PBCore/PREMIS Implementations and Mappings

Because PREMIS and PBCore have different scopes, they are more commonly used together rather than having data in one format transformed into data in the other format. This documentation focuses mainly on outlining various options for implementing PBCore and PREMIS structured data together; however, it does also include mappings between PBCore and PREMIS concepts, as well as direct mappings between PREMIS semantic units and PBCore elements and attributes, where they exist.

Recommended options for implementing PBCore and PREMIS together

Option 1:
If your repository is built more closely around the PBCore schema, you can embed PREMIS data within the PBCore records. PREMIS individual semantic units or entire XML documents can be stored in a PBCore instantiationExtension element. Using the extensionEmbedded element, the PREMIS XML can be carried along inside the PBCore XML document. Alternatively, if you only want to bring a few PREMIS elements into your PBCore, you can use an extentionWrap element. For more on using extensions in PBCore, see our tutorial.

This method can specifically be used to store PREMIS Events, Agents, or Rights related to a specific asset or instantiation with the record for that asset or instantiation.

XML Example: PBCore_with_PREMIS_Events.xml

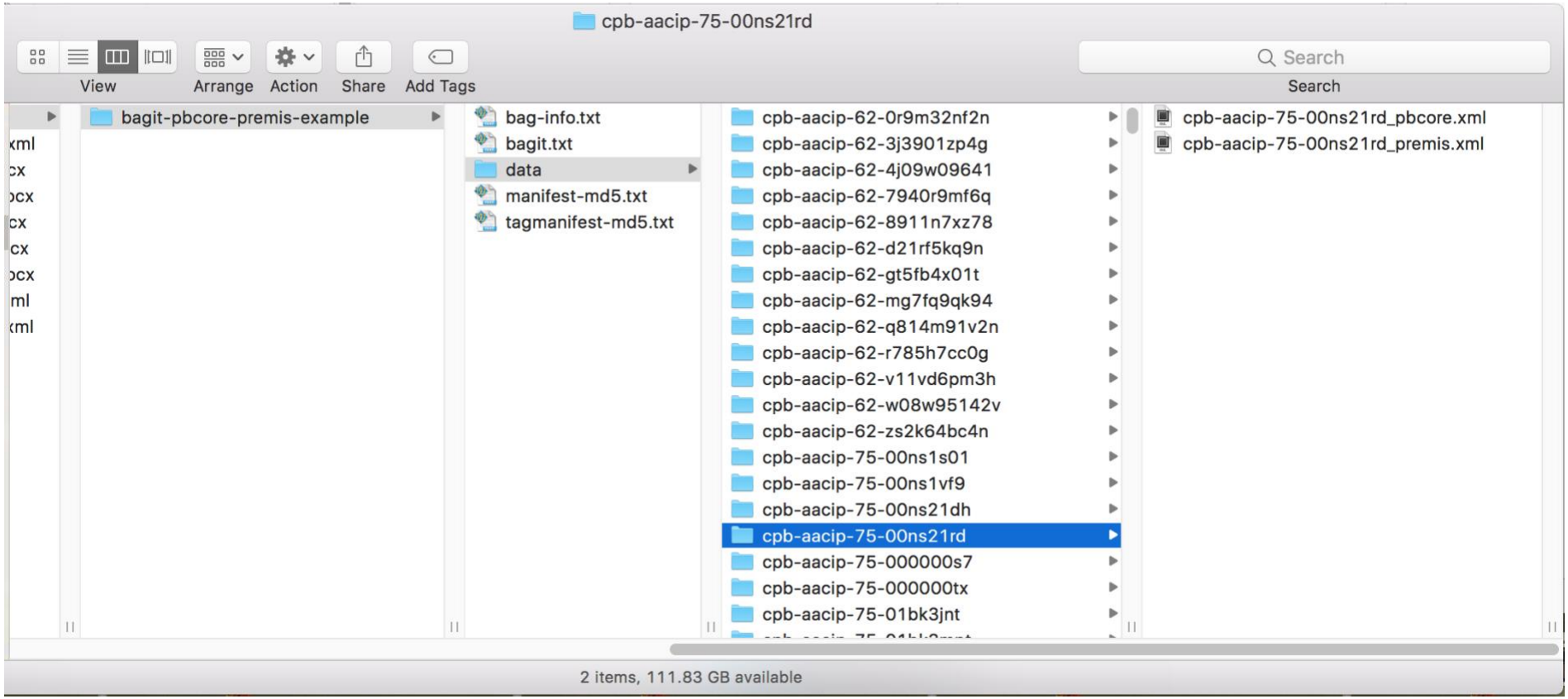
Option 2:
If you have a preservation repository built around the PREMIS schema, but you would like to include structured technical metadata about your items or files, you can use the PREMIS objectCharacteristicsExtension to store a PBCore instantiationDocument describing that file. PREMIS specifically created the objectCharacteristicsExtension semantic unit to store data using a format specific standard, like PBCore.

