or not the preservation format is immediately renderable or is transformed for dissemination is an implementation choice. For example, if the preservation format is a TIFF image, one preservation repository might create a dissemination version (say a JPEG image) on the fly for user access, while another repository might deliver the TIFF master. A third repository might store and process both the TIFF master and the JPEG access copy.

The Data Dictionary does not address the creation of metadata objects that are not stored in a preservation repository. Although the group agreed that dissemination format is important to a repository operationally, it is not core to preservation processes.

Embedded metadata: One implementation used a metadata flag to indicate whether a file object contained embedded metadata. The group agreed to leave this indicator out of the Data Dictionary for now, with the understanding that this will probably have to be revisited in the next several years as more and more formats include embedded metadata. For the time being if embedded metadata is extracted and stored elsewhere, there is no need to note the existence of embedded metadata in the file.

The group also discussed the distinction between standard embedded metadata defined by a file format and locally defined metadata that might be inserted into a file header. Any local divergences from standard formats will likely need to be documented as anomalies.

Event type: The semantic unit *eventType* is core, but not all types of events were considered core, and some were deliberately omitted from the list of suggested values provided in the Data Dictionary. Among these, the group agreed that microfilming (preservation reformatting), moving a file offline, and media refreshment were not core events. Events likely to be handled by a storage system, such as mirroring or the creation of backup copies, would probably be recorded in a system log and are not raised to the level of an event that has metadata associated with it.

Event next occurrence: Many actions taken by a preservation repository are performed periodically, for example, daily or weekly monitoring actions. It could be useful to record an action date or "tickler" for the next scheduled occurrence of an event. This was considered a matter of repository policy and implementation, and not a core property of Events.

File pathname/URI: This element was seen as both implementation specific and system dependent. It was not seen as information that would be explicitly recorded in a repository. Often the pathname or location of an object is not known in a content management system; only the unique object identifier of the asset is known and needed for retrieval. Alternatively, in some systems such as the Handle system, the *objectIdentifier* alone is usually sufficient for retrieving the file. Therefore, a broader, less system-dependent semantic unit was defined: *contentLocation* can be interpreted narrowly (a value could be an exact path or a "fully qualified" path or filename) or broadly (any information needed to retrieve a file from a storage system, which may include information used by a resolution system such as the Handle system).

Global identifier: The *Framework* included a "Global Identifier" defined as an identifier known outside of the repository system. The group did not consider the distinction between an externally known identifier and an internally known identifier to be significant. An internal identifier could easily become known outside of the repository and then would be a global identifier. The issue was raised whether internal identifiers would be sufficiently unique in an external context to function as a global identifier. However, as the *objectIdentifier* always includes an identifier type as well as value, the combination of type and value would be unique even if the type were some local repository scheme.

The *Framework* also implied that a Global Identifier would be a standard identifier such as ISBN or ISSN. However, because these schemes designate an abstract bibliographic entity or set of items, not the specific content data object in the preservation repository, they are really descriptive metadata rather than preservation metadata. ISBNs, ISSNs, and similar standard identifiers are likely to refer to many different representations held in many different preservation repositories, with no way to distinguish between them. Therefore, the identifier used by the repository must in practice be the "global" identifier.

MIME type: The Internet Media Type and SubType (commonly called "MIME type") was subsumed under *formatIdentification*. Format identification is intended to be more granular and precise than MIME type and includes multiple format identification schemes, of which MIME type can be one. A MIME type alone is not rigorous enough to identify formats for digital preservation—not all formats have MIME types, it is too coarse a typing mechanism, it is not necessarily current, and it provides no versioning information. Good practice is to include format name and version and use MIME type only if no other data is available.

Modification date: The PREMIS data model asserts that metadata describes only one object at any given time. If an object is changed or modified, a new object is created that is related to the previous one. Each object then has its own set of metadata, and the relationship between the two is also described. The model does not allow for modifying an object and keeping a set of metadata that describes a history of changes about that object. Therefore, there would be no modification date of an object, only a creation date for the new object. The act of modification (e.g., migration, normalization) is documented as an Event and is linked to the object that is created as a result of these processes. Modification date was considered by the group in the context of an Event record that is associated with an Object, rather than a date associated with a history of changes to the metadata associated with an object.

Object type: The group discussed the desirability of having a semantic unit for a genre or media type that would classify objects on a much higher level than format. There is such an element in the METS schema, but currently there is no controlled vocabulary defined for its value. The group argued that object type is useful information to know at the system level (for example, for performing preservation actions on an entire class of materials) and possibly for categorizing objects in terms of how they are rendered in certain environments. High-level object typing is probably more useful for exchange and access to objects than for preservation purposes. However, developing a universally acceptable list of object types is beyond the PREMIS's scope and, without an authority list of types, this element would not be entirely useful outside of the repository. This element might be recorded in descriptive metadata.

Permanence levels: The group discussed how the National Library of Medicine's Permanence ratings intersected with PREMIS work.[13] The permanence-level rating appeared to be less a property of an Object entity than a property of an entity defining business rules. The group had already decided that business rules were out of scope.

Profile conformance: A "profile" can be seen as a subtype or refinement of a format; for example, the GeoTIFF specification can be seen as a profile of TIFF. There was a question of whether profile conformance should be seen as something separate from format validation. The decision to recommend recording only a single format at the most specific level obviated the need to define a separate semantic unit for profile conformance.

