PREMIS Data Dictionary for Preservation Metadata

version 3.0

This is an excerpt from the PREMIS version 3.0 document. It includes only the Data Dictionary section. For references to the introduction and supplementary material consult the full document, which is available online from:

http://www.loc.gov/standards/premis/v3

PREMIS Editorial Committee

June 2015 – Revised November 2015

CONTENTS

ACKNOWLEDGMENTS

PREMIS Editorial Committee members

Rebecca Guenther, Library of Congress, *Chair*
Karin Bredenberg, Riksarkivet, Swedish National Archives
Angela Dappert, University of Portsmouth
Angela Di Iorio, Sapienza Università di Roma
Leslie Johnston, U.S. National Archives and Records Administration
Devon Landes, HBO
Peter McKinney, National Library of New Zealand
Evelyn McLellan, Artefactual Systems
Tracy Meehleib, Library of Congress
Sébastien Peyrard, Bibliothèque nationale de France
Pauline Sinclair, Preservica
Eld Zierau, Royal Library of Denmark

Special thanks

The following contributed their expertise to previous versions as former members of the PREMIS Editorial Committee:

Steve Bordwell, General Register Office for Scotland
Yair Brama, ExLibris
Olaf Brandt, Koninklijke Bibliotheek, Netherlands
Priscilla Caplan, Florida Center for Library Automation (co-chair of original PREMIS Working Group)
Gerard Clifton, National Library of Australia
Markus Enders, British Library
Noreen Hill, Library and Archives Canada
Karsten Huth, Sächsisches Staatsarchiv, Saxon State Archives
David Lake, U.S. National Archives and Records Administration
Brian Lavoie, OCLC
Yaniv Levi, ExLibris
Bill Leonard, Library and Archives Canada
Rory McLeod, British Library
Robert Sharpe, Preservica
Robert Wolfe, HBO
Zhiwu Xie, Los Alamos National Laboratory
Sally Vermaaten, Statistics New Zealand
Kate Zwaard, U.S. Government Printing Office, Library of Congress

In addition to Editorial Committee members, the following contributed their expertise to the work on the Environment Working Group for PREMIS 3:

Conçalo Antunes, Instituto de Engenharia de Sistemas e Computadores
Artur Caetano, Instituto de Engenharia de Sistemas e Computadores
Carol Chou, Florida Virtual Campus
Janet Delve, University of Portsmouth
Martin Neumann, University of Karlsruhe
Michael Nolan, Intel

In addition to Editorial Committee members, the following contributed their expertise to the work on the Conformance Statement for PREMIS 3:

Jay Gattuso, National Library of New Zealand
Jan Hutař, Archives New Zealand
Amy Kirchhoff, ITHAKA
Joseph Pawletko, New York University

The following people were the original ***Pre****servation* ***M****etadata:* ***I****mplementation* ***S****trategies* (PREMIS) Working Group that developed version 1 of the Data Dictionary:

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.

PREMIS WEB SITES AND E-MAIL

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

PREMIS Implementers' Group discussion list: pig@loc.gov. To subscribe, send an e-mail to listserv@loc.gov with the message, "subscribe pig [your name]"

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

THE PREMIS DATA DICTIONARY VERSION 3.0

The PREMIS Data Dictionary includes semantic units for Objects, Events, Agents, and Rights. 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 section "Special Topics" on page **Error! Bookmark not defined.**.

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:

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. The PREMIS Data Dictionary does not specify what this authority list should be, and it is assumed that different repositories will use different vocabularies. In general, when a value is taken from a controlled vocabulary, the source of the vocabulary should be recorded. A mechanism to record the source is provided in the PREMIS XML schemas.

Object category: Whether the unit applies to an Intellectual Entity, Representation, File, or Bitstream Object. Semantic units that apply to Representations also apply to Intellectual Entities and semantic units that apply to Files also apply to Bitstreams (see page **Error! Bookmark not defined.**).

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 units 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[1], MODS[2], the Dublin Core Metadata Element Set, the Content Standard for Digital Geospatial Metadata[3], the VRA Core[4], the Encoded Archival Description (EAD)[5], and the Data Documentation Initiative[6] 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, for example, 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[7] MADS[8], 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 primarily defines characteristics of Rights and permissions concerned with preservation activities, not those associated with access and/or distribution. The semantic units allow for extensibility to use an external Rights metadata scheme.

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. An extensibility mechanism is provided by including the semantic unit *objectCharacteristicsExtension*, which may be used with an external technical metadata scheme.

Media details: The working group did not attempt to define metadata for detailed documentation of media. 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 characteristics to specialists in these areas.

---

[1] MARC 21, http://www.loc.gov/marc/.
[2] Metadata Object Description Schema (MODS), http://www.loc.gov/standards/mods/.
[3] *Content Standard for Digital Geospatial Metadata*, FGDC-STD-001-1998, http://www.fgdc.gov/metadata/csdgm/.
[4] VRA Core 4.0, http://www.vraweb.org/projects/vracore4/.
[5] Encoded Archival Description (EAD), http://www.loc.gov/ead/.
[6] Data Documentation Initiative (DDI), http://www.ddialliance.org/.
[7] vCard, http://www.imc.org/pdi/.
[8] Metadata Authority Description Schema (MADS), http://www.loc.gov/standards/mads/.

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, although this is not included in the current revision.

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 units that apply to all categories of Object (Intellectual Entity, Representation, File, and Bitstream) are *objectIdentifier and objectCategory*.

Entity types

Intellectual Entity: A set of content that is considered a single intellectual unit for purposes of management and description: for example, a particular book, map, photograph, database, or piece of hardware or software. An Intellectual Entity can include other Intellectual Entities; for example, a web site can include a web page; a web page can include an image. An Intellectual Entity may have one or more digital representations.

Representation: A digital or physical Object instantiating or embodying an Intellectual Entity. A digital representation is the set of stored digital files and structural metadata needed to provide a complete and reasonable rendition of the Intellectual Entity. A physical representation is an item such as a manuscript, video cassette, or printed document.

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.

Links between entities may be recorded from either direction and need not be bi-directional.

Entity semantic units

NB: Semantic units are applicable for Intellectual Entities, Representations, Files and Bitstreams unless otherwise indicated.

1.1 objectIdentifier (M, R)
- 1.1.1 objectIdentifierType (M, NR)
- 1.1.2 objectIdentifierValue (M, NR)

1.2 objectCategory (M, NR)

1.3 preservationLevel (O, R) [Intellectual Entity, Representation, File]
- 1.3.1 preservationLevelType (O, NR) [Intellectual Entity, Representation, File]
- 1.3.2 preservationLevelValue (M, NR) [Intellectual Entity, Representation, File]
- 1.3.3 preservationLevelRole (O, NR) [Intellectual Entity, Representation, File]
- 1.3.4 preservationLevelRationale (O, R) [Intellectual Entity, Representation, File]
- 1.3.5 preservationLevelDateAssigned (O, NR) [Intellectual Entity, Representation, File]

1.4 significantProperties (O, R)
- 1.4.1 significantPropertiesType (O, NR)
- 1.4.2 significantPropertiesValue (O, NR)
- 1.4.3 significantPropertiesExtension (O, R)

1.5 objectCharacteristics (M, R) [File, Bitstream]
- 1.5.1 compositionLevel (O, NR) [File, Bitstream]
- 1.5.2 fixity (O, R) [File, Bitstream]
  - 1.5.2.1 messageDigestAlgorithm (M, NR) [File, Bitstream]
  - 1.5.2.2 messageDigest (M, NR) [File, Bitstream]
  - 1.5.2.3 messageDigestOriginator (O, NR) [File, Bitstream]
- 1.5.3 size (O, NR) [File, Bitstream]
- 1.5.4 format (M, R) [File, Bitstream]
  - 1.5.4.1 formatDesignation (O, NR) [File, Bitstream]
    - 1.5.4.1.1 formatName (M, NR) [File, Bitstream]
    - 1.5.4.1.2 formatVersion (O, NR) [File, Bitstream]
  - 1.5.4.2 formatRegistry (O, NR) [File, Bitstream]
    - 1.5.4.2.1 formatRegistryName (M, NR) [File, Bitstream]
    - 1.5.4.2.2 formatRegistryKey (M, NR) [File, Bitstream]
    - 1.5.4.2.3 formatRegistryRole (O, NR) [File, Bitstream]
  - 1.5.4.3 formatNote (O, R) [File, Bitstream]
- 1.5.5 creatingApplication (O, R) [File, Bitstream]
  - 1.5.5.1 creatingApplicationName (O, NR) [File, Bitstream]
  - 1.5.5.2 creatingApplicationVersion (O, NR) [File, Bitstream]
  - 1.5.5.3 dateCreatedByApplication (O, NR) [File, Bitstream]
  - 1.5.5.4 creatingApplicationExtension (O, R) [File, Bitstream]
- 1.5.6 inhibitors (O, R) [File, Bitstream]

- 1.5.6.1 inhibitorType (M, NR) [File, Bitstream]
- 1.5.6.2 inhibitorTarget (O, R) [File, Bitstream]
- 1.5.6.3 inhibitorKey (O, NR) [File, Bitstream]

1.5.7 objectCharacteristicsExtension (O, R) [File, Bitstream]

1.6 originalName (O, NR) [Intellectual Entity, Representation, File]

1.7 storage (O, R) [Representation, File, Bitstream]

- 1.7.1 contentLocation (O, NR) [Representation, File, Bitstream]
  - 1.7.1.1 contentLocationType (M, NR) [Representation, File, Bitstream]
  - 1.7.1.2 contentLocationValue (M, NR) [Representation, File, Bitstream]
- 1.7.2 storageMedium (O, NR) [Representation, File, Bitstream]

1.8 signatureInformation (O, R) [File, Bitstream]

- 1.8.1 signature (O, R) [File, Bitstream]
  - 1.8.1.1 signatureEncoding (M, NR) [File, Bitstream]
  - 1.8.1.2 signer (O, NR) [File, Bitstream]
  - 1.8.1.3 signatureMethod (M, NR) [File, Bitstream]
  - 1.8.1.4 signatureValue (M, NR) [File, Bitstream]
  - 1.8.1.5 signatureValidationRules (M, NR) [File, Bitstream]
  - 1.8.1.6 signatureProperties (O, R) [File, Bitstream]
  - 1.8.1.7 keyInformation (O, NR) [File, Bitstream]
- 1.8.2 signatureInformationExtension (O, R) [File, Bitstream]

1.9 environmentFunction (O, R) [Intellectual Entity of type environment]

- 1.9.1 environmentFunctionType (M, NR) [Intellectual Entity of type environment]
- 1.9.2 environmentFunctionLevel (M, NR) [Intellectual Entity of type environment]

1.10 environmentDesignation (O, R) [Intellectual Entity of type environment]

- 1.10.1 environmentName (M, NR) [Intellectual Entity of type environment]
- 1.10.2 environmentVersion (O, NR) [Intellectual Entity of type environment]
- 1.10.3 environmentOrigin (O, NR) [Intellectual Entity of type environment]
- 1.10.4 environmentDesignationNote (O, R) [Intellectual Entity of type environment]
- 1.10.5 environmentDesignationExtension (O, R) [Intellectual Entity of type environment]

1.11 environmentRegistry (O, R) [Intellectual Entity of type environment]

- 1.11.1 environmentRegistryName (M, NR) [Intellectual Entity of type environment]
- 1.11.2 environmentRegistryKey (M, NR) [Intellectual Entity of type environment]
- 1.11.3 environmentRegistryRole (O, NR) [Intellectual Entity of type environment]

1.12 environmentExtension (O, R) [Intellectual Entity of type environment]

1.13 relationship (O, R)

- 1.13.1 relationshipType (M, NR)
- 1.13.2 relationshipSubType (M, NR)
- 1.13.3 relatedObjectIdentifier (M, R)
  - 1.13.3.1 relatedObjectIdentifierType (M, NR)
  - 1.13.3.2 relatedObjectIdentifierValue (M, NR)
  - 1.13.3.3 relatedObjectSequence (O, NR)

- 1.13.4 relatedEventIdentifier (O, R)
  - 1.13.4.1 relatedEventIdentifierType (M, NR)
  - 1.13.4.2 relatedEventIdentifierValue (M, NR)
  - 1.13.4.3 relatedEventSequence (O, NR)
- 1.13.5 relatedEnvironmentPurpose (O, R)
- 1.13.6 relatedEnvironmentCharacteristic (O, NR)

1.14 linkingEventIdentifier (O, R)

- 1.14.1 linkingEventIdentifierType (M, NR)
- 1.14.2 linkingEventIdentifierValue (M, NR)

1.15 linkingRightsStatementIdentifier (O, R)

- 1.15.1 linkingRightsStatementIdentifierType (M, NR)
- 1.15.2 linkingRightsStatementIdentifierValue (M, NR)

Semantic unit	**1.1 objectIdentifier**		
Semantic components	1.1.1 objectIdentifierType 1.1.2 objectIdentifierValue		
Definition	A designation used to identify the Object uniquely within the preservation repository system in which it is stored.		
Rationale	Each Object held in the preservation repository must have a unique identifier to allow other entities to refer to it and to relate it to descriptive, technical, and other metadata unambiguously.		
Data constraint	Container		
Object category	**Intellectual Entity / 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 generated automatically or manually.		
Usage notes	The *objectIdentifier* is mandatory for all Objects stored. The *objectIdentifier* is repeatable in order to allow both repository-assigned and externally-assigned identifiers to be recorded. See "Creation/Maintenance" note above. Primary identifiers must be unique within the repository. They may be preexisting, and in use in other digital object management systems. Ideally, secondary identifiers should also be unique but sometimes this is not possible (e.g., if the values are inherited from a legacy system which did not enforce this or only identified items at a higher level). Identifiers for each item must be sufficient to identify the item uniquely at the appropriate level of aggregation. For example, an Intellectual Entity that represents all books in the same edition could use an ISBN but this would be insufficient to identify a particular copy of that book. 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 recorded explicitly. Similarly, if the repository uses only one type of identifier, the type can be assumed and does not need to be recorded.		

