Archive Design Based on Planets Inspired Logical Object Model

Eld Zierau and Anders Sewerin Johansen

The Royal Library of Denmark, P.O. BOX 2149, 1016 Copenhagen K, Denmark
elzi@kb.dk, asjo@kb.dk

Abstract. This paper describes a proposal for a logical data model based on preliminary work within the Planets project. In OAIS terms the main areas discussed are related to the introduction of a logical data model for representing the past, present and future versions of the digital object associated with the Archival Storage Package for the publications deposited by our client repositories.

1 Introduction

This paper describes a logical data model for representing several concrete versions of multi-file digital objects in a trusted institutional repository. Our goals are to preserve and uniquely identify digitally-born publications past, present and future, and being able to disseminate representations or versions of these publications on request.

2 The PINDAR Project

PINDAR is to be a Trusted Institutional Repository [1] for publications from Danish universities. It is funded by Denmark's Electronic Research Library (http://www.deff.dk), and is a joint project of the Royal Danish Library in Copenhagen and the State and University Library in Aarhus. The primary goal is to preserve one or more versions of publications and associated metadata from Institutional Repositories (IRs), which can be requested at a later stage.

Our investigations show that the structure of the submitted publications (Digital Objects, DO) is relatively simple. Most are single files (PDF, Word documents...), but some will be structured, multi-file objects (e.g. TeX projects). The requirement that PINDAR must accept several versions of an identified object ("a publication") dictates that we must consider a submission as a concrete version of a *logical* object that can be identified in a globally unique fashion in the future. This means that our Archival Information Package (AIP) must either implement this concept directly, or employ other components that do so.

In the OAIS model [2] the AIP is explicitly identified as a logical package, rather than an easily identified physical entity e.g. a single row of data in a specific database. We think it is best to maintain that the AIP is a rather abstract entity that should not contain direct references to concrete entities such as files and bitstreams. We therefore

B. Christensen-Dalsgaard et al. (Eds.): ECDL 2008, LNCS 5173, pp. 37–40, 2008.

propose to introduce a logical object for the purpose of managing the various physical manifestations (files and bitstreams) of the known version of the "intellectual entity".

3 Designing a Logical Object Model

The logical object model must be sufficiently expressive to represent the logical and digital objects submitted by clients or generated internally for preservation purposes. We will focus on the ability to represent the logical object level and its manifestations. A manifestation is an interpretation of the object. It can either be ingested from a producer IR ("the first version"), a substitution for previous delivered version ("a new version"), a dissemination version or as the result of a migration for preservation purposes. We will not consider the digital file level, as we will use the Netarchive-Suite (http://netarchive.dk/suite/) bit-preservation module for this task.

In the following we investigate the Planets (http://www.planets-project.eu/) project Conceptual Data Model (PCDM) as a prospective solution. Our analysis is based on the current internal review version (version 4). The distinctive features of the Planets model are:

- *Separation of logical and physical objects*
  Many archives have an operational requirement for maintaining several active manifestations of a logical object (see below). This separation provides a common reference point for the manifestations.
- *Several manifestations of objects*
  This is useful in logical preservation, as it allows us to maintain different, but non-exclusive interpretations of the object for e.g. preservation purposes.
- *N-to-m migrations of concrete parts (files and bitstreams)*
  The PCDM is designed to accommodate not only 1-to-1 transformations (1 file becomes 1 file), but also 1-to-n, n-to-1 and n-to-m transformations.

In PINDAR we shall interpret a manifestation in a broader sense than the PCDM does. The PCDM assumes that two deposits are represented in two objects, and hence two delivered versions are not part of the same logical object. In PINDAR we *can* identify two deliveries as related as they will have the same globally uniquely identifier (GUID) associated with the original IR identifier (IR-ID). This can be achieved by using the entities at manifestation level in a slightly different fashion than the regular PCDM does.

The PCDM is a rather complex model that contains many functional groups, and dozens of entities. We ignore functionality that is irrelevant for PINDAR, and instead focus on the core model that deals directly with representing logical and concrete digital object. This lets us retain some compatibility with Planets tools, nomenclature and concepts, while reducing the complexity of the implementation significantly.

In summary we will include: Information on the ***Logical Object*** structure and its ***Manifestations***. Data on depositing events for different versions will be placed as metadata on event information and documentation of validation properties at depositing events such as ***Rendering Information*** and ***Characterisation***.

In a future extension of PINDAR there may be included more ***Metadata*** or the metadata may be represented in other systems, but for now the delivered technical and bibliographic metadata from the IRs will simply be stored as a file in the digital object

Digital File Representation is a separate issue. Only references are relevant here. ***Preservation Metadata*** is not a major concern at this stage. At a later stage we can either implement a PINDAR specific solution, or e.g. use PREMIS [3], which is conceptually compatible with the PCDM with regards to e.g. object model.

4 The Concrete Implementation in Pindar

Below we illustrate our proposal graphically, and discuss the entities, their intended use and their relationships.

