**Guidelines for using PREMIS with METS for exchange**

Revised January 2017

This document offers guidance for using the PREMIS schema(s) as METS extensions. It is intended to suggest common practices for encoding METS documents with PREMIS metadata for exchange purposes. A recommended use for packaging PREMIS in METS is the preparation of Submission Information Packages (SIPs), Dissemination Information Packages (DIPs) or Archival Information Packages (AIPs). Use of METS with PREMIS is optional, and some judgment is needed in considering the purposes for the METS document. A METS document designed as an exchange object intended for display and delivery may require different encoding decisions from one designed as an object of preservation. An implementation using METS for a SIP or an AIP may include more, and more diverse, data elements than one using it for a DIP, which may require a more restrictive approach and more authoritative metadata.

**Background: PREMIS and METS schemas**

***PREMIS***. In the XML implementation that was made available by the PREMIS Editorial Committee, PREMIS version 1 was implemented as 5 separate schemas that reflected the PREMIS data model. In PREMIS version 2 and above it is implemented as one schema with first level elements for each entity in the data model.

The element <object> in versions 2 and 3 (and the schema Object.xsd in version 1) includes data elements contained in the PREMIS Data Dictionary under the Object entity. Object aggregates information about a digital object held by a preservation repository or, additionally, in PREMIS version 3, about intellectual entities represented by digital objects, and describes those characteristics relevant to preservation management. Those characteristics are properties of the Object, which can be at the level of an Intellectual Entity, Representation (the set of files needed to provide a complete and accurate rendition of an Intellectual Entity), File, or Bitstream.

The element <event> in versions 2 and 3 (and the schema Event.xsd in version 1) includes data elements contained in the PREMIS Data Dictionary under the Event entity. Event aggregates information about an action that involves one or more Objects.

The element <agent> in versions 2 and 3 (and the schema Agent.xsd in version 1) includes data elements contained in the PREMIS Data Dictionary under the Agent entity. Agent aggregates information about attributes or characteristics of agents (persons, organizations, software, hardware) associated with rights management and preservation events in the life of a data object.

The element <rights> in versions 2 and 3 (and the schema Rights.xsd in version 1) includes data elements that are related to statements of rights and permissions. Rights are entitlements allowed

to agents by copyright, intellectual property law, licenses, statutes and other provisions. Permissions are powers or privileges granted by rights holders to other parties.

**METS**. The METS schema specifies an administrative metadata section (amdSec) with the following subelements:
techMD
rightsMD
sourceMD
digiprovMD

The element <premis> in versions 2 and 3 (and the schema PREMIS.xsd in version 1) provides for a container element that may be used to keep all PREMIS metadata together. This allows an implementer to put all the PREMIS metadata in one METS section if appropriate, since METS requires using one of the subelements in the administrative metadata section.

**Guidelines for using PREMIS with METS**

**1. Using PREMIS in METS sections**

If not keeping all PREMIS metadata together in the same METS section under amdSec, PREMIS first level data elements should be used in the METS sections as follows.

*premis:object* under techMD or dmdSec
If separating between METS sections, Object metadata should be in techMD for Representation, File and Bitstream levels. Object metadata for environment Intellectual Entities should be in techMD; Intellectual Entities that are not environments should be placed in the dmdSec. Implementations may wish to include information about Intellectual Entities in the object schema under techMD, but this is a local decision which may depend on the processing model used. The important consideration is that the structMap and fileSec allow the user to understand what Representations, Files and Bitstreams are being referenced and what metadata apply to them (see also section 5, below).

*premis:event* under digiprovMD

*premis:rights* under rightsMD

*premis:agent* under either digiprovMD (if given in the context of an event) or rightsMD (if given in the context of a rights statement).

If using all PREMIS units together the entire package goes in digiprovMD with the <premis> element as a container.

Whether to use one amdSec with repeating subelements (techMD, etc.) or repeat amdSec for each METS subelement is an implementer's preference. These are semantically equivalent provided that the sections are referenced appropriately within the METS document from the fileSec (see section 5, below, for information on using METS ID/IDREFS to link sections within a METS document). If referencing the amdSec, you are also referencing the children (i.e. it is a shortcut to referencing all the children). It is therefore not mandated whether or not to repeat amdSec.

The *premis:agent* should be given its own digiprovMD or rightsMD section to minimize redundancy, since the same Agent may be involved in various Events and Rights statements.

Technical metadata from different schemas (including PREMIS) may be given in separate techMD sections, with the metadata type indicated using the MDTYPE or OTHERMDTYPE attribute, or may be included within PREMIS metadata under <objectCharacteristicsExtension> in *premis:object* (versions 2 and 3).

**2. Use of PREMIS container**

If distributing PREMIS metadata among METS administrative subsections, do not use the PREMIS container. If an implementation wants to keep all PREMIS metadata together the PREMIS container is used and the PREMIS package is in digiprovMD.