	A persistent identifier should be used, but the particular identifier scheme is an implementation-specific decision.

Semantic unit	**1.1.1 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. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/identifiers.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	ISBN (Intellectual Entity) DOI (Intellectual Entity) DLC DRS hdl:4263537	DLC DRS hdl:4263537	DLC DRS hdl:4263537
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	**1.1.2 objectIdentifierValue**		
Semantic components	None		
Definition	The value of the *objectIdentifier*.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	0 00 221804-6 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	**1.2 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 (Intellectual Entities, Representations, Files, and Bitstreams) differently in terms of metadata and data management functions, it is therefore important to differentiate between the categories.		
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/objectCategory.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	intellectual entity representation	file	bitstream
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	A filestream should be considered a file.		

Semantic unit	**1.3 preservationLevel**		
Semantic components	1.3.1 preservationLevelType 1.3.2 preservationLevelValue 1.3.3 preservationLevelRole 1.3.4 preservationLevelRationale 1.3.5 preservationLevelDateAssigned		
Definition	Information indicating the decision or policy on the set of preservation functions to be applied to an object and the context in which the decision or policy was made.		
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. In such circumstances the *preservationLevelValue* that applies may need to be directly associated with an Object. The choice of a particular preservation option for an object may also require further explanation. This can depend on the preservation functions expected to be applied to the object (which can be described by assigning a *preservationLevelType*) and/or the context in which a set of preservation options is applicable (which can be described by assigning a *preservationLevelRole*). The distinction between *preservationLevelType* and *preservationLevelRole* can be illustrated by examples. One possible preservation level type is "Bit preservation level". This might have values of 'Low', 'Medium' or 'High', where, for example, in 2015 technology examples for: • 'Low' means ordinary on-site backup • 'Medium' means two copies on different media types with a minimum of 150 km distance between the stored copies, with separate checksums that are integrity checked annually • 'High' means solutions a minimum of 5 independent copies on a variety of storage media distributed over different organizations in several continents with quarterly integrity checks. The *preservationLevelRole* can then be used to distinguish, for example, if this level is an aim or has actually been achieved. It is also possible to assign the rationale for choosing the value (which can be described by adding a *preservationLevelRationale*).		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Not applicable

Repeatability	Repeatable	Repeatable	
Obligation	Optional	Optional	
Creation / Maintenance notes	The preservation level may be assigned by the repository or requested by the depositor and submitted as metadata. The repository may also choose to record additional metadata indicating the context for the assignment of the preservation level.		
Usage notes	If the repository offers only a single preservation level or the preservation level can be calculated externally (e.g., based on the information in a technical registry or by the type of collection), this value does not need to be explicitly recorded with Objects within the repository. Application of a particular set of *preservationLevel* semantic units may only cover a single Representation of an object: Representations in other technical forms or serving other functions may have a different *preservationLevel* applied. The container may be repeated if a preservation level value needs to be recorded in additional contexts (see *preservationLevelRole*, page 24) or part of a context (see *preservationLevelType*, page 22).		

Semantic unit	**1.3.1 preservationLevelType**		
Semantic components	None		
Definition	A value indicating the type of preservation functions expected to be applied to the Object for this preservation level.		
Rationale	Digital preservation functionality can be typed to express various aspects of the preservation. For instance bit-safety can represent a preservation level type, where values of this type can be used to express degree of replication and independence between copies. Likewise logical preservation can represent a type, where values of this type express whether functional preservation is done via format migration, emulation etc.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Not applicable
Examples	Bit preservation Logical/functional preservation	Bit preservation Logical/functional preservation	
Repeatability	Not repeatable	Not repeatable	
Obligation	Optional	Optional	
Creation / Maintenance notes	The *preservationLevelType* may be assigned by the repository or requested by the depositor and submitted as metadata.		
Usage notes	Only one *preservationLevelType* may be recorded per *preservationLevel* container. If a further *preservationLevelType* applies to the Object in a different context, a separate *preservationLevel* container should be repeated.		

Semantic unit	**1.3.2 preservationLevelValue**		
Semantic components	None		
Definition	A value indicating the set of preservation functions expected to be applied to the object.		
Rationale	Allows a value to be assigned to the *preservationLevel*.		
Data constraint	Value should be taken from a controlled vocabulary. If *preservationLevelType* and/or *preservationLevelRole* are used, then the available controlled vocabulary should be dependent on the values set for each of these types.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Not applicable
Examples	For *preservationLevel Type* "Logical Preservation": • Migration • Emulation	For *preservationLevel Type* "Bit Preservation": • Low (e.g. backup) • Medium (e.g. min. 2 copies and integrity check) • High (e.g. min 5 copies, integrity check and high independence)	
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	Only one *preservationLevelValu*e may be recorded per *preservationLevel* container. If a further *preservationLevelValue* applies to the Object in a different context, a separate *preservationLevel* container should be repeated.		

Semantic unit	**1.3.3 preservationLevelRole**		
Semantic components	None		
Definition	A value indicating the context in which a set of preservation options is applicable.		
Rationale	Repositories may assign *preservationLevelValues* in different contexts which must be differentiated, and may need to record more than one context. This allows a distinction, for example, between the intended preservation level and the current achievable preservation level. Note: This is distinct from the *preservationLevelType* which distinguishes the purpose of the *preservationLevel*.		
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary list is available at: http://id.loc.gov/vocabulary/preservation/preservationLevelRole.html .		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Not applicable
Examples	requirement intention capability	requirement intention capability	
Repeatability	Not repeatable	Not repeatable	
Obligation	Optional	Optional	
Usage notes	This optional semantic unit qualifies the sense or context in which the *preservationLevelValue* in the current *preservationLevel* container is applied. For example, a repository may have a legislated obligation to “fully preserve” object X (which is of format F) but is presently only capable of preserving objects of format F at a “bit-level”. The repository may need to record both the required or intended level of preservation (e.g. *preservationLevelRole*=“requirement”) and the current capability (e.g. *preservationLevelRole*=“capability”). In transferring custody of material from one repository to another, it may also be important for the receiving repository to know the sense in which *preservationLevelValue* should be understood. A receiving repository may not need to know a “capability” preservation level of which the transferring repository was capable (as this will have little bearing on its own capabilities), but it needs to know any		

	preservation level “requirements” for material for which it is now taking responsibility. It is good practice to specify *preservationLevelRole* for clarity even if the repository only assigns *preservationLevelValue* in one sense or context. If more than one *preservationLevel* is recorded with the same *preservationLevelType*, *preservationLevelRole* should always be supplied. If more than one sense or context needs to be expressed for the same object, (e.g. both the “requirement” and “capability” are recorded), separate *preservationLevel* containers should be used.

Semantic unit	**1.3.4 preservationLevelRationale**		
Semantic components	None		
Definition	The reason a particular *preservationLevelValue* was applied to the object.		
Rationale	Application of a particular *preservationLevelValue* may require justification, especially if it differs from that usually applied according to repository policy.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Not applicable
Examples	user pays legislation	defective file bit-level preservation only available for this format	
Repeatability	Repeatable	Repeatable	
Obligation	Optional	Optional	
Usage notes	This optional semantic unit records the reason for applying the *preservationLevelValue*. This information can be particularly important when the assigned *preservationLevelValue* differs from usual repository policy. For example, a repository may normally assign a *preservationLevelValue* of “full preservation” for JPEG2000 files, but detects that a particular file is defective. This may mean that the repository’s preservation strategy for JPEG2000 may not be effective for this particular file, so the repository may assign a *preservationLevelValue* of “bit-level preservation” to this file, recording “defective file” as the rationale. Similarly, legislative requirements or contractual agreements may require a higher level of preservation to be assigned to a particular object than would be assigned to that class of object according to usual policy. In this case, the rationale for the assignment may be recorded as “legislation” or “user pays”, for example. *preservationLevelRationale* may be repeated if more than one reason needs to be recorded.		

Semantic unit	**1.3.5 preservationLevelDateAssigned**		
Semantic components	None		
Definition	The date, or date and time, when a particular *preservationLevelValue* was assigned to the object.		
Rationale	The *preservationLevel* applicable to an object is expected to be reviewed and changed over time, in response to changes in repository preservation requirements, policies, or capabilities relevant to the object. The date that the current *preservationLevelValue* was assigned aids review of decisions.		
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Not applicable
Examples	2007-11-05 2007-11-05T08:15:30-05:00 20080315	2007-11-05 2007-11-05T08:15:30-05:00 20080315	
Repeatability	Not repeatable	Not repeatable	
Obligation	Optional	Optional	

Semantic unit	**1.4 significantProperties**		
Semantic components	1.4.1 significantPropertiesType 1.4.2 significantPropertiesValue 1.4.3 significantPropertiesExtension		
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	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
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	At least one of the *significantPropertiesValue* and *significantPropertiesExtension* subunits must be present if this container is included, or both may be used. The choice of whether a property is significant is subjective. Some of these properties can be directly measured while others can only be determined subjectively. 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 that the repository plans to preserve these properties across time and requires them to survive preservation action acceptably; for example, to be maintained during emulation or after format migration. It also implies that the repository would note when preservation action results in modification of significant properties. In practice, significant properties might be used as measures of		

	preservation success, as part of quality checking the results of a preservation action or evaluating the efficacy of a preservation method. For example, if the listed significant properties are not maintained after application of a particular preservation method, it may indicate a failure of the process or that the method is not well suited to the type of material. More experience with digital preservation is needed to determine the best ways of representing significant properties in general, and of representing modification of significant properties. The semantic units included in the *significantProperties* container aim to provide a flexible structure for describing significant properties, allowing general types of aspects, facets or attributes of an object to be declared and to be paired with specific significant details about the object pertaining to that aspect, facet or attribute. For example, some repositories may define significant properties for objects related to facets of content, appearance, structure, behavior, and context. Examples of facet:detail pairs in this case are as follows (note that each facet:detail pair should be contained in a separate, repeated *significantProperties* container): *significantPropertiesType* = "content" *significantPropertiesValue* = "all textual content and images" *significantPropertiesType* = "behavior" *significantPropertiesValue* = "editable" Other repositories may choose to describe significant properties at a more granular attribute level; for example: *significantPropertiesType* = "page count" *significantPropertiesValue* = "7" *significantPropertiesType* = "page width" *significantPropertiesValue* = "210 mm" Further work on determining and describing significant properties may yield more detailed schemes to facilitate general description. Representing modification of significant properties as a result of preservation action also requires further work. One possible way involves the use of Object and Event information: Object A has the 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	**1.4.1 significantPropertiesType**		
Semantic components	None		
Definition	The aspect, facet, or attribute of an object which is being recorded as a significant property.		
Rationale	Repositories may choose to describe significant properties based on a particular aspect or attribute of an object.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	content structure behavior page count page width typeface hyperlinks image count	content structure behavior page count page width typeface	[for an embedded image] color space
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Optional	Optional	Optional
Usage notes	This semantic unit is optional and may be used as part of a facet:detail pair with *significantPropertiesValue*.		

Semantic unit	**1.4.2 significantPropertiesValue**		
Semantic components	None		
Definition	Description of a characteristic of a particular object which has been determined subjectively to be important to maintain through preservation actions.		
Rationale	Repositories may choose to describe significant properties based on a particular aspect or attribute of an object.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	[for a Web page containing animation that is not considered essential] Content only. [For detail associated with a *significantProperties Type* of "behavior"] hyperlinks traversable	[for a word processed document with embedded links that are not considered essential] Content only. [For detail associated with a *significantProperties Type* of "behavior"] editable [For detail associated with a *significantProperties Type* of "page width"] 210 mm	[for a PDF with an embedded graph, where the lines' color determines the lines' meaning] Color. [For detail associated with a *significantProperties Type* of "appearance"] Color.
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	At least one of the *significantProperties Value* and *significantProperties Extension* subunits must be present if this container is included, or both may be used	At least one of the *significantProperties Value* and *significantProperties Extension* subunits must be present if this container is included, or both may be used	At least one of the *significantProperties Value* and *significantProperties Extension* subunits must be present if this container is included, or both may be used
Usage notes	If facet:detail pairs are used, the content of		

	significantPropertiesValue should describe the significant properties of the object relevant to the aspect, facet, or attribute declared in the *significantPropertiesType* with which it is paired. If facet:detail pairs are not used, *significantPropertiesValue* may be used to freely to describe any characteristic of an object. *significantPropertiesValue* is not repeatable. Multiple significant properties should be described in separate, repeated *significantProperties* container units.

Semantic unit	**1.4.3 significantPropertiesExtension**		
Semantic components	Defined externally		
Definition	A container to include semantic units defined outside of PREMIS for significant properties.		
Rationale	There may be a need to replace or extend PREMIS defined units.		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	At least one of the *significantProperties Value* and *significantProperties Extension* subunits must be present if the *significantProperties* container is included, or both may be used	At least one of the *significantProperties Value* and *significantProperties Extension* subunits must be present if the *significantProperties* container is included, or both may be used	At least one of the *significantProperties Value* and *significantProperties Extension* subunits must be present if the *significantProperties* container is included, or both may be used
Usage notes	For more granularity or use of externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata scheme may be included in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility," page 27. All of this semantic unit's subunits are optional. At least one of the *significantPropertiesValue* and *significantPropertiesExtension* subunits must be present if the *significantProperties* container is included. If the *significantPropertiesExtension* container needs to be associated explicitly with any PREMIS subunit under *significantProperties*, the container *significantProperties* is repeated. Also, if extensions from different external schemas are needed, *significantProperties* should be repeated. It is recommended to give information about the metadata used in *significantPropertiesExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.		