Reason for creation: This metadata element was defined in the *Framework*. The working group concluded that for objects created by the preservation repository (e.g., a normalized version of a file) the reason for creation could be recorded as part of the *eventDetail* for the event of creation. However, the group did not consider at length events or processes that occur before ingest and was not convinced that these were core knowledge for a preservation repository. Some of the context surrounding object creation may be documented in relation to the Object entity in *creatingApplication*. The group expressed some reservations about the life-cycle model used by the *Framework* (origin, pre-ingest, ingest, archival retention, etc.) as being too restrictive.

Sibling relationships: The group discussed whether sibling relationships (children of the same parent) should be made a separate category of relationship. It was agreed that sibling relationships always have a structural relationship (and may possibly also have a derivation relationship), and should therefore fall under these relationship categories. What renders them potentially confusing is that the parent is not always stored within the repository system. For example, a report created using Microsoft Word might be processed to create a PDF version for printing and an HTML version for online display. If both of these representations were stored in the preservation archive without the original Word file, it might not be obvious that the two representations have a sibling relationship.

5. METHODOLOGY

The Core Elements Subgroup began by analyzing the Preservation Description Information recommendations of the earlier Preservation Metadata Framework working group. In OAIS, Preservation Description Information includes reference information (identifiers and bibliographic information), context information (how objects are related to each other), provenance information (the history of digital content), and "fixity" information. Members of the subgroup from institutions actively running or developing preservation repositories mapped elements from the *Framework* to those in use in their own systems. The subgroup also reviewed published specifications from organizations and projects that did not have representatives on the PREMIS working group.

It became clear that the prototype elements detailed in the *Framework* did not always correspond to elements implemented in practice. However, the exercise provided a common denominator for diverse implementations; the group discussed each element in conference calls to discover commonality in usage. Widely used elements formed the beginning of a set of core elements, which were then mapped to appropriate entity types as the data model evolved.

In the OAIS and the *Framework*, technical metadata is considered Representation Information rather than Preservation Descriptive Information. Because there are few technical metadata elements in the *Framework*, the working group compiled a list of potential technical metadata based on specifications for the proposed Global Digital Format Registry (GDFR), supplemented by data elements used in the repository systems of members' institutions.[14] Each element on the list was then discussed at some length, and any element that was format specific or implementation specific was regarded as non-core. In some cases outside experts were asked to help with particularly difficult areas, including formats, hardware and software environment information, and digital signatures.

The process for determining which semantic units were core involved analysis and discussion of a selection of elements from various sources and a determination of whether they were in scope. In general, the working group excluded these candidates from the Data Dictionary:

- Metadata elements that could be grouped into broader categories.
- Format-specific, implementation-specific, or policy-driven elements.
- Elements outside the PREMIS scope.
- Elements for which information could be obtained easily and reliably from the object itself or other sources.

This page intentionally blank.

6. IMPLEMENTATION CONSIDERATIONS

PREMIS conformance

PREMIS conformance requires a preservation repository to follow the specifications outlined in the Data Dictionary. For example, if the repository claiming to be PREMIS-conformant implements a metadata element sharing the name of a semantic unit in the Data Dictionary, it is expected that the repository's metadata element will also share the definition of the semantic unit. Metadata not defined in the Data Dictionary may certainly be used, but non-PREMIS elements should not conflict with or overlap with PREMIS semantic units. In other words, local metadata can be used to extend but not modify the PREMIS semantic units. Data constraints and applicability guidelines in the Data Dictionary must also be adhered to. For repeatability and obligation, PREMIS conformance permits more stringent but not more liberal application. That is, a semantic unit defined in the Data Dictionary as repeatable can be treated as not repeatable within a repository, but not vice versa.

The PREMIS Data Dictionary designates some semantic units as mandatory when describing representations, files, and/or bitstreams. The mandatory semantic units represent the minimum amount of information 1) necessary to support the long-term preservation of digital objects, and 2) that must accompany a digital object as it is transferred from the custody of one preservation repository to another. There is no prescribed strategy for collecting, storing, or managing the mandatory semantic units within the repository's internal systems. Nor is there a minimum level of information that must be explicitly recorded and maintained locally by the repository. In general, the mandatory semantic units of the Data Dictionary represent the information that a preservation repository must be able to associate with any archived digital object in its possession. The specific means of association (e.g., local metadata storage, shared registries, etc.) are implementation issues and outside the scope of the Data Dictionary.

When a digital object is exchanged between two preservation repositories, the repository sending the object must be able to extract from its systems or from other sources the information needed to populate the semantic units marked mandatory in the Data Dictionary. This information must conform to the specifications in the Data Dictionary and must be packaged with the digital object before its transfer to the second repository. The PREMIS working group believes that this information represents the minimum amount for the second repository to accept custody of the digital object and assume responsibility for its long-term preservation.

Some PREMIS semantic units are equivalent to metadata elements in other metadata schemas. If metadata is taken from other schemas to populate PREMIS semantic units, care must be taken to ensure that this information conforms to the requirements and constraints associated with the corresponding semantic unit in the PREMIS Data Dictionary. Harmonizing the PREMIS Data Dictionary with other metadata schemas in cases where they overlap would help minimize conformance issues. For example, the Z39.87 metadata standard (Technical Metadata for Digital Still Images) revised some of its elements to harmonize them with equivalent semantic units in the PREMIS Data Dictionary.

6. Implementation Considerations

Sometimes a preservation repository exchanges digital objects with parties that are not themselves preservation repositories. When a party submits an object to a preservation repository for archival retention, it is unlikely that the submitter will be in a position to supply the full range of information needed to populate the mandatory semantic units. Instead, it will supply a subset of this information whose extent, ideally, is determined by prior arrangement between the submitter and the repository. Whatever the extent of this subset, any information supplied by the submitter should conform to the Data Dictionary. The repository's ingest process would then supply the rest of the information for the mandatory semantic units.

