Angela Dappert
Rebecca Squire Guenther
Sébastien Peyrard *Editors*

Digital Preservation Metadata for Practitioners

Implementing PREMIS

Springer

Digital Preservation Metadata for Practitioners

Angela Dappert · Rebecca Squire Guenther
Sébastien Peyrard
Editors

Digital Preservation Metadata for Practitioners

Implementing PREMIS

Springer

Editors
Angela Dappert
The British Library
London
UK

Rebecca Squire Guenther
Consultant
New York, NY
USA

Sébastien Peyrard
Bibliothèque nationale de France
Site François-Mitterrand
Paris
France

ISBN 978-3-319-43761-3 ISBN 978-3-319-43763-7 (eBook)
DOI 10.1007/978-3-319-43763-7

Library of Congress Control Number: 2016949612

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Digital Preservation Metadata for Practitioners: Implementing PREMIS is intended for anyone who needs to preserve digital assets in any form over a long period of time. It considers the steps involved in determining what information one needs to keep, together with one's digital assets, so that they can be understood and used in the long term.

The information that ensures the long-term usability of digital objects to keep them accessible in some form in the future is referred to as digital preservation metadata. This is particularly important for *repositories*, places where information objects are stored and managed for a long time. Simply storing digital objects on a data carrier is not enough to keep them usable. They need to be managed in a repository so that they are protected from accidental or intentional damage and so that a full computing environment can be created in which they can be accessed and understood when they are needed.

In the early 2000s it became clear that a shared community metadata standard was needed to ensure long-term preservation of our ever-increasing digital resources. Experts from key memory institutions and repository developers joined together to define it, resulting in *The PREMIS Data Dictionary for Preservation* Metadata, which has become a de facto standard that defines *core* metadata needed by most preservation repositories. There is a large variety of information object content types, such as documents, images, audiovisual material, web pages, spreadsheets, and business management files in proprietary formats. One needs additional application or content-type specific metadata that go beyond this core metadata, to achieve long-term usability of their specific features. Format-specific metadata are defined in other standards that can be combined with the PREMIS core preservation metadata standard. The use of standards is important as it supports the development of a community of best practice; it helps practitioners learn from the insights of others, so that they do not inadvertently overlook key metadata in their own practice; it allows for development of tools to make metadata creation and management easier; and it enables organizations to more easily exchange information with each other.

Because standards are broadly applicable and flexible, they need to be customized to fit the context of an individual organizational situation. The PREMIS Data Dictionary provides the basic building blocks for creating one's preservation metadata, intending to be applicable to diverse environments and content types. It provides a data model consisting of basic entities (objects, agents, events, and rights) and basic properties (called '*semantic units*') that describe them. The practitioner then needs to determine how this is applied to their specific context, i.e., which specific entities, relationships, and vocabulary are needed to support their repository functionality. The dictionary provides considerable freedom in how this is done.

This book helps readers understand which options need to be considered in specifying the digital preservation metadata that is needed to customize to their individual content types, technical infrastructure, and organizational needs. It provides practical guidance examples and raises important considerations. It does not provide a full-fledged implementation solution that can, by definition, only be specific to a given preservation context. As such, the book forms the bridge between the formal specifications provided in a standard, such as the *PREMIS Data Dictionary*, and a specific implementation.

The book gives an introduction to fundamental issues related to digital preservation metadata and then proceeds to develop an in-depth understanding of issues related to its practical use and implementation. It should be of use to beginners and current practitioners. It is equally targeted at digital preservation repository managers and metadata analysts who are responsible for digital preservation metadata, as it is at students in Library, Information and Archival Science degree programs or related fields. It can be used at the conception stage of a digital preservation system or for self-audit of an existing system.

The book explores the following topics:

Chapter 1 gives an end-to-end overview of the steps involved in determining what information one needs to keep, together with one's digital assets, so that they can be understood and used in the long term.

Chapter 2 explains how risk and requirements analysis methodologies can be used as the basis for determining the required metadata. Standards give guidance for implementers, but they need to be tailored to specific needs so that the metadata supports requirements. The chapter proposes important questions that help to break down the task of determining these requirements into more manageable subtasks.

Chapter 3 reviews the development of PREMIS (now in version 3.0), explains the goals and principles behind the *PREMIS Data Dictionary*, reviews some of its features, and puts it in the context of the OAIS model.

Chapters 4 through 12 explain the methodology for designing an application profile and illustrate implementation choices that have been made by leading institutions from around the world for specific entities and content types. These specific types include moving images, web archives, e-books, removable disk images, archival collections, and computing environments; specific entities include events, agents, and rights.

Chapter 13 explains the technical options for serializing data conformant to the PREMIS Data Dictionary. Serialization is the process of mapping a data model into formatted bits; serialization is generally required to facilitate transmission, storage, or computation on the data. Common choices which are explored include XML, RDF, and relational database implementations.

Chapter 14 discusses how to make different metadata frameworks work together in an institutional ecosystem. In most contexts, PREMIS will not be the only metadata standard implemented in an institution. Different standards aim at covering complementary functionality and can be combined in a modular fashion.

Chapters 15 through 17 provide a snapshot of tools and systems that can be used to produce, consume, or manage PREMIS metadata. The tools included in the chapters are diverse in focus and functionality; some were designed for PREMIS metadata specifically, while others are more general with broader usage. There are also studies of PREMIS implementations in open-source digital preservation systems and in a local repository.

Chapter 18 reviews the PREMIS Conformance Statement, which defines what can be considered a well-executed implementation. It explores the value of conformance, how best to achieve it, and how conformance could be linked to assertions of best practice and certification.

London, UK — Angela Dappert
New York, NY, USA — Rebecca Squire Guenther
Paris, France — Sébastien Peyrard

Acknowledgements

The editors would like to thank Ralf Gerstner for suggesting that the book should be written, Clément Oury for supporting the idea, the PREMIS Editorial Committee for advice and contributions, and Adam Farquhar for assisting with reviewing.

Contents

Contributors

Daniel Gelaw Alemneh University of North Texas, UNT Libraries, Denton, TX, USA

Karl-Rainer Blumenthal New York Art Resources Consortium, New York, NY, USA

Karin Bredenberg Riksarkivet, Avdelningen För Offentlig Informationshantering/ Public Information Management, Stockholm, Sweden

Alexandra Chassanoff University of North Carolina, Chapel Hill, NC, USA

Carol Chou Florida Virtual Campus, Gainesville, FL, USA

Euan Cochrane Yale University Library, New Haven, CT, USA

Angela Dappert The British Library, London, UK

Adam Farquhar The British Library, London, UK

Andrea Goethals Harvard Library, Cambridge, MA, USA

Rebecca Squire Guenther Consultant (Formerly Library of Congress), New York, NY, USA

Thomas Habing Library, University of Illinois at Urbana-Champaign, Champaign, IL, USA

Mark Jordan W.A.C. Bennett Library, Simon Fraser University, Burnaby, BC, Canada

Amy Kirchhoff ITHAKA, Princeton, NJ, USA

Christopher A. Lee University of North Carolina, Chapel Hill, NC, USA

Peter McKinney National Library of New Zealand/Te Puna Mātauranga O Aotearoa, Wellington, New Zealand

Evelyn McLellan Artefactual Systems Inc., New Westminster, BC, Canada

Sheila M. Morrissey ITHAKA, Princeton, NJ, USA

Clément Oury International ISSN Centre, Paris, France

Sébastien Peyrard Bibliothèque Nationale de France, Paris, France

Mark Edward Phillips University of North Texas, UNT Libraries, Denton, TX, USA

Julie Seifert Harvard Library, Cambridge, MA, USA

Kara Van Malssen AVPreserve, Brooklyn, NY, USA

Kam Woods University of North Carolina, Chapel Hill, NC, USA

Eld Zierau The Royal Library of Denmark, Copenhagen K, Denmark

Chapter 1
An Introduction to Implementing Digital Preservation Metadata

Angela Dappert, Sébastien Peyrard and Rebecca Squire Guenther

1.1 Introduction

Digital Preservation Metadata for Practitioners: *Implementing PREMIS* is written for anybody who needs to care for digital assets in any form over a long period of time. There are many decisions involved in this task, such as choosing data storage, backups and replication for safekeeping, or choosing file formats that can be read in the future. One key challenge that is addressed in this book is the question of what information one needs to keep, together with one's digital assets, so that they can be understood and used in the long-term. In other words, what metadata does one need?

Metadata are structured data that describe information objects, such as books, images, and maps, but also other objects. They are a key vehicle for accessing, managing, and understanding these objects. The properties that they describe are carefully chosen so that they are most helpful for the tasks they need to support. Librarians use metadata to create catalogs that help readers discover collection items by a variety of search criteria. The title of a book is the prototypical piece of metadata that comes to most people's minds. It helps you find the book and it helps you to decide whether it might be of interest to you. Online shoppers use metadata

A. Dappert (✉)
The British Library, 96 Euston Rd, London NW1 2DB, UK
e-mail: angela.dappert@bl.uk

S. Peyrard
Bibliothèque nationale de France, Site François-Mitterrand,
Quai François-Mauriac, 75706 Paris Cedex 13, France
e-mail: sebastien.peyrard@bnf.fr

R.S. Guenther
Consultant (Formerly Library of Congress), 215 W. 75th St.,
New York, NY 10023, USA
e-mail: rguenther52@gmail.com

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_1

to find items to purchase and to decide whether the item meets their needs. Store managers use metadata to control their stock and to identify which items are in high demand. Photographers use metadata, embedded in digital images, to trace the history of an image or to inform editing decisions based on technical image information. In this book, we focus on metadata for a broad range of information objects.

1.2 Digital Preservation Metadata: Useful Information for Long-Term Access to Digital Objects

In addition to these search, discovery, access, rights, management, provenance, or technical metadata, we need to ask ourselves what metadata we need in order to keep digital information objects accessible over a long time—that is, to ensure their digital preservation. One mostly does not need to think about how to open a digital image, access an internet page, or edit a document when one accesses it on the same generation of computer that was used to create it. The file and the computer environments are compatible, the software needed to render or execute the file is installed, licenses are current, the software is supported by the software vendor, and you do not need much additional information about the file or the software or hardware it depends on. This is not the case if the data carrier on which the file is stored is unusual or outdated; if the file format is proprietary, rare or older; if the software or hardware is no longer supported; or if the digital object has undergone changes over time.

If we want to ensure the long-term usability of digital objects, it is necessary to gather enough information so that we can keep them accessible in some form in the future. This information is referred to as digital preservation metadata. This is particularly important for *repositories*, places where information objects are stored and managed for a long time. Simply storing digital objects on a data carrier is not enough to keep them usable. They need to be managed in a repository so that they are protected from accidental or intentional damage and so that a full computing environment can be created in which they can be accessed and understood when they are needed.

1.3 Standards for Digital Preservation Metadata

Over the last decade independent community activities emerged that initially defined their own metadata needs. But it quickly became clear that it was much better if the main actors shared the effort and worked toward developing a shared standard. Experts from key memory institutions and repository developers joined together to form the PREMIS Working Group to do exactly this. *The PREMIS Data*

Dictionary for Preservation Metadata [1], a de facto standard, now defines *core* metadata that are needed by most preservation repositories. There is a large variety of information object content types, such as documents, images, audiovisual material, web pages, spreadsheets, and business management files in proprietary formats. One needs additional application or content type specific metadata that go beyond this core metadata, to achieve long-term usability of their specific features. These forms of metadata are defined in additional standards that can be combined with the PREMIS core preservation metadata standard.

Use of standards is important as it supports the development of a community of best practice; it helps you learn from the insights of others, so that you do not inadvertently overlook key metadata in your own practice; it allows for development of tools to make metadata creation and management easier; and it enables organizations to more easily exchange information with each other.

1.4 How to Develop a Digital Preservation Metadata Profile

Because standards are broadly applicable and flexible they need to be customized to fit the context of an individual organizational situation. The PREMIS Data Dictionary, as the de facto digital preservation metadata standard, provides a data model consisting of basic entities (objects, agents, events, and rights) and basic properties (called '*semantic units*') that describe them. Understanding the dictionary alone does not teach you how to create your preservation metadata. It is like a language definition. It gives you a basic grammar and vocabulary. But it does not give you the sentences that tell your story. You use its constructs in order to write your own story. You have considerable freedom in how you do this. You will not use all of the words in the language; you choose how to structure the world you are describing; you may define some custom vocabulary for your own specific domain; you may fall back on foreign languages to express specific parts of your story, you may create your own dialect or accent to make this language serve your needs.

The result of customizing a set of standards to your needs is called a *metadata* (or *application*) *profile*. An application profile can be defined as follows:

> A set of metadata elements, policies, and guidelines defined for a particular application. The elements may be from one or more element sets, thus allowing a given application to meet its functional requirements by using metadata from several element sets including locally defined sets [2].

This book helps readers understand which options need to be considered in specifying a digital preservation metadata profile so that it is customized to their individual content types, technical infrastructure, and organizational needs. It provides practical guidance examples and raises important considerations. It does not provide a full-fledged implementation solution that can, by definition, only be specific to a given preservation context. As such, the book forms the bridge

between the formal specifications provided in a standard, such as the *PREMIS Data Dictionary* [1], and a specific implementation. First, it explains the thought-processes needed to decide what digital preservation metadata are needed and how they should be organized. Once this step has been accomplished, we can turn to the task of identifying how the needed metadata can be obtained. Will it be extracted automatically from the digital objects or from the metadata provided by, say, their publisher? Will it be created manually? Will it be submitted in various forms from various sources and need to be brought into a uniform format? One can then go on to choosing the most suitable standards for implementing the metadata and then, finally, to determining how to implement the specification in the chosen standards and serialization format. All of these topics are addressed in this book.

The book gives an introduction to fundamental issues related to digital preservation metadata and then proceeds to develop an in-depth understanding of issues related to its practical use and implementation. It should be of use to beginners and current practitioners. It is equally targeted at digital preservation repository managers and metadata analysts who are responsible for digital preservation metadata, as it is at students in Library, Information and Archival Science degree programs, or related fields. It can be used at the conception stage of a digital preservation system or for self-audit of an existing system.

This book is usable independent of the chosen standard or the version of the chosen standard. Rather than giving instructions on how to implement with a specific version it is about how to specify and implement your own digital preservation metadata profile to match your content type and organization. At the point of publication of this book PREMIS version 3.0 has been released but most existing implementations still use earlier versions. This is a normal aspect of using a standard since standards develop with user needs. Examples in this book are given in a specific version but usually can easily be translated to newer versions.

1.5 Reading Guide to This Book

Chapter 2 explains how risk and requirements analysis methodologies can be used as the basis for determining the required metadata.

The PREMIS Data Dictionary is generally applicable to all digital preservation scenarios, but it cannot give specific solutions on how it should be implemented for a particular application or for a specific content type. This means that you need to create a customized, content type specific and organization specific profile that specifies the required entities, structural relationships between them, and their properties. In other words, you define a specific profile of the generic Data Dictionary. In fact, this profile specifies those components that are *required* for your specific scenario based on a risk and functionality-driven requirements analysis.

Preservation risks are, among others, organizational, policy, economic, legal, or technological risks. The risk mitigation strategies that will be implemented in the repository depend on the availability of the appropriate metadata.

Requirements can also be defined based on

- the basic functions to be performed on a repository,
- the need for integration with existing systems, and
- the need for integration with related existing metadata and their standards.

The requirements analysis should always be agnostic of the eventual implementation solution and focus just on the required functionality. But sometimes the metadata choice is determined by what types of workflows can be implemented, and whether they may have to be manual or automatic. There are also nonfunctional requirements concerning cost and efficiency that can affect the choice of implemented metadata.

Requirements analysis in the domain of digital preservation does not need to start from scratch. Frameworks exist that

- describe the current best practice of digital preservation functionality that is to be supported in a repository, such as OAIS [3];
- an understanding of the basic preservation goals that need to be achieved, such as the ones defined by Caplan [4];
- risk analysis approaches, such as SPOT [5] or DRAMBORA [6]; and
- agreed core digital preservation metadata, such as is defined in the PREMIS Data Dictionary [1].

They all can inform the requirements definition process, but all of them need to be customized to the specific situation.

Chapter 3 explains the principles behind the PREMIS Data Dictionary and puts it in the context of the OAIS model.

The Data Dictionary specifies

- the data model,
- the basic categories of preservation metadata,
- the principles behind its design,
- how to apply it in practice, and
- the bodies and activities needed to ensure that the standard evolves together with its user community's needs.

OAIS [3] is a fundamental framework for long-term repositories, but PREMIS goes beyond OAIS in supporting the whole life-cycle of digital objects.

Chapters 4 through 12 explain the methodology for designing an application profile and illustrate implementation choices that have been made by leading institutions from around the world for specific entities and content types. This discussion is technically neutral and illustrates a variety of aspects that need to be considered in their context.

Chapter 4 discusses the general methodology in designing the specific logical data model for the context. Once the metadata requirements are known, one can specify the data model so that it supports the implementation of these requirements. The PREMIS Data Dictionary defines the general entities of the basic data model.

One needs to determine how they are to be tailored to the specific scenario. Which entities are significant and implemented depends on the functionality that is to be supported through the metadata. For example, for an '*e-journal*' scenario there may be '*journal*', '*issue*', '*article*', and '*figure*' objects that are particular subentities of the PREMIS Object. Do they all need to be implemented? For which of them do we need to create *Intellectual Entity* objects, so that we can capture descriptive metadata for them, to support search and access? Which of them have concrete digital realizations in the form of *File* or *Bitstream* objects that need to be described by technical metadata? Which of those have *Representations* that consist of several files for a single rendition of the object? Which events, agents and rights need to be captured to provide evidence of the digital assets' authenticity over time? Additionally, one needs to determine how the chosen entities are related to each other. Having access to an '*article*' object, for examples, makes it possible to follow its linking relationships to related '*prepublication*' objects and to the events that have affected this object over time.

Rather than defining a data model from scratch one can also reuse other people's profiles or customize default profiles that come with a digital preservation software solution. The general methodology for creating an application profile can equally be applied to this task of customization.

The case studies in Chaps. 5 through 12 illustrate how '*object*'-specific issues have been decided

- for different *entity types*, such as objects, events and rights;
- for different *content types*, such as web archives, audiovisual or e-book materials; and
- for different *organization types*, such as archives as opposed to libraries.

They include the choice of data models; the needs of the designated user communities; purposes in different communities, such as archives, libraries, museums; purposes of the collections; the functions the metadata need to support, such as storage, search, browsing, access, exchange, data management or preservation actions; intended scales for the size of the collection, IT resources, and human resources; what is particular about the content type that might impact the way you implement PREMIS descriptions of it; what other metadata systems were integrated and how that influenced PREMIS choices; what is in the scope of PREMIS and what is not; how the repository architecture influences metadata decisions; what policies or regulations affect implementation choices; pros and cons of choices; pitfalls and lessons learned. Chapters 11 and 12 discuss the use of event, agent and rights metadata.

Chapter 13 explains the serialization options, with XML, RDF and relational database implementations as examples of three common choices.

Once the basic entities in the logical data model and their relationships are defined, and the relevant properties that describe them are decided, one can go on to design the physical data model. A serialization is

> the process of translating data structures or object state into a format that can be stored (for example, in a file or memory buffer, or transmitted across a network connection link) and reconstructed later in the same or another computer environment [7].

The serialization of the metadata depends on the chosen metadata standards. On the other hand, a preferred serialization choice may influence which standard to choose.

It is important to note that PREMIS is completely implementation independent. It is a data dictionary—that is, a way of organizing your domain model with applicable semantic units that describe the entities in the model. It helps you think about your domain and its requirements. It does not at all specify how you implement it and almost everything is optional rather than mandatory—so that you can choose only what is needed by you.

So how, then, do we decide what serialization to choose? How do we combine different standards? In what way are different metadata uses supported by different serializations? And how can one reuse existing schemas and controlled vocabularies? Different choices of serialization have different advantages and disadvantages and support different functionalities. Factors that are relevant for the decision-making are: how serialization choices for multiple implemented standards complement each other; available IT skills; scalability problems; response times; how indexed metadata and administrative metadata that are associated with individual digital objects support search of the content; and the impact of the ubiquity and popularity of the serialization type on its tool support and continued usability. Optimization potential can be important, for example, to avoid repeating shared information over and over. In addition, existing and planned storage solutions are closely linked to serialization choices and impact data management and the way metadata are created, read, updated and deleted.

Chapter 14 discusses how to make different metadata frameworks work together in an institutional ecosystem.

It was already mentioned that it is generally necessary to combine several metadata specifications in order to support the complete functionality of the system. Extension schemas let you embed content type or file format specific metadata within a more general, core metadata framework. Metadata container formats, such as METS [8], have the specific purpose of tying different metadata frame works together. And packaging container formats are intended to package multiple content objects together with their metadata into an aggregate archival file, such as WARC [9]. Domain-specific metadata schemas that are typically descriptive and are used in a specific context, such as for libraries or archives, can be used to complement core preservation metadata.

When combining different frameworks one has to consider how to deal with possible redundancies when the same information is repeated in several places, and how this may affect the scalability, processing, versioning, and synchronization of the redundant information.

Chapters 15 through 17 discuss tools and systems.