Semantic unit	**1.5 objectCharacteristics**		
Semantic components	1.5.1 compositionLevel 1.5.2 fixity 1.5.3 size 1.5.4 format 1.5.5 creatingApplication 1.5.6 inhibitors 1.5.7 objectCharacteristicsExtension		
Definition	Technical properties of a file or bitstream that are applicable to all or most formats.		
Rationale	There are some important technical properties that apply to objects of any format. Detailed definition of format-specific properties is outside the scope of this Data Dictionary, although such properties may be included within objectCharacteristicsExtension.		
Data constraint	Container		
Object category	**Intellectual Entity / 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.		

Semantic unit	**1.5.1 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 (or “unknown”)		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		0 1 2 Unknown	0 1 2 Unknown
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
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 decrypt file C to create file B and then decompress 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. The *compositionLevel* should be set whenever possible, however it is permitted to omit (or use “unknown”) if it not possible to calculate		

	this. Use 0 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 **Error! Bookmark not defined.**.

Semantic unit	**1.5.2 fixity**		
Semantic components	1.5.2.1 messageDigestAlgorithm 1.5.2.2 messageDigest 1.5.2.3 messageDigestOriginator		
Definition	Information used to verify whether an object has been altered in an undocumented or unauthorized way.		
Data constraint	Container		
Object category	**Intellectual Entity / 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. (Note that the terms “message digest” and “checksum” are commonly used interchangeably. However, the term “checksum” is more correctly used for the product of a cyclical redundancy check (CRC), whereas the term “message digest” refers to the result of a cryptographic hash function, which is what is referred to here.) 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.		

	For more information, see the special topic on “Fixity, integrity, authenticity,” page **Error! Bookmark not defined.**.

Semantic unit	**1.5.2.1 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. A controlled vocabulary list is available at: http://id.loc.gov/vocabulary/preservation/cryptographicHashFunctions.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		MD5 Adler-32 HAVAL SHA-1 SHA-256 SHA-384 SHA-512 TIGER WHIRLPOOL	MD5 Adler-32 HAVAL SHA-1 SHA-256 SHA-384 SHA-512 TIGER WHIRLPOOL
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	**1.5.2.2 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	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		7c9b35da4f2ebd436f1cf88e5a39b3a257edf4a22be3c955ac49da2e2107b67a1924419563	7c9b35da4f2ebd436f1cf88e5a39b3a257edf4a22be3c955ac49da2e2107b67a1924419563
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	**1.5.2.3 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 what Agent calculated the initial value of the message digest.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		DRS A0000978	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., “A0000978” taken here to be an *agentIdentifierValue*).		

Semantic unit	**1.5.3 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 has 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	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		2038937	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	**1.5.4 format**		
Semantic components	1.5.4.1 formatDesignation 1.5.4.2 formatRegistry 1.5.4.3 formatNote		
Definition	Identification of the format of a file or bitstream where format is defined as 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 to a format registry, should be sufficient to associate the object with more detailed format information.		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	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 filename extension, recommended practice is to identify the format independently by parsing the file when possible. If the format cannot be identified at the time of ingest, it is valid to record that it 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 from 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 object. 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. At least one subunit (i.e. either *formatDesignation* or *formatRegistry*) must be present if this container is included, or both may be used. If the subunit (*formatDesignation* or *formatRegistry*) needs to be repeated, the entire *format* container is repeated. This allows for		

	association of format designation with a particular set of format registry information. For example, if the precise format cannot be determined and two *format* designations are recorded, each is given within a separate *format* container. In such cases the *formatNote* element can be used to distinguish between the cases where either (i) it is known that the file complies with multiple format definitions or (ii) it is known that the file complies with one of these formats but there is insufficient knowledge to distinguish between them. The *format* container may also be repeated for multiple format registry entries. If a file or bitstream is in an unknown format then a *formatDesignation* element should be added with *formatName* set to unknown. See "Format information," page **Error! Bookmark not defined.**.

Semantic unit	**1.5.4.1 formatDesignation**		
Semantic components	1.5.4.1.1 formatName 1.5.4.1.2 formatVersion		
Definition	An identification of the format of the object.		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		At least one subunit must be present if this container is included.	At least one subunit must be present if this container is included.
Usage notes	Either *formatDesignation* or at least one instance of *formatRegistry* is required. Both may be included. The most specific format (or format profile) should be recorded. A repository (or format registry) may wish to use multipart format names (e.g., “TIFF_GeoTIFF” or “WAVE_MPEG_BWF”) to achieve this specificity. For any given file or bitstream, the most specific format identified by the repository should be recorded. A restricted or modified version of a format is considered more specific than the format; for example, GeoTIFF is more specific than TIFF; BWF is more specific than WAVE. If a file or bitstream conforms to more than one format of equal specificity, each should be recorded in separate *format* containers.		

Semantic unit	**1.5.4.1.1 formatName**		
Semantic components	None		
Definition	A commonly accepted name for the format of the file or bitstream.		
Data constraint	Value should be taken from a controlled vocabulary.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		Text/sgml image/tiff/geotiff Adobe PDF DES PGP base64 unknown	LaTex
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	For unidentified formats, *formatName* may be recorded as "unknown". Whenever possible, controlled vocabularies for *formatName* should come from format or technical registries.		

Semantic unit	**1.5.4.1.2 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	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		6.0 2003	7.1
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Usage notes	If the format is versioned, *formatVersion* should be recorded where applicable. It can be either a numeric or chronological designation.		

Semantic unit	**1.5.4.2 formatRegistry**		
Semantic components	1.5.4.2.1 formatRegistryName 1.5.4.2.2 formatRegistryKey 1.5.4.2.3 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	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		At least one subunit must be present if this container is included.	At least one subunit must be present if this container is included.
Usage notes	Either *formatDesignation* or at least one instance of *formatRegistry* is required. If more than one *formatRegistry* needs to be recorded the format container should be repeated to include each additional set of *formatRegistry* information.		

Semantic unit	**1.5.4.2.1 formatRegistryName**		
Semantic components	None		
Definition	A designation identifying the referenced format registry.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		PRONOM http://www.nationalarchives.gov.uk/pronom Representation Information Registry Repository	PRONOM http://www.nationalarchives.gov.uk/pronom
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	This can be a formal name, internally used name, or URI.		

Semantic unit	**1.5.4.2.2 formatRegistryKey**		
Semantic components	None		
Definition	The unique key used to reference an entry for this format in the specified format registry.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		fmt/155	fmt/155
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	**1.5.4.2.3 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 / information about software support and dependencies. 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. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/formatRegistryRole.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		Specification Validation profile	Specification Validation profile
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional

Semantic unit	**1.5.4.3 formatNote**		
Semantic components	None		
Definition	Additional information about format.		
Rationale	Qualifying information may be needed to supplement format designation and registry information or to record a status for identification.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		tentative identification disjunction multiple format identifications found extension mismatch	tentative identification disjunction multiple format identifications found
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	The *formatNote* may contain free text, a reference pointer, or a value from a controlled list.		

Semantic unit	**1.5.5 creatingApplication**		
Semantic components	1.5.5.1 creatingApplicationName 1.5.5.2 creatingApplicationVersion 1.5.5.3 dateCreatedByApplication 1.5.5.4 creatingApplicationExtension		
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. It is also useful to determine what rendering software is available for the digital object. For example, if you know that the Distiller program created the PDF file, you know it will be renderable with (among other programs) Adobe Reader.		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		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 both to objects created outside of the repository and subsequently ingested, and to objects created by the repository, for example, through migration Events. The *creatingApplicatio*n 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." The container 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 ingest. *CreatingApplication* is a frequently used semantic unit; however, with the introduction of more detailed environment capability, best practice should be to use an environment Object rather than this semantic unit. The use of environment Objects allows repositories to build complex relationships between objects and their creating or rendering environments. It also allows for the recording of relationships between environments themselves, for example, to expose dependencies between multiple components of complex software stacks. For more information on using environment Objects, see section “Special Topics” on page 251. However, unlike environment Objects, *creatingApplication* enables the description of different creating applications at different composition levels of the Object. It is also supported for backward compatibility. Rather than having each repository record this locally, it is preferable to share this information using environment registries.

Semantic unit	**1.5.5.1 creatingApplicationName**		
Semantic components	None		
Definition	A designation for the name of the software program that created the object.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		MS Word	MS Excel
Repeatability		Not repeatable	Not repeatable
Obligation		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 MS Word, not the Ingest program.		

Semantic unit	**1.5.5.2 creatingApplicationVersion**		
Semantic components	None		
Definition	The version of the software program that created the object.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		2000	1.4
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional

Semantic unit	**1.5.5.3 dateCreatedByApplication**		
Semantic components	None		
Definition	The actual or approximate date and time the object was created.		
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		2000-12-01 20030223T151047	2000-12-01 20030223T151047
Repeatability		Not repeatable	Not repeatable
Obligation		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*. The creation date, in PREMIS terms, of an Object is the time that it was last modified; that is, the last time the document was saved. This is discussed in the *1:1 Principle* section. If the application is a Web harvester capturing an object at a point in time, use the date captured as the creation date.		

Semantic unit	**1.5.5.4 creatingApplicationExtension**		
Semantic components	Defined externally		
Definition	Creating application information using semantic units defined externally to PREMIS.		
Rationale	There may be a need to supplement or replace PREMIS defined units.		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	For more granularity or to use externally defined semantic units, extensibility is provided. This could include a reference to an external technical registry describing the creating application. Either local semantic units or metadata using another specified metadata scheme may be included instead of or in addition to PREMIS-defined semantic units. When using an externally defined schema, a reference to that schema must be provided. See further guidance on "Extensibility," page 27. If multiple extensions are needed (e.g., for different purposes), the container *creatingApplication* is repeated. Also, if extensions from different external schemas are needed, *creatingApplication* should be repeated. It is recommended to give information about the metadata used in *creatingApplicationExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.		

Semantic unit	**1.5.6 inhibitors**		
Semantic components	1.5.6.1 inhibitorType 1.5.6.2 inhibitorTarget 1.5.6.3 inhibitorKey		
Definition	Features of the object that 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	**Intellectual Entity / 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 hard to detect during ingest if a file is encrypted, but this may be known from the context in which it is ingested. 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 in local applications. If the inhibitor key information is identical to key information in digital signatures, use those semantic units.		

Semantic unit	**1.5.6.1 inhibitorType**		
Semantic components	None		
Definition	The inhibitor method employed.		
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/inhibitorType.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		DES PGP Blowfish Password protection	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	**1.5.6.2 inhibitorTarget**		
Semantic components	None		
Definition	The content or function protected by the inhibitor.		
Data constraint	Values should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/inhibitorTarget.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		All content Function: Play Function: Print	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	**1.5.6.3 inhibitorKey**		
Semantic components	None		
Definition	The decryption key or password.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		[DES decryption key]	[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 plain text in an unsecure database.		

Semantic unit	**1.5.7 objectCharacteristicsExtension**		
Semantic components	Defined externally		
Definition	A container to include semantic units defined outside of PREMIS.		
Rationale	There may be a need to replace or extend PREMIS defined units.		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	For more granularity or use of externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata scheme may be included in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility," page 27. *objectCharacteristicsExtension* is used for additional object characteristics not covered by PREMIS, for instance format-specific metadata that is defined externally. It is not a replacement for units specified in PREMIS. If multiple extensions are needed (e.g., for different purposes), the container *objectCharacteristics* needs to be repeated. Also, if extensions from different external schemas are needed, *objectCharacteristics* should be repeated. It is recommended to give information about the metadata used in *objectCharacteristicsExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.		

Semantic unit	**1.6 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	This unit allows preservation system to rename files (for internal management and storage purposes) but still retain the name of the object that was submitted. 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	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Not applicable
Examples	“Animal Antics”	N419.pdf	
Repeatability	Not repeatable	Not repeatable	
Obligation	Optional	Optional	
Creation / Maintenance notes	This value would always be supplied to the repository by the submitter or harvesting application. How much of the file path to preserve would be up to the repository.		
Usage notes	This is designed to be used to hold the name supplied by the immediate source before ingest by the repository. This may or may not be the initial name that the object had (e.g., the source could have renamed a file when it ingested it from a previous source and not retained its initial name). In addition it may or may not be the name that the object was called in the immediate source before the repository (e.g., the source system might hold it with one name but rename it upon export to, say, its truly original name). The original name of the object must be designated in the Submission Information Package (SIP) (since the repository cannot in general work it out by other means) although it may or may not be the primary identifier of the object in the SIP. The object may have other names in different contexts. When two repositories are exchanging content, it would be important for the receiving repository to know and record the name of the Intellectual Entity or Representation at the originating repository. In the case of IEs or representations, this may be a directory name.		

Semantic unit	**1.7 storage**			
Semantic components	1.7.1 contentLocation 1.7.2 storageMedium			
Definition	Information about how and where an entity can be found. For bitstreams this means the location within a file. For files this means the physical location in one or more storage systems. Starting with PREMIS version 3.0, *storage* is applicable to representations. For physical representations such as system disks or printed versions of digital files, this means the location of the physical object, such as a shelf location. For digital representations this means the location of the digital object in a storage system, if all files in the representation can be found in this same location.			
Rationale	It is necessary for a repository to know where to locate objects and to associate the *contentLocation* of digital objects with the *storageMedium*.			
Data constraint	Container			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Not Applicable	Applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable	Repeatable
Obligation		Optional	Optional	Optional
Usage notes	For digital representations and files, the *storage* container 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. To use this repetition, the copies must have a single *objectIdentifier* and be managed as a single object by the repository. Note that many storage systems (e.g., hierarchical storage management systems, cloud storage providers etc.) will provide only a single reference to an object even though it may store multiple copies. For bitstreams, the *storage* container could be repeated if there is more than one way of getting to the start of the bitstream within a file (e.g., byte offset from start or byte offset from end). For representations of physical Objects, the storage for both physical copies of content (for example, a paper copy of a digital object) and for physical environment instances (for example, a custom computer chip) can be specified. See the "Environment" section in "Special Topics" on page 251 below for more information on physical Environments.			