When a repository disseminates an archived digital object to a user, it is unlikely that the user will be interested in the full range of mandatory semantic units associated the archived object. Instead, the user would be provided with a subset of these semantic units. As in the case of submission, whatever the extent of this subset, any information supplied by the repository should conform to the Data Dictionary.

Achieving interoperability across a network of preservation repositories and other stakeholders requires a shared view of the metadata needed to support long-term preservation, formalized as an implementable schema. PREMIS conformance and the mandatory semantic units are intended to fill this need.

Implementation of the data model

The PREMIS data model is meant to clarify the meaning and use of the semantic units in the Data Dictionary. It is not intended to prescribe an architecture for implementation.

The working group believed that most preservation repositories will need to deal in some way with conceptual entities Objects, Agents, Events, and Rights, and found it useful to distinguish between the properties of subclasses of objects, such as files and filestreams, bitstreams, and representations. A particular repository implementation, however, may need to be more or less granular or define different categories of entity altogether. PREMIS recommends that any data model used be clearly defined and documented, and that metadata decisions be consistent with the data model.

Sets of semantic units may be grouped and related indirectly to particular entities. For example, *environment* is a property of Objects. Logically, each file has one or more associated environments. However, in many cases the environment is determined by the file format; that is, all files of a particular format will have the same environment information. This could be handled in many different ways by different implementations. Three examples:

- Repository 1 uses a relational database system for metadata. It has a file table with a row describing each file object; one column in the file table is format. A format table has a row for each file format; columns in the format table store environment information associated with that format.

- Repository 2 also uses a relational database system. It has a file table with a row for each file object, and an environment table with a row for each unique set of environment information.

The file table has a column for a pointer to the appropriate environment information for each file.

- Repository 3 uses a system that models representations as containers and files as objects within those containers. Each object consists of a set of property/typed value pairs. Properties define roles for values. Property and type descriptions are themselves objects whose identifiers are drawn from the same namespace as other object identifiers. A file object may include a format property. Because format description is also an object, it could include an environment property, which in turn would point to an environment description object. Alternatively, a file object could include an environment property directly.

Storing metadata

The survey by the Implementation Strategies Subgroup showed that repositories have implemented several different architectures for storing metadata. Most commonly, metadata is stored in relational database tables. It is also common to store metadata as XML documents in an XML database, or as XML documents stored with the content data files. Other methods include proprietary flat file formats and object-oriented databases. Most respondents were using two or more of these methods. (For more information, see the *Implementation Survey Report*.)

Storing metadata elements in a database system has the advantages of fast access, easy update, and ease of use for query and reporting. Storing metadata records as digital objects in repository storage along with the digital objects the metadata describes also has advantages: it is harder to separate the metadata from the content, and the same preservation strategies that are applied to the content can be applied to the metadata. Recommended practice is to store critical metadata in both ways.

Supplying metadata values

Most preservation repositories will deal with large quantities of materials, so it is desirable to automate the creation and use of metadata as much as possible. The values of many PREMIS semantic units can be obtained by parsing files programmatically, or can be supplied as constants by repository ingest programs. In cases where human intervention might be unavoidable, the group tended to pair a semantic unit requiring a coded value with a second semantic unit allowing a textual explanation.

When information is supplied by the individual or organization submitting the objects to the repository, recommended practice is for the repository to attempt to verify this information by program whenever possible. For example, if a filename includes a file type extension, the repository should not assume the file extension necessarily indicates the format and should attempt to verify the format of the file before recording this as metadata.

To facilitate automatic processing, the use of controlled vocabularies is recommended whenever applicable. PREMIS assumes that repositories will adopt or define controlled vocabularies useful to them; only a small number of semantic units require values defined in the Data Dictionary. However, the use of many different vocabularies will impede interoperability. Recommended practice is for a repository to note the source of each controlled vocabulary used when exporting

metadata for exchange. The group expects that as more experience is gained in digital preservation, and as repositories begin to exchange PREMIS-conformant metadata some dominant vocabularies may emerge.

In Resource Description Framework (RDF), use of resource URIs as property values is encouraged, and many XML Schemas require attribute values to be URIs.[15] For example, in the *XML Signatures*, the value of the signature method algorithm must be a URI, such as "http://www.w3.org/2000/09/xmldsign#dsa-sha1".

In general, resource URIs are allowable as values for semantic units in the PREMIS Data Dictionary, unless some noted constraint would disallow this. However, the working group was wary of recommending this practice for preservation. Resolution of URIs depends on a protocol that while currently ubiquitous is outside the control of the preservation repository. Also, the group felt strongly that any information needed for long-term preservation should be stored within the repository itself. If this information is stored as a preservation object, it is best referenced by the repository's *objectIdentifier*. Information stored otherwise should still be under the direct control of the repository. Therefore, most examples in the Data Dictionary are names of values rather than resource URIs. The equivalent of the example above might be simply "DSA-SHA1," which should be assumed to be a constant whose meaning is known to the repository through some table or other documentation under the control of the repository organization.