Digital asset repositories can be developed in-house from scratch; or they can be commercial or open-source systems that are customized to varying degrees; or they can be commercial services, possibly offered in the cloud. The choices are determined by a variety of factors: What degree of customizability is required? What amounts of digital objects need to be accommodated? How does the solution fit in with the existing infrastructure? How much in-house IT support is available? What are the specific characteristics of the digital objects, in terms of size, access, or viewer requirements? What cost models are most appropriate for the intended use patterns? All of these requirements may determine the choice of implementation solution. Typically the system architecture is modular and layered and several system solutions are combined to accommodate all the required functionality. This includes

- pre-ingest functions, such as virus checking and metadata creation, through metadata extracting characterization tools;
- preservation actions, such as the creation of derivative formats;
- workflow issues;
- authorization and authentication;
- asset management, including the management of persistent, unique identifiers, the creation, update or deletion of digital objects or their metadata, their versioning, and the assurance of their integrity;
- persistent storage in nonproprietary forms;
- the creation of indexes to support discovery or search;
- cross-walking between different metadata formats;
- access through browsing, search, facets, or appropriate visualizations; and finally the
- viewing or delivery of the digital object.

For all of these life-cycle functions various tools may be combined into one system. They may be supported by registries that share important information, such as

- file format registries that help identify the file types of the files in the repository;
- controlled vocabulary registries that share vocabulary for enumerative data types in the domain, such as checksum algorithms in use, or typical preservation event types;
- software registries that describe, or even preserve, the software that is needed to render or execute a file.

Many of these tools, systems, and registries implement PREMIS and are shared by the digital preservation community.

Chapter 18 discusses what it means to be conformant with PREMIS.

It was said above that, whenever possible, repository systems should produce metadata that conform with standards. Conformance and compliance are however not always straightforward, especially if there are requirements to keep a standard as flexible as possible, so that it is widely usable.

1.6 Conclusion

Practitioners are sometimes intimidated by the prospect of having to specify their metadata profile. In the end metadata exist in order to support the implementation of system functions. Their specification is a part of the overall system requirements and implementation process. Tools and the metadata that support them need to meet your needs. Even if you cannot foresee the future and cannot now know what metadata you will wish you would have kept, it is better to get started. With the help of established systems and standards you can create satisfactory solutions from which you can learn and continuously improve your organization's services. Additional support comes from community development, through user groups, mailing lists, implementation registers, and reusable profiles. There is a lively user community of people who are concerned about the long-term preservation of digital objects. Practitioners are not alone.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 24 Apr 2016
2. Dublin Core Metadata Initiative (DCMI) (2001) Dublin Core glossary. http://dublincore.org/documents/2001/04/12/usageguide/glossary.shtml#A. Accessed 24 Apr 2016
3. Consultative Committee for Space Data Systems (2012) Reference model for an Open Archival Information System (OAIS). CCSDS Secretariat CCSDS 650.0-M-2. Magenta Book, Washington, DC, Issue 2, June 2012
4. Caplan P (2008) The preservation of digital materials. Libr Technol Rep 44(2):9
5. Vermaaten A, Lavoie B, Caplan P (2012) Identifying threats to successful digital preservation: the SPOT model for risk assessment, D-Lib Magazine, September/October 2012. http://www.dlib.org/dlib/september12/vermaaten/09vermaaten.html. Accessed 24 Apr 2016
6. DCC (2015) DRAMBORA: Digital Repository Audit Method Based On Risk Assessment. Welcome to DRAMBORA interactive: log in or register to use the toolkit. http://www.repositoryaudit.eu/. Accessed 24 Apr 2016
7. Wikipedia (2015) Serialization. https://en.wikipedia.org/w/index.php?title=Serialization&oldid=715389105. Accessed 24 Apr 2016
8. Digital Library Federation (2010) <METS> Metadata Encoding and Transmission Standard: primer and reference manual, version 1.6 revised. http://www.loc.gov/standards/mets/METSPrimerRevised.pdf. Accessed 24 Apr 2016
9. ISO 28500:2009 (2016) Information and documentation—WARC file format, 2009. http://www.iso.org/iso/catalogue_detail.htm?csnumber=44717. Accessed 24 Apr 2016

Chapter 2
How to Develop a Digital Preservation Metadata Profile: Risk and Requirements Analysis

Sébastien Peyrard, Angela Dappert and Rebecca Squire Guenther

2.1 Introduction

In order to decide how to implement preservation metadata for digital objects in a specific situation, one needs to know what the specific requirements are. There is no off-the-shelf solution when implementing preservation metadata—even when using a standard, such as PREMIS [1]. PREMIS provides core information elements which are intended to accommodate a wide range of contexts, providing general implementation guidance. Therefore, PREMIS needs to be tailored to specific needs so that the implemented preservation metadata supports all relevant requirements, making the most appropriate decisions in a constrained context. Risk-oriented frameworks, such as the SPOT model [2], are efficient tools to start a requirements analysis for digital preservation metadata.

S. Peyrard (✉)
Bibliothèque nationale de France, Site François-Mitterrand,
Quai François-Mauriac, 75706 Paris Cedex 13, France
e-mail: sebastien.peyrard@bnf.fr

A. Dappert
The British Library, 96 Euston Rd, London NW1 2DB, UK
e-mail: angela.dappert@bl.uk

R.S. Guenther
Consultant (Formerly Library of Congress), 215 W. 75th St.,
New York, NY 10023, USA
e-mail: rguenther52@gmail.com

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_2

2.2 Why Define Requirements?

One needs to know one's requirements for many reasons.

- You need to know what purpose the metadata is going to achieve, that is, what functions in the preservation of a digital asset it supports. This determines what crucial, specific pieces of information are needed. The functions to consider include data management, access and preservation commitments, the mandate of your organization, and user needs.
- You will likely have other existing metadata formats. This may be MARC or EAD descriptive records in a library or in archives. You might already use METS to describe and manage digital objects in your online library, which includes some preservation metadata. Requirements should specify how your preservation metadata, probably PREMIS-based, will interoperate with those metadata formats, and how you will handle redundancies. Considerations about this are also provided in Chap. 14 of this book.
- If you have an operational digital preservation solution it may constrain the functions you would like to perform and influence the requirements you need to define for metadata solutions. If you are preparing to tender for one, you need to understand your requirements to help select the appropriate one, as off-the-shelf solutions may not be able to satisfy them. If you are in the definition phase for custom development, requirements are essential for shaping it. In any case, they impact cost and future extensibility of functions and architectural solutions.
- Digital preservation metadata does not presuppose use in complex automatic, customized IT solutions. It can document and support long-term preservation for manual processing or for piloting workflows.
- Different serializations (the form in which metadata is stored) have different strengths and support different use cases. For example, XML is human- and machine-readable; relational databases can be queried efficiently; and RDF files in an RDF store are well-suited to support linking with other metadata sets. Those choices can only be made if you know the use cases and underlying requirements and should therefore be made after you have determined how you want to handle metadata. This aspect is further discussed in Chap. 13 of this book.

2.3 Metadata Requirements Analysis as Part of the Digital Preservation System Specification

Recording metadata to safeguard your digital objects is only one possible answer to the multiple threats to digital objects. Therefore, most of the time, requirements analysis for defining preservation metadata is part of a wider requirements analysis for your digital preservation solution, and should not be performed as isolated work. The key questions you should ask yourself to define your digital preservation solution can be the following:

- What *functions and processes* will your organization support? Will they be manual, automated, or semi-automated? Thus, you need to define how you are going to implement Ingest, Storage, Data management, Preservation and Access, and what workflows and tools you are going to use for the overall systems management and organization, and to execute preservation plans. Outside the preservation system, you also have to define how your digital preservation solution will be integrated with other systems or activities within your institution, e.g., assets management tools, catalogs or finding aids, different acquisition and production workflows, and access tools.
- What are the *objects* that you want to preserve? Are they digitized or born-digital? Are they text, image, audio, video, databases, software? Are they publications or archive records? What is their legal status, and what is your institutional mandate to preserve them? The risks, requirements, and constraints may be different depending on the content type. Considerations about how PREMIS is applied to specific content types are discussed in Chaps. 5 to 9 of this book.
- What are the *risks* to your digital objects? What is your priority for mitigating them?
- At what level do you need to describe objects, events, agents, and rights involved in the domain so that they can provide information for the functions you need to support? How do they relate to each other? This analysis will result in the *data model* that will be implemented. Each entity needs to be described by an associated set of semantic units (properties) and relationships. Considerations about data modeling can be found in Chap. 4.

Such questions should help to answer which information you need to record about your digital objects so that they can be managed and preserved. Implementing preservation metadata is obviously focused on the preservation aspect; however, you also need to address other aspects:

- You will have to know how you will *manage the metadata*. Some relevant questions are: Will you import from existing descriptions in different formats or create the metadata from scratch? If it needs to be created, who is going to create it or will it be generated automatically? Will you index the metadata in a knowledge base where it can be queried? Will you store the metadata with the data objects of a particular Information Package? What metadata are you going to distribute or display with your objects when you disseminate them? And finally, how are you going to update your preservation descriptions? You also need to know how this metadata is going to interoperate with other kinds of metadata curated externally to your repository.
- The requirements, and the underlying data model for your metadata, should be as *generic* as possible to guarantee maximum maintainability over the long term, but *specific* enough to enable you to ingest, preserve, and disseminate them.
- Your institution will have limited human, IT, and financial resources. Therefore, not all metadata requirements may be feasible to implement right away. *Risk analysis* is powerful because it helps you *prioritize* your metadata requirements

in MoSCoW terms (Must, Should, Could, Would be nice[1]) so that you can know what to postpone or discard.

2.4 How to Get to Know Your Requirements

How are you going to define your requirements? You can draw on three important sources: principles, people, and documentation, and two key methods: analysis of your specific situation combined with best practice in the preservation domain.

2.4.1 Principles: Set Global Guidelines

First, you need to define a clear set of principles that will serve as guidance for the requirements definition. Here are some key principles for any digital preservation project that can be applied to digital objects and are equally important for their preservation metadata:

- *Understandability*. Being able to render and understand the primary digital objects is the key goal of the OAIS model [3] and of digital preservation in general. This also applies to metadata and the need to understand it over time. When something can be stated simply, keep it *simple*. Most of the time, a preservation action needs to gather sets of *information packages* that share common characteristics. This means your data model and metadata statements should be *consistent*, at a particular point in time and over time. Whether you use in-house or shared publicly available codes or controlled vocabularies, you need to record precisely what a particular code or value means so that it can be interpreted later. In other terms, *document*!
- *Reversibility*. In digital preservation, whenever possible in a given context, it is recommended to provide the ability to reverse to a previous state of your information package. This means that you need to document the events that have altered your information package and keep earlier versions of digital objects (unless they can be reliably and losslessly reconstructed from the event information). This takes us back to the *traceability* principle discussed earlier.
- Because you have to migrate data from obsolete systems to more current ones you have to define requirements that are *technology independent*, suggesting the use of and compliance with open standards. The principle is that the data is there forever, but the system that manages it evolves over time. As an open, technically- and domain-neutral standard PREMIS achieves this for your metadata:

[1]This method is associated with Rapid Application Development and agile development methods in general, but can be applied to other contexts, including this one. See [4].

Fig. 2.1 Preservation goals for successful digital preservation

you can implement it in different ways and remain PREMIS compliant.[2] What is more, PREMIS can be used as a common language between different preservation systems, at a point in time (interoperable repositories) and over time (moving from one preservation solution to another one).

- Further principles apply to the preservation of the digital objects in your custody, as illustrated by Fig. 2.1. These principles are discussed by Caplan [5]. Metadata is intended to support those preservation goals.

2.4.2 People: Ask the Experts

You need to identify the stakeholders of your OAIS repository. Who are the Producers,[3] who produce or collect the digital assets that you are to preserve, the Consumers, the repository owners and managers? Those persons are experts in their domains: they are the ones who define the original intent for the digital objects that you preserve, what actions and information will be necessary to manage and preserve them, and what access and use requirements will need to be satisfied. They are the key persons to help you define what you need to record and the key pieces of information that one needs in order to use the objects. Your role is to understand whether such information is relevant for preservation. During meetings, brainstorming sessions or the like, you may want to obtain answers to questions such as "If you irreparably lose certain information (metadata) about this digital object,

[2]See Chap. 18 of this book for considerations about PREMIS conformance.

[3]Producer: "The role played by those persons or client systems that provide the information to be preserved. This can include other OAISes or internal OAIS persons or systems." OAIS, page 1–14, part 1.7.2: Terminology.

what would happen?" Such questions help to prioritize among different metadata requirements. What is more, they will improve your own awareness of the usefulness of metadata for the long-term preservation and management of the objects. If some of those key metadata elements are missing right now, this will be a good way to convince the stakeholders to incorporate their creation into their own business processes. You may also find yourself in the typical preservation situation where little has been documented formally and is only known to stakeholders, or the documentation is hard to find as it is buried in folders scattered across different computer systems. At the time of metadata requirements analysis you need to document what is missing; gather the documentation while it exists, and raise awareness among your coworkers.

2.4.3 *Best Practice: Implement the State-of-the-Art in Digital Preservation*

A metadata standard, such as PREMIS, is not an aim in itself. It is a means to preserving collections in the best way possible, given your constraints. But even though you will not implement every entity or semantic unit defined in PREMIS you can use PREMIS to help you define your own requirements. The PREMIS Data Dictionary can be particularly useful if you do not have existing digital preservation metadata expertise in your institution as it implicitly suggests high-level requirements that one can adapt to one's specific context. It has preidentified, through community consensus, what information is expected to be relevant for digital preservation by summarizing state-of-the-art core preservation metadata; it documents rationales for why they have been added as core metadata and suggests contexts in which they should be implemented. Its set of semantic units can be used as a checklist. For example, you can check the cardinalities of each semantic unit: if PREMIS states that a semantic unit is mandatory, it means that the information should be available somewhere in your repository even if it is not explicitly recorded as metadata accompanying the digital object. If it is optional, it means it may be dispensable in some contexts—but might be vital in others.

Similarly, the OAIS framework defines shared requirements for preservation repositories as well as an underlying information model.

But, because they are context-agnostic, neither the OAIS model nor the PREMIS Data Dictionary provides you with a ready-to-use metadata specification. This means that you need to tailor them to your specific context. To understand your context and to tailor OAIS and PREMIS to your situation, risk-driven analysis proves to be very useful. DRAMBORA [6][4] specializes the general risk management framework of ISO 31000:2009 [7] for the digital preservation context. It is a perfect complement for tailoring the OAIS information model to your specifics.

[4]DRAMBORA: Digital Repository Audit Method Based On Risk Assessment.

Both DRAMBORA and the OAIS model apply to the whole domain of preservation repositories and go beyond a preservation-specific information model. The SPOT model for Risk Assessment[5] [2] is more suited to metadata requirements analysis. SPOT focuses on the digital objects themselves, for which it identifies properties for successful digital preservation. This can be a good entry point for preservation metadata requirements analysis and for tailoring PREMIS to specific needs. The OAIS information model and the SPOT model are further discussed in the following:

2.5 Two Reference Frameworks to Get Started: OAIS Information Model, SPOT Risk Assessment Model

2.5.1 The OAIS Information Model

The OAIS information model defines, at a high-level, the categories of metadata needed to preserve digital objects. This list of information types is basic to any activity involving digital preservation metadata in general. It can be used as a core checklist to define metadata requirements

- *Representation Information*: "The information that maps a Data Object into more meaningful concepts."[6] It comprises all information that is needed to make sense of the digital object, including its file format, technical characteristics, documentation that helps to understand its content, or the software that is required to render it.
- *Fixity Information*: "The information which documents the mechanisms that ensure that the Content Information object has not been altered in an undocumented manner."[7] It captures metrics of the physical state of the object at the point of ingest into the repository. They are recalculated periodically and compared against the original values to ensure that no inadvertent changes have taken place. The size of a digital object in bytes and its checksums are examples of fixity information.
- *Reference Information*: "The information that is used as an identifier for the Content Information.[8] It also includes identifiers that allow outside systems to refer unambiguously to a particular Content Information."[9] Reference Information is about identifiers for the PREMIS *Representations*, *Files* and *Bitstreams* (objects directly handled by the preservation repository) and

[5]Simple Property-Oriented Threat Model for Risk Assessment.

[6]OAIS, page 1–14, part 1.7.2: Terminology.

[7]OAIS, page 1–11, part 1.7.2: Terminology.

[8]Content information is the primary information that is the target of preservation. It consists of the digital object together with its Representation Information.

[9]OAIS, page 1–14, part 1.7.2: Terminology.

PREMIS *Intellectual Entities* (descriptive information generally handled outside the preservation repository).

- *Provenance Information*: "The information that documents the history of the Content Information."[10] Examples of provenance information are operations (events) that were performed on the digital object before and upon ingest into the preservation repository, and information that helps to understand the transfers of responsibility for the digital object across different actors over time. It helps users to understand the state of the digital object when they access it. Once digital objects are ingested into your repository, you need to record any significant action that you perform upon them that has an impact on their form or primary content (e.g., migration) or on its metadata (e.g., addition or update of the Representation Information). Approved current best practice might be considered bad practice some decades from now: people preserving your digital objects in the future will need to know precisely what actions you took on your objects, to make the best informed curatorial decisions. To that purpose it is also useful to preserve policies that applied at the time.
- *Context Information*: "The information that documents the relationships of the Content Information to its environment. This includes why the Content Information was created and how it relates to other Content Information objects." Preservation policies are typical examples of Context Information. Depending on the context, this can be repository-level policies or policies that are differentiated for different content types and intended preservation strategies. PREMIS, with few exceptions, does not cover business rules and policies, although they are essential for preservation planning and for interpreting the historical context of the objects' curation.
- *Access Rights Information*: "The information that identifies the access restrictions pertaining to the Content Information, including the legal framework, licensing terms, and access control." Examples of such information can include the rights pertaining to the digital object; and statutes, copyright terms, or licenses negotiated with rights holders that affect which preservation actions the organization is allowed to perform on the digital object.

All of these information types help to achieve the goal of a preservation repository, which is to ensure that the content remains accessible and understandable to a designated community. All of them are captured in PREMIS, but for requirements analysis purposes it may be beneficial to step back and consider the purpose that the metadata has to fulfill.

[10]OAIS, page 1–14, part 1.7.2: Terminology.

2.5.2 *The Core Aspects that Mitigate Risks on Digital Objects: The SPOT Model*

The SPOT (Single Property-Oriented Threat) Model is a risk assessment framework that allows you to identify threats to digital assets through a very straightforward methodology. Similar to the preservation goals depicted in Fig. 2.1, SPOT is structured along six main facets that should be considered when preserving digital objects: *availability*, *identity*, *persistence*, *renderability*, *understandability*, and *authenticity* of the digital object.

Using the SPOT framework one can translate the generic OAIS information categories introduced above into the concrete, implementable semantic units that make up the PREMIS Data Dictionary. Used in this way, SPOT is a good framework for refining preservation metadata requirements. Digital preservation is concerned with mitigating risks for digital objects which is partially, but significantly, achieved through recording knowledge about the object. SPOT helps digital curators ask important questions. Each set of properties can be viewed as a set of questions to ask yourself concerning a particular digital object you want to preserve, helping you identify *threats* to those properties, and identifying metadata elements needed to mitigate the threats.

- Availability: Is my digital object under my control, e.g., ingested in my repository and can it be disseminated to end-users? Do I have the right to preserve it?
- Identity: Can I uniquely and persistently find this digital object in my repository and distinguish it from similar objects?
- Renderability: Can the content of the object be displayed or executed in a way that does not alter it significantly?
- Understandability: Can the content of the object be understood and used by the designated community?
- Authenticity: Is there evidence that the digital object is what it purports to be, and what its chain of custody has been?

2.6 Key Questions at the Level of Each Metadata Element

Once you have identified a list of metadata elements considered important for each of the entities identified in your data model, you should answer the following questions for each one:

- Is the information already available somewhere, or do I have to generate it? In some cases, the information cannot be reproduced outside the context in which it initially became available. This means it may be crucial to capture it as early as possible or at a point at which it is still known. This may be at the time of

creation, before ingest into a repository, or at the time of a preservation action. Examples are operations performed on the digital object prior to ingest, or checksums that have been calculated when the digital object was initially produced.

- What information about the digital objects is especially important to know? In most cases, this will concern Representation Information. It may, for a particular type of content (audio, video…), need to be recorded at a finer grain than the high-level semantic units in PREMIS.
- Is there potential for optimizing the implementation to, for example, avoid duplication or to improve access speed to frequently required metadata? Is the information shared by multiple instances? For example, have the checksums for many files been calculated in the same batch event linking to the same software agent description? Can preservation policies be recorded at a single place where they can be factored out for all relevant digital objects? When objects are preserved in a standard file format for which many interoperable rendering environments are available, one may choose to rely on external sources about the rendering software. For example, one may rely on external registries for metadata about it, and on backward-compatible broadly available rendering software. If the object is preserved in a proprietary format or has a complex behavior, then recording information about the software used to create or render the object, and even preserving it as an environment object, may be crucial.
- If you use multiple metadata standards that have overlapping metadata information, do you store this information in PREMIS or another metadata format, or both?