Semantic unit	**1.7.1 contentLocation**			
Semantic components	1.7.1.1 contentLocationType 1.7.1.2 contentLocationValue			
Definition	Information needed to retrieve a physical item from its physical storage location or a file from the storage system, or to access a bitstream within a file.			
Data constraint	Container			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Not Applicable	Applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable	Not repeatable
Obligation		Optional	Optional	Optional
Creation / Maintenance notes	A preservation repository will normally refer to content that it controls. Therefore, it is assumed that the repository will normally assign the *contentLocation*, probably by a 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	**1.7.1.1 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. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/contentLocationType.html.			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Not Applicable	Applicable	Applicable	Applicable
Examples		Physical storage location Shelfmark RFID number	URI handle NTFS EXT3	Byte offset
Repeatability		Not repeatable	Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory	Mandatory

Semantic unit	**1.7.1.2 contentLocationValue**			
Semantic components	None			
Definition	The reference to the location of the content used by the storage system.			
Data constraint	None			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Not Applicable	Applicable	Applicable	Applicable
Examples		Shelf 4, box 3 DA592.B76 [Shelfmark] 3054257BF4C21B4000001ABF [RFID number] http://wwasearch.loc.gov/107th/200212107035/http://house.gov/langevin/ hdl:loc.pnp/cph.3b34188 c:\apache2\htdocs\index.html /home/web/public_html/index.html	http://wwasearch.loc.gov/107th/20021210703 5/http://house.gov/langevin/ hdl:loc.pnp/cph.3b34188 c:\apache2\htdocs\index.html /home/web/public_html/index.html	64 [offset from start of file c:\apache2\htdocs\image\logo.gif]
Repeatability		Not repeatable	Not repeatable	Not repeatable
Obligation		Mandatory	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 native 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 within a file. It is up to the repository to determine the level of granularity that should be recorded.			

Semantic unit	**1.7.2 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 if it is to know how and when to do media refreshment and media migration.			
Data constraint	Value should be taken from a controlled vocabulary. Example controlled vocabularies are available at: http://id.loc.gov/vocabulary/preservation/storageMedium.html. and http://metadataregistry.org/concept/list/vocabulary_id/145.html.			
Object category	**Intellectual Entity**	**Representation (for digital Representations only*)**	**File**	**Bitstream**
Applicability	Not Applicable	Applicable	Applicable	Applicable
Examples		Magnetic tape Hard disk TSM	Magnetic tape Hard disk TSM	Magnetic tape Hard disk TSM
Repeatability		Not repeatable	Not repeatable	Not repeatable
Obligation		Optional	Optional	Optional
Usage notes	In some cases this can be masked from direct repository management by storage management systems. In such cases responsibility to manage for technological obsolescence of storage media must be delegated to the storage system. Digital preservation practitioners must maintain responsibility to manage storage media and system obsolescence within the combination of repository and storage systems. Usually this is done at system level well above the details of individual objects. *Storage media are only recorded for digital items. The storage locations of physical items will be managed through traditional estate management, and media management is not part of the digital repository.			

Semantic unit	**1.8 signatureInformation**		
Semantic components	1.8.1 signature 1.8.2 signatureInformationExtension		
Definition	A container for PREMIS-defined and externally-defined digital signature information, used 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	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	Either *signature* or *signatureInformationExtension* may be used. Use of *signatureInformationExtension* with the schema defined in W3C's *XML-Signature Syntax and Processing* (http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/) is encouraged when applicable. See the discussion of digital signatures on page **Error! Bookmark not defined.** for more information on use of both PREMIS-defined and externally-defined semantic units.		

Semantic unit	**1.8.1 signature**		
Semantic components	1.8.1.1 signatureEncoding 1.8.1.2 signer 1.8.1.3 signatureMethod 1.8.1.4 signatureValue 1.8.1.5 signatureValidationRules 1.8.1.6 signatureProperties 1.8.1.7 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	**Intellectual Entity / 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 http://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 **Error! Bookmark not defined.**.)		

Semantic unit	**1.8.1.1 signatureEncoding**		
Semantic components	None		
Definition	The encoding used for the values of *signatureValue*, *keyInformation*.		
Rationale	The values of *signatureValue* and *keyInformation* cannot be interpreted correctly if the encoding is unknown.		
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/signatureEncoding.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		Base64 Ds:CrytoBinary	Base64 Ds:CrytoBinary
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory

Semantic unit	**1.8.1.2 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	**Intellectual Entity / 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	**1.8.1.3 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. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/signatureMethod.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Examples		DSA-SHA1 RSA-SHA1	DSA-SHA1 RSA-SHA1
Repeatability		Not repeatable	Not repeatable
Obligation		Mandatory	Mandatory
Usage notes	Recommended practice is to use the following naming convention: name the encryption algorithm first, followed by a hyphen, followed by the name of the hash (message digest) algorithm.		

Semantic unit	**1.8.1.4 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	**Intellectual Entity / 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=	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	**1.8.1.5 signatureValidationRules**		
Semantic components	None		
Definition	The operations to be performed in order to validate the digital signature.		
Rationale	The repository should not assume that the procedure for validating any particular signature will be known many years in the future without documentation.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	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	**1.8.1.6 signatureProperties**		
Semantic components	None		
Definition	Additional information about the generation of the signature.		
Data constraint	None		
Object category	**Intellectual Entity / 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	**1.8.1.7 keyInformation**		
Semantic components	Extensible container		
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	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Not repeatable	Not repeatable
Obligation		Optional	Optional
Usage notes	Different types of keys will have different structures and parameters. PREMIS does not define the structure of this container. Recommended practice is to represent key values as defined for "KeyInfo" in the W3C's *XML-Signature Syntax and Processing* (http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/).		

Semantic unit	**1.8.2 signatureInformationExtension**		
Semantic components	Defined externally		
Definition	Digital signature information using semantic units defined outside of PREMIS.		
Rationale	There may be a need to replace or extend PREMIS defined units.		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Not applicable	Applicable	Applicable
Repeatability		Repeatable	Repeatable
Obligation		Optional	Optional
Usage notes	For more granularity or use of externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata scheme may be included instead of or in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility," page 27. If the *signatureInformationExtension* container needs to be associated explicitly with any PREMIS subunit under *signatureInformation*, the container *signatureInformation* is repeated. Also, if extensions from different external schemas are needed, *signatureInformation* should be repeated. Use of the W3C's *XML-Signature Syntax and Processing* (http://www.w3.org/TR/2002/REC-xmldsig-core-20020212/) is encouraged when applicable. It is recommended to give information about the metadata used in *signatureInformationExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.		

Semantic unit	**1.9 environmentFunction**			
Semantic components	1.9.1 environmentFunctionType 1.9.2 environmentFunctionLevel			
Definition	A hierarchical description of the function of the environment used to render or execute an object.			
Rationale	This information describes the inherent nature of the environment which, in turn, specifies its function. It will not change over time and is not dependent on the way the environment is used. Types, such as software, documentation, hardware, must be described through an extensible vocabulary. It inherits from the previous PREMIS environment *software* and *hardware* semantic unit containers that specified fine-grained *swType* and *hwType* vocabulary. It is important to capture the type and use of an environment to express: • What is the type of the environment, such as hardware, software or documentation. • A refinement of the type to isolate where in the rendering stack the Intellectual Entity is used, such as plugin, driver, application, peripheral or specification.			
Data constraint	Container			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Repeatability	Repeatable			
Obligation	Optional			
Usage notes	*environmentFunction* can be used to describe the nature of an environment (whether it is hardware, software or documentation). This can be done on multiple, increasingly specific, hierarchical levels until the appropriate level of granularity has been reached. Multiple, hierarchical levels of description should be described in separate, repeated *environmentFunction* semantic unit containers. This can also be done by recording only the most specific level of granularity. For example, an environment Intellectual Entity describing a particular version of an operating system is also of the more generic type "software;" thus two levels of description ("software", then "operating system") describe the function. If it proved necessary, a third-level description could be added.			

	There is no limit to the number of levels, although it is unlikely that a deep hierarchy will be necessary.

Semantic unit	**1.9.1 environmentFunctionType**			
Semantic components	None			
Definition	A description of the environment at a given level within the environmental stack.			
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/environmentFunctionType.			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	Level 1 types: software, hardware. Level 2 types: software application, software library, software driver, operating system, plugin, hardware architecture, hardware peripheral, chip.			
Repeatability	Not Repeatable			
Obligation	Mandatory			
Usage notes	The type values used at the higher levels are more generic and those used at the lower levels are more specific.			

Semantic unit	**1.9.2 environmentFunctionLevel**			
Semantic components	None			
Definition	Level of the environment within an environmental stack.			
Rationale	Distinguishes hierarchical levels of an environmental stack.			
Data constraint	Positive integer			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	1, 2			
Repeatability	Not Repeatable			
Obligation	Optional if there is only one level; otherwise mandatory.			

Semantic unit	**1.10 environmentDesignation**			
Semantic components	1.10.1 environmentName 1.10.2 environmentVersion 1.10.3 environmentOrigin 1.10.4 environmentDesignationNote 1.10.5 environmentDesignationExtension			
Definition	An identification of the environment used to render or execute an object.			
Rationale	Information identifying the environment using human-readable language which can be expected to be understood outside of a digital repository. This information is expected to be distinct from the *objectIdentifier* which uses a formal identification scheme to identify an object within the context of a repository and may not be readily understood outside of that context.			
Data constraint	Container			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Repeatability	Repeatable			
Obligation	Optional			

Semantic unit	**1.10.1 environmentName**			
Semantic components	None			
Definition	A commonly accepted name used to describe the environment.			
Data constraint	It is recommended to take this information from a controlled vocabulary.			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	Ubuntu			
Repeatability	Not Repeatable			
Obligation	Mandatory			
Usage notes	Will be unique when combined with the other elements of *environmentDesignation*.			

Semantic unit	**1.10.2 environmentVersion**			
Semantic components	None			
Definition	Version of the environment.			
Data constraint	None			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	Version 12.04			
Repeatability	Not Repeatable			
Obligation	Optional			
Usage notes	Environments can be specified at various levels of generalization. A single environment instance might be a specific software release or cover a range of versions. For a detailed discussion on environment versions, see section “Special Topics” on page 251.			

Semantic unit	**1.10.3 environmentOrigin**			
Semantic components	None			
Definition	The origin of the environment referenced in *environmentName*.			
Rationale	Software or hardware objects are products of an organization or an individual. This information is useful for preservation to further specify which software / hardware object is being described.			
Data constraint	Value should be taken from a controlled vocabulary.			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	Microsoft, The Document Foundation, IBM			
Repeatability	Not Repeatable			
Obligation	Optional			

Semantic unit	**1.10.4 environmentDesignationNote**			
Semantic components	None			
Definition	Any further information required to enhance the correct specification of the environment.			
Rationale	This semantic unit allows further information to be added in an unstructured way. This information could include restrictions, known issues, issues fixed in this release, information on execution time, etc.			
Data constraint	None			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	32 bit			
Repeatability	Repeatable			
Obligation	Optional			
Usage notes	The Data Dictionary does not prescribe when to use a designation note and when to include the note in the name or version. For example, it is possible to describe the “Ubuntu 12.04 32 bit” operating system in a number of ways including: • Name= “Ubuntu”, Version = “12.04”, Designation Note = “32 bit” • Name= “Ubuntu 32 bit” Version = “12.04” • Name= “Ubuntu” Version = “12.04 32-bit” • Name= “Ubuntu 12.04 32 bit” All of these are acceptable, and which is used will depend on naming conventions in use within the repository, common usage by the individual or organization producing the environment, and other factors.			

Semantic unit	**1.10.5 environmentDesignationExtension**			
Semantic components	None			
Definition	A container for describing the designation of an environment using semantic units defined outside of PREMIS.			
Rationale	There may be a need to supplement or replace PREMIS defined units.			
Data constraint	None			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Repeatability	Repeatable			
Obligation	Optional			
Usage notes	For more granularity or use of externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata scheme may be included in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility," page 27. If *environmentDesignationExtension* needs to be associated explicitly with any PREMIS subunit under *environmentDesignation*, the container *environmentDesignation* is repeated. Also, if extensions from different external schemas are needed, *environmentDesignation* should be repeated. *environmentDesignationExtension* is used for additional information about the environment designation beyond name and version in a structured way. This can be either additional information (such as release date, support dates for the software, manufacturing period for hardware etc.) or a finer-grained expression of version information (such as build number, serial number or installation guidelines). It is recommended to give information about the metadata used in *environmentDesignationExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.			

Semantic unit	**1.11 environmentRegistry**			
Semantic components	1.11.1 environmentRegistryName 1.11.2 environmentRegistryKey 1.11.3 environmentRegistryRole			
Definition	Identifies details about the registry where further information about the environment can be found.			
Rationale	If external registries are available to the preservation repository, they may provide an excellent way of referencing detailed information about the environment.			
Data constraint	Container			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Repeatability	Repeatable			
Obligation	Optional			

Semantic unit	**1.11.1 environmentRegistryName**			
Semantic components	None			
Definition	A designation identifying the referenced external registry.			
Data constraint	None			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	PRONOM			
Repeatability	Not Repeatable			
Obligation	Mandatory			

Semantic unit	**1.11.2 environmentRegistryKey**			
Semantic components	None			
Definition	The unique key used to reference an entry for this environment in an external registry.			
Data constraint	None			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	sfw/2 x-sfw/255			
Repeatability	Not Repeatable			
Obligation	Mandatory			

Semantic unit	1.11.3 environmentRegistryRole			
Semantic components	None			
Definition	The purpose or expected use of the external registry.			
Rationale	Registries could be used for different purposes or may not provide an identical match to the Intellectual Entity described in PREMIS. For example, one registry may only provide skeleton records that are sufficient to allow an environment to be formally identified (or might only describe a family of environments while the PREMIS object describes a more specific version within that family). An alternative registry may provide an exact match and provide much more detailed information needed to actually use the environment.			
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/environmentRegistryRole.			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Examples	Identification Specification			
Repeatability	Not Repeatable			
Obligation	Optional			

Semantic unit	**1.12 environmentExtension**			
Semantic components	None			
Definition	A container to include semantic units defined outside of PREMIS.			
Rationale	There may be a need to extend PREMIS defined units.			
Data constraint	None			
Object category	**Intellectual Entity**	**Representation**	**File**	**Bitstream**
Applicability	Applicable	Not applicable	Not applicable	Not applicable
Repeatability	Repeatable			
Obligation	Optional			
Usage notes	For more granularity or use of externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata scheme may be included in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility," page 27. If the extension is about the name, version or overall designation of the environment, use *environmentDesignationExtension*. *environmentExtension* is used for additional environment characteristics not covered by PREMIS, for instance environment-specific metadata that is defined externally. It is not a replacement for units specified in PREMIS. If multiple extensions are needed (e.g., for different purposes), the container *environmentExtension* is repeated. Also, if extensions from different external schemas are needed, *environmentExtension* should be repeated. It is recommended to give information about the metadata used in *environmentExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.			