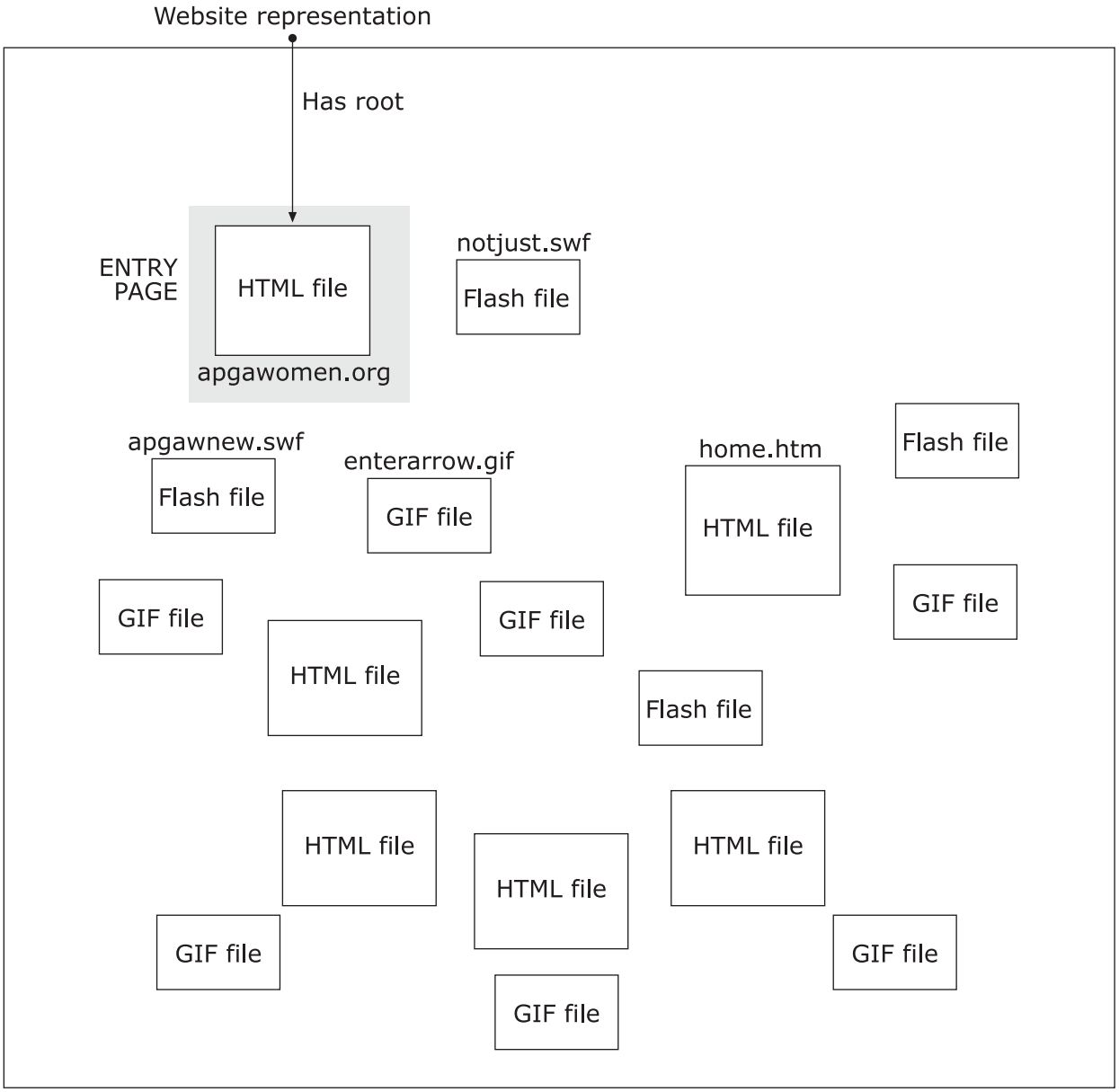
Preservation metadata for Web sites and Web pages

The PREMIS working group had several discussions about the peculiarities of Web sites and Web pages that are archived for preservation purposes. Many of the current projects archiving Web sites have dealt with them in terms of access rather than preservation, so there is little experience in applying preservation metadata. A particular problem with Web sites and Web pages is the difficulty viewing them in an implementation-neutral way.

- **Objects:** Since Web sites are complex, with many layers of component objects and relationships, there are various interpretations of what really constitutes a site or page as an object of preservation. Some harvesters aggregate objects into larger files. For example, the harvester used by the Internet Archive aggregates objects into an ".arc" or ".warc" file. One of these files may contain page components from more than one Web site, and pages from one Web site may be spread over multiple .arc files. In this case an additional tool is needed to bring together all the pieces of a logical Web page. With this situation the repository must decide whether it considers its object of preservation to be the .arc file or the conceptual Web site. This is an implementation decision, similar to that made about other file types that package filestreams together, like ZIP files. The working group thought it may be advantageous to consider the object of preservation the conceptual Web site. Web harvesting programs will undoubtedly evolve, and any particular container format is not likely to have a long life span.

- **Relationships among components:** There are several ways to model the relationships between the components of a Web site, even leaving aside the complexities of multiple captures. Assume a Web site consists of a finite number of pages that in turn consist of one or more files; for example, one page may be simply a PDF file, while another may be an HTML file, several images, and a Flash animation. At one extreme, a repository could designate one file as the parent for each page and describe the others as ordered or unordered children, duplicating relationship information stored internally within files as metadata. At the other extreme, the Web site as a whole could be described as a representation, and the many files that make up the site could be described as file Objects having an "is part of" relationship to the Web site representation (see page 6-6). Alternately, the repository could consider each page a representation as well, with the file Objects composing each page having an "is part of" relationship to the page, and the representation Object for each page having an "is part of" relationship to the Web site representation (see page 6-7) Working with a representation of the whole Web site means the repository does not need to maintain more hierarchical relationship information because linking information is contained within the files themselves. All three models, however, require the designation of one file as the root file, the file you go to first in order to reassemble the representation.