2.7 Conclusion

Non-standard digital preservation scenarios or those that need to customize standard solutions cannot rely on out-of-the-box profiles. People sometimes feel overwhelmed when they start defining digital preservation requirements in these situations. The long-term goal can be perceived as very abstract. Requirements analysis is essential to tailoring digital preservation best practice solutions to specific circumstances. This chapter proposes important questions that help to break down the task into more manageable subtasks.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 29 Dec 2015

2. Vermaaten A, Lavoie B, Caplan P (2012) Identifying threats to successful digital preservation: the SPOT model for risk assessment, D-Lib Magazine, September/October 2012. doi:10.1045/september2012-vermaaten
3. CCSDS (2012) Reference model for an Open Archival Information System (OAIS). CCSDS 650.0-M-2. Magenta Book, Issue 2, June 2012
4. Clegg D, Barker R (2004) Case method fast-track: a RAD approach. Addison-Wesley, New York
5. Caplan P (2008) The preservation of digital materials. Libr Technol Rep 44(2):9
6. McHugh A, Ross S, Ruusalepp R, Hofman H (2007) The digital repository audit method based on risk assessment (DRAMBORA). Digital Curation Centre. Welcome to DRAMBORA interactive: log in or register to use the toolkit. http://www.repositoryaudit.eu/. Accessed 1 Feb 2016
7. Leitch M (2010) ISO 31000:2009—the new international standard on risk management. Risk Anal 30:887–892. doi:10.1111/j.1539-6924.2010.01397.x

Chapter 3
An Introduction to the PREMIS Data Dictionary for Digital Preservation Metadata

Rebecca Squire Guenther, Angela Dappert and Sébastien Peyrard

3.1 Introduction

Digital objects are not viewable on a shelf. Because of their nature they require an intermediary device to use and understand them and are dependent on the physical medium on which they are stored. Metadata is one key to discovery, access, management, and preservation. Enter the PREMIS Data Dictionary for Preservation Metadata, which tells you what you need to know to preserve your digital material for the long term—and thereby standardizes what we call preservation metadata.

Convened by OCLC and the Research Libraries Group in the U.S., the PREMIS Working Group developed the *PREMIS Data Dictionary for Preservation Metadata*, which was based on a deep pool of institutional experiences in setting up and managing operational capacity for digital preservation. Since its initial release in 2005, it has undergone several revisions that included changes and enhancements informed by experimentation in implementing preservation metadata in digital repositories. It has since become a mature and widely implemented specification and a key piece of the digital preservation infrastructure.

Chapter 2 details methodologies for determining what metadata is needed based on a requirements analysis and categorizes the types of information needed to

R.S. Guenther (✉)
Consultant (Formerly Library of Congress), 215 W. 75th St.,
New York, NY 10023, USA
e-mail: rguenther52@gmail.com

A. Dappert
The British Library, 96 Euston Rd, London NW1 2DB, UK
e-mail: angela.dappert@bl.uk

S. Peyrard
Bibliothèque nationale de France, Site François-Mitterrand,
Quai François-Mauriac, 75706 Paris Cedex 13, France
e-mail: sebastien.peyrard@bnf.fr

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_3

mitigate preservation risks. This chapter introduces the *PREMIS Data Dictionary for Preservation Metadata* (referred to as "PREMIS") [1] and how it may be used as a common core set of metadata elements, i.e., the key things you want to say about your digital objects so that they can be preserved for the long term. Using an agreed-upon standard benefits all those involved in the preservation process. PREMIS attempts to provide a mechanism to minimize the threats to preserving your important digital assets.

3.2 The PREMIS Data Dictionary

The PREMIS Working Group built on the work of an earlier group, the Preservation Metadata Framework Working Group (PMF), which based its 2002 report [2] on the Open Archival Information System's (OAIS) Information and Reference model, informed by previous work on preservation metadata schemes. OCLC and RLG established a new working group in 2002: the Preservation Metadata: Implementation Strategies (PREMIS) Working Group. The membership included experts who were active in the field of digital preservation in a variety of types of institutions (including libraries, archives, and museums) and countries.

The *Data Dictionary for Preservation Metadata: Final Report of the PREMIS Working Group* (widely known simply as "PREMIS") gained wide attention after its release in May of 2005 as version 1.0 [3], because many institutions investigating their readiness for preserving their digital objects saw a critical need for this sort of specification in helping them to set up viable preservation implementation policies and strategies. It quickly became the de facto standard for preservation metadata.

Version 2.0 of the PREMIS Data Dictionary followed in 2008 [4]. A change in number indicates substantial revisions that affect existing PREMIS metadata as opposed to incremental version changes (e.g., version 2.1 [5], 2.2 [6]) that include additional semantic units or minor changes. Version 2 (2.0 issued in 2008, with subsequent revisions in 2011 and 2012) greatly enhanced the ability to provide metadata for expressing rights information. Version 3.0 [1], issued in June 2015, revised the data model to include *Intellectual Entities* as another level of Object and to be able to more fully describe environments. All of the changes that have been incorporated into later versions of the Data Dictionary have been based on implementation experience using PREMIS and demonstrated use cases.

Implementation experience will be crucial to evaluating the changes and whether they result in a more effective mechanism to understand metadata needed to use digital objects in the future. When we talk about digital preservation in general, it is difficult to prove what is the "right" way to do something, since it is only after we "commit an act of preservation" that we can make an evaluation—and at that point it could be too late to save what we were attempting to preserve.

3.3 The PREMIS Maintenance Activity

After the initial release of the PREMIS Data Dictionary in 2005, the Library of Congress established a permanent Web presence and maintenance activity responsible for maintaining, supporting, promoting, and coordinating future revisions. The activity consists of a Managing Agency (the Library of Congress) which handles inquiries, hosts the website [7], maintains the discussion lists and generally supports the activity; the PREMIS Editorial Committee [8], which sets directions and priorities and coordinates and approves revisions to the specification; and the PREMIS Implementers' Group, which discusses implementation experiences, proposes changes, and solicits clarifications.

The PREMIS Editorial Committee has broad membership across different types of institutions and internationally (as of this writing from eight different countries). It consists of members who are implementing PREMIS in their institutions and software products; they act as information providers about the standard, take a leadership role in their countries in using and implementing PREMIS, and participate in monthly meetings concerning maintenance and revision of the Data Dictionary and accompanying documentation. The Committee also forms Working Groups on special topics as needed and this has resulted in supplementary documentation. Recent working groups have included PREMIS Conformance, Environment, and the PREMIS OWL ontology working groups. Members of working groups provide expertise in specific areas and do not have to be members of the Editorial Committee.

Tutorials have been given in a number of countries throughout the world, sometimes in conjunction with other related preservation events. More focused discussion for implementers, take place in a series of workshops called PREMIS Implementation Fairs [9]. These have taken place annually since 2009 in conjunction with the International Conference on the Preservation of Digital Objects (iPres) and have resulted in discussion leading to revisions of the Data Dictionary, clarification of implementation approaches, and the formation of working groups on special issues. This annual event gives implementers a chance to compare experiences, discuss problems and solutions, and get updates on future work.

The PREMIS Maintenance Activity website maintains an Implementation Registry [10], based on information provided by PREMIS implementers. Although not comprehensive (since it depends on institutions submitting the information to the Library of Congress), the listings include information on the types of content being preserved, relevant dates, implementation details such as type of repository software used, tools developed, PREMIS entities supported and examples of use.

The PREMIS Working Group received the 2005 Digital Preservation Award (awarded by the Digital Preservation Coalition in the UK and part of the Conservation Awards) [11] for its work on the *Data Dictionary* and the following year the Society of American Archivists awarded it the 2006 Preservation Publication Award. In addition it was on the short list for the 2012 Digital Preservation award *for the most outstanding contribution to digital preservation in the last decade* [12].

3.4 OAIS and PREMIS

The first edition of the *Reference Model for an Open Archival Information System (OAIS)* was developed by the Consultative Committee for Space Data Systems (CCSDS) in cooperation with the International Organization for Standardization (ISO) between 1997 and 2000 [13, 15]. Its purpose was to provide a standard framework for defining the functional components and information needed for long-term preservation of digital objects. Its second edition was released as ISO standard 14721:2012 in 2012 [14] and was widely accepted as the standard that all digital repositories needed to follow. Chapter 2 on Methodology discusses the categories of information in the OAIS information model and how they relate to PREMIS and to the common goal of mitigating preservation risks. PREMIS was developed with OAIS as its context and with the assumption that digital preservation repositories will comply with the functionality and information that OAIS specifies. However, it goes beyond OAIS in providing discrete pieces of information that you need in order to preserve your digital objects, while OAIS provides broader categories of such information. Additionally, PREMIS allows for recording information about digital objects that occur prior to ingest in the repository, which is not covered in OAIS. In PREMIS version 3.0 with the revision of the data model and the way you record information about software and hardware environments that support the rendering of digital objects, PREMIS now also covers the description of physical objects such as hardware or physical copies of data. In other words, although influenced strongly by OAIS, PREMIS provides key pieces of information that cover the whole life-cycle of digital objects that are going beyond the scope of the preservation repository.

3.5 PREMIS Data Model

The PREMIS Data Dictionary [1] defines a data model as a tool for thinking about the information needed to achieve digital preservation goals. That data model defines what things need to be described (the PREMIS entities) and what you need to say about them (the PREMIS semantic units). It also tells you how different PREMIS entities relate to one another (relationships). The Data Dictionary is organized around the data model and implementations, such as the associated default XML schema, can be conveniently structured following it.

The core entities include Objects, Events, Agents, and Rights Statements.

Objects are the target of preservation activities and what the repository actually preserves. Objects have four levels of description

- *Representation*: the set of *Files*, including structural metadata, needed for a complete rendition of an *Intellectual Entity*
- *File*: a named and ordered sequence of bytes that is known to an operating system

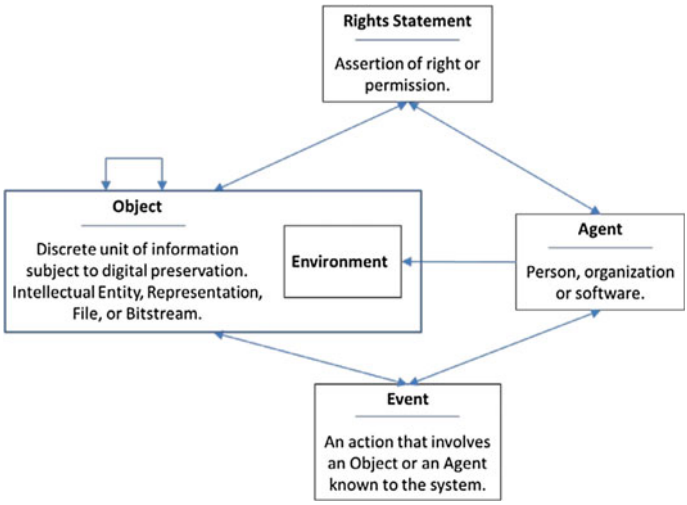


Fig. 3.1 PREMIS data model version 3. *Source* [1: p. 6]

- *Bitstream*: contiguous or non-contiguous data within a *File* that has meaningful common properties for preservation purposes
- *Intellectual Entity*: a distinct intellectual or artistic creation that is considered relevant to a designated community in the context of digital preservation. It could be an abstract concept (e.g., a work in terms of the Functional Requirements for Bibliographic Records entity-relationship model or FRBR [16]), an information resource affixed to a particular carrier (a manifestation in terms of the FRBR model), or it could be represented by descriptive metadata that serves as a surrogate for the *Intellectual Entity*. *Intellectual Entities* were independent Entities in the early PREMIS versions and became a fourth object category starting with version 3 of the Data Dictionary.

In version 3 all the semantic units that are applicable to *Representations* are also applicable to *Intellectual Entities*. *Intellectual Entities* also offer semantic units that describe computing environments.[1]

The PREMIS Data Model in version 3 can be summarized in Fig. 3.1.

3.6 PREMIS Goals and Principles

PREMIS was developed to provide the community with a common data model for organizing and thinking about preservation metadata. The challenges in preserving digital objects given changing technological environment may be overwhelming for

[1]See Chap. 10 for more on how environments are handled in PREMIS 3.

anyone with the responsibility to preserve and manage digital objects, and it gives specific guidance for implementations. PREMIS is based on and has evolved with the contemporary understanding of digital preservation best practice over time. PREMIS can provide a blueprint for the information which is needed to prevent the inability to use and understand our valuable information resources into the future. As a shared community standard it provides for interoperability among repositories of digital objects, systems that support the preservation process, and data that is exchanged and reused such that institutions do not have to define their own metadata specifications.

3.6.1 Scope

There are some limitations to the scope of what PREMIS covers. In order to be flexible and applicable in a variety of contexts, the Data Dictionary does not address all the information that is needed for long-term preservation. It has little information about business rules and institutional policies, which is a key to local implementations. Decisions need to be made about issues such as levels of preservation, how to collect or generate metadata, which identifier systems to use, what preservation strategies might be employed, where metadata is stored, etc. These become part of the day-to-day workings of an institutions' preservation repository. However, PREMIS has minimal information about an institution's preservation rules or policies, only including information about levels at which objects are preserved (e.g., bit-level, full level), even though a preservation repository needs to keep information about its policies for carrying out its responsibilities during the life cycle of the object. Another limitation in scope is that the technical metadata, i.e., intrinsic technical characteristics about the object, often related to its encoding or file format, only includes what applies to all or most format types. PREMIS does not suggest that format-specific technical metadata (e.g., technical characteristics that pertain to still images or moving images but not to other format types) is not important or core for preservation, but often format experts must be relied upon to define these and remain current with emerging file formats.

3.6.2 Free and Open

PREMIS is intended to be a community resource which is free and open for anyone to use who is preserving digital objects. It provides guidance to persons and institutions who are establishing or managing a repository of digital objects that need to be preserved for the long term. As such, anyone can use the specifications and in doing so, others will understand the same language and be able to exchange information in a common way.

3.6.3 Technical Neutrality

PREMIS tells you the information you need to know to preserve your digital objects but does not specify how you know it or what encoding you will use to express it. It uses the term "semantic unit" for a piece of information you need to know, rather than the term "metadata element," which implies a defined way of representing that information in a metadata record. Thus the information it tells you to record is independent of any particular technology or system and may be used in a variety of contexts and implementations. It should be understood as defining the pieces of information that are needed to achieve preservation goals; in order to meet these goals it requires implementing a method for expressing that information in a particular system that provides the functionality needed for preservation. There is considerable detail in the Data Dictionary about the semantic units so that there is sufficient guidance for such implementations.

3.6.4 Extensibility

Because PREMIS is designed to be flexible in how it is implemented, but also has limitations in its scope, extensibility is a key principle in terms of meeting its goals. Some PREMIS semantic units are defined as extensions that allow for using non-core or more granular metadata along with what PREMIS provides. In these cases, a container is defined (its name ends in "extension," e.g., *rightsExtension*) where implementations may include either local elements or elements from another metadata standard. This is particularly important in areas that are complex. For instance, *significantPropertiesExtension* allows for extending the ability to say what significant characteristics need to be maintained through preservation actions, information that may be subjective and need more or less detail. This is also an area where considerable research has been done, but standards to express it have not yet emerged. Since the scope of PREMIS does not include format-specific technical metadata, *objectCharacteristicsExtension* provides a place to plug in that information following a local extension model or external metadata standard. The PREMIS semantic units that allow for extensibility are summarized in the PREMIS Data Dictionary [1: pp. 27–29].

The extensibility principle means that PREMIS is often combined with other standards to cover complimentary functionalities that are supported by different standards. Since PREMIS does not cover the management of metadata or a mechanism to package together metadata and content beyond the use of identifiers, container standards such as the Metadata Encoding and Transmission Standard (METS) [17] are frequently used to create an OAIS-compliant information package. Thus, when METS is used as a container to include the digital objects or links to those objects along with their associated descriptive, technical, digital provenance, source and rights metadata it can be considered a Submission, Archival or

Dissemination Information Package. As such this provides a standard for interoperability between preservation repositories. At the same time, METS includes a rich mechanism for defining the structure of the object in question, i.e., the structural metadata that is crucial to understanding the content that is being preserved and how the parts fit into the whole.[2] Because of its flexibility and technical neutrality, PREMIS may also be used with other container formats, such as the MPEG 21 Digital Item Declaration [18].

3.6.5 Degrees of Freedom

As a result of the flexibility and technical neutrality built into PREMIS, the PREMIS Editorial Committee issued a statement of conformance in October 2010, which was revised in April 2015. The statement explained what PREMIS implementers were free to adapt in their particular implementations and how to do this while still remaining conformant. These include the freedom to use different semantic unit names than is specified in the Data Dictionary; the freedom to change the repeatability, obligation, or applicability of a semantic unit to make it more stringent (but not more relaxed); the freedom to implement only those levels of object that it wants to control; the freedom not to record a mandatory semantic unit within a record as long as it can be obtained from within the repository system; and, the freedom to enhance or extend PREMIS semantic units that are extensible. The statement of conformance is discussed further in Chap. 18.

3.7 Semantic Units

As mentioned above, PREMIS uses the term "semantic unit" for a discrete piece of information that a preservation repository needs to know. This is what you say about the entities in the PREMIS Data Dictionary. A "semantic unit container" bundles together different related pieces of information; a semantic unit container does not take a value, but its component semantic units do. There may be containers within containers as necessary to associate different pieces of information together. Note that semantic unit containers are used for organizing related semantic units, but it is not intended that a hierarchical implementation such as XML must be used to retain those containers. The Data Dictionary gives sufficient guidance for a repository to define semantic units as metadata elements in an implementation (see example in Fig. 3.2), including the following:

[2]For more on combining PREMIS with METS, see Chap. 14.

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.		

Fig. 3.2 Example of a semantic unit from the PREMIS Data Dictionary version 3 [1: p. 68]

- Whether it is a container for nested semantic units or a component that takes a value.
- A definition and rationale for why it is included.
- What type of value it takes (e.g., string, controlled vocabulary, integer).
- What levels of objects it applies to (for the Object entity).
- Whether it is mandatory or optional; repeatable or not repeatable.
- Examples.
- Where the values might come from.
- Any additional guidance on usage.

The emphasis is on automatic generation of values based on metadata extraction from the objects themselves or from their production systems, using automated tools (such as format characterization tools), and using defaults that are rules or policies of a repository. This expectation is what makes PREMIS "implementable," since hand crafted metadata creation is expensive and impractical given the number

of digital objects we collect and preserve. In addition, best practice is to store metadata that are needed to support repository management functions; other metadata may be extracted on demand.

The Data Dictionary is organized around the core entities in the data model, i.e., Objects, Events, Agents, and Rights Statements. Links between entities are made through linking identifiers, e.g., a link from an Event to the Agent responsible for it (*linkingAgentIdentifier*). The Object entity aggregates information about the intrinsic technical characteristics of an Object (as defined above under Data model), so includes information such as size, format, significant properties, fixity, etc. The Event entity aggregates information on preservation related actions performed on one or more Objects, such as event type, date and time, outcomes, etc. The Agent entity, although not fully described in the Data Dictionary, aggregates information about agents responsible for preservation events or rights management in the life cycle of an object. Agents include persons, organizations, software or hardware. The Rights entity aggregates information about rights (entitlements granted by law, including copyright) and permissions (entitlements granted by agreements between a rights holder and another party) in relation to objects in a repository. Examples are the basis that is asserted for the right, specific actions that are allowed or restricted, applicable dates, and supporting documentation.

Figure 3.2 illustrates an entry for a semantic unit in the Data Dictionary.

Many semantic units specify that the value should be taken from a controlled vocabulary, since predictable values allow for systems to automatically act on data (i.e., to carry out a preservation function) and facilitate the generation of metadata values. The Library of Congress makes available controlled vocabularies for use with PREMIS semantic units except for those that are applicable only in a local context [19]. Consequently, in the Data Dictionary the semantic units specifying use of a controlled vocabulary give a few examples and reference the applicable vocabulary in the LC system. In providing the terms in these vocabularies as Linked Open Data, the vocabularies are made widely available so that preservation repositories do not all have to create their own lists of controlled vocabulary terms. In addition, as Linked data the vocabularies could be used with other Linked Data compatible metadata specifications, e.g. the Unified Digital Format Registry (UDFR) [20]. As a byproduct of its exposure as Linked Data, each vocabulary as well as each term in the vocabulary has a URI that identifies it, enabling linking and referencing on the Web. However, implementations are free to use their own vocabularies if desired. An example of a controlled vocabulary term is illustrated in Fig. 3.3.