Semantic unit	**1.13 relationship**		
Semantic components	1.13.1 relationshipType 1.13.2 relationshipSubType 1.13.3 relatedObjectIdentifier 1.13.4 relatedEventIdentifier 1.13.5 relatedEnvironmentPurpose 1.13.6 relatedEnvironmentCharacteristic		
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), rigorously track digital provenance (derivation relationships) and document the links between parts of a rendering stack in a Representation Information Network (dependency and documentation links). Documentation about relationships between different objects is crucial to these purposes.		
Data constraint	Container		
Object category	**Intellectual Entity / 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. In complex scenarios, PREMIS might not be able to express rich enough structural relationships to be the only source of structural metadata. Many formats for representing structural information may be used instead of the semantic units specified here. This information must be recorded, and some implementations may record it by using other structures (e.g. METS). Structural relationships between the file and representation 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. Structural relationships between Intellectual Entities may be used to record logical containment, such as between an article and an issue,		

	or physical containment, such as between a page and a book. Derivative relationships at the file, representation and Intellectual Entity level are important for documenting digital provenance.

Semantic unit	**1.13.1 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. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/relationshipType.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	derivation structural	derivation structural	derivation structural
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	It is recommended to further categorize the relationship using *relationshipSubType*.		

Semantic unit	**1.13.2 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. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/relationshipSubType.html.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	Has root	Is source of Has part	Has source Is part of
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory
Usage notes	A repository may find it necessary to define more or less granular relationships than those values in the available controlled vocabulary. For derivation relationships, note that the precise relationship may be indicated by the type of the related Event. 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	**1.13.3 relatedObjectIdentifier**		
Semantic components	1.14.3.1 relatedObjectIdentifierType 1.14.3.2 relatedObjectIdentifierValue 1.14.3.3 relatedObjectSequence		
Definition	The identifier and sequential context of the related Object.		
Data constraint	Container		
Object category	**Intellectual Entity / 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 external Object. Internal and external references should be unambiguous.		

Semantic unit	**1.13.3.1 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	**Intellectual Entity / 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 semantic unit should be set to the value of that Object's *objectIdentifierType*.		

Semantic unit	**1.13.3.2 relatedObjectIdentifierValue**		
Semantic components	None		
Definition	The value of the related Object identifier.		
Data constraint	None		
Object category	**Intellectual Entity / 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 semantic unit should set to the value of that Object's *objectIdentifierValue*.		

Semantic unit	**1.13.3.3 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 the pages of a book, it is necessary to know the order of files representing pages.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	1 2 3	1 2 3	1 2 3
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Optional	Optional	Optional
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 website. It is acceptable to either use the “dummy” sequence number zero, or to omit completely. This semantic unit is applicable only for structural relationships and is thus optional.		

Semantic unit	**1.13.4 relatedEventIdentifier**		
Semantic components	1.13.4.1 relatedEventIdentifierType 1.13.4.2 relatedEventIdentifierValue 1.13.4.3 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	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	For derivative relationships between Objects *relatedEventIdentifier* must be recorded.		

Semantic unit	**1.13.4.1 relatedEventIdentifierType**		
Semantic components	None		
Definition	The *eventIdentifierType* of the related Event.		
Data constraint	Must be an existing *eventIdentifierType* value.		
Object category	**Intellectual Entity / 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 its own internal numbering system. It can be implicit within the system and provided explicitly only if the data is exported.		

Semantic unit	**1.13.4.2 relatedEventIdentifierValue**		
Semantic components	None		
Definition	The *eventIdentifierValue* of the related Event.		
Data constraint	Must be an existing *eventIdentifierValue* value.		
Object category	**Intellectual Entity / 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	**1.13.4.3 relatedEventSequence**		
Semantic components	None		
Definition	The order of the related Event.		
Data constraint	None		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	1 2 3	1 2 3	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	**1.13.5 relatedEnvironmentPurpose**		
Semantic components	None		
Definition	The use(s) supported by the related 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 from the environment needed to render it.		
Data constraint	Values should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/environmentPurpose.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	render edit create	render edit create	render edit create
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/or software environment information, which might be the submitter, the repository, or an environment registry.		
Usage notes	Other values might indicate the ability to transform, print, and manipulate by program.		

Semantic unit	**1.13.6 relatedEnvironmentCharacteristic**		
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 between them.		
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/environmentCharacteristic.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	unspecified (Representation) minimum (Representation) known to work (Representation) recommended (Representation)	unspecified minimum known to work recommended	unspecified minimum known to work recommended
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 environment registry, *relatedEnvironmentCharacteristic* might also be obtained from the registry. Note however that the criteria for "recommended" may be different for different repositories.		
Usage notes	If an environment could be described as 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	**1.14 linkingEventIdentifier**		
Semantic components	1.14.1 linkingEventIdentifierType 1.14.2 linkingEventIdentifierValue		
Definition	The *eventIdentifier* of an Event associated with the Object.		
Data constraint	Container		
Object category	**Intellectual Entity / 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. Linking semantic units are mandatory in the sense that a repository needs to know the information, but are defined as optional because PREMIS does not specify in which direction the linkage should be.		

Semantic unit	**1.14.1 linkingEventIdentifierType**		
Semantic components	None		
Definition	The *eventIdentifierType* value of the related Event.		
Data constraint	Must be an existing *eventIdentifierType* value.		
Object category	**Intellectual Entity / 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

Semantic unit	**1.14.2 linkingEventIdentifierValue**		
Semantic components	None		
Definition	The *eventIdentifierValue* value of the related Event.		
Data constraint	Must be an existing *eventIdentifierValue* value.		
Object category	**Intellectual Entity / 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	**1.15 linkingRightsStatementIdentifier**		
Semantic components	1.15.1 linkingRightsStatementIdentifierType 1.15.2 linkingRightsStatementIdentifierValue		
Definition	An identifier for a Rights statement associated with the object.		
Rationale	A repository may choose to link from a Rights statement to an object or from an object to a Rights statement or both.		
Data constraint	Container		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Repeatability	Repeatable	Repeatable	Repeatable
Obligation	Optional	Optional	Optional
Usage notes	Linking semantic units are mandatory in the sense that a repository needs to know the information, but are defined as optional because PREMIS does not specify in which direction the linkage should be.		

Semantic unit	**1.15.1 linkingRightsStatementIdentifierType**		
Semantic components	None		
Definition	A designation of the domain within which the *linkingRightsStatementIdentifier* is unique.		
Data constraint	Must be an existing *rightsStatementIdentifierType* value.		
Object category	**Intellectual Entity / Representation**	**File**	**Bitstream**
Applicability	Applicable	Applicable	Applicable
Examples	URI LCCN	URI LCCN	URI LCCN
Repeatability	Not repeatable	Not repeatable	Not repeatable
Obligation	Mandatory	Mandatory	Mandatory

Semantic unit	**1.15.2 linkingRightsStatementIdentifierValue**		
Semantic components	None		
Definition	The value of the *linkingRightsStatementIdentifier*.		
Data constraint	Must be an existing *rightsStatementIdentifierValue* value.		
Object category	**Intellectual Entity / 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*, and *eventDateTime*.

Entity properties

Must be related to one or more Objects.

Can be related to one or more Agents.

Links between entities may be recorded from either direction and need not be bi-directional.

Entity semantic units

2.1 eventIdentifier (M, NR)
- 2.1.1 eventIdentifierType (M, NR)
- 2.1.2 eventIdentifierValue (M, NR)

2.2 eventType (M, NR)

2.3 eventDateTime (M, NR)

2.4 eventDetailInformation (O, R)
- 2.4.1 eventDetail (O, NR)
- 2.4.2 eventDetailExtension (O, R)

2.5 eventOutcomeInformation (O, R)
- 2.5.1 eventOutcome (O, NR)
- 2.5.2 eventOutcomeDetail (O, R)
  - 2.5.2.1 eventOutcomeDetailNote (O, NR)
  - 2.5.2.2 eventOutcomeDetailExtension (O, R)

2.6 linkingAgentIdentifier (O, R)
- 2.6.1 linkingAgentIdentifierType (M, NR)
- 2.6.2 linkingAgentIdentifierValue (M, NR)
- 2.6.3 linkingAgentRole (O, R)

2.7 linkingObjectIdentifier (O, R)
- 2.7.1 linkingObjectIdentifierType (M, NR)
- 2.7.2 linkingObjectIdentifierValue (M, NR)
- 2.7.3 linkingObjectRole (O, R)

Semantic unit	**2.1 eventIdentifier**
Semantic components	2.1.1 eventIdentifierType 2.1.2 eventIdentifierValue
Definition	A designation used to identify the Event uniquely within the preservation repository system.
Rationale	Each Event recorded by the preservation repository 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 these identifiers. The identifier is therefore not repeatable.

Semantic unit	**2.1.1 eventIdentifierType**
Semantic components	None
Definition	A designation of the domain within which the Event identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary.
Examples	FDA Stanford Repository Event ID UUID local
Repeatability	Not repeatable
Obligation	Mandatory
Creation / Maintenance notes	For most preservation repositories, the *eventIdentifierType* will be its own internal numbering system. It can be implicit within the system and provided explicitly only if the data is exported.

Semantic unit	2.1.2 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	**2.2 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. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/eventType.
Examples	E77 [a code used within a repository for a particular Event type] ingestion migration validation
Repeatability	Not repeatable
Obligation	Mandatory
Usage notes	Some *eventType* values may be more generic than others: migration, normalization, and replication are in some instances more precise subtypes of the creation *eventType* value. “Creation” can be used when more precise terms do not apply, for example, when a digital object was first created by scanning from paper. In general, 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 *eventDetailInformation* rather than using a very granular value for *eventType*.

Semantic unit	2.3 eventDateTime
Semantic components	None
Definition	The single date and time, or date and time range, at or during which the Event occurred.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
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	**2.4 eventDetailInformation**
Semantic components	2.4.1 eventDetail 2.4.2 eventDetailExtension
Definition	Additional information about the Event.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	All subunits of this semantic unit are optional but at least one subunit (i.e. *eventDetail* and/or *eventDetailExtension*) must be present if this container is included.

Semantic unit	**2.4.1 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 necessarily intended to be processed by machine. It may record any information about an Event and/or point to information stored elsewhere. When multiple details about the same Event need to be recorded, the *eventDetailInformation* container must be repeated. *eventDetailExtension* allows more expressivity if required.

Semantic unit	2.4.2 eventDetailExtension
Semantic components	Defined externally
Definition	A container to include semantic units defined outside of PREMIS.
Rationale	There may be a need to replace or extend PREMIS defined units.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	For more granularity or use of externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata schema may be included instead of or in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility", page 27. If more than one extension needs to be associated explicitly with *eventDetail*, *eventDetailExtension* is repeated. However, if extensions from different external schemas are needed or if the extension is not associated explicitly with *eventDetail*, the *eventDetailInformation* container should be repeated instead. It is recommended to give information about the metadata used in *eventDetailExtension*, including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.

Semantic unit	**2.5 eventOutcomeInformation**
Semantic components	2.5.1 eventOutcome 2.5.2 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. All subunits of this semantic unit are optional. At least one subunit (i.e. *eventOutcome* or *eventOutcomeDetail*) must be present if this container is included.

Semantic unit	2.5.1 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 a controlled vocabulary that a system can act upon automatically. Because this is inherently a local vocabulary, there is no general one available. 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	2.5.2 eventOutcomeDetail
Semantic components	2.5.2.1 eventOutcomeDetailNote 2.5.2.2 eventOutcomeDetailExtension
Definition	A 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	Container
Repeatability	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. All subunits of this semantic unit are optional. At least one subunit (i.e. *eventOutcomeDetailNote* and/or *eventOutcomeDetailExtension*) must be present if this container is included.

Semantic unit	2.5.2.1 eventOutcomeDetailNote
Semantic components	None
Definition	A detailed description of the result or product of the Event in textual form.
Rationale	Additional information in textual form may be needed about the outcome of the Event.
Data constraint	None
Examples	LZW compressed file Non-standard tags found in header
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	2.5.2.2 eventOutcomeDetailExtension
Semantic components	Defined externally
Definition	A container to include semantic units defined outside of PREMIS.
Rationale	There may be a need to replace or extend PREMIS defined units.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	For more granularity or use of externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata schema may be included instead of or in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility," page 27. If more than one extension needs to be associated explicitly with *eventOutcomeDetailNote*, *eventOutcomeDetailExtension* is repeated. However, if extensions from different external schemas are needed or if the extension is not associated explicitly with *eventOutcomeDetailNote*, the *eventOutcomeDetail* container should be repeated instead. It is recommended to give information about the metadata used in *eventOutcomeDetailExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.

Semantic unit	**2.6 linkingAgentIdentifier**
Semantic components	2.6.1 linkingAgentIdentifierType 2.6.2 linkingAgentIdentifierValue 2.6.3 linkingAgentRole
Definition	Identification of one or more Agents associated with the Event.
Rationale	Digital provenance often requires 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. Linking semantic units are mandatory in the sense that a repository needs to know the information, but are defined as optional because PREMIS does not specify in which direction the linkage should be.