Website representation
internal relationships
option 1



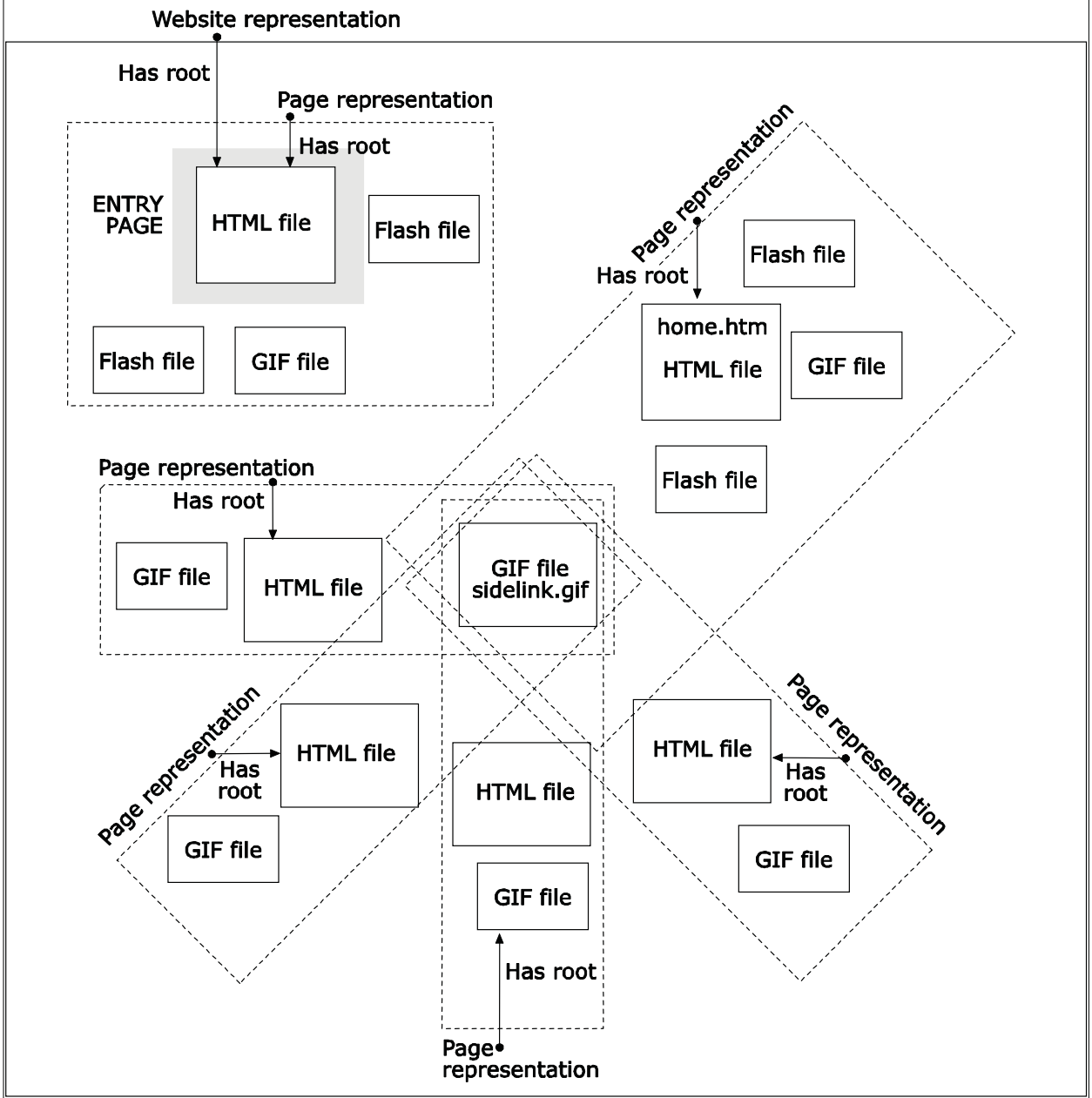
Notes:

1) Only relationship information is that all are part of the website representation

2) All objects have "Is part of" relationship to the website representation

3) HTML file serving as entry page is designated as root file

Website representation
internal relationships
option 2



Notes:

1) Relationships: All objects are part of representation.

2) Separate representations for each page (all files)

3) HTML designated "root" for each

4) Each object has "Is part of" relationship to its page

5) One GIF file (sidelink.gif) has "Is part of" relationship to each page representation in site

- **Relationships among captures:** There appears to be no widely accepted model for representing the relationships between different captures of a Web site. If different time-dependent captures of a Web site or Web page are treated as formally different Intellectual Entities, like different editions of a work, the metadata describing these relationships would formally be descriptive metadata and outside the scope of PREMIS. However, a repository might prefer to treat all captures of a Web site or Web page as Objects related temporally. Implementers of this approach might consider recording temporal as well as structural and derivative relationships as metadata.

- **Capture date:** Another point of discussion was how to treat the date the Web site or Web page was captured. Since different captures could be considered different Intellectual Entities, the date captured could be considered descriptive metadata and therefore out of scope for PREMIS. On the other hand there is a semantic unit for recording the date that an application created an object (*dateCreatedByApplication*) that would pertain to an aggregate created by the harvester, like the Internet Archive's .arc files. If the harvested files were altered in any way by the harvester (that is, if they are not exact copies of the source files), this element should be used for the date of capture, since the harvester is literally the creating application of the harvested files. This creates another problem: where to record the creating application and create date of the source files. In any case, the act of capture can be recorded as a pre-ingest event.