3.8 Mapping Preservation Goals to OAIS and PREMIS Semantic Units

As described in Chap. 2 on risk and requirements analysis the pieces of information that repositories need to know to fulfill their functions in preserving digital objects may be mapped to preservation goals and identified threats. The SPOT model [21]

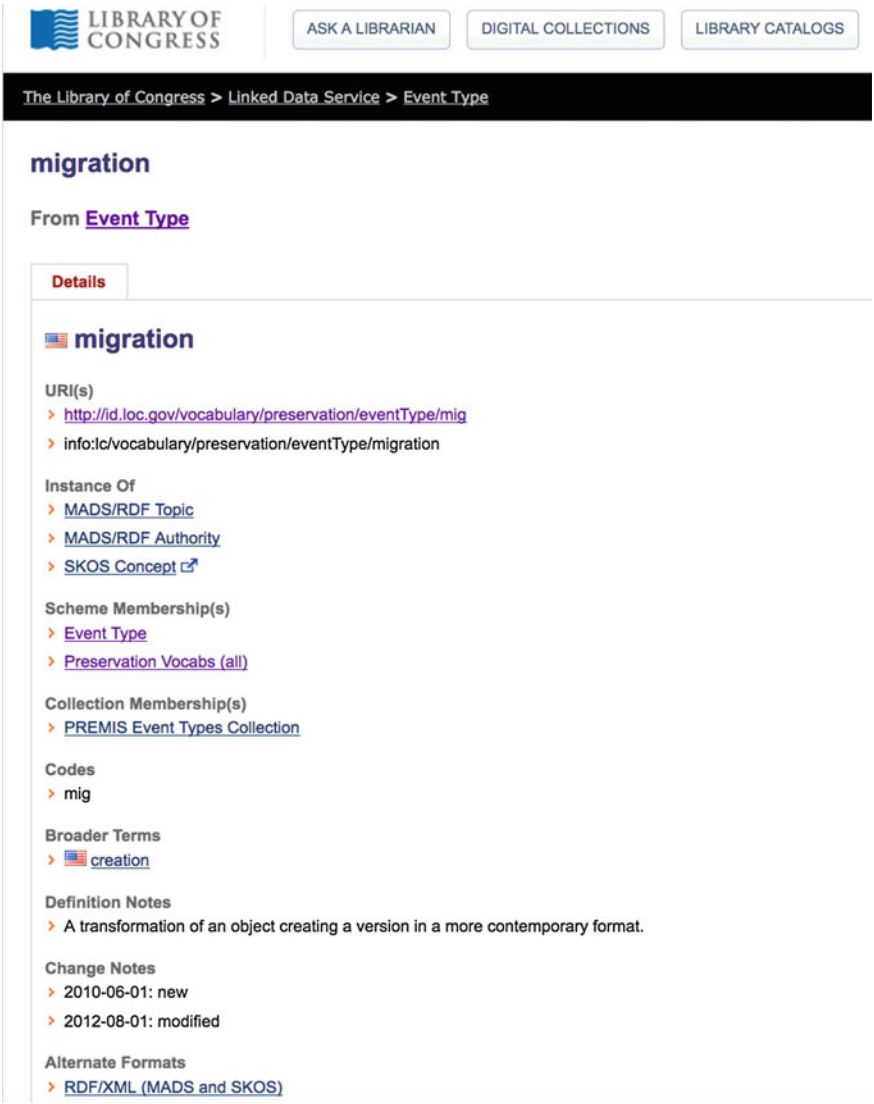


Fig. 3.3 Controlled vocabulary term "migration" from event type vocabulary at http://id.loc.gov/preservationdescriptions/

introduced in Chap. 2 uses categories of threats to digital objects, such as threats to availability, identity, persistence, renderability, understandability, and authenticity. These may also be understood in terms of the categories of information included in the OAIS Information and Reference Model and likewise mapped to PREMIS semantic units. Figure 3.4 illustrates these relationships between preservation goals, OAIS information categories, and PREMIS semantic units.

SPOT property	OAIS information	PREMIS entities and semantic units
Availability	Reference Information Context information Access Rights Information	Rights entity, and Agents related to Rights Object entity: *objectIdentifier*, *storage*, *preservationLevel*
Identity Persistence	Reference Information Fixity information	Object entity: *objectIdentifier*, *originalName* Events that change the form of the object or check its integrity (e.g. fixity checks, format and media migration, and Agents related to these Events
	Reference information Provenance Information	Object entity: *preservationLevel*, *storage*, *fixity*, *size*, *format* Environment information
Renderability Understandability	Structural and Other Representation Information Provenance Information Semantic Representation Information Transformational Information Property	Object entity: *preservationLevel*, *format*, *environment*, *creatingApplication*, structural *relationships*, *objectCharacteristics* Events that change the format of the object Rights granted for preservation actions Object: *relatedObjectIdentifier*: points to a description of a related *Intellectual Entity* *significantProperties* external descriptive metadata for *Intellectual Entity* Object
Authenticity	Fixity Information	Object entity: *objectCharacteristics*, esp. *size* and *fixity*, *signatureInformation*

Fig. 3.4 Requirements-driven PREMIS metadata. Inspired from PHC pilot report [22]

3.9 Conclusion

PREMIS defines, to the best of our shared current knowledge, what institutions need to know for preservation repositories to perform their preservation functions. It is impossible to predict the future and exactly how technology will change to affect our use of digital data. There is no way to prove that the information being provided now will in fact protect against loss of those functions which define our preservation goals: authenticity, renderability, viability, fixity, understandability, identity, and availability. However, as more institutions have implemented preservation repositories using PREMIS as the standard for the metadata they need

to know, there has been a convergence around it as a key piece of the preservation infrastructure. It has become the de facto standard for preservation metadata, institutions have requested more stringent methods for asserting conformance to its specifications, and it is built into many open source and commercial preservation repository systems. Version 3 introduces substantial changes that have broadened its scope to describing *Intellectual Entities* and provides a powerful means to describe software and hardware environments, which are so necessary for long-term preservation and use of digital objects. The latter will have particular benefits for complex types of objects that have multiple components and require a combination of platforms for future rendering and use, especially for multimedia objects.

It is likely that the PREMIS OWL ontology (under revision as of this writing) will be used more widely in the future[3] given the growth in making data available as Linked Open Data and other tools and metadata being made available in this way (e.g., Unified Digital Formats Registry, LC's Library of Congress Linked Data Service for Authorities and Vocabularies, repository tools such as Fedora 4, Linked Data compatible bibliographic data). As the Semantic web grows preservation metadata will need to interact more closely with other forms of metadata.

The PREMIS Data Dictionary provides a comprehensive, widely implemented and accepted specification that is revised based on concrete experience and changing technological environments. The subsequent chapters of this book detail implementation experiences for different types of content and in different systems with varying tools, how PREMIS is integrated with other standards, and what constitutes conformance.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 05 Jan 2016
2. OCLC/RLG Working Group on Preservation Metadata (2002) Preservation metadata and the OAIS information model: a metadata framework to support the preservation of digital objects. http://www.oclc.org/content/dam/research/activities/pmwg/pm_framework.pdf. Accessed 05 Jan 2016
3. PREMIS Working Group (2005) Data dictionary for preservation metadata version 1.0. http://www.loc.gov/standards/premis/v1/premis-dd_1.0_2005_May.pdf. Accessed 05 Jan 2016
4. PREMIS Editorial Committee (2008) PREMIS data dictionary for preservation metadata version 2.0. http://www.loc.gov/standards/premis/v2/premis-2-0.pdf. Accessed 05 Jan 2016
5. PREMIS Editorial Committee (2011) PREMIS data dictionary for preservation metadata version 2.1. http://www.loc.gov/standards/premis/v2/premis-2-1.pdf. Accessed 05 Jan 2016
6. PREMIS Editorial Committee (2012) PREMIS data dictionary for preservation metadata version 2.2. http://www.loc.gov/standards/premis/v2/premis-2-2.pdf. Accessed 05 Jan 2016
7. Library of Congress, Preservation Metadata Maintenance Activity (2016) PREMIS website. www.loc.gov/standards/premis/. Accessed 05 Jan 2016

[3]See Chap. 13 for more on the PREMIS Ontology.

8. Library of Congress, PREMIS Maintenance Activity (2016) PREMIS Editorial Committee. https://www.loc.gov/standards/premis/premis-editorial-committee.html. Accessed 05 Jan 2016
9. Library of Congress, PREMIS Maintenance Activity (2014) PREMIS implementation fairs. http://www.loc.gov/standards/premis/implementation-fairs.html. Accessed 05 Jan 2016
10. Library of Congress, PREMIS Maintenance Activity (2014) PREMIS implementation registry. http://www.loc.gov/standards/premis/registry/. Accessed 05 Jan 2016
11. Digital Preservation Coalition (2005) Second digital preservation award 2005: press release number two, November 22, 2005. http://www.dpconline.org/newsroom/not-so-new/110-awards-2005. Accessed 05 Jan 2016
12. Digital Preservation Coalition (2010) Saving the digital decade: DPC rewards organizations helping to safeguard our national memory. http://www.dpconline.org/newsroom/latest-news/945-saving-the-digital-decade-dpc-recognizes-major-accomplishments-to-safeguard-our-digital-memory. Accessed 23 Jan 2016
13. Consultative Committee for Space Data Systems, International Organization for Standardization (2002) Reference model for an Open Archival Information System (OAIS): Issue 1. http://public.ccsds.org/publications/archive/650x0b1s.pdf. Accessed 03 Jan 2016
14. Consultative Committee for Space Data Systems, International Organization for Standardization (2012) Reference model for an Open Archival Information System (OAIS): Issue 2. http://public.ccsds.org/publications/archive/650x0m2.pdf. Accessed 03 Jan 2016
15. Lavoie B (2014) The Open Archival Information System reference model: introductory guide, 2nd edn. DPC Technology Watch Series Report 14-02. http://www.dpconline.org/component/docman/doc_download/1359-dpctw14-02. Accessed 05 Jan 2016
16. IFLA Study Group on the Functional Requirements for Bibliographic Records (1998) Functional requirements for bibliographic records. http://www.ifla.org/publications/functional-requirements-for-bibliographic-records. Accessed 05 Jan 2016
17. Library of Congress, METS Editorial Board (2015) Metadata Encoding and Transmission Standard (METS): official web site. http://www.loc.gov/standards/mets. Accessed 05 Jan 2016
18. ISO/IEC 21000-2:2005 information technology—multimedia framework (MPEG-21). Part 2: digital item declaration. http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=41112. Accessed 23 Jan 2016
19. Library of Congress (2016) Linked data service, authorities and vocabularies: preservation schemes. http://id.loc.gov/preservationdescriptions/. Accessed 05 Jan 2016
20. University of California (2012) Unified Digital Format Registry (UDFR): a semantic registry for digital preservation. http://UDFR.cdlib.org/. Accessed 05 Jan 2016
21. Caplan P, Lavoie B, Vermaaten S (2012) Identifying threats to successful digital preservation: the SPOT model for risk assessment. D Lib Mag 18(9/10):4. doi:10.1045/september2012-vermaaten
22. Kool W, Lavoie B, Van der Werf T (2014) Preservation health check: monitoring threats to digital repository content. http://www.oclc.org/content/dam/research/publications/library/2014/oclcresearch-preservation-health-check-2014.pdf. Accessed 05 Jan 2015

Chapter 4
How to Develop a Digital Preservation Metadata Profile: Data Modeling

Angela Dappert

4.1 Introduction

Digital preservation metadata profiles vary because of different content types held in the repository, different functions performed on them, different organizational mandates and processes, different policies, different technical platforms, and other reasons. Because of this, one important step in their development is the definition of a *logical data model*. The logical data model declares the key context-specific entities for which metadata needs to be created, the relationships between them and the specific metadata properties that should be captured for them. This chapter describes the principles of how to create a logical data model. Chapters 5 through 12 go on to present a number of case studies that illustrate how specific data model issues have been decided for different entity types, for different content types, such as web archives, audiovisual or e-book materials, and for different organization types. Issues around the creation of a *physical data model* that builds on top of the logical data model are discussed briefly below and are followed up in more depth in Chap. 13 on the Serialization of PREMIS.

One of the first steps in the process of defining a digital preservation metadata profile is the risk- and function-based requirements analysis that was discussed in Chap. 2 of this book. Once the digital preservation metadata requirements are known, one can define a specific logical data model that is applicable and relevant to the context so that it supports them. The data model in the PREMIS Data Dictionary is a general data model. It determines what the basic structure of the resulting profile data model will look like. The specific solution shares this basic

A. Dappert (✉)
The British Library, 96 Euston Rd, London NW1 2DB, UK
e-mail: angela.dappert@bl.uk

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_4

structure with other PREMIS conformant implementations. The requirements analysis determines how the basic model is fine-tuned to the specific situation and may differ from other implementations.[1]

Typically one determines the data model for the application profile by analyzing the specific scenario in several ways. This chapter explores the main steps of doing this.

4.2 Identifying Entities and Entity Types

To start with, you need to *understand the entities that are applicable and relevant.* This depends on the functionality that is to be supported through the metadata. At what level do you need to describe objects, events, agents, and rights involved in the domain so that they can provide information for the functions you need to support? How should you use the entities provided by the PREMIS Data Dictionary for the specific context?

The entities that need to be described may be 'objects.' Objects are either content objects, or they may be computing environment objects that are needed to render a content object. Either of them may need to be described at several levels, called object categories: their intellectual description, their *Representations*,[2] their *Files*, and *Bitstreams*. Furthermore, entities may be 'agents,' such as institutions, persons or software agents; 'events,' such as events that transform a digital object, for example by deriving a write-protected, page-oriented PDF from an editable Word document; or events that record activities that have been performed to assure the digital object's integrity; and 'rights' that are held by 'agents' or associated with 'objects.'

For example, for an 'e-journal' scenario there may be 'journal,' 'issue,' 'article,' and 'figure' objects. They are specific sub-entities of the PREMIS 'object' entity. In the example, 'journal' and 'issue' objects have no files associated with them. The software requirements state that they will only be used to support search and access of articles. Because of this they are implemented as *Intellectual Entity* objects so that they can capture descriptive metadata to support search and access. But 'article' and 'figure' objects have actual file objects associated with them. They may be implemented as *Intellectual Entity* objects so that they can capture descriptive metadata about the articles and figures. Additionally they may be implemented as *Representation* objects so that one can capture which files together make up one article rendition. And they may be implemented as *File* objects so that one can capture technical metadata about them. Typically, in an e-journal setting one would

[1]The concepts of an application profile, requirements analysis, and the PREMIS data model are covered in the first three chapters of this book.

[2]"The set of Files, including structural metadata, needed for a complete rendition of an Intellectual Entity" [1].

not need to implement a *Bitstream* object. There is typically no need to access parts of a file, as one might do in an A/V scenario, where one might want to access subchannels or time segments of the content separately.

It is not compulsory to create these four object sub-entities. If the repository, for example, uses e-journal issues solely to support search, then you may choose not to implement 'issue' objects, but rather just record the needed issue information with every article. This creates a simpler model, but introduces repetition of the same information in multiple places. A different solution may decide the opposite, and combine all articles in one 'issue' object. This, again, creates a simpler model, but creates big objects that always need to be transferred and processed together, even if just one article is needed.

Let us assume that in this example every article consists of exactly one file. This raises a further question related to which object categories should be implemented for an article. Should one then create an *Intellectual Entity* for the article that holds the descriptive metadata, and link it directly to the article's *File* object? Or should one create a separate *Representation* object that links to the *Intellectual Entity* object it represents and also to the *File* object of which it is composed, even though it always consists of only one file? An answer may be that the second solution provides more flexibility. If the file formats should change and the repository, at some point, needs to ingest articles that are composed of several files, then the system will be ready to handle this new use case without further software development effort. The future expansibility comes at the slight cost of a more complex model. This in turn implies some extra initial software development effort, creates metadata that are strictly speaking not (yet) needed, and a somewhat increased potential for introducing errors with the increasing model complexity. Again, different solutions may make different choices.

Another modeling question is how generic and reusable the model should be. If you need to manage very heterogeneous content types in the same repository (e-journals, e-books, digitized books, archive records, web archives, A/V documents, etc), you may wish to implement a generic data model that is to be shared by several content types. A general, reusable model, among other things, decreases the software development effort, creates more consistent metadata, supports sharing of software routines among content types, and makes the model more maintainable. But it may make it hard for repository developers to know how to map a new content type to the existing model. It may sometimes force them to map a specific concept to the existing data model in an unnatural way that also prevents specialized use of the concept. And it means that the model cannot enforce content type specific validation rules in a generic model. Most often, in a repository, a general, reusable data model will be combined with content type specific metadata extensions.

Similar considerations apply to events, agents, rights, and computing environments. The organization's digital preservation policy is an important source for requirements. It may, for example, inform the decision about which events should be recorded for preservation and provenance purposes. Do you need to record every time a checksum check has been performed on a digital object? The answer will

vary from organization to organization. So will the decision on whether this information should be recorded for each individual file or for a set of files. And, hence, the resulting data model will vary from organization to organization, even for the same content type.

4.3 Describing the Entities

Once the entities that need to be implemented have been identified, each entity needs to be described by an associated set of semantic units (properties). The decision about which semantic units to include is also requirements based. Core semantic units may be adopted from PREMIS; if they are not contained in PREMIS, they can be implemented through another, related standard by extending PREMIS. The options for combining PREMIS with external metadata standards are discussed in Chap. 14 of this book. In the e-journal example, the 'figure' objects may require a variety of image-specific technical metadata, such as the image resolution. This is important preservation metadata that is needed to render the image properly in the future, but it is also in common use for other image-related purposes and can be adopted from an image-specific standard.

4.4 Relating Entities

You also need to determine how the identified entities are related to each other. Not all existing or possible relationships are included in the data model. It only includes the ones that support functions that were identified in the requirements analysis. In the e-journal example, you may want to inquire whether article *Representations* were derived from each other. For example, PDF *Representations* may have been derived from the original LaTex *Representations*. In this case you may wish to record a *derivation* relationship between these two 'article' *Representations*. This goes hand-in-hand with the opportunity to also record information about the event in which the derived *Representation* was created, and information about the agents (people, organizations and computing tools) that were involved in its creation. This is important provenance information for determining why the derivatives were created, what the outcome was, what characteristics of the object may have been altered, and identifying which objects were affected by the use of a given software tool.

Again, the decision is based on the requirements analysis. The requirements ensure implementation of the functionality that is to be supported for the specific scenario. In the e-journal example, if the user needs to be able to determine which *Representations* are derived from others then this results in a requirement to capture the *derivation* relationship between *Representations*. The repository software then needs to create such a relationship whenever a migration of file formats happens, or whenever files that are derivatives of each other are ingested from an external source.

In addition to the *derivation* relationship between *Representations* it may be useful to record a *prepublication* relationship between articles at an *Intellectual Entity* level, which reflects evolving information content. It makes it possible to link from an article to its related prepublished versions and to events that happened in the publication workflow of the article over time. Whether you create a separate *Intellectual Entity* that separates the prepublished *Representations* from the published *Representations* again depends on your policies and processes.

4.5 Completing the Logical Data Model

Once the basic data model is defined, it needs to be finished up by specifying which values are mandatory to record. You also need to specify data types for metadata values. For example, when semantic units are adopted that take values from a controlled vocabulary, then this vocabulary needs to be specified from a preexisting vocabulary or it needs to be created locally for the specific situation.

PREMIS gives you five notable degrees of freedom when designing the data model:

1. A repository is free to implement semantic units using names different from those defined in the Data Dictionary.
2. A repository is free to implement semantic units at higher or lower granularity than defined in the Data Dictionary.
3. An implementation may extend PREMIS semantic units.
4. Controlled vocabulary is recommended but not compulsory.
5. An implementation does not have to record mandatory metadata explicitly (as long as they can be generated for exchange or export).

The Archival Information Package (AIP) is stored inside the repository. The Dissemination Information Package (DIP) specifies the metadata that is to be exported from the repository. It is sensible that the implementation of the AIP and DIP metadata profiles may have to be different. As far as the DIP is concerned, PREMIS does not dictate in what form one stores preservation metadata in the AIP, as long as the DIP can be produced in an export format that can be mapped to the Data Dictionary.

4.6 The Physical Data Model

Defining entities, semantic units, and relationships results in the logical data model that will be implemented. The logical data model does not specify any implementation decisions. The physical data model, which defines how metadata is to be physically implemented and then serialized, is developed in a separate step. For

example, in the logical data model one does not yet decide whether the metadata should be implemented in XML or in an Excel spreadsheet. Even if one knows that XML will be used one would not specify, at this point, whether a semantic unit would be implemented through a property or an attribute. Another example of a situation that is not specified as part of the logical data model is whether relationships should be implemented to be uni- or bidirectional. In the e-journal article example above, you could choose to link from the original to the derived *Representation*, or vice versa, or even both ways. The logical model may specify in which direction the linkage will be required. For example, it may require that you should be able to tell what older versions exist for an article, but may not require that the system also show newer versions. But the actual implementation is likely left as a physical data modeling decision. To some degree, how objects are physically linked with software pointers is a separate decision from their logical relationships. For example, if one uses bidirectional relationships then they need to be kept synchronized in both objects if any updates to the relationship are performed. Otherwise the metadata may end up contradicting itself. Bidirectional pointers mean that updates become slower and somewhat more error-prone, but access to a related object may be more flexible and faster. This decision is also based on a requirements analysis, but this analysis is oriented toward the technical solution. Again, the implemented solutions may vary. This topic is further discussed in Chap. 13 of this book on serialization choices and issues.