**3. Redundancies between PREMIS and METS**

Redundancies may occur when capturing technical metadata in PREMIS and METS and when using other technical metadata schemas which may be referenced by or wrapped in the METS document.

There may be redundancies when elements are defined both as PREMIS elements and in the METS schema (generally as attributes). Examples include file size, which in PREMIS is included in <size> under <objectCharacteristics> and in METS as an attribute of <file> in the <fileGrp>; checksum information, included in PREMIS <fixity> under <objectCharacteristics> and in METS as attributes of <file>; and format information, captured in PREMIS in <format> under <objectCharacteristics> and as MIMETYPE in METS as an attribute of <file>. Additionally, there may be redundancies between both PREMIS and METS and other technical metadata schemas such as MIX and EBUCore elements contained within their own techMDs or added to the PREMIS <objectCharacteristicsExtension> field.

An implementer may decide whether it is easier to include the information redundantly, based on how the data will be used and/or supplied. Implementers should consider the use of the metadata

(e.g. display or preservation) and whether the METS (for display) or PREMIS (for preservation) is primary when deciding which to use and whether to record redundantly. Another consideration is which schema is more expressive. In most cases (except for structural metadata in METS vs. PREMIS) the PREMIS element is more detailed. For example, it is possible to capture more information about a file format using PREMIS <format> than METS <file> attributes. When the use of the metadata is not clear or display and preservation are of equal importance, it is advisable to err on the side of redundancy and include both PREMIS and METS constructs. Implementers should document how their application handles redundancies in a profile.

Redundancies may also occur between METS ID/REFS and PREMIS identifier elements. See section 5, below, for a discussion of this topic.

## 4. METS structMap and PREMIS structural relationship elements

Since the METS structMap is designed to support the representation of hierarchies, hierarchical structural relationships should be detailed as nested structMap  elements. If the scope of exchange objects is preservation, implementers should also use the PREMIS relationship elements in the Object schema for structural relationships (for example, has Part or is Part of). PREMIS relationship elements should always be used for relationships that are not structural (derivative, dependency, reference, etc.).

## 5. METS ID/IDREF and PREMIS identifier elements

METS provides ID/IDREFs that link between files and their related metadata in the appropriate sections. However, when PREMIS metadata is used outside of the METS container, METS ID/IDREFs will break, so PREMIS identifier elements may be needed. PREMIS provides separate data elements for identifiers (objectIdentifier, eventIdentifier, agentIdentifier, and rightsStatementIdentifier) along with linking identifiers for linking from one PREMIS entity to another (linkingObjectIdentifier, linkingEventIdentifier, linkingAgentIdentifier and linkingRightsStatementIdentifier).

Implementers need to rely on METS ID/IDREFs when referencing a METS section (e.g. techMD) from a file in the fileSec when mdWrap is used. Links should be made at the highest level possible, usually pointing to the first level subelement under amdSec. Linkages between metadata sections for different PREMIS entity metadata using the PREMIS first level elements (or the different PREMIS schemas in version 1.1) will also use the METS ID/IDREF constructs, rather than the PREMIS ID/IDREFs. As described above, Implementers should also include the explicit PREMIS linking identifier elements in the event that the PREMIS metadata is used outside of the METS document, but linking using PREMIS constructs may also provide more detailed semantics than the METS linking itself. In these cases two-way linking (i.e. using both METS ID/IDREFs and PREMIS identifier elements) may be necessary.

When establishing the values of METS IDs, it is important that implementers create appropriate IDs by assuring that they are unique within the instance and that they maintain their persistence. There should not be any orphaned elements that cannot be ultimately be linked back to the fileSec or structMap; however, a METS ID attribute can be used to link METS sections to one another without referencing the fileSec or structMap directly. For example, the fileSec could include an ADMID reference to a techMD containing metadata about a PREMIS File, and that techMD could contain an ADMID reference to another techMD containing metadata about a related Intellectual Entity. Note that the METS fileSec provides a means of referencing PREMIS Bistreams through the <stream> element beneath the <file> element. Any sections with PREMIS elements should be referenced; although orphaned elements are not expressly prohibited, they risk being overlooked by systems relying on METS IDs to provide links between digital objects and their preservation metadata. Note that these considerations hold for METS in general with or without PREMIS as an extension schema.

## 6. Use of METS profiles

Implementations are encouraged to establish profiles that specify options chosen for particular applications or where this document is not prescriptive and, where there are redundancies, state whether elements in PREMIS or METS are primary. Profiles may also suggest policy for resolving conflicts between metadata included redundantly using both PREMIS and METS elements or PREMIS and format-specific elements.

## 7. Examples of PREMIS in METS documents

Sample PREMIS in METS documents are linked to the relevant PREMIS Data Dictionary versions at http://www.loc.gov/standards/premis/.

## 8. Acknowledgements

These guidelines were developed by the PREMIS Editorial Committee in collaboration with the METS Editorial Board.