The PREMIS working group thought it would most flexible to provide a few alternatives for describing the complex relationships between files that constitute Web sites. Best practices are likely to emerge after some further experimentation.

7. GLOSSARY

Early in its work, the PREMIS working group realized the need for a glossary, since a common vocabulary seemed to be lacking in discussions about preservation metadata. This glossary defines a number of terms used in this report; the working group recognizes that in some cases other groups may have given different meanings to some of these terms. Terms were selected for inclusion in the glossary on the basis of their relative importance or frequency of occurrence in the report and Data Dictionary, and/or the potential for ambiguity or confusion in their interpretation.

Terms that are capitalized are defined elsewhere in the glossary.

Actionable: Property of a Semantic Unit indicating that the Semantic Unit is recorded/coded in such a way as to be machine processable.

Agent: Actor (human, machine, or software) associated with Events occurring over the course of a Digital Object's life cycle.

Anomaly: Property of a Digital Object that does not meet the specification for the Digital Object.

Authenticity: Property that a Digital Object is what it purports to be.

Bit-Level Preservation: Preservation strategy in which the sole objective is to ensure that a Digital Object remains fixed (unaltered) and viable (readable from media). No effort is made to ensure that the Digital Object remains renderable or interpretable by contemporary technology.

Bitstream: Contiguous or non-contiguous data within a file that has meaningful common properties for preservation purposes. A Bitstream cannot be transformed into a standalone File without the addition of file structure (headers, etc.) and/or reformatting the Bitstream in order to comply with some particular Format. Note that this definition is more specific than the common definition of "bitstream" used in computer science.

Business Rules: Policies and other restrictions, guidelines, and procedures governing the administration and operation of a Preservation Repository.

Byte: A component in the machine data hierarchy usually larger than a bit and smaller than a word; now most often eight bits and the smallest addressable unit of storage. A byte typically holds one character. (From FOLDOC: foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?query=byte)

Capture: Process by which a Preservation Repository actively obtains Digital Objects for long-term retention, for example, a harvesting program that collects Web sites. Note that the Capture process precedes the Ingest process.

Complex Object: See Compound Object.

7. Glossary

Compound Object: Digital Object composed of multiple Files, for example, a Web page composed of text and image files.

Compression: Process of coding data to save storage space or transmission time. Although data is already coded in digital form for computer processing, it can often be coded more efficiently (using fewer bits). For example, run-length encoding replaces strings of repeated characters (or other units of data) with a single character and a count. There are many compression algorithms and utilities. Compressed data must be decompressed before it can be used. (From FOLDOC: foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?query=compression)

Container: In the Data Dictionary, a Semantic Unit used to group other related Semantic Units. A container Semantic Unit takes no value of its own.

Core Preservation Metadata: Semantic Units that most Preservation Repositories will need to know in order to support the digital preservation process. Core Preservation Metadata should be independent of factors such as specific preservation strategy, type of archived content, and institutional context.

Data File: See File.

Data Object: See Digital Object.

Deaccession: Process of removing a Digital Object from the inventory of a Preservation Repository.

Decompression: Process of reversing the effects of data Compression. (From FOLDOC: foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?decompress)

Decryption: Process of employing any procedure used in cryptography to convert ciphertext (encrypted data) into plaintext. (From FOLDOC: foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?decryption)

Deletion: Process of removing a Digital Object from repository storage.

Dependency Relationship: Relationship where one Digital Object requires another Digital Object to support its function, delivery, or coherence of content.

Derivation Relationship: Relationship between Digital Objects where one Object is the result of a Transformation performed on the other Object.

Descriptive Metadata: Metadata that serves the purposes of discovery (how one finds a resource), identification (how a resource can be distinguished from other, similar resources), and selection (how to determine that a resource fills a particular need, for example, for the DVD version of a video recording). (From Caplan, *Metadata Fundamentals for All Librarians*, ALA Editions, 2003)

Digital Migration: See Migration.

Digital Object: Discrete unit of information in digital form. A Digital Object can be a Representation, File, Bitstream, or Filestream. Note that the PREMIS definition of Digital Object differs from the definition commonly used in the digital library community, which holds a digital object to be a combination of identifier, metadata, and data.

Digital Provenance: Documentation of processes in a Digital Object's life cycle. Digital Provenance typically describes Agents responsible for the custody and stewardship of Digital Objects, key Events that occur over the course of the Digital Object's life cycle, and other information associated with the Digital Object's creation, management, and preservation.

Digital Signature Validation: Process of determining that a decrypted digital signature matches an expected value when the correct keys, algorithms, and parameters have been used. Validation confirms the originator and Fixity of the signed Digital Object.

Dissemination: Process of retrieving a Digital Object from the Preservation Repository's archival storage and making it available to users. In the context of OAIS, Dissemination involves transforming one or more Archival Information Packages (AIP) into a Dissemination Information Package (DIP) and making it available in a form suitable for the Preservation Repository's Designated Community.