4.7 Customizing Data Models

To continue the 'metadata standard as a language' simile in Chap. 1 of this book: creating an application profile from scratch is like creative writing. But you can also quote other people's profiles. Quoting others is a great idea. Many digital preservation metadata implementers have shared their own metadata profiles. When implementers with similar organizational needs use shared profiles, they get a boost to their development speed, they benefit from others' lessons learned, and they improve interoperability with partner institutions. Similarly, both commercial and open-source repository software packages often offer default profiles for established content types. But implementers still need to understand whether somebody else's profile suits their own needs and how they can customize it to their own. The general methodology for creating an application profile can equally be applied to these customization tasks.

4.8 Case Studies

The case studies in Chaps. 5 through 12 of the book illustrate how specific data model issues have been decided for different entity types, such as objects, events and rights; for different content types, such as web archives, audiovisual, or e-book materials; and for different organization types, such as archives as opposed to libraries. They include:

- the choice of data models;
- the needs of the designated user communities;
- purposes in different communities, such as archives, libraries, museums;
- purposes of the collections;
- the functions the metadata needs to support, such as storage, search, browsing, access, exchange, data management, or preservation actions;
- intended scales for the size of the collection, IT resources, and human resources;
- what is particular about the content type that might impact the way you implement PREMIS descriptions of it;
- what other metadata systems were integrated and how that influenced PREMIS choices;
- what is in the scope of PREMIS and what is not;
- how the repository architecture influences metadata decisions;
- what policies or regulations affect implementation choices;
- pros and cons of choices;
- pitfalls and lessons learned.

An in-depth case study for the creation of a data model for e-journals can be found in [2].

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 22 Feb 2016
2. Dappert A, Enders M (2008) Using METS, PREMIS and MODS for archiving eJournals. D-Lib Magazine, 14(9/10), ISSN 1082-9873. http://www.dlib.org/dlib/september08/dappert/09dappert.html. Accessed 22 Feb 2016

Chapter 5
Digital Preservation Metadata Practice for Audio-Visual Materials

Kara Van Malssen

5.1 Introduction

Digital moving images objects in some ways are just like other digital objects of any type. General applications of PREMIS [1] apply just as well to moving image objects: logging fixity checks, describing events such as repository Ingest, and documenting the agents responsible for those events. In many preservation environments, the local application of PREMIS is general enough to support preservation documentation of moving images and other object types equally, allowing for efficient processing of objects.

In other ways, moving image objects are quite unique. Particularly in a preservation context, aspects of the creation of moving image objects and their characteristics are important to capture in a way that can be easily interpreted over time so that they can be appropriately monitored for obsolescence, rendered and transformed in a way that preserves significant characteristics, and so that they can be understood by end users. In addition, it is important to note that moving image materials fall into the category of content that MUST be preserved digitally in order to be accessed over time, due to technical dependencies for renderability and the frequent obsolescence of those technologies.

PREMIS provides a framework for describing these objects and the events that occur throughout their lifecycle, as well as semantics to support expression of these.

K. Van Malssen (✉)
AVPreserve, 253 36th Street, Suite C302, Brooklyn, NY 11232, USA
e-mail: kara@avpreserve.com

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_5

5.2 Composition of Moving Image Objects

5.2.1 Structure

Digital moving image objects, particularly digital video objects, are complex in composition and structure. In the most basic form, digital video can be understood as an individual *File* object which is composed of a container and one or more tracks. Tracks found within video containers typically include:

- At least one video track.
- Zero or more audio tracks.
- Zero or more text tracks, sometimes referred to as data tracks.

Multiple audio tracks may be used for multi-channel audio experiences. Stereo sound may be represented by either two tracks or two channels within one track, and files that support surround sound may contain five or more audio tracks. Different audio tracks may contain different language variants of the content. Similarly, text tracks could contain subtitles in one or more languages, closed caption, or teletext data.

Some digital video file containers allow for additional data to be wrapped in the object. These data might include:

- Other file objects or attachments (e.g., images of cover art).
- Embedded descriptive, rights, or other metadata.
- Menus, such as those found on DVDs.
- Chapter data.
- Timecode.

5.2.2 Characteristics

The time-based nature of digital video, as well as aspects related to the resolution and presentation of the content, point to additional characteristics of video objects that are important to understand for long-term content preservation. These include:

- Encoding format/codec: algorithm used to compress the data found in tracks.
- Bitrate: amount of data, expressed in bits per second, delivered over time.
- Frame rate (video only): number of frames delivered per second
- Duration: run time of the content.
- Sampling rate (audio only): number of bits sampled per second, expressed in Hz.
- Bit depth: number of bits sampled per pixel, or for audio, per second.
- Frame size: the width and height of the frame, expressed in pixels.
- Aspect ratio: the ratio of width to height.
- Colorspace: format of the color data.
- Chroma subsampling: the encoding of chrominance and luminance.
- Pixel format: the format and sequence of color pixels.

Properties such as the encoding format and bitrate help identify the resolution of the video and audio data in their current form. Properties such as aspect ratio, frame size, and frame rate are characteristics inherent to the video that is important to preserve over time.

5.3 Use Cases

Any aspect of the PREMIS Data Dictionary could be applicable in preservation environments that hold moving image content. Usage largely depends on local requirements and business processes. For instance, a repository may have a business rule that says all objects must have a PREMIS Rights statement or that file-level fixity data must be expressed using the semantic units for *fixity* in order to be ingested. These are not unique to moving image content, and so in this sense, moving images can be treated just as any other content type.

However, the characteristics described above, requirements for describing creation, and potentially specific rendering environments, do call for some specific uses of PREMIS. This chapter describes several use cases for metadata capture and expression relevant in the context of moving image preservation, and approaches for utilizing PREMIS to fulfill these.

5.3.1 Use Case 1: Describe Events and Corresponding Agents in the Process of Reformatting Physical Moving Image Material

The preservation of moving image material requires that content contained on physical carriers be reformatted to the file-based domain sooner or later (depending on the obsolescence risk of the carrier). Therefore, the reformatting process is part of the preservation lifecycle, and is important to document so that future caretakers and users are aware of the decisions that went into the process of creating digital objects, and understand why the content looks the way it does when it is viewed.

The PREMIS data model provides an ideal framework for capturing events related to the preservation reformatting process. Many organizations tasked with the preservation of video content utilize PREMIS for expressing reformatting events, and the agents responsible for these. These events may include:

- *Cleaning*: of the original carrier, which may involve running it through a cleaning machine. Cleaning is important to remove oxide shed on digital tape and dust and other particulate on film. Lack of cleaning can at times result in distortion of the image or sound, the source of which may be unclear once the video is accessed in the file-based domain.

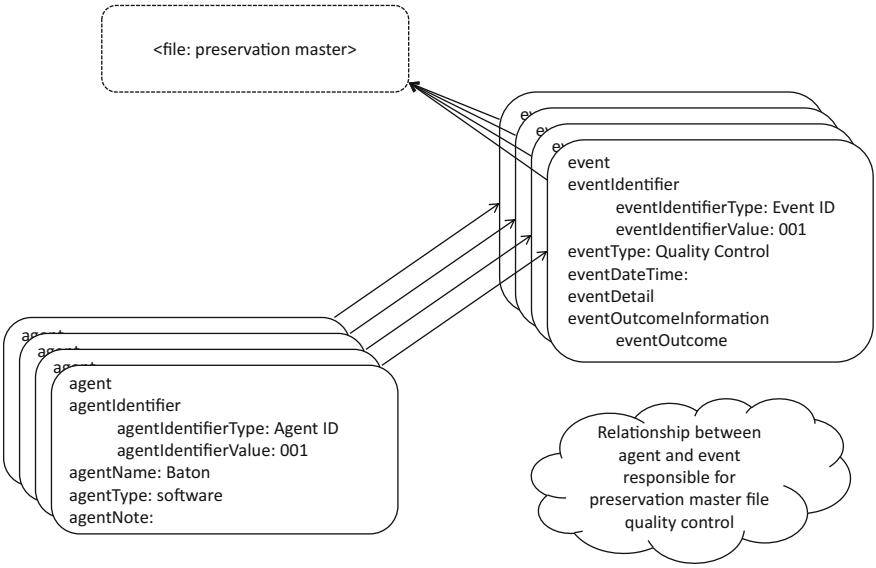


Fig. 5.1 Use case 1: agent to event relationships

- *Repair/rehousing*: Damaged carriers may at times need to be repaired or rehoused, such as in a new tape cassette shell.
- "*Baking*": Deteriorated magnetic tapes suffer from oxide shed or binder degradation due to hydrolysis, which can cause a build-up of particulate matter during playback that can severely damage both machines and tapes. In order to prevent or correct this problem, magnetic tapes are sometimes "baked" or dehydrated using specialized equipment, which allows them to be temporarily played back. It is important for caretakers to know if baking has occurred because this can cause the tape to become brittle, meaning it may not be playable again.
- *Migration*: The act of migrating the physical source to the digital target, through a series of processes, which may include: playback, time-based correction, color correction, analog-to-digital conversion, encoding, and capture.
- *Quality Control*: Machine and/or human inspection, and reporting of any errors or problematic aspects of the video signal.

Each of these events can be expressed using the PREMIS Event entity, and is related to their corresponding agents and preservation *File* objects. For most, the relationship is one-to-one (one event to one agent). An example is provided in Fig. 5.1.

However, the migration event is somewhat more complex, as it involves a device chain, or multiple agents to complete the one event. This can best be represented using PREMIS environment to capture the multiple software or hardware agents that constitute the environment associated with the migration event. Depending on

local data models and business rules, there are several approaches that could be used to express this within PREMIS v2, including:

- *environment* semantic container with multiple agents embedded in *environmentExtension*
- *environment* semantic container with descriptors for multiple *software* and *hardware* containers
- *environment* semantic container contained within *creatingApplicationExtension*
- *environment* semantic container with *environmentExtension* used to contain elements from an external schema.

An example of the last option is provided below, in this case, to continue with the approach used in the previous example, embedded in an agent entity, which would be related to a reformatting event. This approach uses the reVTMD schema from the US National Archives and Records Administration [2], which was specifically developed for expressing process history of reformatted video content (Fig. 5.2).

Version 3 of the PREMIS Data Dictionary introduced more expressive ways for describing environments, as shown in use case 4 below. See Chap. 10 for more on PREMIS environments.

This same approach can be used to express future migrations, either for preservation (e.g., format migration) or for the creation of new access derivatives, which tend to occur more frequently as video on the web evolves and users' needs and expectations change.

5.3.2 Use Case 2: Describe the Creation of Object Tiers and Their Structural Relationships

Digital video preservation often involves the creation of tiered stacks of objects that play different roles. Preservation master files, whether digitized or born-digital, are often quite large in size, and require significant resources to move, store, playback, and edit. As a result, these files may simply fulfill a preservation role only, and be stored in lower cost and less accessible storage (e.g., on data tape). In these cases, organizations managing digital video will typically create mezzanine files (sometimes referred to as edit masters or other local term) in a format and resolution that is high enough quality to create nearly all derivatives, but significantly smaller in size and less demanding of computing resources to use. These can be used for the creation of any number of access copies.

A package containing a preservation master and one or more derivative objects, their creation, and the relationships between *File* objects may be described using PREMIS Object, event, and agent entities. Using an *Intellectual Entity* object as the parent, several *File* objects, and potentially even *Bitstreams* for tracks, the relationships between objects can be described as illustrated in Fig. 5.3.

```
<?xml version="1.0" encoding="UTF-8"?>
<agent xmlns=" http://www.loc.gov/premis/v3"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:revtmd="http://www.archives.gov/preservation/products/"
   xsi:schemaLocation=" http://www.loc.gov/premis/v3
http://www.loc.gov/standards/premis/v3/premis-v3-0.xsd
   http://www.archives.gov/preservation/products/
https://www.archives.gov/preservation/products/reVTMD.xsd">
   <agentIdentifier>
      <agentIdentifierType>Agent ID</agentIdentifierType>
      <agentIdentifierValue>Agent 002</agentIdentifierValue>
   </agentIdentifier>
   <agentType>software</agentType>
   <agentExtension>
      <environment>
         <environmentPurpose>U-Matic reformatting</environmentPurpose>
         <environmentExtension>
            <revtmd:captureHistory>
               <revtmd:codingProcessHistory>
                  <revtmd:codingProcessHistory>
                     <revtmd:role>Playback</revtmd:role>
                     <revtmd:description>VTR used to playback original
                        tape</revtmd:description>
                     <revtmd:manufacturer>Sony</revtmd:manufacturer>
                     <revtmd:modelName>BVU-950</revtmd:modelName>
                     <revtmd:signal>composite</revtmd:signal>
                     <revtmd:serialNumber>ABC123</revtmd:serialNumber>
                  </revtmd:codingProcessHistory>
                  <revtmd:codingProcessHistory>
                     <revtmd:role>TBC</revtmd:role>
                     <revtmd:description>Time-base correction</revtmd:description>
                     <revtmd:manufacturer>DPS</revtmd:manufacturer>
                     <revtmd:modelName>290</revtmd:modelName>
                     <revtmd:signal>composite</revtmd:signal>
                     <revtmd:serialNumber>67890</revtmd:serialNumber>
                  </revtmd:codingProcessHistory>
                  <revtmd:codingProcessHistory>
                     <revtmd:role>Signal conversion</revtmd:role>
                     <revtmd:description>Converts VTR signal output to
                        SDI</revtmd:description>
                     <revtmd:manufacturer>Blackmagic Design</revtmd:manufacturer>
                     <revtmd:modelName>Analog-to-SDI miniconverter</revtmd:modelName>
                     <revtmd:signal>SDI</revtmd:signal>
                     <revtmd:serialNumber>V3G3902970NKG0</revtmd:serialNumber>
                  </revtmd:codingProcessHistory>
                  <!-- shortened for readability -->
```

Fig. 5.2 Use case 1: description of the reformatting environment

The creation process for each object can then be expressed in a way that is similar to the above example (use case #1), where the hardware, software (such as transcoding tools or nonlinear editing tools) or other devices used in the creation of derivative *File* objects are captured as agents, and related to the events, which could be of type “derivative creation,” or an equivalent controlled term.

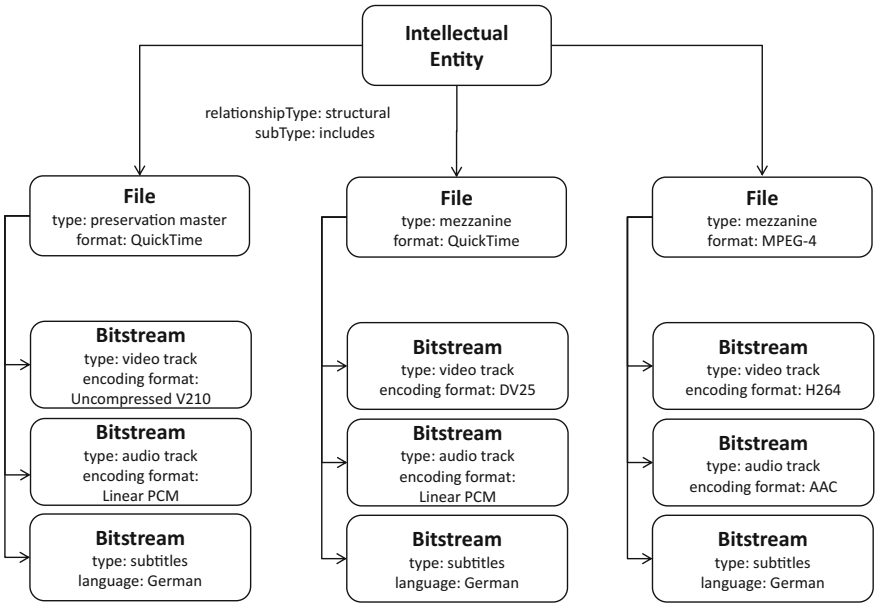


Fig. 5.3 Use case 2: structural relationships between objects

5.3.3 Use Case #3: Describe Significant Properties of Digital Video Objects

As noted above, digital video objects are rather complex in their structure and characteristics. When accessed, the content of these objects must be delivered to the user over time, in the correct sequence, and with the right characteristics so that the content is displayed without abnormalities or distortion, in current and future technological environments. Therefore, understanding the characteristics, or significant properties, of digital video is critical to long-term content preservation.

There are several ways PREMIS can support the description of digital video objects. These approaches could be used together or independently from one another.

5.3.3.1 Identification and Characterization of a File Object

A very simple and common approach to support this use case is to use the *objectCharacteristicsExtension* semantic unit to contain the output of a characterization or technical metadata extraction tool, which may or may not be normalized to a standard schema. Figure 5.4 shows an un-normalized example, which uses the native output of the AV identification and characterization tool MediaInfo [3]. In this example, the entire output of the tool is contained as a descriptor for one *File* object.

Fig. 5.4 Use case 3: video *File* object characteristics

```
<?xml version="1.0" encoding="UTF-8"?>
<object xmlns="http://www.loc.gov/premis/v3"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.loc.gov/premis/v3
   http://www.loc.gov/standards/premis/v3/premis-v3-0.xsd"
   xsi:type="file">
   <objectIdentifier>
      <objectIdentifierType>SAMPLE-ID</objectIdentifierType>
      <objectIdentifierValue>122345</objectIdentifierValue>
   </objectIdentifier>
   <objectCharacteristics>
      <compositionLevel>3</compositionLevel>
      <format>
         <formatDesignation>
            <formatName>MPEG-4</formatName>
            <formatVersion>QuickTime</formatVersion>
         </formatDesignation>
      </format>
      <objectCharacteristicsExtension>
<Mediainfo version="0.7.64">
   <File>
      <track type="General">
         <Complete_name>fileID.mov</Complete_name>
         <Format>MPEG-4</Format>
         <Format_profile>QuickTime</Format_profile>
         <Codec_ID>qt</Codec_ID>
         <File_size>191 MiB</File_size>
         <Duration>49mn 18s</Duration>
         <Overall_bit_rate>541 Kbps</Overall_bit_rate>
         <Encoded_date>UTC 2013-04-22 18:27:57</Encoded_date>
         <Writing_library>Apple QuickTime</Writing_library>
      </track>
      <track type="Video">
         <ID>1</ID>
         <Format>H264</Format>
         <Codec_ID>H264</Codec_ID>
         <Duration>49mn 18s</Duration>
         <Bit_rate>407 Kbps</Bit_rate>
         <Width>854 pixels</Width>
         <Height>480 pixels</Height>
         <Display_aspect_ratio>16:9</Display_aspect_ratio>
         <Frame_rate_mode>Variable</Frame_rate_mode>
         <Frame_rate>30.000 fps</Frame_rate>
         <!-- shortened for readability -->
      </track>
      <track type="Audio">
         <ID>2</ID>
         <Format>AAC</Format>
         <Format_Info>Advanced Audio Codec</Format_Info>
         <Codec_ID>40</Codec_ID>
         <Duration>49mn 18s</Duration>
         <Source_duration>49mn 18s</Source_duration>
         <Bit_rate>128 Kbps</Bit_rate>
         <Channel_count>2 channels</Channel_count>
         <Channel_positions>Front: L R</Channel_positions>
         <Sampling_rate>44.1 KHz</Sampling_rate>
         <Compression_mode>Lossy</Compression_mode>
         <!-- shortened for readability -->
```

There are other possible options that could be taken in this example. The file size, for instance, could be extracted and put into a PREMIS *size* semantic unit, if that is required for consistency with other objects in the preservation environment. Likewise, the encoded date information could be captured in a PREMIS Event.

5.3.3.2 Identification and Characterization of File Object and Contained Tracks as Bitstream Objects

Because each track is encoded differently, identifying the "format" of a video file actually requires identifying both the container and track encoding formats. The level of identification may be represented in PREMIS using both the *File* and *Bitstream* object types, and the format identification semantic units to identify the format of each.

In this implementation, the *objectCharacteristicsExtension* may again be used, but in this case it would contain only the data relevant to the applicable *File* or *Bitstream*.

This approach might make for easier and more consistent parsing of format identification data for indexing by applications that might provide search, analytic, or obsolescence monitoring services, among others.

5.3.4 *Use Case #4: Describe Rendering Environments*

Often digital video files will play using standard consumer software, such as QuickTime [4] or VLC Player [5]. In other cases, such as for video content created in some broadcast, cinema, or fine art contexts, it may be ideal to document specific required rendering environments.

As an example, camera originals created by professional cameras may require very specific software or hardware in order to be rendered properly. PREMIS environment provides an effective way to describe data about rendering environments known to work.

The example below is the directory structure output by a Sony XDCAM camera [6]. It is important to note that if any component were to be changed (e.g., a file is removed, or a folder is renamed), the structure would break and the video would not be renderable (Fig. 5.5).