Semantic unit	2.6.1 linkingAgentIdentifierType
Semantic components	None
Definition	A designation of the domain in which the linking Agent identifier is unique.
Data constraint	Must be an existing *agentIdentifierType* value.
Examples	[see examples for *agentIdentifierType*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**2.6.2 linkingAgentIdentifierValue**
Semantic components	None
Definition	The value of the linking Agent identifier.
Data constraint	Must be an existing *agentIdentifierValue* value.
Examples	[see examples for *agentIdentifierValue*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	2.6.3 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. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/eventRelatedAgentRole.
Examples	authorizer implementer validator executing program
Repeatability	Repeatable
Obligation	Optional

Semantic unit	2.7 linkingObjectIdentifier
Semantic components	2.7.1 linkingObjectIdentifierType 2.7.2 linkingObjectIdentifierValue 2.7.3 linkingObjectRole
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
Usage notes	Linking semantic units are mandatory in the sense that a repository needs to know the information, but are defined as optional because PREMIS does not specify in which direction the linkage should be.

Semantic unit	2.7.1 linkingObjectIdentifierType
Semantic components	None
Definition	A designation of the domain in which the linking Object identifier is unique.
Data constraint	Must be an existing *objectIdentifierType* value.
Examples	[see examples for *objectIdentifierType*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**2.7.2 linkingObjectIdentifierValue**
Semantic components	None
Definition	The value of the linking Object identifier.
Data constraint	Must be an existing *objectIdentifierValue* value.
Examples	[see examples for *objectIdentifierValue*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	2.7.3 linkingObjectRole
Semantic components	None
Definition	The role of the Object associated with an Event.
Rationale	Distinguishes the role of the Object in relation to an Event. If this is not explicit it is necessary to analyze the relationship between Objects in the Object metadata.
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/eventRelatedObjectRole.
Examples	source outcome
Repeatability	Repeatable
Obligation	Optional

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.

A piece of software or hardware that is captured as an Agent can also be preserved in a repository and described as an environment Object, for example, as source code or an ISO disk image. In this case, implementers may choose to relate the Agent to the environment Object using the *linkingEnvironmentIdentifier* semantic unit. This relationship can support the ability of a repository to record in considerable detail the interactions between software or hardware Agents and the preserved digital objects with which they interact, and even to reproduce these interactions if desired.

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 through an Event or with respect to a Rights statement.

Entity semantic units

3.1 agentIdentifier (M, R)
- 3.1.1 agentIdentifierType (M, NR)
- 3.1.2 agentIdentifierValue (M, NR)

3.2 agentName (O, R)

3.3 agentType (O, NR)

3.4 agentVersion (O, NR)

3.5 agentNote (O, R)

3.6 agentExtension (O, R)

3.7 linkingEventIdentifier (O, R)
- 3.7.1 linkingEventIdentifierType (M, NR)
- 3.7.2 linkingEventIdentifierValue (M, NR)

3.8 linkingRightsStatementIdentifier (O, R)
- 3.8.1 linkingRightsStatementIdentifierType (M, NR)
- 3.8.2 linkingRightsStatementIdentifierValue (M, NR)

3.9 linkingEnvironmentIdentifier (O, R)
- 3.9.1 linkingEnvironmentIdentifierType (M, NR)
- 3.9.2 linkingEnvironmentIdentifierValue (M, NR)
- 3.9.3 linkingEnvironmentRole (O, R)

Semantic unit	3.1 agentIdentifier
Semantic components	3.1.1 agentIdentifierType 3.1.2 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 Rights statements.
Data constraint	Container
Repeatability	Repeatable
Obligation	Mandatory
Creation / Maintenance notes	An identifier may be created by the repository system, or it may be created or assigned outside of the repository. Similarly, identifiers can be automatically or manually generated. Recommended practice is for repositories to use an identifier automatically created by the repository as the primary identifier in order to ensure that identifiers are unique and usable by the repository. Externally assigned identifiers can be used as secondary identifiers in order to link an Agent to information held outside the repository.
Usage notes	Identifiers must be unique within the repository. The *agentIdentifier* is repeatable in order to allow both repository-assigned and externally-assigned identifiers to be recorded. See Creation/Maintenance notes, above.

Semantic unit	3.1.1 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 DLC URI local
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	3.1.2 agentIdentifierValue
Semantic components	None
Definition	The value of the *agentIdentifier*.
Data constraint	Value may be taken from a controlled vocabulary. If *agentType* is organization, a controlled vocabulary is available at: http://id.loc.gov/vocabulary/organizations.
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	**3.2 agentName**
Semantic components	None
Definition	A text string which could be used in addition to *agentIdentifier* to identify an Agent.
Rationale	This semantic unit provides a reader-friendly label for the Agent identified by the *agentIdentifier*.
Data constraint	None
Examples	Erik Owens Woodyard PC JHOVE
Repeatability	Repeatable
Obligation	Optional
Usage notes	The value is not necessarily unique. If *agentType* is software, *agentVersion* can be used to refine *agentName*.

Semantic unit	3.3 agentType
Semantic components	None
Definition	A high-level characterization of the type of Agent.
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/agentType.html.
Repeatability	Not repeatable
Obligation	Optional
Examples	person organization software hardware

Semantic unit	3.4 agentVersion
Semantic components	None
Definition	The version of the Agent referenced in *agentName*, if *agentType* is software or hardware.
Rationale	Software or hardware Agents can behave very differently from one version to another.
Data constraint	None
Examples	1.6 2.1.0 20120521
Repeatability	Not repeatable
Obligation	Optional
Usage notes	This semantic unit only applies to machine Agents (hardware or software). If there is no formal version, the date of issuance may be used. The distinction between *agentName* and *agentVersion* can be fuzzy in some cases. Sometimes, a new version of a product becomes a standalone product of its own. For instance, even though JHOVE 2 declares itself as the second major version of the JHOVE file analysis software application, it is a very different product with a different architecture, features and maintaining agency; currently, JHOVE 1 and JHOVE 2 continue to evolve in parallel. For those reasons, it would be better to have JHOVE 2 in *agentName*, and record the specific build number of the software used in the repository in *agentVersion*. In any case, implementers should define their own naming policies to express name and version information and apply it consistently for all their Agent descriptions. If the Agent is also described as an environment Object, implementers may choose to link the Agent to the environment Object using the *linkingEnvironmentIdentifier* semantic unit. In this case, the information in *agentVersion* and *environmentVersion* should be consistent.

Semantic unit	3.5 agentNote
Semantic components	None
Definition	Additional information about the Agent.
Rationale	Additional information may be needed to describe or disambiguate the Agent.
Data constraint	None
Examples	--prefix=/opt/local [configuration options used with a software Agent]
Repeatability	Repeatable
Obligation	Optional
Usage notes	Use *agentNote* when relatively limited information must be supplied. If extensive additional information is required, consider using an externally defined schema with *agentExtension* instead.

Semantic unit	3.6 agentExtension
Semantic components	Defined externally.
Definition	A container to include semantic units defined outside of PREMIS.
Rationale	There may be a need to replace or extend PREMIS defined units.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	For more granularity or use of externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata scheme may be included instead of or in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility," page 27. It is recommended to give information about the metadata used in *agentExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.

Semantic unit	**3.7 linkingEventIdentifier**
Semantic components	3.7.1 linkingEventIdentifierType 3.7.2 linkingEventIdentifierValue
Definition	The *eventIdentifier* of an Event associated with the Agent.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	Linking semantic units are mandatory in the sense that a repository needs to know the information, but are defined as optional because PREMIS does not specify in which direction the linkage should be.

Semantic unit	**3.7.1 linkingEventIdentifierType**
Semantic components	None
Definition	The *eventIdentifierType* value of the related Event.
Data constraint	Must be an existing *eventIdentifierType* value.
Examples	[see examples for *eventIdentifierType*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**3.7.2 linkingEventIdentifierValue**
Semantic components	None
Definition	The *eventIdentifierValue* value of the related Event.
Data constraint	Must be an existing *eventIdentifierValue* value.
Examples	[see examples for *eventIdentifierValue*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	3.8 linkingRightsStatementIdentifier
Semantic components	3.8.1 linkingRightsStatementIdentifierType 3.8.2 linkingRightsStatementIdentifierValue
Definition	An identifier for a Rights statement associated with the Agent.
Rationale	A repository may choose to link from a Rights statement to an Agent or from an Agent to a Rights statement or both.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	Linking semantic units are mandatory in the sense that a repository needs to know the information, but are defined as optional because PREMIS does not specify in which direction the linkage should be.

Semantic unit	**3.8.1 linkingRightsStatementIdentifierType**
Semantic components	None
Definition	A designation of the domain within which the *linkingRightsStatementIdentifier* is unique.
Data constraint	Must be an existing *rightsStatementIdentifierType* value.
Examples	URI LCCN
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**3.8.2 linkingRightsStatementIdentifierValue**
Semantic components	None
Definition	The value of the *linkingRightsStatementIdentifier*.
Data constraint	Must be an existing *rightsStatementtIdentifierValue* value.
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**3.9 linkingEnvironmentIdentifier**
Semantic components	3.9.1 linkingEnvironmentIdentifierType 3.9.2 linkingEnvironmentIdentifierValue 3.9.3 linkingEnvironmentRole
Definition	The *objectIdentifier* of an environment Object associated with the Agent.
Rationale	Allows implementers to link the Agent to an environment Object, if the Agent is hardware or software that has been preserved in some way in the repository and has been described as an environment Object.
Data constraint	Container
Examples	Link from the JHOVE Agent used in a characterization Event to the JHOVE source code captured in the repository as an environment Object.
Repeatability	Repeatable
Obligation	Optional
Usage notes	Although the semantic unit is named *linking**Environment**Identifier*, it links an Agent to an **Object**. *LinkingEnvironmentIdentifierType* and *linkingEnvironmentIdentifierValue* must therefore match *objectIdentifierType* and *objectIdentifierValue* in the related environment Object.

Semantic unit	**3.9.1 linkingEnvironmentIdentifierType**
Semantic components	None
Definition	A designation of the domain within which the *linkingEnvironmentIdentifier* is unique.
Data constraint	Must be an existing *objectIdentifierType* value.
Examples	[see examples for *objectIdentifierType*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**3.9.2 linkingEnvironmentIdentifierValue**
Semantic components	None
Definition	The value of the *linkingEnvironmentIdentifier*.
Data constraint	Must be an existing *objectIdentifierValue*.
Examples	[see examples for *objectIdentifierValue*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	3.9.3 linkingEnvironmentRole
Semantic components	None
Definition	The role of the *environment* Object associated with this Agent.
Rationale	A piece of software or hardware that is captured as an Agent can also be preserved in a repository and described as an environment Object: for example, as source code or an ISO disk image. The role allows one to record in what form the software or hardware has been preserved in the repository. .
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/linkingEnvironmentRole.html.
Examples	source code bytecode disk image
Repeatability	Repeatable
Obligation	Optional
Usage notes	Where applicable, implementers may to choose to relate the Agent to a top-level Intellectual Entity, which in turn has different representations (e.g. source code and bytecode representations).

Rights Entity

For the purpose of the PREMIS Data Dictionary, statements of legal rights and permissions are taken to be constructs that can be described as the Rights entity. Rights are entitlements granted 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 Rights or permissions a repository has to carry out actions related to objects within the repository. These may generally be granted by copyright law, by statute, or by a license agreement with the rightsholder. In some situations the basis for the rights is for other reasons, for instance institutional policy.

If the repository records Rights information, either *rightsStatement* or *rightsExtension* must be present.

Entity properties

May be related to one or more Objects.

May be related to one or more Agents.

Links between entities may be recorded from either direction and need not be bi-directional.

Entity semantic units

4.1 rightsStatement (O, R)
- 4.1.1 rightsStatementIdentifier (M, NR)
  - 4.1.1.1 rightsStatementIdentifierType (M, NR)
  - 4.1.1.2 rightsStatementIdentifierValue (M, NR)
- 4.1.2 rightsBasis (M, NR)
- 4.1.3 copyrightInformation (O, NR)
  - 4.1.3.1 copyrightStatus (M, NR)
  - 4.1.3.2 copyrightJurisdiction (M, NR)
  - 4.1.3.3 copyrightStatusDeterminationDate (O, NR)
  - 4.1.3.4 copyrightNote (O, R)
  - 4.1.3.5 copyrightDocumentationIdentifier (O, R)
    - 4.1.3.5.1 copyrightDocumentationIdentifierType (M, NR)
    - 4.1.3.5.2 copyrightDocumentationIdentifierValue (M, NR)
    - 4.1.3.5.3 copyrightDocumentationRole (O, NR)
  - 4.1.3.6 copyrightApplicableDates (O, NR)
    - 4.1.3.6.1 startDate (O, NR)
    - 4.1.3.6.2 endDate (O, NR)