Emulation: Preservation strategy for overcoming technological obsolescence of hardware and software by developing techniques for imitating obsolete systems on future generations of computers. (From DPC: www.dpconline.org/graphics/intro/definitions.html)

Encryption: Process of employing any procedure used in cryptography to convert plaintext into ciphertext (encrypted message) in order to prevent any but the intended recipient from reading that data. Schematically, there are two classes of encryption primitives: public-key cryptography and private-key cryptography; they are generally used complementarily. Public-key encryption algorithms include RSA; private-key algorithms include the obsolescent Data Encryption Standard, the Advanced Encryption Standard, as well as RC4. (From FOLDOC: foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?query=encryption)

Entity: Abstraction for a set of "things" (agents, events, etc.) described by the same properties. The PREMIS data model defines five types of Entities: Intellectual Entities, Objects, Agents, Rights, and Events.

Event: Action that involves at least one Digital Object and/or Agent known to the Preservation Repository.

File: Named and ordered sequence of Bytes that is known by an operating system. A File can be zero or more Bytes, has access permissions, and has file system statistics such as size and last modification date. A File also has a Format.

Filestream: Embedded Bitstream that can be transformed into a standalone File without adding any additional information, for example, a TIFF image embedded within a tar file, or an encoded EPS within an XML file.

7. Glossary

Fixity: Property that a Digital Object has not been changed between two points in time.

Fixity Check: Process of verifying that a File or Bitstream has not been changed during a given period. A common Fixity Check method is to compute a message digest ("hash") at one point and recalculate the message digest at a later point; if the digests are identical, the object has not been altered.

Format: Specific, preestablished structure for the organization of a File, Bitstream, or Filestream.

Format Migration: See Migration.

Forward Migration: See Migration.

Granularity: Relative size, scale, level of detail, or depth of penetration that characterizes an object or activity. "Level of granularity" is often used to refer to the level of focus in a hierarchy; for example, in a hierarchy of entity types from largest to smallest, collection, intellectual entity, representation, and file would be different levels of granularity. In the context of preservation metadata specifically and metadata generally, granularity is important in defining at what level a particular metadata element or Semantic Unit applies, for example, to a Representation, to a File, or to a Bitstream.

Ingest: Process of adding objects to a Preservation Repository's storage system. In the context of OAIS, Ingest includes services and functions that accept Submission Information Packages (SIP) from Producers, and transforms them into one or more Archival Information Packages (AIP) for long-term retention.

Inhibitor: Feature of a Digital Object intended to inhibit access, copying, Dissemination, or Migration. Common Inhibitors are Encryption and password protection.

Intellectual Entity: Coherent set of content that is described as a unit, for example, a book, a map, a photograph, a serial. An Intellectual Entity can include other Intellectual Entities; for example, a Web site can include a Web page, a Web page can include a photograph. An Intellectual Entity may have one or more Representations.

Media Migration: Form of Replication, in which a Digital Object is copied onto a different type of digital storage medium because the original medium is in danger of obsolescence.

Media Refreshment: Form of Replication, in which a Digital Object is copied onto a different unit of storage of the same or similar medium as the original. Note: Media Refreshment is used in preference to the definition of "refreshment" in the *OAIS Reference Model*. OAIS defines refreshment as a "Digital Migration where the effect is to replace a media instance with a copy that is sufficiently exact that all Archival Storage hardware and software continues to run as before."

Message Digest Calculation: Process by which a message digest ("hash") is created for a Digital Object residing in a Preservation Repository. See also Fixity Check.

Migration: Preservation strategy in which a Transformation creates a version of a Digital Object in a different Format, where the new Format is compatible with contemporary software and hardware environments. Ideally, Migration is accomplished with as little loss of content, formatting and functionality as possible, but the amount of information loss will vary depending on the Formats and content types involved. Also called "format migration" and "forward migration."

Note: Migration and Media migration are used in preference to the definition of "digital migration" in the *OAIS Reference Model*. OAIS defines digital migration as the "transfer of digital information, while intending to preserve it, within the OAIS. It is distinguished from transfers in general by three attributes: 1) a focus on the preservation of the full information content; 2) a perspective that the new archival implementation of the information is a replacement for the old; and 3) an understanding that full control and responsibility over all aspects of the transfer resides with the OAIS."

Namespace: Set of names in which all names are unique. (From FOLDOC: foldoc.doc.ic.ac.uk/foldoc/foldoc.cgi?namespace)

Normalization: Form of Migration in which a version of a Digital Object is created in a new Format with properties more conducive to preservation treatment. Normalization is often implemented as part of the Ingest process.

Object: See Digital Object.

Permission: Agreement between a rights holder and a Preservation Repository, allowing the Preservation Repository to undertake some action.

Pre-Ingest: Period in the life cycle of a Digital Object *before* it is Ingested into a Preservation Repository.

Preservation Metadata: Information a Preservation Repository uses to support the digital preservation process.

Preservation Repository: Repository that, either as its sole responsibility or as one of multiple responsibilities, undertakes the long-term preservation of the Digital Objects in its custody.

Profile: Specification for a particular implementation of a Format. For example, GeoTIFF is a profile of TIFF.

Quirk: Any loss in functionality or change in the look and feel of a Digital Object resulting from the preservation processes and procedures implemented by a Preservation Repository. (See also the definition supplied by the National Library of Australia www.nla.gov.au/preserve/pmeta.html#14)

Refreshment: See Media Refreshment.

Relationship: Statement about an association between instances of Entities.

Render: To make a Digital Object perceptible to a user, by displaying (for visual materials), playing (for audio materials), or other means appropriate to the Format of the Digital Object.

Replication: Process of copying a Digital Object so that the copy is bit-wise identical to the original. Media Migration and Media Refreshment are specific types of Replication.