A very specific application is required to package this into a self-contained video file for editing and playback, in this case Sony XDCAM Transfer, and version

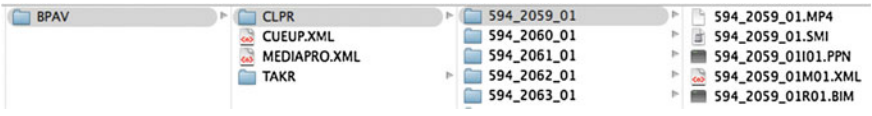


Fig. 5.5 Use case 4: XDCAM structure

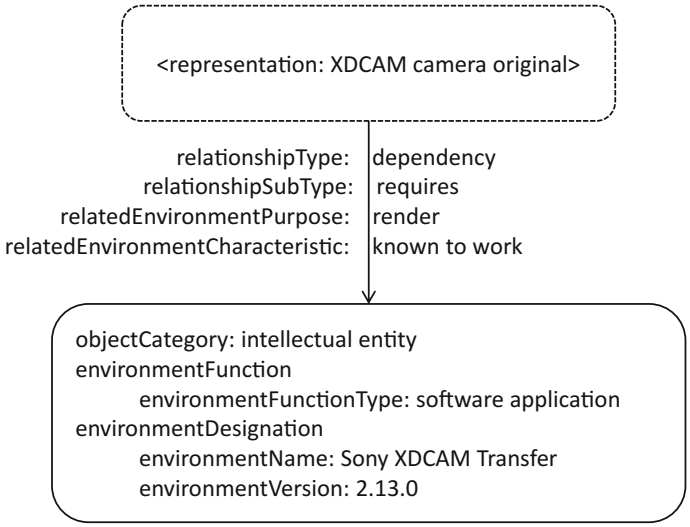


Fig. 5.6 Use case 4: expressing rendering environment

2.13.0 is known to work for this particular object. PREMIS 3.0 offers an elegant solution for expressing the role of the software and the nature of its relationship to the camera original package.

In PREMIS 3, the XDCAM Transfer software can be expressed as an object in its own right, specifically an *Intellectual Entity* object of the type environment. In addition, the relationship between the camera original package (*Representation*) and the rendering software (environment) can be used to document the dependent nature of the *Representation* on the environment, as demonstrated in Fig. 5.6.

This model supports the approach of preserving the rendering environment itself in addition to just the content object.

Digital cinema and video art may present even more complex examples whose rendering environments could be described using multiple PREMIS environment entities as needed. Preservation repositories may elect to describe rendering environments for all video objects in their care, although if attempting to be comprehensive, this approach can be hard to maintain over time through version updates and across operating systems.

5.4 Implementation Approaches and Complementary Standards

Implementation approaches for PREMIS, including how it is used with or in relation to other standards, vary greatly depending on the organizational context in which moving image content is being managed for preservation purposes. Below are a few examples.

5.4.1 Reformatting Services

Because reformatting is such a critical part of the preservation lifecycle for moving image content, it is important to capture the process history from the reformatting service, whether internal or outsourced. Reformatting service providers are used to providing data about their process to clients in a format that best serves the client's needs. Content holders often request PREMIS compliant metadata to be delivered in spreadsheet form, which, when stored as CSV, can easily be transformed to the content holder's local data model. If available, and if desired by the content holder, reformatting services may be able to provide process history data in other formats, such as XML. It is up to the content holder to request this data in the form that best suits their needs, and provide templates, vocabularies, and a data dictionary as needed. A pilot project with the vendor can help identify any irregularities or problems with the data, which can be resolved before full production begins. In scenarios where reformatting is being performed at scale (i.e., hundreds or thousands of moving image objects being processed as part of one project), which is not uncommon as obsolescence of physical media and demand for digital content increase, documentation can be very efficient for vendors as the process is similar across batches of objects.

5.4.2 Libraries and Archives

Preservation repositories in library and archive environments, such as within a research library setting, are often responsible for managing a variety of content types, rather than exclusively dealing with moving image materials. In these cases, moving image content will likely be processed through the same pipeline as other content types, with only a few variations in tools and output to ensure efficient processing and reduce the level of custom handling or development required. This approach scales effectively to ingest, storage, and delivery of large volumes of mixed content types in a uniform environment.

Preservation repositories in libraries and archives will often develop content models for preservation objects in their care that enable processing according to content type. Creating application or process history information from content creators (including reformatting services) might be required for the Submission Information Package (SIP) [7: pp. 1–15] for the moving image content type according to a structure and vocabulary that is already compliant with or could easily be transformed to PREMIS. Preservation repositories may choose to express process history metadata using the reVTMD schema from the US National Archives and Records Administration [2], which can be packaged with additional PREMIS and other metadata.

Another common scenario is for SIPs identified as holding moving image content (whether through automated or manual identification) to trigger business rules that

send video objects through a set of tools or services that best support actions such as characterization for those formats, such as Mediainfo [3] or ffprobe [8].

Libraries and archives are most likely to express PREMIS in XML for ingest and Archival Information Package (AIP) [7: pp. 1–9] storage. Often, PREMIS XML will be embedded in METS [9] containers, which, depending on the local rules for METS, will allow compliance for video to be aligned with other content types. At the time of writing, libraries and archives may be likely to take advantage of the PREMIS ontology [10] as data models move toward the architecture of the Semantic web.[1]

5.4.3 Broadcast Archives

Broadcast archives are typically in active production environments, where content creation is complex and fast-paced, and users often need immediate access to content. PREMIS metadata may be created and captured throughout the production, broadcast, and archival lifecycle of broadcast content, ideally in an automated manner.

While broadcast environments may not generate and store PREMIS metadata in XML form, they may use the Data Dictionary as a guideline when modeling archival databases and business processes. PREMIS compliant metadata may be stored within relational databases.

Broadcast file wrappers, such as Material Exchange Format (MXF) [11] also offer the potential for metadata to be embedded within the *File* object itself. MXF is commonly found within production environments today, both as an acquisition format as well as a mastering format. Because MXF allows metadata to be embedded within the file header in XML form, it has the potential to contain PREMIS metadata. However, it is not recommended that this be the sole place where preservation metadata is stored as it tightly couples the content object and the metadata, and thereby increases challenges to data accessibility. In addition, because formats like MXF are quite flexible, files in this format with embedded metadata might not be supported equally by all vendors, especially over time as the standard evolves. Therefore, it is not recommended to only store metadata that broadcast applications might consider “exotic” within the container, as this may cause the information to be discarded.

5.4.4 Fine Arts Museums

Fine arts museums are less focused on high volume processing of a large number of preservation objects, and more on documentation of the components, composition,

[1]See Chap. 14 for more information on METS and PREMIS, and Chap. 13 about the PREMIS XML Schema and the PREMIS ontology.

characteristics, and dependencies of digital video objects required for conservation and ultimately exhibition. Museums are likely to have multi-channel video art works in their collections, in which the structural relationships between individual *File* objects are incredibly important to document. They are also more likely to have video works that depend on specific hardware or software environments for display and exhibition. Often, video components may be only a part of a more complex installation. Because the time-based media conservator's goal is to ensure that the content is exhibited in its original form to the greatest extent possible, documentation practices tend to be extremely granular for each work of art.

PREMIS offers a flexible model for expressing the nature of complex structural relationships, dependencies, and detailed characteristics. Museums may use the *Representation* object to describe a series of *File* and *Bitstream* objects associated with an artwork entity. Museums are also likely to require documentation about dependent software and/or hardware environments. The PREMIS 3.0 Data Dictionary and schema provides improved support for these use cases.

PREMIS may be used strictly in the museum context, such as through the creation and storage of valid PREMIS XML in a conservation repository. It also may provide a framework for data modeling in relational databases.

5.5 Conclusion

Moving images are complex in structure and require recording varied and detailed technical characteristics in order to ensure their long-term preservation. In a context where there are multiple generations and formats of moving image objects that require the use of complex combinations of software and hardware, the nuanced features of the PREMIS Data Dictionary can be used to their fullest. The flexibility and extensibility of PREMIS, as well as the ability to express complex software and hardware environments and the relationships between their components in version 3 make PREMIS an ideal metadata standard to use for providing the information needed to preserve moving images.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 06 Jan 2016
2. The U.S. National Archives and Records Administration. reVTMD XML Schema. https://www.archives.gov/preservation/products/reVTMD.xsd. Accessed 06 Jan 2016
3. MediaArea.net SARL (2015) MediaInfo. https://mediaarea.net/en/MediaInfo. Accessed 06 Jan 2016
4. Apple Inc. (2015) Quicktime. http://www.apple.com/fr/quicktime/. Accessed 06 Jan 2016

5. VideoLAN Organization (2015) VLC media player. https://www.videolan.org. Accessed 06 Jan 2016
6. Sony Corporation (2016) XDCAM. https://pro.sony.com/bbsc/ssr/micro-xdcam/. Accessed 06 Jan 2016
7. Consultative Committee for Space Data Systems, International Organization for Standardization (2012) Reference model for an Open Archival Information System (OAIS): Issue 2. http://public.ccsds.org/publications/archive/650x0m2.pdf. Accessed 06 Jan 2016
8. FFmpeg Team (2016) ffprobe documentation. https://ffmpeg.org/ffprobe.html. Accessed 06 Jan 2016
9. Library of Congress, METS Editorial Board (2015) Metadata Encoding and Transmission Standard (METS): official web site. http://www.loc.gov/standards/mets. Accessed 06 Jan 2016
10. Library of Congress (2013) Preservation Metadata: Implementation Strategies (PREMIS) ontology. http://www.loc.gov/premis/rdf/v1#. Accessed 06 Jan 2016
11. Wikimedia Foundation (2015) Material exchange format. https://en.wikipedia.org/wiki/Material_Exchange_Format. Accessed 06 Jan 2016

Chapter 6
Digital Preservation Metadata Practice for Web Archives

Clément Oury, Karl-Rainer Blumenthal and Sébastien Peyrard

6.1 Introduction

Web archiving is the series of processes through which digital resources published online are identified, harvested, stored, indexed, made available to end users, and preserved over the long term. Web archiving may follow different goals and be performed by various actors: self-archiving, archiving by third parties, or by scientific and cultural institutions; for legal purposes; for research and data analysis, and for the preservation of online heritage.[1] Legal deposit of the Internet is now part of the regulations of several countries; in that case it is generally performed by national libraries [6]. Crawls performed by search engine companies may also be considered as short-term web harvesting, even though only the latest versions of harvested files are stored, indexed, and displayed as a "cache" copy.

This paper will mainly focus on web archiving performed for the purpose of preserving cultural heritage, as it is the field where preservation issues are more

[1]It is not in the scope of this chapter to present the legal, scientific or cultural goals of web archiving, nor to discuss web archiving examples and use cases in detail. Interested readers are invited to look at [1, 2] for examples of collection policies in national libraries. Readers looking for examples of data mining and research projects performed on web archives should see [3–5].

C. Oury (✉)
International ISSN Centre, 45, rue de Turbigo, 75003 Paris, France
e-mail: clement.oury@issn.org

K.-R. Blumenthal
New York Art Resources Consortium, 1 E 71st Street,
New York, NY 10021, USA
e-mail: karl.blumenthal@gmail.com

S. Peyrard
Bibliothèque nationale de France, Site François-Mitterrand,
Quai François-Mauriac, 75706 Paris Cedex 13, France
e-mail: sebastien.peyrard@bnf.fr

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_6

significant. Preservation-oriented web harvesting started in the middle of the 1990s: the pioneer institutions were the Internet Archive, a not-for-profit foundation hosted in California, and the national libraries of Sweden and Australia. Following their experience, a growing number of public institutions experimented with web archiving principles and tools: for example, web archiving efforts started in 1998–1999 at the National Library of France (Bibliothèque nationale de France or BnF), and first crawls were performed in 2002. The Library of Congress began a Web Archiving program in 2000 with the Election 2000 Collection. These institutions quickly became aware that international cooperation was a key to addressing the challenges they were facing: this is the reason why eleven national libraries and the Internet Archive founded the International Internet Preservation Consortium [7] in 2003. The IIPC currently brings together around 50 heritage and research organizations worldwide.

6.2 How Do We Archive Websites?

6.2.1 Complementary Approaches

Since its inception, two models emerged concerning the selection of websites to be archived:

- *broad or bulk crawls*, typically launched on a national Top-Level Domain or TLD (.dk,.fr…). They are mostly performed by national libraries with a clearly defined legal deposit mandate. Their goal is to harvest as many domain names as possible within their national scope, irrespective of "quality" criteria (e.g., intellectual, artistic value). Broad crawls intend to take a representative snapshot of a national presence on the web.
- *selective or focused crawls* of websites or web resources nominated by librarians or other professionals. They are generally performed at greater "depth" (i.e., more URLs are crawled) or higher frequency; therefore they collect more efficiently very large websites or websites whose content changes regularly.
- Finally, some institutions are *mixing both approaches*: broad and focused crawls complement each other. This is the case since 2004 at the BnF: broad harvests of the .fr domain are performed every year with 4 million domain names harvested in 2013. This is complemented by more frequent harvests of approximately 30,000 websites that are individually selected by BnF librarians or external partners, which amount to around 2.4 billion URLs and 100 TB. The same year, the British Library changed its original selective model into a mixed approach: thanks to a newly enacted legal deposit law, the Library performed its first .uk domain crawl.

Institutions seeking to archive websites may be faced with different situations:

- National institutions often have a legal mandate to archive web content; legal deposit has indeed been extended to the Internet in several countries. The right to archive web content without authorization of rights holders is then very

broad; but there are generally strong constraints on access (e.g., access restricted to the reading rooms of the institution or of few partner institutions).
- Without the legal mandate, a permission-based system can be implemented: here the authorization to crawl a website is granted by rights holders. When institutions rely on the permission-based system, they are forced to adopt the selective model.
- Finally, some institutions (notably in the USA, where there is not a legal mandate) archive websites on an "opt-out" basis, where they are crawled without previous authorization but are removed from public access in case of complaints by right holders.

6.2.2 Tools for Web Archiving

Several methods and tools may be used in order to archive web content. However, most institutions rely on crawling techniques.

6.2.2.1 Crawling Robots to Collect Web Content

The heart of the crawling system is a harvesting robot, i.e., software that successively requests URLs, copies and stores resources, and parses the resulting resource for further URLs. The crawler starts from a list of URLs, generally the website home pages: these source URLs are called the seeds. The robot behaves like an automated Web user and can theoretically follow interlinked Internet resources almost indefinitely. Its scope is defined or limited by crawl parameters, such as crawling depth or budget (number of URLs per domain or host), or the number of links that the crawler is allowed to follow. However, robots may encounter technical obstacles during the harvesting process (inability to crawl interactive animations, streaming videos, etc.), thus leading to unintended incompleteness of website archives.

6.2.2.2 Curator Tools to Manage the Workflow

In order to manage the whole process, institutions frequently use curator tools, which are software that runs on top of a web crawler. They may cover different functions according to the institution's needs: the most common ones are websites selection, management of permissions (when applicable), definition of the target[2] to harvest, scheduling, quality control, and sometimes ingest into the digital repository. Curator tools also enable the management of metadata generated before, during, or after the harvesting process.

[2]A target is a seed or a set of seeds with crawling instructions.

6.2.2.3 Access Tools

Several methods and tools are used to give access to web archives:

- The most common access mechanism is the *replay of web archives* to make them browsable and readable the same way as when they were online. The access interface offers a search by URL and by capture date. The user can then browse the archive "through space" by following hyperlinks and "through time" by choosing previous or following capture dates of a given host.
- Many *other indexing techniques* can be offered to the user: full-text indexing, map projection of web corpora, text or link mining—it is not in the scope of this contribution to describe them.

6.2.2.4 Repositories

For heritage institutions the last step of the web archiving process is the ingest into a shared digital repository. The ingest step should ideally perform all functions defined in the OAIS model [8], notably SIP validation, metadata extraction, and process monitoring. Some commercial repository software, such as Rosetta [9] provided by Ex-Libris, or Preservica [10], are offering ingest functions for web archives. Several institutions have, however, developed their own repositories (see below).

6.2.2.5 A Choice of Open Source Tools

Most IIPC institutions, like the Internet Archive or the BnF, rely on open source tools: Heritrix as a crawler and the Wayback Machine [11] as an access and display tool. There is more diversity in the choice of curator tools, as their features depend on the legal mandate and the internal organization of the harvesting institution. The British Library is using the Web Curator Tool [12], an open source software it developed in collaboration with the National Library of New Zealand. The BnF adopted the NetarchiveSuite [13] in 2010, a tool originally developed by the National Library of Denmark and the State and University Library in Aarhus. Archive-It [14] is a special case of curator tool, offered by the Internet Archive as a service to its partners to let them select content and check the outcome of the harvest.

Finally, several institutions have implemented their own digital repository, such as SPAR, the Scalable Preservation and Archiving Repository, for the BnF [15].

Figure 6.1 shows as example the BnF web archiving workflow, as presented in [16].

- The steps of the workflow are depicted as oval circles, with the color representing the professional roles in charge of the tasks (white: content curators; light gray: digital curators; dark gray: engineers).
- The tools used are depicted as rectangles.

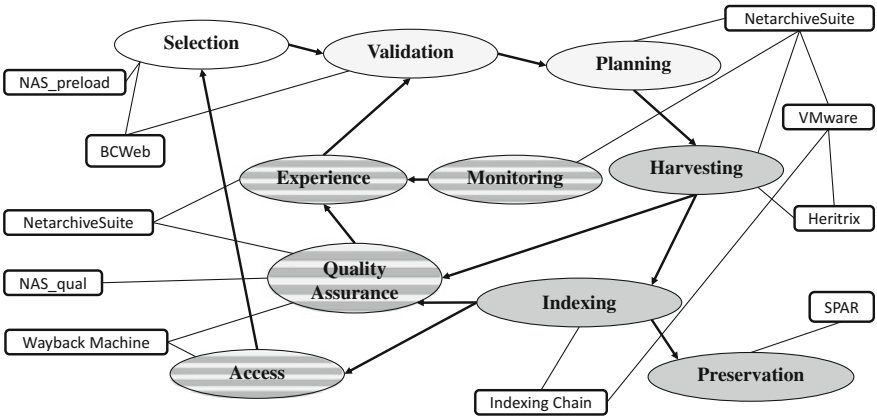


Fig. 6.1 BnF web archiving workflow

6.3 Technical Specifications of Web Archives

6.3.1 The Common Use of Container Formats

As the files harvested on the web are to be counted in hundreds of millions, when not in billions, solutions are needed in order to store and handle this huge number of resources.

6.3.1.1 The ARC and WARC File Formats

Web archives are generally stored in WARC files: WARC [17] is a standardized format for storing web content (ISO 28500:2009). It is a successor of the ARC format, designed in 1996 by the Internet Archive for its web crawls [18].

A WARC file consists of the concatenation of an arbitrary number of "WARC records". There are several types of WARC records[3]:

- warcinfo records, at the beginning of each WARC file, contain information describing the crawl process (operator, crawl settings, etc.);
- request records contain the request sent by the robot to the server hosting the content;
- response records contain a single file harvested on the web, with http protocol response and harvesting metadata (URL, capture date, checksum, MIME type, etc.);

[3]ARC files have only one type of record, which basically corresponds to the WARC response record as it contains the file harvested on the web together with harvesting metadata. The metadata header at the beginning of each ARC file also plays the role of a warcinfo record.

- metadata records may contain any additional metadata as decided by the crawling institution;
- resource records may store additional files, not harvested on the web;
- revisit records are the outcomes of the deduplication process (when a webfile is not collected because it has been already archived by the same institution);
- conversion records store the results of format migrations;
- continuation records allow for segmenting very large files harvested on the web (e.g., for streaming media) when they exceed the target size of the container WARC file.

One of the main advantages of the ARC and WARC files (hereafter collectively referred to as W/ARC files) is indeed that it is possible, as part of the crawler configuration, to define a target size: the crawler stops writing web files in a W/ARC file when it reaches a chosen size limit. This facilitates the storage, transfer, or ingest of the files. The target size of a WARC file is generally 1 or 2 GB while it was around 100 MB for an ARC file.

W/ARC files are generally compressed, in order to save storage space. Most IIPC institutions use the GZIP compression scheme, which is recommended in the appendices of the WARC standard.

6.3.2 Numerous but Often Uncontrolled Metadata

One of the main advantages of the WARC format, compared to its predecessor, is the fact that it offers better ways of storing metadata and to distinguish between different kinds of metadata. Information about web archives is gathered, created, or extracted at different steps of the archiving process.

6.3.2.1 Context and Provenance Information Prior to Ingest

Certain operations performed before ingest by the crawling institution are especially important to record, as they are essential to preserving the captured digital objects and render them over the long term.

The information exchanged between the requesting robot and the server hosting the content through the HTTP protocol indicates how the data was captured and what its status was at the time of capture.[4] In a sense, it is the most important information, as it cannot be recreated or rediscovered later. As explained before, these metadata are written in the header fields of W/ARC records.

[4]For example, an HTML page for which an HTTP 404 response code is attached will be an error message, while an HTML page with an HTTP 200 response code will contain the expected content. Even 404 pages are useful: they can be used as evidence that a URL was “broken” at the time of the harvest.

A second type of metadata consists of all information related to the purpose and context of the crawl, as well as details about the harvesting institution and tools. This kind of information is either recorded in the header of the container format, or in the configuration and log files produced by the robot. It may be critical in different cases:

- *For legal purposes*: it is notably important to record which institution performed the crawl, e.g., whether an organization in charge of legal deposit directly crawled the data, whether it was achieved on its behalf by a third-party archiving company, or whether the data was crawled in the context of another mandate and given to the hosting institution afterward.
- *For technical reasons*: for example, keeping the name and version of the crawling robot helps tracking potential errors of container formats.
- *For documentary evidence*: keeping the configuration files helps to explain why some web files have been collected and others not. This is important for the web harvesting team performing quality assurance, but also for future readers that need to understand how the collection was put together in order to analyze it.
- For the sake of *preserving user experience*. When a robot crawls a website, it does not generally declare itself as an indexing robot, but as a particular family of browsers, in order to archive content tailored for display in a specific browsing environment. Keeping this information (the "user agent") helps in choosing the best browser to render the archive in the technical context in which it was originally offered to the user.