- 4.1.4 licenseInformation (O, NR)
  - 4.1.4.1 licenseDocumentationIdentifier (O, R)
    - 4.1.4.1.1 licenseDocumentationIdentifierType (M, NR)
    - 4.1.4.1.2 licenseDocumentationIdentifierValue (M, NR)
    - 4.1.4.1.3 licenseDocumentationRole (O, NR)
  - 4.1.4.2 licenseTerms (O, NR)
  - 4.1.4.3 licenseNote (O, R)
  - 4.1.4.4 licenseApplicableDates (O, NR)
    - 4.1.4.4.1 startDate (O, NR)
    - 4.1.4.4.2 endDate (O, NR)
- 4.1.5 statuteInformation (O, R)
  - 4.1.5.1 statuteJurisdiction (M, NR)
  - 4.1.5.2 statuteCitation (M, NR)
  - 4.1.5.3 statuteInformationDeterminationDate (O, NR)
  - 4.1.5.4 statuteNote (O, R)
  - 4.1.5.5 statuteDocumentationIdentifier (O, R)
    - 4.1.5.5.1 statuteDocumentationIdentifierType (M, NR)
    - 4.1.5.5.2 statuteDocumentationIdentifierValue (M, NR)
    - 4.1.5.5.3 statuteDocumentationRole (O, NR)
  - 4.1.5.6 statuteApplicableDates (O, NR)
    - 4.1.5.6.1 startDate (O, NR)
    - 4.1.5.6.2 endDate (O, NR)
- 4.1.6 otherRightsInformation (O, NR)
  - 4.1.6.1 otherRightsDocumentationIdentifier (O, R)
    - 4.1.6.1.1 otherRightsDocumentationIdentifierType (M, NR)
    - 4.1.6.1.2 otherRightsDocumentationIdentifierValue (M, NR)
    - 4.1.6.1.3 otherRightsDocumentationRole (O, NR)
  - 4.1.6.2 otherRightsBasis (M, NR)
  - 4.1.6.3 otherRightsApplicableDates (O, NR)
    - 4.1.6.3.1 startDate (O, NR)
    - 4.1.6.3.2 endDate (O, NR)
  - 4.1.6.4 otherRightsNote (O, R)
- 4.1.7 rightsGranted (O, R)
  - 4.1.7.1 act (M, NR)
  - 4.1.7.2 restriction (O, R)
  - 4.1.7.3 termOfGrant (O, NR)
    - 4.1.7.3.1 startDate (M, NR)
    - 4.1.7.3.2 endDate (O, NR)
  - 4.1.7.4 termOfRestriction (O, NR)
    - 4.1.7.4.1 startDate (M, NR)

4.1.7.4.2 endDate (O, NR)
4.1.7.5 rightsGrantedNote (O, R)
4.1.8 linkingObjectIdentifier (O, R)
4.1.8.1 linkingObjectIdentifierType (M, NR)
4.1.8.2 linkingObjectIdentifierValue (M, NR)
4.1.8.3 linkingObjectRole (O, R)
4.1.9 linkingAgentIdentifier (O, R)
4.1.9.1 linkingAgentIdentifierType (M, NR)
4.1.9.2 linkingAgentIdentifierValue (M, NR)
4.1.9.3 linkingAgentRole (O, R)
4.2 rightsExtension (O, R)

Semantic unit	**4.1 rightsStatement**
Semantic components	4.1.1 rightsStatementIdentifier 4.1.2 rightsBasis 4.1.3 copyrightInformation 4.1.4 licenseInformation 4.1.5 statuteInformation 4.1.6 otherRightsInformation 4.1.7 rightsGranted 4.1.8 linkingObjectIdentifier 4.1.9 linkingAgentIdentifier
Definition	Documentation of the repository's Rights or indeed restrictions to perform one or more acts.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	This semantic unit is optional because in some cases Rights may be unknown. Institutions are encouraged to record Rights information when possible. Either *rightsStatement* or *rightsExtension* must be present if the Rights entity is included. The *rightsStatement* should be repeated when the act(s) described has (have) more than one basis, or when different acts have different bases.

Semantic unit	**4.1.1 rightsStatementIdentifier**
Semantic components	4.1.1.1 rightsStatementIdentifierType 4.1.1.2 rightsStatementIdentifierValue
Definition	The designation used to identify the Rights statement uniquely within a preservation repository system.
Rationale	Each statement of Rights associated with the preservation repository must have a unique identifier to allow it to be related to Events and Agents.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Mandatory
Creation/ maintenance notes	The *rightsStatementIdentifier* is likely to be system generated. There is no global scheme or standard for these identifiers. The identifier is therefore not repeatable.
Usage notes	Identifiers must be unique within the repository.

Semantic unit	4.1.1.1 rightsStatementIdentifierType
Semantic components	None
Definition	A designation of the domain within which the Rights statement identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/identifiers.html.
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.1.2 rightsStatementIdentifierValue
Semantic components	None
Definition	The value of the *rightsStatementIdentifier*.
Data constraint	None
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**4.1.2 rightsBasis**
Semantic components	None
Definition	Designation of the basis for the right or permission identified by the *rightsStatementIdentifier*.
Data constraint	Values should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/rightsBasis.html.
Repeatability	Not repeatable
Obligation	Mandatory
Examples	copyright license statute other
Usage Notes	When *rightsBasis* is “copyright”, *copyrightInformation* should be provided. When *rightsBasis* is “license”, *licenseInformation* should be provided. When *rightsBasis* is “statute”, *statuteInformation* should be provided. When *rightsBasis* is “other”, *otherRightsBasis* (in *otherRightsInformation* container) should be provided. If the Rights for the item are public domain, use “copyright”. If they are Fair Use, use “statute”. If more than one basis applies, the entire Rights entity should be repeated.

Semantic unit	4.1.3 copyrightInformation
Semantic components	4.1.3.1 copyrightStatus 4.1.3.2 copyrightJurisdiction 4.1.3.3 copyrightStatusDeterminationDate 4.1.3.4 copyrightNote 4.1.3.5 copyrightDocumentationIdentifier 4.1.3.6 copyrightApplicableDates
Definition	Information about the copyright status of the object(s).
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional
Usage notes	When *rightsBasis* is “copyright”, *copyrightInformation* should be provided. Repositories may need to extend this with more detailed information. See the California Digital Library's copyrightMD schema (http://www.cdlib.org/groups/rmg//) for an example of a more detailed scheme.

Semantic unit	4.1.3.1 copyrightStatus
Semantic components	None
Definition	A coded designation for the copyright status of the object at the time the Rights statement was recorded.
Data constraint	Values should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/copyrightStatus.html.
Repeatability	Not repeatable
Obligation	Mandatory
Examples	copyrighted = Under copyright. publicdomain = In the public domain. unknown = Copyright status of the resource is unknown.

Semantic unit	4.1.3.2 copyrightJurisdiction
Semantic components	None
Definition	The country whose copyright laws apply.
Rationale	Copyright law can vary from country to country.
Data constraint	Values should be taken from ISO 3166.
Example	us de
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.3.3 copyrightStatusDeterminationDate
Semantic components	None
Definition	The date that the copyright status recorded in *copyrightStatus* was determined.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	20070608
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	4.1.3.4 copyrightNote
Semantic components	None
Definition	Additional information about the copyright status of the object.
Data constraint	None
Examples	Copyright expiration expected in 2010 unless renewed. Copyright statement is embedded in file header.
Repeatability	Repeatable
Obligation	Optional

Semantic unit	**4.1.3.5 copyrightDocumentationIdentifier**
Semantic components	4.1.3.5.1 copyrightDocumentationIdentifierType 4.1.3.5.2 copyrightDocumentationIdentifierValue 4.1.3.5.3 copyrightDocumentationRole
Definition	A designation used to identify documentation supporting the specified Rights granted according to copyright uniquely within the repository system.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	This semantic unit is intended to refer to a document detailing the granting of permission when the Rights basis is copyright. If repeated, use *copyrightDocumentationRole* to distinguish the role of the given documentation.

Semantic unit	4.1.3.5.1 copyrightDocumentationIdentifierType
Semantic components	None
Definition	A designation of the domain within which the copyright documentation identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/identifiers.
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**4.1.3.5.2 copyrightDocumentationIdentifierValue**
Semantic components	None
Definition	The value of the *copyrightDocumentationIdentifier*.
Data constraint	None
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.3.5.3 copyrightDocumentationRole
Semantic components	None
Definition	A value indicating the purpose or expected use of the documentation being identified.
Rationale	This information distinguishes the purpose of the supporting documentation especially when there are multiple documentation identifiers.
Examples	accession record copyright statement
Data constraint	Value should be taken from a controlled vocabulary.
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	4.1.3.6 copyrightApplicableDates
Semantic components	4.1.3.6.1 startDate 4.1.3.6.2 endDate
Definition	The date range during which the particular copyright applies or is applied to the content. This is distinct from *termOfGrant*, which applies to a particular act expressed in *rightsGranted* and may differ from the period of time the license, statute, or other basis applies to the content.
Rationale	Specific dates may apply to the particular copyright granted.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional
Usage notes	The repository may wish to retain the history of Rights and restrictions associated with the content over time. Associating active dates with particular Rights bases allows applications to identify which of several *rightsStatements* are in force at a given time.

Semantic unit	4.1.3.6.1 startDate
Semantic components	None
Definition	The date the granted copyright commences.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2006-01-02 20050723
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	4.1.3.6.2 endDate
Semantic components	None
Definition	The date the granted copyright expires.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2010-01-02 20120723
Repeatability	Not repeatable
Obligation	Optional
Usage notes	Use “OPEN” for an open ended term of restriction. Omit *endDate* if the ending date is unknown or the permission statement applies to many objects with different end dates.

Semantic unit	**4.1.4 licenseInformation**
Semantic components	4.1.4.1 licenseDocumentationIdentifier 4.1.4.2 licenseTerms 4.1.4.3 licenseNote 4.1.4.4 licenseApplicableDates
Definition	Information about a license or other agreement granting permissions related to an object.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional
Usage notes	When *rightsBasis* is “license”, *licenseInformation* should be provided.

Semantic unit	4.1.4.1 licenseDocumentationIdentifier
Semantic components	4.1.4.1.1 licenseDocumentationIdentifierType 4.1.4.1.2 licenseDocumentationIdentifierValue 4.1.4.1.3 licenseDocumentationRole
Definition	A designation used to identify uniquely documentation supporting the specified Rights granted by license within the repository system.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	This semantic unit is intended to refer to a document recording the granting of permission when the Rights basis is license. For some repositories this may be a formal signed contract with a customer. If the granting agreement is verbal, this could point to a memo by the repository documenting the verbal agreement. The identifier is optional because the agreement may not be stored in a repository with an identifier. For example, in the case of a verbal agreement the entire agreement may be included or described in the *licenseTerms*. If repeated, use *licenseDocumentationRole* to distinguish the role of the given documentation. This replaces the semantic unit in PREMIS version 2.1 which was called *licenseIdentifier*.

Semantic unit	4.1.4.1.1 licenseDocumentationIdentifierType
Semantic components	None
Definition	A designation of the domain within which the license documentation identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/identifiers.
Example	Digital Object Identifier
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.4.1.2 licenseDocumentationIdentifierValue
Semantic components	None
Definition	The value of the *licenseDocumentationIdentifier*.
Data constraint	None
Example	http://nrs.harvard.edu/urn-3:HUL.DRS.OBJECT:6789
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.4.1.3 licenseDocumentationRole
Semantic components	None
Definition	A value indicating the purpose or expected use of the documentation being identified.
Rationale	This information distinguishes the purpose of the supporting documentation especially when there are multiple documentation identifiers.
Data constraint	Value should be taken from a controlled vocabulary.
Example	donor agreement
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	4.1.4.2 licenseTerms
Semantic components	None
Definition	Text describing the license or agreement by which permission was granted.
Data constraint	None
Repeatability	Not repeatable
Obligation	Optional
Usage notes	This could contain the actual text of the license or agreement, or a paraphrase or summary of it.

Semantic unit	4.1.4.3 licenseNote
Semantic components	None
Definition	Additional information about the license.
Data constraint	None
Example	License is embedded in XMP block in file header.
Repeatability	Repeatable
Obligation	Optional
Usage notes	Information about the terms of the license should go in *licenseTerms*. *licenseNote* is intended for other types of information related to the license, such as contact persons, action dates, or interpretations. The note may also indicate the location of the license, for example, if it is available online or embedded in the object itself.

Semantic unit	4.1.4.4 licenseApplicableDates
Semantic components	4.1.4.4.1 startDate 4.1.4.4.2 endDate
Definition	The date range during which the license applies or is applied to the content. This is distinct from termOfGrant, which applies to a particular act expressed in rightsGranted and may differ from the period of time the license, statute or other basis applies to the content.
Rationale	Specific dates may apply to the particular Rights granted.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional
Usage notes	The repository may wish to retain the history of Rights and restrictions associated with the content over time. Associating active dates with particular Rights bases allows applications to identify which of several rightsStatements are in force at a given time.

Semantic unit	4.1.4.4.1 startDate
Semantic components	None
Definition	The beginning date of the Rights granted.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2006-01-02 20050723
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	**4.1.4.4.2 endDate**
Semantic components	None
Definition	The ending date of the Rights granted.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2010-01-02 20120723
Repeatability	Not repeatable
Obligation	Optional
Usage notes	Use “OPEN” for an open ended term of restriction. Omit *endDate* if the ending date is unknown or the permission statement applies to many objects with different end dates.

Semantic unit	**4.1.5 statuteInformation**
Semantic components	4.1.5.1 statuteJurisdiction 4.1.5.2 statuteCitation 4.1.5.3 statuteInformationDeterminationDate 4.1.5.4 statuteNote 4.1.5.5 statuteDocumentationIdentifier 4.1.5.6 statuteApplicableDates
Definition	Information about the statute allowing use of the object.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	When *rightsBasis* is “statute”, *statuteInformation* should be provided.

Semantic unit	4.1.5.1 statuteJurisdiction
Semantic components	None
Definition	The country or other political body enacting the statute.
Rationale	The connection between the object and the Rights granted is based on jurisdiction.
Data constraint	Values should be taken from ISO 3166 if applicable.
Example	us de
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.5.2 statuteCitation
Semantic components	None
Definition	An identifying designation for the statute.
Data constraint	None
Example	Legal Deposit (Jersey) Law 200- National Library of New Zealand (Te Puna Mātauranga o Aotearoa) Act 2003 no 19 part 4 s 34
Repeatability	Not repeatable
Obligation	Mandatory
Usage notes	Use standard citation form when applicable.