Representation: Digital Object instantiating or embodying an Intellectual Entity. A Representation is the set of stored Files and structural metadata needed to provide a complete and reasonable rendition of the Intellectual Entity.

Rights: Assertions of one or more rights or permissions pertaining to a Digital Object and/or an Agent.

Root: The File that must be processed first in order to render a Representation correctly.

Semantic Component: Semantic Unit grouped with one or more other Semantic Units within a Container. A Semantic Component may itself be a Container.

Semantic Unit: Property of an Entity. Note: The PREMIS Data Dictionary makes a distinction between a Semantic Unit and a metadata element. A Semantic Unit is information that a Preservation Repository needs to know; a metadata element is how that information is actually recorded. So in practice there could be a one-to-one relationship between a Semantic Unit and its associated metadata element; a one-to-many relationship; or even a many-to-one relationship. Ultimately, the translation of a set of Semantic Units into a corresponding set of metadata elements is an implementation issue.

Simple Object: Digital Object consisting of a single File, for example, a technical report complete in one PDF file.

Store: Write a File to some non-volatile storage device such as disk, tape, or DVD.

Structural Relationship: Relationship between parts of a Digital Object.

Technical Metadata: Information describing physical (as opposed to intellectual) attributes or properties of Digital Objects. Some Technical Metadata properties are Format specific (that is, they pertain only to Digital Objects in a particular Format, for example, color space associated with a TIFF image), while others are Format independent (that is, they pertain to all Digital Objects regardless of Format, for example, size in bytes).

Transformation: Process performed on a Digital Object that results in one or more new Digital Objects that are not bit-wise identical to the source Digital Object. Examples of Transformation include Migration and Normalization.

Validation: Process of comparing a Digital Object with a standard or benchmark and noting compliance or exceptions. For example, a File can be validated against a file format specification or profile; a Representation can be validated against criteria for completeness.

Viability: Property of being readable from media.

Virus Check: Process of scanning a File for malicious programs designed to corrupt Digital Objects and systems.

Web Page: "Page" of the World Wide Web, usually in HTML/XHTML format (the file extensions are typically .htm or .html) and with hypertext links to enable navigation from one page or section to another. Web pages often use associated graphics files to provide illustration, and these too can be clickable links. (From Wikipedia: en.wikipedia.org/wiki/Web_page)

Web Site: A collection of Web Pages, that is, HTML/XHTML documents accessible via HTTP on the Internet; all publicly accessible Web Sites in existence comprise the World Wide Web. The pages of a Web Site will be accessed from a common root URL, the home page, and usually reside on the same physical server. The URLs of the pages organize them into a hierarchy, although the hyperlinks between them control how the reader perceives the overall structure and how the traffic flows between the different parts of the Web Site. (From Wikipedia: en.wikipedia.org/wiki/Web_site)

This page intentionally blank.

NOTES

[1] *A Metadata Framework to Support the Preservation of Digital Objects* (Dublin, Ohio: OCLC Online Computer Library Center, 2002), www.oclc.org/research/projects/pmwg/pm_framework.pdf.
[2] *Implementing Preservation Repositories for Digital Materials: Current Practice and Emerging Trends in the Cultural Heritage Community* (Dublin, Ohio: OCLC Online Computer Library Center, 2004), www.oclc.org/research/projects/pmwg/surveyreport.pdf.
[3] *Reference Model for an Open Archival Information System (OAIS)* (Washington, DC: Consultative Committee for Space Data Systems, 2002), ssdoo.gsfc.nasa.gov/nost/wwwclassic/documents/pdf/CCSDS-650.0-B-1.pdf.
[4] Other preservation metadata initiatives have developed other models. The National Library of New Zealand defines four types of entity: objects, files, processes, and metadata modification. *Metadata Standards Framework—Preservation Metadata (Revised)* (Wellington: National Library of New Zealand, June 2003), www.natlib.govt.nz/files/4initiatives_metaschema_revised.pdf.
[5] Note that the PREMIS definition of an Object entity differs from the definition of digital object commonly used in the digital library community, which holds a digital object to be a combination of identifier, metadata, and data. This is not intended to be a conflict. The Object entity in our model is an abstraction defined only to cluster attributes (semantic units) and clarify relationships.
[6] IFLA, *Functional Requirements for Bibliographic Records* (Munich: K.G. Saur, 1998), www.ifla.org/VII/s13/frbr/frbr.pdf.
[7] Wikipedia, the free encyclopedia, en.wikipedia.org/wiki/Main_Page.
[8] See, for example, the proposed Global Digital Format Registry at hul.harvard.edu/gdfr/.
[9] JHOVE - JSTOR/Harvard Object Validation Environment, hul.harvard.edu/jhove/.
[10] Digital Preservation Coalition Handbook, www.dpconline.org/graphics/intro/definitions.html.
[11] *XML-Signature Syntax and Processing: W3C Recommendation 12 February 2002*, www.w3.org/TR/xmldsig-core/.
[12] *Data Dictionary—Technical Metadata for Digital Still Images*, NISO Z39.87-2002/AIIM 20-2002, www.niso.org/standards/resources/Z39_87_trial_use.pdf.
[13] Margaret Byrnes, *Assigning Permanence Levels to NLM's Electronic Publications* (presented at 2000 Preservation: An International Conference on the Preservation and Long Term Accessibility of Digital Materials), www.rlg.org/en/page.php?Page_ID=244.
[14] Global Digital Format Registry (GDFR) Data Model v.3, hul.harvard.edu/gdfr/DataModel_v3.doc.
[15] Resource Description Framework (RDF), www.w3.org/RDF/.