6.3.2.2 Representation Information Extracted by Format Analysis Tools

When web archives are ingested into a digital repository, several analyses and controls are typically performed. One of the key steps is to extract file format information. This should be done at two levels: the container (W/ARC) file on one hand, and the contained harvested files on the other hand. Institutions have indeed no control over the format of the files they are crawling on the web: a single website hosts generally tens of different formats; and a broad crawl can gather hundreds or even thousands of formats [19]. Besides, very little information is available, except the MIME type sent by the server, which is frequently wrong. To this end, JHOVE2 [20] modules for ARC, WARC, and GZIP have been developed, thanks to BnF and IIPC funds. These modules are able to identify and characterize W/ARC files but also the format of the files contained within them. JHOVE2 is the new version of the widely adopted JHOVE validation and characterization tool. Compared to JHOVE, this new version offers two main features that are especially of interest for web archives. First, it distinguishes the format identification of the file from the more detailed validation and characterization steps. Characterization and validation steps are performed by specific JHOVE2 modules (a new module has to be developed for each format), whereas format identification is performed by other

tools (notably DROID [21]). Second, JHOVE2 is able to perform analyses at different levels when it deals with container formats. For W/ARC files, this allows detailed analysis, including format version and technical characteristics, at the level of the contained files [20, 22].

6.3.3 A Complex Granularity

The technical specificities explained above show that one should deal with different levels of granularities when trying to render or preserve web archives.

- At the lowest level, there is the *harvested file*.
- Several harvested files may make up a *harvested page*.
- At an intermediate level, we can identify the logical unit of the *website*. This level of granularity is not as obvious as it may seem, as there is no agreed definition of a website: is it the set of online resources available on the same host? On the same domain name? For example, do we consider wordpress.com as a single website, or is each blog on wordpress.com a distinct website? How to deal with resources displayed on a website but hosted on a different domain, e.g., images or videos hosting platforms cited on a blog?
- Third, there is the level of the harvesting process or *harvest instance*: the set of web files crawled by a single robot at the same time. For selective crawls, it is possible to launch a robot on a unique website, hence allowing a one-to-one relationship between the website and the harvest instance. For scalability reasons, this is not an achievable goal in case of broad crawls where hundreds or thousands of domain names are captured by the same robot.
- At the highest level, several harvest instances will group into a *collection*. For example, all daily crawls of newspapers websites will represent the newspapers "collection". All the harvest instances launched each year to crawl the domain names of a national top-level domain should be considered the yearly domain crawl. Note, however, that the collections may be recursive (the set of all yearly domain crawls represent the "domain crawls" collection); and that collections may be constituted afterward.
- Finally, the *W/ARC container itself* represents another (complex) level of granularity. It is a subdivision of the harvest instance (W/ARC files hold only content coming from the same robot). However, a W/ARC file does not generally correspond to a website, except when a single robot is launched on a unique website. W/ARC files may contain files from numerous websites. On the other hand, resources from a single website should be retrieved from different W/ARCs to be displayed by archive rendering tools. Finally, W/ARC files can also be used to bundle the configuration, log and report files generated by a particular crawl—this is the solution adopted by the NetarchiveSuite curator tool [13]. Those "W/ARC metadata files" are a digital artifact documenting a harvest instance.

When specifying how web archives should be ingested and preserved in a digital repository, and when defining the data model of submission, archival and dissemination packages, institutions need to decide at what level of granularity the "content information" should be managed. What is their primary preservation target? Web files, websites, container files, harvest instances or collections? These questions have been a long-standing subject of discussion and debate between IIPC members, especially within its Preservation Working Group [23]. Diverse solutions have been implemented, one of the most prominent difference being manifested in the choice of logical modeling or container modeling.

6.4 Container Modeling

6.4.1 What Is Container Modeling?

An answer to web archive preservation challenges

After this overview of web archive characteristics and specifics, we can summarize the main challenges arising when working on their long-term preservation:

- Preservation systems are supposed to be able to manage *large-scale* collections: hundreds of terabytes, billions of files. Metadata management should also be at scale: each single harvested file is accompanied by several metadata elements; each crawl generally creates numerous configurations, report and log files.
- Metadata are numerous but often unreliable: it is necessary to decide which *metadata* should be recorded for preservation and which metadata should be discarded.
- Institutions that have adopted a mixed model of harvesting (broad and selective crawls) should be able to ingest both kinds of collections in a consistent way.

Container modeling is an attempt to face these three challenges.

Container modeling considers that the basic unit of preservation, the Content Information, is the W/ARC container file, with the harvest instance as a higher level of granularity.

6.4.2 Container Modeling at the BnF

6.4.2.1 BnF Ingest Step: A Functional Overview

The ingest step implemented for the BnF SPAR repository is an example of container modeling [24].

- When a harvest instance is finished,[5] all metadata describing the crawling process (configuration, report and log files) are recorded in a dedicated W/ARC "metadata file", which is created by the NetarchiveSuite tool [13].
- The metadata file in W/ARC format is then ingested in SPAR as a single SIP. Two levels of granularity are generated: the container level and the harvest instance level.
- All W/ARC files created by the harvest instance are then ingested as SIPs.

For each W/ARC file (that can contain data or metadata), a technical analysis is performed, thanks to JHOVE2 [20, 22] modules, to validate and characterize the W/ARC files and to identify the formats of the contained harvested files. When errors are discovered in the format of W/ARC files, an individual analysis is performed in order to decide if the file should be accepted in spite of its formal errors, normalized to a valid W/ARC file, or rejected.

6.4.2.2 Reasons for Choosing Container Modeling at BnF

BnF has chosen container modeling for practical reasons. First, other levels of granularity were not considered fit to be the basic unit of preservation. The level of the contained web files was not considered practical, as there were too many of them: it was not scalable to characterize and validate them individually. The level of the website has also been rejected, as there was no one-to-one relationship between a website and a W/ARC file or a harvest instance. For BnF broad and even selective crawls, robots are capturing at the same time tens, hundreds, or thousands of websites.

Second, the W/ARC level had obvious practical advantages. This choice allows applying differentiated policies on the container format, at which level the repository performs strict format validation, and on the contained files level, at which level any file format is accepted and plain identification is performed, thanks to a file Unix command [25].

Besides, container formats are supposed to have a predefined size, which simplifies the technical management of ingest.

However, container modeling was not only chosen for pragmatic reasons: it reflects the very nature of web archiving, as seen by BnF web archiving and digital preservation teams. Web archives are not websites: they are frozen copies of dynamic resources at a certain period of time.[6] Websites are more than an interlinked set of publications: the web is a space where people discuss, interact, live. Web archives can only keep the tracks of this living space, as archeological remains keep the track of former human activities.

[5] That is, when the activity of a robot is ended and all harvested web files are stored in W/ARCs.

[6] See [26] for further discussions about the ontological nature of web archives and the potential differences between live websites and their archives generated by crawlers.

Container modeling starts from this postulate: it does not try to recreate a logical level of a website, but considers web archives as the product of the activity of the crawl engine. They are artifacts whose original producer[7] is not the website publisher, but the robot which copied resources during its online journey, following more or less the rules decided by humans. From this point of view, it is critical to maintain the intelligibility of the harvesting context; therefore it makes sense to consider as the target of preservation the set of data and metadata the crawler produces, in its original form—the container file.

6.4.2.3 Matching Container Modeling to PREMIS Concepts at BnF

The PREMIS [27] objects that match the container modeling approach were the following:

- The W/ARC file is a PREMIS *File*.
- As every W/ARC file is a discrete AIP, it is also considered a single PREMIS *Representation*.
- Due to performance and file management issues, the level of the harvested file is not expressed directly in PREMIS or METS. We rely on the W/ARC structure to reference the file. Preservation metadata is aggregated using the containerMD [28] metadata format, expressed at the level of the W/ARC container file.
- Context information was added to explicitly differentiate between W/ARC data and metadata files: a link between the two *Representations* stated that any W/ARC "data" AIP was documented in the corresponding "metadata" AIP. The harvest instance was not considered a discrete structural level that had to be described in its own sake, as this level only corresponded to a grouping of W/ARC files sharing the same Provenance Information prior to ingest.
- As we will see further below, the collection can be considered an *Intellectual Entity*.

This can be summarized in Fig. 6.2.

We can see we have two kinds of links: a one-to-one relationship for single-file packages (one *File*, one *Representation*); a one-to-many relationship between a crawl and the W/ARC files that it produced (harvested data; configuration, logs, and reports metadata). The harvest instance is represented on a separate level.

[7] In OAIS terms, the "Producer" of the web archive is the crawling institution, or the entity in charge of web crawls within the institution. This "Producer" entitles the "Archive" to be in charge of the preservation of the content. Here however, we use the term "producer" as understood by archival science: the agent that created or gathered the content for a specific purpose: i.e., the crawling robot.

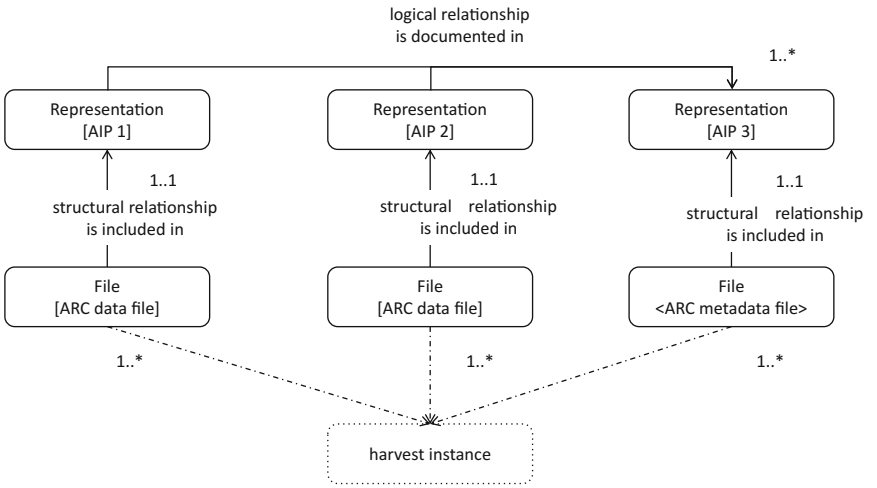


Fig. 6.2 Articulation between W/ARC data and metadata files

Apart from the traditional ingest and preservation events defined in the SPAR repository, PREMIS Events were also used to capture web archive specific Provenance Information:

- the *harvest* event was expressed at the level of the W/ARC "metadata" file, and records the harvest event of the whole harvest instance. This event records the specific outcome of the harvest process: automatically finished, aborted by an administrator, or crashed. The *reports* on the produced W/ARC files and crawled hosts were defined as outcomes of the harvest Event. The need to express them in a structured fashion to allow complex queries led to the use of PREMIS *eventOutcomeDetailExtensions*. The related agents recorded are: the harvesting institution, the operator administrating the harvest, and the job that triggered the process.
- for practical reasons, we also recorded the harvest of data contained in a single W/ARC file with a "*web files harvest*" event, related to the W/ARC "data" *File*. The *robots policy* was considered as a particular *eventOutcome* of the web files harvest; however, if the (non-repeatable) *eventDetail* semantic unit is not already reserved for another use, it is recommended to use it instead. The related agents are: the harvesting and curation software used, and the virtual server that runs the harvest. The *user agent* was also modeled as a PREMIS agent involved in the web files harvest event, distinct from the crawler because the same crawler could declare itself under different identities.

- the creation of the W/ARC container file itself, was documented in a *serialization* event related to the corresponding W/ARC file; the related agents are: the organization that created the W/ARC file, the software used to produce it and the virtual server on which it ran.
- the parameters of a single harvest can change, but documenting all the setting changes as PREMIS Events would have been too verbose and counterproductive (because almost unmanageable). For that reason we decided to record the last state of the crawler parameters in a "*harvest profile processing*" event, which documents the last harvesting parameters defined by web curators, and used by the harvesting tool.

PREMIS metadata was itself wrapped in a METS wrapper. This METS file declared *premis*:*representations* and *premis:files* in *techMD* sections. At the level of the W/ARC file, it also aggregated preservation information about the content files in containerMD. The events are documented in *digiprovMD*.

At the level of the contained files, the descriptions were not handled by discrete PREMIS Objects, but were implemented as a containerMD section at the level of the W/ARC container file: containerMD was leveraged for scalability reasons, in order to keep METS/PREMIS files at a manageable size. The aggregation mechanisms are summarized in Table 6.1.

A summary of the various information types and where they are stored is summarized in Table 6.2.

6.4.3 *Logical Modeling for Web Archives at BnF*

As BnF's preservation strategy was heavily focused on the container and content files structure, logical modeling mostly focused on the only level it considered persistent and relevant for curatorial purposes, which was the collection level (see above). The descriptive metadata used to describe them are outside the scope of this book; we can merely say that we use identifiers for each harvest instance, and that we commit to make them persistent to provide a stable link between our SPAR preservation repository and the NetarchiveSuite curation tool. Collection descriptive metadata is handled at a higher level than each W/ARC file, and is therefore implemented as an Archival Information Collection[8] with its own separate metadata. This can be summarized in Fig. 6.3.

[8]"An Archival Information Package whose Content Information is an aggregation of other Archival Information Packages" [8, pp. 1–9]. This OAIS definition matched our concept of a web archive "collection", which has a clearly defined content from the curators' perspective, but is embodied in several W/ARC data files holding the primary content and W/ARC metadata files holding the configurations, logs and report files for each specific harvest.

Table 6.1 PREMIS information aggregation strategy

Information type	PREMIS corresponding semantic unit	ContainerMD aggregation in <entriesInformation>	Reduction method
Existence of an entry	One premis:object per entry	One entriesInformation for all the content files	Count of all the contained files.
Creation dates	One *dateCreatedByApplication* unit per object	firstDateTime and lastDateTime attributes	Only the min and max values are kept
File format	One formatName and *formatVersion* per object	formats container element. For each name and (if any) version, one format child element captures the number and global size of the concerned files	Aggregation, with count of the number of objects and sum of their size
Encodings (compression, encryption) at entry level	For each object, *compositionLevel* with compressed/encrypted file characteristics: *size*, *formatName*, etc	Encodings element. For each encoding type and method, one encoding child element captures the number and global size of concerned files	Aggregation, with count of the number of objects and sum of their size
Host, declared MIME type and protocol response information	Web archive specific information: for each harvested object, *objectCharacteristicsExtension* container providing its corresponding host, MIME type and response information	ARC and WARC extension elements. For each discrete piece of information, one element captures the value, with the number and global size of concerned files	Aggregation, with count of objects and sum of their size

6.5 Describing the Harvested File for Preservation

6.5.1 Why Describe at the Harvested File Content Level?

While the container model of web archives enables bit-level preservation of W/ARCs and all of their component records in a digital repository, it cannot ensure the viability of those records to replay, using either current or future access and display tools. Format obsolescence threatens the replay viability of any digital file encoded to standards and conventions no longer supported by their primary access tools [29]. In the case of web archives, this means that components vital to the functional and/or aesthetic completeness of a website or websites that are nonetheless superseded and unsupported by current web browsing environments (and the crawling robots based upon them) may in turn fail to render in any fashion understandable to either an automated system or human observer. Given the incendiary pace of change to formats and features on the web in particular [30], obsolescence is not a distant risk, rather it can affect web-based content before, during, and after the harvesting stage of web archiving.

Outside the specific web archiving context, digital archivists and preservationists broadly manage the risk of format obsolescence with one or both of the following strategies:

Table 6.2 Summary of the main information types in web archives and where they are recorded according to BnF's data model

Information type	Where is it recorded?	W/ARC data AIP METS manifest	W/ARC metadata AIP METS manifest	AIC METS manifest
Collection description	METS collection level dmdSec Dublin Core metadata (title, description)			X
Package content classification	METS dmdSec at the level of the described package (type): collection, web data, or harvest metadata	X	X	X
Links between web data AIPs and their harvest metadata AIP	Logical relationship from web data AIP-level *Representation* to harvest metadata AIP-level *Representation* (*relationshipSubType*: "is documented in"	X		
Harvest date	"harvest" event: whole harvest, related to the corresponding W/ARC metadata *File* "web files harvest" event: extreme capture dates of the content of a W/ARC file, related to the corresponding W/ARC data *File*	X	X	
Status of the harvest job	"harvest" event: *eventOutcome* (finished; aborted; crashed)		X	
Crawling institution	"harvest" and "web files harvest" events: linkingAgent, role: performer	X	X	
Crawling operator	Harvest and harvest profile processing events: linkingAgent, role: manager		X	
Crawling software	"harvest" and "web files harvest" events: linkingAgent, role: performer	X	X	
Crawling process itself	Harvest and web files harvest events: linkingAgent, role: trigger	X	X	
User agent	"web files harvest" event: linkingAgent, role: user	X		
Host performing the crawling operations	Web files harvest event: linkingAgent, role: issuer		X	
Creation date of the W/ARC file	"serialization" event, related to the corresponding W/ARC *File*	X	X	
Host packaging the W/ARC files	Web files harvest event: linkingAgent, role:			
Policy towards robots.txt	"Web files harvest" event: *eventOutcome* (could be *eventDetail*) Sample values: "classic" (complies with the protocol), "ignore" (does not follow the protocol)	X		
W/ARC files reports	Harvest event: *eventOutcomeDetailExtension*		X	
W/ARC content files technical information	W/ARC file level technical information WARC metadata files contain technical metadata about crawl/log/configuration files W/ARC data files contain technical metadata about harvest files	X	X	

(continued)

Table 6.2 (continued)

Information type	Where is it recorded?	W/ARC data AIP METS manifest	W/ARC metadata AIP METS manifest	AIC METS manifest
Hosts report[a]	Harvest event: *eventOutcomeDetailExtension*		X	
W/ARC files list[b]	Harvest event: *eventOutcomeDetailExtension*		X	
Last harvesting parameters used by the crawler	"Harvest profile processing" event, related to the ARC metadata *File*		X	

[a]Number of URLs crawled per DNS host
[b]List of W/ARC files collected by a particular host, used to verify that the collection has been fully ingested

- *Format migration*: By which *access copies* of digital files are regularly migrated from their original formats to those supported in present computing environments, while *preservation copies* may be maintained in their original formats.
- *Emulation*: By which access tools are engineered to reproduce the computing environments required to successfully render digital files according to their original formats.

Both strategies, applied at any scale, require the same kind of rich technical metadata, describing the native computing environments of archived digital files, that W/ARC records currently lack. If recorded systematically, however, PREMIS-conformant manifests of technical metadata specific to the object and environment characteristics of the harvested files within a larger W/ARC container could enable uninterrupted access to web archives into the future.

6.5.2 A Case for Content Level Description: NYARC and MoMA.Org

6.5.2.1 Web Archiving at the New York Art Resources Consortium (NYARC)

The New York Art Resources Consortium (NYARC) is a resource-sharing collaboration among three leading art museum libraries in the city of New York—the Brooklyn Museum Library & Archives, the Frick Art Reference Library, and the Museum of Modern Art Library [31]. Since 2013, NYARC has archived the web presences of its member institutions and selectively archived web-based content complementary to those institutions' respective traditional strengths in collecting specialist art historical resources.

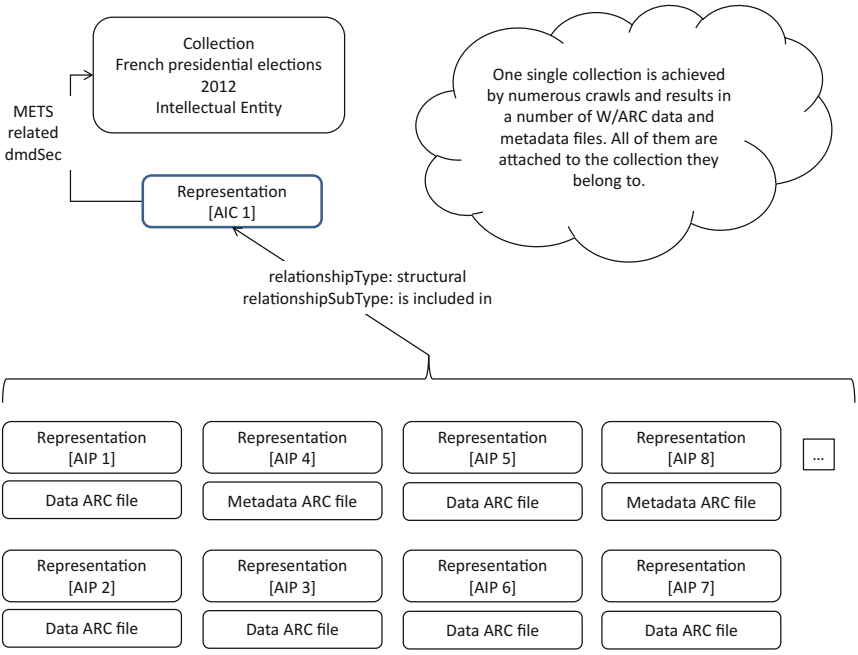


Fig. 6.3 Relationship between the W/ARC *File* level information packages and collection-level information

NYARC relies primarily on Archive-It [14], the subscription-based service provided by the Internet Archive, for access to the Heritrix crawling robot; to a deployment of the Wayback Machine optimized for Archive-It's subscribers; to the Internet Archive's digital storage network for W/ARC files [32]; and to the related curator tools in Archive-It's web-based software application, in order to build, manage, and preserve collections of web archives. It has, however, additionally employed the private web archiving service Hanzo Archives to harvest especially dynamic and subsequently challenging websites within its collecting scope [33].