Semantic unit	4.1.5.3 statuteInformationDeterminationDate
Semantic components	None
Definition	The date that the determination was made that the statute authorized the permission(s) noted.
Rationale	The permission in question may be the subject of some interpretation. These assessments are made within a specific context and at a specific time. At another time the context, and therefore the assessment, could change. For this reason it can be important to record the date of the decision.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2007-12-01 20040223151047.0
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	4.1.5.4 statuteNote
Semantic components	None
Definition	Additional information about the statute.
Data constraint	None
Example	Applicability to web-published content sent for review by general counsel 9/19/2008.
Repeatability	Repeatable
Obligation	Optional

Semantic unit	**4.1.5.5 statuteDocumentationIdentifier**
Semantic components	4.1.5.5.1 statuteDocumentationIdentifierType 4.1.5.5.2 statuteDocumentationIdentifierValue 4.1.5.5.3 statuteDocumentationRole
Definition	A designation used to uniquely identify documentation supporting the specified Rights granted by statute within the repository system.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	This semantic unit is intended to refer to a document detailing the granting of permission when the Rights basis is statute. If repeated, use *statuteDocumentationRole* to distinguish the role of the given documentation.

Semantic unit	4.1.5.5.1 statuteDocumentationIdentifierType
Semantic components	None
Definition	A designation of the domain within which the statute documentation identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/identifiers.
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**4.1.5.5.2 statuteDocumentationIdentifierValue**
Semantic components	None
Definition	The value of the *statuteDocumentationIdentifier*.
Data constraint	None
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.5.5.3 statuteDocumentationRole
Semantic components	None
Definition	A value indicating the purpose or expected use of the documentation being identified.
Rationale	This information distinguishes the purpose of the supporting documentation especially when there are multiple documentation identifiers.
Data constraint	Value should be taken from a controlled vocabulary.
Examples	law application decree case law
Repeatability	Not repeatable
Obligation	Optional
Usage notes	For a particular law one might want to link to various sources of documentation, e.g. the law publication itself (role: law), the application decree that enforces it (role: application decree) or some additional text refining the law by showing a real world verdict (role: case law).

Semantic unit	**4.1.5.6 statuteApplicableDates**
Semantic components	4.1.5.6.1 startDate 4.1.5.6.2 endDate
Definition	The date range during which the statute applies or is applied to the content. This is distinct from *termOfGrant*, which applies to a particular act expressed in *rightsGranted* and may differ from the period of time the license, statute or other basis applies to the content.
Rationale	Specific dates may apply to the particular Rights granted.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional
Usage notes	The repository may wish to retain the history of Rights and restrictions associated with the content over time. Associating active dates with particular Rights bases allows applications to identify which of several *rightsStatements* are in force at a given time.

Semantic unit	4.1.5.6.1 startDate
Semantic components	None
Definition	The date the granted statute commences.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2006-01-02 20050723
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	4.1.5.6.2 endDate
Semantic components	None
Definition	The date the granted statue expires.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2010-01-02 20120723
Repeatability	Not repeatable
Obligation	Optional
Usage notes	Use “OPEN” for an open ended term of restriction. Omit *endDate* if the ending date is unknown or the permission statement applies to many objects with different end dates.

Semantic unit	4.1.6 otherRightsInformation
Semantic components	4.1.6.1 otherRightsDocumentationIdentifier 4.1.6.2 otherRightsBasis 4.1.6.3 otherRightsApplicableDates 4.1.6.4 otherRightsNote
Definition	Information about the Rights (other than copyright, license, or statute) that apply to the object(s).
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional
Usage notes	This semantic unit is used to supply information about Rights granted when the basis is something other than copyright, license or statute.

Semantic unit	**4.1.6.1 otherRightsDocumentationIdentifier**
Semantic components	4.1.6.1.1 otherRightsDocumentationIdentifierType 4.1.6.1.2 otherRightsDocumentationIdentifierValue 4.1.6.1.3 otherRightsDocumentationRole
Definition	A designation used to uniquely identify documentation supporting the specified Rights within the repository system, when the basis for these Rights is something other than copyright, license or statute.
Rationale	This semantic unit provides a mechanism to link to documentation for *rightsBasis* values other than those granted through copyright, license or statute. The Rights basis may be specified in *otherRightsBasis*.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Creation/ maintenance notes	The semantic unit is repeatable because there may be more than one document that provides information about the Rights.
Usage notes	This semantic unit is intended to refer to a document recording the granting of permission, the expression of requirements or restrictions, or other information supporting the specified *rightsBasis*. If repeated, use *otherRightsDocumentationRole* to distinguish the role of the given documentation.

Semantic unit	4.1.6.1.1 otherRightsDocumentationIdentifierType
Semantic components	None
Definition	A designation of the domain within which the Rights statement documentation identifier is unique.
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/identifiers.
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**4.1.6.1.2 otherRightsDocumentationIdentifierValue**
Semantic components	None
Definition	The value of the *otherRightsDocumentationIdentifier*.
Data constraint	None
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.6.1.3 otherRightsDocumentationRole
Semantic components	None
Definition	A value indicating the purpose or expected use of the documentation being identified.
Rationale	This information distinguishes the purpose of the supporting documentation especially when there are multiple documentation identifiers.
Examples	institutional policy deed of gift
Data constraint	Value should be taken from a controlled vocabulary.
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	**4.1.6.2 otherRightsBasis**
Semantic components	None
Definition	Designation of the basis for the other right or permission described in the *rightsStatementIdentifier*.
Data constraint	Values should be taken from a controlled vocabulary.
Example	Harvard policy
Repeatability	Not repeatable
Obligation	Mandatory
Usage notes	Use this semantic unit for specific Rights bases other than *copyrightInformation*, *licenseInformation* or, *statuteInformation*. When this semantic unit is used, should be set to 4.1.2 *rightsBasis* is "other". The Rights basis may be specific to the repository, but it is recommended to use a value from a local or globally controlled vocabulary for machine actionability. If more than one basis applies, the entire Rights entity should be repeated.

Semantic unit	4.1.6.3 otherRightsApplicableDates
Semantic components	4.1.6.3.1 startDate 4.1.6.3.2 endDate
Definition	The date range during which the particular right apply or applied to the content. This is distinct from *termOfGrant*, which applies to a particular act expressed in *rightsGranted* and may differ from the period of time the license, statute or other basis applies to the content.
Rationale	Specific dates may apply to the particular Rights granted.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional
Usage notes	The repository may wish to retain the history of Rights and restrictions associated with the content over time. Associating active dates with particular Rights bases allows applications to identify which of several *rightsStatements* are in force at a given time.

Semantic unit	4.1.6.3.1 startDate
Semantic components	None
Definition	The date the granted right commences.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2006-01-02 20050723
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	**4.1.6.3.2 endDate**
Semantic components	None
Definition	The date the granted right expires.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2010-01-02 20120723
Repeatability	Not repeatable
Obligation	Optional
Usage notes	Use “OPEN” for an open ended term of restriction. Omit *endDate* if the ending date is unknown or the permission statement applies to many objects with different end dates.

Semantic unit	4.1.6.4 otherRightsNote
Semantic components	None
Definition	Additional information about the Rights of the object.
Data constraint	None
Examples	80-year rule
Repeatability	Repeatable
Obligation	Optional

Semantic unit	**4.1.7 rightsGranted**
Semantic components	4.1.7.1 act 4.1.7.2 restriction 4.1.7.3 termOfGrant 4.1.7.4 termOfRestriction 4.1.7.5 rightsGrantedNote
Definition	The action(s) that the granting agency has allowed the repository.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional

Semantic unit	**4.1.7.1 act**
Semantic components	None
Definition	The action the preservation repository is allowed to take.
Data constraint	Value should be taken from a controlled vocabulary. A controlled vocabulary is available at: http://id.loc.gov/vocabulary/preservation/actionsGranted.html.
Repeatability	Not repeatable
Obligation	Mandatory
Examples	replicate modify use disseminate
Usage notes	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	**4.1.7.2 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	**4.1.7.3 termOfGrant**
Semantic components	4.1.7.3.1 startDate 4.1.7.3.2 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	Optional

Semantic unit	4.1.7.3.1 startDate
Semantic components	None
Definition	The date the granted permission commences.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2006-01-02 20050723
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**4.1.7.3.2 endDate**
Semantic components	None
Definition	The date the granted permission expires.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2010-01-02 20120723
Repeatability	Not repeatable
Obligation	Optional
Usage notes	Use “OPEN” for an open ended term of grant. Omit *endDate* if the ending date is unknown or the permission statement applies to many objects with different end dates.

Semantic unit	**4.1.7.4 termOfRestriction**
Semantic components	4.1.7.4.1 startDate 4.1.7.4.2 endDate
Definition	The time period for the restriction granted.
Rationale	The current definition of *termOfGrant* is "time period for the permissions granted". This allows repositories to express information about the Rights granted, but some repositories may need to express time-bounded restrictions like embargoes. To express such restrictions, use the semantic units of termOfRestriction: *startDate* for the beginning of the embargo and *endDate* for the end of the embargo.
Data constraint	Container
Repeatability	Not repeatable
Obligation	Optional

Semantic unit	4.1.7.4.1 startDate
Semantic components	None
Definition	The date the restriction commences.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2006-01-02 20050723
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.7.4.2 endDate
Semantic components	None
Definition	The date the restriction expires.
Data constraint	To aid machine processing, value should use a structured form. To facilitate exchange of PREMIS-conformant metadata, use of standard conventions, for instance as used in the date elements in the PREMIS schema, is recommended.
Examples	2010-01-02 20120723
Repeatability	Not repeatable
Obligation	Optional
Usage notes	Use “OPEN” for an open ended term of restriction. Omit *endDate* if the ending date is unknown or the permission statement applies to many objects with different end dates.

Semantic unit	4.1.7.5 rightsGrantedNote
Semantic components	None
Definition	Additional information about the Rights granted.
Rationale	A textual description of the granted Rights 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.

Semantic unit	**4.1.8 linkingObjectIdentifier**
Semantic components	4.1.8.1 linkingObjectIdentifierType 4.1.8.2 linkingObjectIdentifierValue 4.1.8.3 linkingObjectRole
Definition	The identifier of an object associated with the Rights statement.
Rationale	Rights statements must be associated with the objects to which they pertain, either by linking from the Rights statement to the object(s) or by linking from the object(s) to the Rights statement. This semantic unit provides the mechanism for the link from the Rights statement to an object.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	Linking semantic units are mandatory in the sense that a repository needs to know the information, but are defined as optional because PREMIS does not specify in which direction the linkage should be. In particular, *linkingObjectIdentifier* is optional because in some cases it will be more practical to link from the object(s) to the Rights statement; for example, a repository may have a single Rights statement covering thousands of public domain objects.

Semantic unit	**4.1.8.1 linkingObjectIdentifierType**
Semantic components	None
Definition	A designation of the domain in which the linking object identifier is unique.
Data constraint	Must be an existing *objectIdentifierType* value.
Examples	[see examples for *objectIdentifierType*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.8.2 linkingObjectIdentifierValue
Semantic components	None
Definition	The value of the *linkingObjectIdentifier*.
Data constraint	Must be an existing *objectIdentifierValue* value.
Examples	[see examples for *objectIdentifierValue*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.8.3 linkingObjectRole
Semantic components	None
Definition	The role of the object associated with an Agent.
Rationale	Distinguishes the role of the object in relation to an Agent. If this is not explicit it is necessary to analyze the relationship between objects in the object metadata.
Data constraint	None
Repeatability	Repeatable
Obligation	Optional
Usage notes	This value need not be supplied in the ordinary case where the role of the linked-to object is to be governed by the Rights statement. If the object has a different relationship to the Rights statement, however, it should be noted here.

Semantic unit	4.1.9 linkingAgentIdentifier
Semantic components	4.1.9.1 linkingAgentIdentifierType 4.1.9.2 linkingAgentIdentifierValue 4.1.9.3 linkingAgentRole
Definition	Identification of one or more Agents associated with the Rights statement.
Rationale	Rights statements may be associated with related Agents, either by linking from the Rights statement to the Agent(s) or by linking from the Agents(s) to the Rights statement. This provides the mechanism for the link from the Rights statement to the Agent.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	Linking semantic units are mandatory in the sense that a repository needs to know the information, but are defined as optional because PREMIS does not specify in which direction the linkage should be. In particular, *linkingAgentIdentifier* is optional because a relevant Agent may be unknown, or no Agent may be relevant. The latter is likely when the Rights basis is statute.

Semantic unit	**4.1.9.1 linkingAgentIdentifierType**
Semantic components	None
Definition	A designation of the domain in which the linking Agent identifier is unique.
Data constraint	Must be an existing *agentIdentifierType* value.
Examples	[see examples for *agentIdentifierType*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	4.1.9.2 linkingAgentIdentifierValue
Semantic components	None
Definition	The value of the *linkingAgentIdentifier*.
Data constraint	Must be an existing *agentIdentifierValue* value.
Examples	[see examples for *agentIdentifierValue*]
Repeatability	Not repeatable
Obligation	Mandatory

Semantic unit	**4.1.9.3 linkingAgentRole**
Semantic components	None
Definition	The role of the Agent in relation to the Rights statement.
Data constraint	Values should be taken from a controlled vocabulary.
Examples	contact creator publisher rightsholder grantor
Repeatability	Repeatable
Obligation	Optional

Semantic unit	4.2 rightsExtension
Semantic components	Defined externally
Definition	A container to include semantic units defined outside of PREMIS.
Rationale	There may be a need to replace or extend PREMIS defined units.
Data constraint	Container
Repeatability	Repeatable
Obligation	Optional
Usage notes	For more granularity or to use externally defined semantic units, extensibility is provided. Either local semantic units or metadata using another specified metadata scheme may be included instead of or in addition to PREMIS defined semantic units. When using an extension schema, a reference to that schema must be provided. See further guidance in "Extensibility," page 27. Either *rightsStatement* or *rightsExtension* must be present if the Rights entity is included. If the *rightsExtension* container needs to be associated explicitly with any PREMIS subunit under *Rights*, the container *Rights* is repeated. Also, if extensions from different external schemas are needed, *Rights* should be repeated. It is recommended to give information about the metadata used in *rightsExtension* including date the metadata was created, status of the metadata, internal linking IDs, type of metadata used and its version, message digest and message digest algorithm of the metadata, and type of identifier for external metadata links.