XML Example: PREMIS_with_PBCore_Instantitaion.xml

Option 3:
The METS (Metadata Encoding & Transmission Standard) schema is used to encode descriptive, administrative and structural metadata about digital objects. METS's structure allows for the use of other schemas that can better describe a certain set of qualities pertaining to a specific item. In this way, a PBCore instantiationDocument can be wrapped in the techMD section, thereby taking advantage of the AV-specific nature of PBCore. Additionally, preservation specific metadata expressed in PREMIS XML can be wrapped in the digiprovMD section.

XML Example: METS_with_PBCore_and_PREMIS.xml

Option 4:
PREMIS and PBCore data don't necessarily need to be in the same XML file to remain associated. Using a convention like BagIt, you can store both a PBCore XML file and a PREMIS XML file together, so their association remains intact. This could look like this:



Mapping PREMIS entities to PBCore root elements/main structural units

PREMIS is structured using 4 entity types: Objects, Events, Agents, and Rights. PBCore does not have a corollary to PREMIS's Events or Agents; however, PREMIS Objects can map to various PBCore root elements and main structural elements, and PREMIS Rights are similar to PBCore's rightSummary elements.

PREMIS Objects can have different types (Intellectual Content, Representations, Files, and Bitstreams). PREMIS's Bitstreams and PBCore's Essence Tracks represent the same concept; while PREMIS Representations and Files can both be analogous to instantiations in PBCore's structure.

PBCore rightsSummary container elements can store some of the data that PREMIS Rights expresses; however, as rightsSummary elements can't stand as their own documents within PBCore, there is no directly corollary between PREMIS Rights and PBCore root elements.

PREMIS specifically and purposely does not support the description of intellectual content through properties such a title, subject heading, etc. The PREMIS editorial committee notes that there are enough standards, such as MARC, Dublin Core, etc. that cover these types of properties. Because PREMIS does not include these properties, there is no direct mapping between PREMIS and the Asset elements in PBCore Description Documents, which is where PBCore stores these types of properties.

Mapping PREMIS semantic units to PBCore elements

Although in most use cases we recommend the above implementation solutions, it may be relevant in some cases to directly compare the two standards, so we have also provided the following mappings. Only semantic units, elements, and attributes that have direct mappings have been stated explicitly. For the rest, any PREMIS semantic unit can be stored in an extensionWrap element and PBCore elements or xml can be stored in either the PREMIS significantProperties or objectCharactertisticsExtension semantic units.

PREMISObject_to_PBCoreInstantiation.xlsx is a mapping from PREMIS Objects that are Representations or Files to PBCore instantiationDocuments.

PREMISObject_to_PBCoreEssenceTrack.xlsx is a mapping from PREMIS Objects that are Bitstreams to PBCore essence tracks. Please note that in PBCore essence tracks are not root elements and can only exist if they are part of a parent instantiation.

PBCoreInstantiation_to_PREMISObject.xlsx is a mapping from a PBCore instantiationDocument to a PREMIS Object.

PBCoreEssenceTrack_to_PREMISObject.xlsx is a mapping from a PBCore essence track to a PREMIS Object.