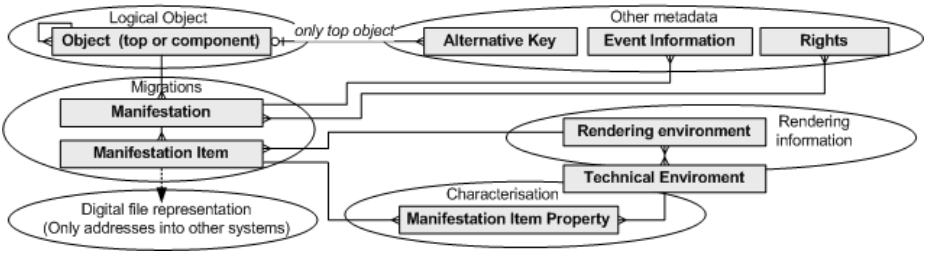

Fig. 1. Entity diagram for logical object model

Note that an object can be of two types: A top object that defines the logical object and shares the *globally unique identification* with the related AIP, or a component of a logical object. This is not immediately obvious from the type diagram. Also note that the ***Technical Environment*** is shared between *Characterisation* and *Rendering Information*. Software tools supply both types of information, and all software tools require a specific technical environment.

Manifestation Item Properties represents ingest information validation results. ***Rights*** and ***Ingest Information*** are simple metadata for the delivered objects. These are the minimal operational metadata required for the initial system. The ***Alternative Key*** entity implements the translation from GUID to IR-ID.

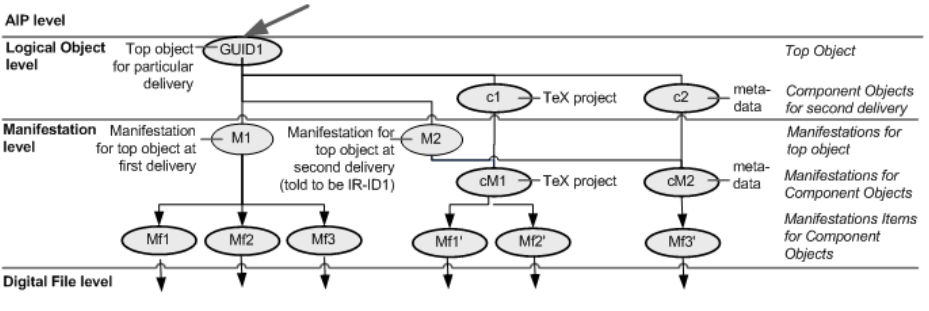

Fig. 2. Instance diagram for logical object model

To complete the picture, the illustration above details an entry/instance diagram for two versions of an article. In the first deposit it was not specified that the digital object consisted of two items (a TeX project and a metadata file). In the second deposit that has been remedied. IR-ID1 identifies our Logical object for the article. The first version is represented by manifestation M1 with manifestation items MI1, MI2, MI3 representing the three files in version 1. The second version is represented by manifestation M2, but here the ingested files are grouped as two components.

5 Discussion

The logical object model is complex. We claim that the complexity is manageable, and that the expressiveness is substantial, as it allows for preserving the original digital objects after migration through versioning, and multiple interpretations that may preserve different aspects [4] of an object.

Diverging from the full PCDM introduces the risk that we loose easy access to tools and practices developed for the PCDM. This would be inconvenient, but not a direct threat to the viability and integrity of the repository. We claim that we have captured the core concepts and entities, and that adding the omitted elements at need is a viable option. Further, the PCDM is currently a work in progress, and many of the omitted functional groups are not yet finalized.

6 Conclusion

We decided to employ a reduced version of the Planets Conceptual Data Model to implement a logical object model. This model is responsible for representing and maintaining several different versions and manifestations of the logical object ("the publication"), which relieves the AIP of these responsibilities. The model is sufficiently expressive to store digital objects that consist of several files, even if the file structure changes from version to version.

References

1. Beagrie, N., Bellinger, M., Dale, R., Doerr, M., Hedstrom, M., Jones, M., Kenney, A., Lupovici, C., Russell, K., Webb, C., Woodyard, D.: Trusted Digital Repositories: Attributes and Responsibilities, RLG-OCLC Report (2002) (retrieved, March 2008), http://www.rlg.org/longterm/repositories.pdf
2. Reference Model for an Open Archival Information System (OAIS) CCSDS 650.0-B-1 BLUE BOOK (January 2002), http://public.ccsds.org/publications/archive/650x0b1.pdf
3. PREMIS (PREservation Metadata: Implementation Strategies) Working Group (2005). Data Dictionary for Preservation Metadata: Final Report of the PREMIS Working Group (May 2005). (retrieved, March 2008), http://www.loc.gov/standards/premis/
4. Clausen, L.R.: Opening Schrödingers Library: Semi-automatic QA Reduces Uncertainty in Object Transformation. In: Kovács, L., Fuhr, N., Meghini, C. (eds.) ECDL 2007. LNCS, vol. 4675, pp. 186–197. Springer, Heidelberg (2007)