6.5.2.2 Archiving MoMA.Org's 20 Years on the World Wide Web

Among the most voluminous and technically sophisticated host domains within NYARC's selective web archiving scope is that of its member institution, the Museum of Modern Art—MoMa.org. In addition to an extensive database of its institutional art holdings, MoMA.org continues to host the exhibition "microsites" and related content that the Museum of Modern Art has published to the web since 1995. In this respect, an archived MoMA.org serves as an historical record of not only the Museum's exhibitions, but of web development and digital file formatting conventions more generally [34].

Format obsolescence threatens to erode the completeness and accuracy of these records. In aggregate, the lack of rich technical metadata recorded at the level of the W/ARCs' harvested file contents prevents the architect of a digital repository from designing systems and routines sufficient to support the full diversity of file formats and versions represented across such a collection of web-native materials. More immediately, the access tools used to render these materials legible to a human viewer today lack the sophistication and/or the requisite information to interpret obsolete file formats and replay them. The harvest tool, replay mechanism, or both, can obscure the format of a file contained within a W/ARC to the point that it is invisible or transformed.

In the case of MoMA.org, experiments with emulated web browsing environments more contemporary to W/ARC contents' time of creation have returned mixed, but encouraging, results. The oldweb.today browser emulation tool [35], created by web archiving software developer Ilya Kreymer with support from the Internet-based arts organization Rhizome, can replay select MoMA.org ARC files more accurately and completely than can the Wayback Machine within a modern browser, as illustrated in Fig. 6.4.

In order to fully mitigate the risk of format obsolescence at even the relatively small scale of the Museum's host domain, however, systems designed to automatically migrate the formats of access files or emulate their native environments would require more comprehensive and consistent delivery of technical metadata than the W/ARC currently enables.

6.5.3 Challenges and Solutions to Describing at the Content Level

6.5.3.1 File Format Analysis at the Scale of Web Archives

Models and schema for manifests that include PREMIS-conformant metadata at the level of harvested file content have been proposed since at least 2006 [36, 37]. However, feasible tools and workflows to generate and manage this metadata have yet to be fully developed. As evidenced by BnF's Scalable Preservation and Archiving Repository (SPAR), this is principally an issue of scale—that is, both the scale of the web archiving model and of the harvested resources to be described.

Large-scale web archiving programs with highly customized, in-house digital repositories such as BnF's may deploy next generation file format analysis tools like JHOVE2 and develop custom metadata schema like containerMD [28] in order to support them. Mixed, selective, or generally otherwise less resourced web archiving programs must rely upon bundled digital preservation microservice platforms in order to ingest their web archives into integrated digital repositories. Whether open source or proprietary, these platforms to date do not use JHOVE2, but still universally rely upon JHOVE, which cannot reliably extract file level

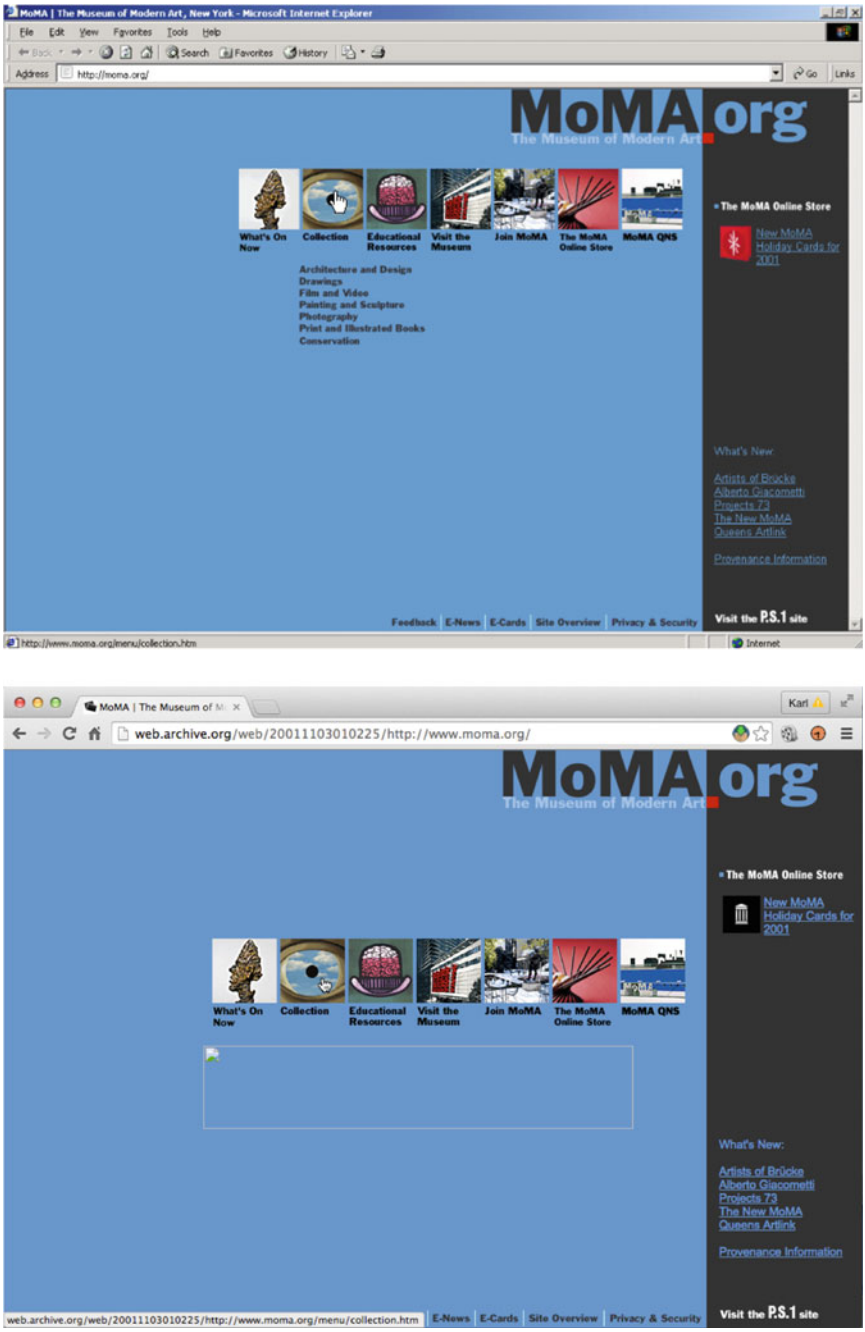


Fig. 6.4 Two views of www.moma.org as it was harvested on November 3, 2001 and preserved by the Internet Archive. *Top* As replayed by oldweb.today in an Internet Explorer for Windows Version 5.5 web browsing environment *Bottom* As rendered by the Wayback Machine in a Google Chrome for Mac Version 47 web browsing environment. In this example, the browsing environment more contemporary to the date of capture of the source material more successfully renders dynamic elements like responsive dropdown menus and animated graphics than can the successor environment

format information from W/ARC containers, in order to produce METS metadata manifests that subsequently cannot be validated as PREMIS-conformant.[9] Regardless of the format analysis tool selected, the BnF case further evidences that conformant manifests at the scale of web archives can be so extensive and voluminous as to be themselves unsustainable in current digital repository environments.

6.5.3.2 Opportunities for Technical Enhancement and Development

Tools and workflows for generating PREMIS-conformant metadata must at the very least address the current problems of analyzing format information accurately and at the scale of all harvested files within a larger W/ARC container file. In these regards, the first and most pivotal challenge to overcome is the current reliance upon MIME type as the standard for identifying and characterizing harvested contents. Servers can, as discussed, provide inaccurate information in the form of MIME type. That information is furthermore inadequate to preservation ends because it lacks sufficient granularity with regards to essential format information, principally version identification. For the latter reason, the PREMIS Data Dictionary rejects MIME as an authority list for format identification unless it is complemented by separately derived version information [27].

Registries with more complete and granular coverage of file format information have been developed since the introduction of MIME and may serve as alternative authority files for web archive preservation metadata. Principal among these is PRONOM, the technical registry developed and maintained by the UK National Archives [39]. PRONOM provides a regularly updated database of digital file formats, their version histories, technical components, and brief descriptions. Structured "signature files" of data specific to all of these formats—more than 700 to date—are in turn derived for use by automated file analysis software tools. One such tool, the UK National Archives' DROID (Digital Record and Object Identification) [21], is already incorporated into the JHOVE2 architecture. Rhizome has used Siegfried [40], the similarly PRONOM-based file identification tool to analyze the same information in the contents of WARC files that it has harvested with the browser-based crawling robot alternative Webrecorder [41, 42]. BnF uses the file Unix command [25] to perform identification at the harvested file level.

[9]Vendors of the proprietary digital preservation platform Preservica claimed in 2015 that their product could perform this format validation and characterization for W/ARCs at the level of the harvest file. However, tests conducted by the New York Art Resources Consortium (NYARC) in 2015 confirmed its reliance upon JHOVE to perform these operations, and its subsequent failure to output valid, PREMIS-conformant METS metadata manifests. See [38].

Table 6.3 PREMIS Object metadata

Information type	PREMIS semantic unit	PRONOM corresponding attribute	Example
Name of file format	1.5.4.1.1 *formatName*	Name	Graphics Interchange Format
Version of file format	1.5.4.1.2 *formatVersion*	Version	89a
Authority file or registry for above information	1.5.4.2.1 *formatRegistryName*	(N/A, not variable)	PRONOM
Unique identifier or key for above file format entry in above authority file or registry	1.5.4.2.2 *formatRegistryKey*	Identifier/PUID	fmt/4

To address issues of scale, preservation metadata generated by such tools at the level of each harvested file could be limited to the required PREMIS Object semantic units in Table 6.3.

In accordance with the container model for web archives, further agent, event, and rights semantic unit metadata could continue to be consolidated at the W/ARC container *File* level.

From an emulation perspective, the original browsing environment can be documented in order to replay captured web content in that original browsing environment. This is a sizeable scalability challenge, as this browsing environment includes not only the web browser but also the browser plug-ins, software libraries and operating system installed on the computer. Compatibility issues make it important to record the version of each of those pieces of software, e.g., to know which browser is compatible with which plugin.

No institutional precedent has yet been set for systematically managing this extensive, highly complex, and expensive operation. However, important information can nonetheless be periodically recorded; web archiving programs may record the operating system, browser version, plug-ins versions, and software libraries that they employ in their efforts. While imperfect, this is a pragmatic way to record, from time to time, one or a select few rendering environments that best support the accurate replay of web content captured at that time. Following this idea, the IIPC members are recording such rendering environments in the context of the Preservation Working Group.[10]

[10]See Sect. 4.2 of [43].

6.6 Conclusion

In a way, preserving web content over the long term illustrates all the greatest preservation challenges: the variety of file formats is potentially infinite; the logical structure of the original web content website is often very different from the way it is physically stored and preserved; most of the time, the preservationist has little or no control of the way the web content is produced; finally, the number of harvested files poses scalability issues, which forces one to adopt realistic answers to such challenges.

The BnF approach is an attempt to face such challenges, but scalability issues still have to be tackled, especially for describing every single file's characteristics. The content model presented in part 4 of this chapter can be used extensively for small collections (e.g., archiving of a single institutional website) but only the upper levels can be fully described for more sizeable collections. Describing and documenting web archives' rendering environments for a certain period of time is a possible answer to face such challenges. The enhancements in PREMIS version 3.0 provide us with the ability to describe complex environments to support future use of web-archived content.

References

1. Bonnel S, Oury C (2014) Selecting websites in an encyclopaedic national library: a shared collection policy for internet legal deposit at the BnF. Paper presented at the 80th IFLA WLIC conference, Lyon, France. http://library.ifla.org/998/. Accessed 22 Mar 2016
2. Hockx-Yu, Helen (2014) Archiving social media in the context of non-print legal deposit. Paper presented at the 80th IFLA WLIC conference, Lyon, France. http://library.ifla.org/999/. Accessed 22 Mar 2016
3. Brügger N (2005) Archiving websites: general considerations and strategies. Center for Internetforskning, Aarhus Universitet, Aarhus, 76 pp
4. Brügger N (2015) Humanities, digital humanities, media studies, internet studies: an inaugural lecture. Center for Internetforskning, Aarhus Universitet, Aarhus, 16 pp
5. Graham S, Milligan I, Weingart S (2015) Exploring big historical data: the historian's macroscope. Imperial College Press, London, 400 pp
6. Illien G, Sanz P, Sepetjan S, Stirling P (2011) The state of e-legal deposit in France: looking back at five years of putting new legislation into practice and envisioning the future. IFLA J 38(1). http://www.ifla.org/files/assets/hq/publications/ifla-journal/ifla-journal-38-1_2012.pdf. Accessed 23 Oct 2016
7. IIPC: International Internet Preservation Consortium (2016) Official web site. http://netpreserve.org. Accessed 01 Feb 2016
8. Consultative Committee for Space Data Systems, International Organization for Standardization (2012) Reference model for an Open Archival Information System (OAIS): issue 2. http://public.ccsds.org/publications/archive/650x0m2.pdf. Accessed 06 Jan 2016
9. http://www.exlibrisgroup.com/category/RosettaOverview
10. Preservica (2016) http://preservica.com/. Accessed 20 Mar 2016
11. The Internet Archive (2016) Wayback machine. http://archive.org/web. Accessed 01 Feb 2016

12. National Library of New Zealand, British Library (2015) Web curator tool. http://sourceforge.net/projects/webcurator/. Accessed 01 Feb 2016
13. Royal Library of Denmark, State and University Library of Aarhus (2015) NetarchiveSuite. https://sbforge.org/display/NAS/NetarchiveSuite. Accessed 01 Feb 2015
14. The Internet Archive (2016) ArchiveIt! https://Archive-It.org/. Accessed 01 Feb 2016
15. Bermès E, Fauduet L, Peyrard S (2010) A data first approach to digital preservation: the SPAR project. Paper presented at the 76th IFLA general conference and assembly, Gothenburg. http://conference.ifla.org/past-wlic/2010/157-bermes-en.pdf. Accessed 01 Feb 2016
16. Le Follic A, Stirling P, Wendland B (2013) Putting it all together: creating a unified web harvesting workflow at the Bibliothèque nationale de France. http://hal-bnf.archives-ouvertes.fr/docs/00/87/37/59/PDF/Putting_it_all_together.pdf
17. BnF (2009) The WARC file format (ISO 28500): information, maintenance, drafts. http://bibnum.bnf.fr/warc. Accessed 01 Feb 2016
18. Burner M, Kahle B (1996) Arc file format. http://archive.org/web/researcher/ArcFileFormat.php. Accessed 01 Feb 2016
19. Oury C (2010) Large-scale collections under the magnifying glass: format identification for web archives. Paper presented at the 7th international conference on preservation of digital objects (iPRES), Vienna. https://hal-bnf.archives-ouvertes.fr/hal-00769091. Accessed 01 Feb 2016
20. Abrams S, Cramer T, Morrissey S (2009) "What? So What": the next-generation JHOVE2 architecture for format-aware characterization. Int J Digit Curation 4(3):132–136. doi:10.2218/ijdc.v4i3.122
21. The National Archives (2016) File profiling tool (DROID). http://www.nationalarchives.gov.uk/information-management/manage-information/policy-process/digital-continuity/file-profiling-tool-droid. Accessed 02 Feb 2016
22. Open Planets Foundation (2014) JHOVE2: the next-generation architecture for format-aware characterization. https://github.com/opf-labs/jhove2. Accessed 01 Feb 2016
23. Gones J, Oury C, Steinke T (2012) Ensuring long-term access to the memory of the web: preservation working group of the international internet preservation consortium. Int Preserv News 28:34–37. http://www.ifla.org/files/assets/pac/ipn/ipn-58.pdf. Accessed 02 Feb 2016
24. Oury C, Peyrard S (2011) From the World Wide Web to digital library stacks. Paper presented at the 8th international conference on preservation of digital objects (iPRES), Singapore. https://halshs.archives-ouvertes.fr/halshs-00868729. Accessed 02 Feb 2016
25. Wikimedia Foundation (2016) File (command). https://en.wikipedia.org/wiki/File_%28command%29. Accessed 13 Feb 2016
26. Hockx-Yu H (2015) The unknown aspects of web archives. https://hhockx.wordpress.com/2015/08/11/7/. Accessed 22 Mar 2016
27. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 05 Jan 2016
28. Bibliothèque nationale de France (2011) ContainerMD. http://bibnum.bnf.fr/containerMD-v1. Accessed 03 Jan 2016
29. Pearson D, Webb C (2008) Defining file format obsolescence: a risk journey. Int J Digit Curation 3(1):89–106. doi:10.2218/ijdc.v3i1.44
30. Jackson A (2015) Ten years of the UK web archive: what have we saved? UK web archive blog. http://britishlibrary.typepad.co.uk/webarchive/2015/09/ten-years-of-the-uk-web-archive-what-have-we-saved.html. Accessed 02 Feb 2016
31. New York Art Resources Consortium (NYARC) (2015) Official website. Accessed 02 Feb 2015
32. Internet Archive Webteam (2015) Archive-it storage and preservation policy. https://web.archive.org/web/20150920070827/https://webarchive.jira.com/wiki/display/ARIH/Archive-It+Storage+and+Preservation+Policy. Accessed 02 Feb 2016

33. Duncan S (2015) Preserving born-digital catalogues raisonnés: web archiving at the New York art resources consortium. Art Librar J 40(2):50–55. http://web.archive.org/web/20151120180335/http://www.nyarc.org/sites/default/files/duncanALJ.pdf. Accessed 02 Feb 2016
34. Persons S (2015) MoMA.org Turns 20: archiving two decades of exhibition sites. INSIDE/OUT, 25 May 2015. http://web.archive.org/web/20150527195327/http://www.moma.org/explore/inside_out/2015/05/25/moma-org-turns-20-archiving-two-decades-of-exhibition-sites?. Accessed 02 Feb 2016
35. Kreymer I (nd) oldweb.today. http://oldweb.today. Accessed 10 Apr 2010
36. University of Illinois at Urbana-Champaign, Grainger Engineering Library Information Center (2006) ECHO Dep METS profile for web site captures. http://www.loc.gov/standards/mets/profiles/00000016.html. Accessed 02 Feb 2016
37. Guenther R, Myrick L (2008) Archiving web sites for preservation and access: MODS, METS and MINERVA. J Arch Organ 4(1–2):141–166. doi:10.1300/J201v04n01_08
38. Blumenthal KR (2015) Preserving NYARC's web archives: a steep towards long-term stewardship. http://web.archive.org/web/20150623183326/http://static1.squarespace.com/static/51c07825e4b0b892821e029d/t/5589a643e4b077187933b441/1435084355126/NYARCWebArchiveStorageandPreservation+%281%29.pdf. Accessed 02 Feb 2016
39. The National Archives (2016) The technical registry PRONOM. https://www.nationalarchives.gov.uk/PRONOM. Accessed 02 Feb 2016
40. Lehane R (2016) Siegfried. http://www.itforarchivists.com/siegfried. Accessed 02 Feb 2016
41. Rhizome (2016) Welcome to webrecorder beta. https://webrecorder.io. Accessed 02 Feb 2016
42. McKeehan M (2015) Preserving variability. The National Digital Stewardship Residency, New York. http://web.archive.org/web/20151205234527/http://ndsr.nycdigital.org/preserving-variability/. Accessed 02 Feb 2016
43. Goethals A, Oury C, Pearson D, Sierman B, Steinke T (2015) Facing the challenge of web archives preservation collaboratively: the role and work of the IIPC Preservation Working Group. D-Lib Mag. doi:10.1045/may2015-goethals

Chapter 7
Digital Preservation Metadata Practice for E-Journals and E-Books

Amy Kirchhoff and Sheila M. Morrissey

7.1 Introduction

Portico [1] is a not-for-profit digital preservation service and is among the largest community-supported digital archives in the world. Working with libraries, publishers, and funders, Portico preserves e-journals, e-books, and other digital scholarly content to ensure researchers and students will have access to these resources in the future.

As of December 2015, Portico was preserving more than 53 million journal articles, more than 530,000 books, and nearly 3.3 million items from digitized historical collections (d-collections, for example, digitized newspapers of the eighteenth century) with more than 21,000 e-journals committed to the archive. Portico's approach to preserving this content addresses the key goals of digital preservation: usability—the intellectual content of the item must remain usable via the delivery mechanism of current technology; authenticity—the provenance of the content must be proven and the content an authentic replica of the original; discoverability—the content must have logical bibliographic metadata so that the content can be found by end users through time; and accessibility—the content must be available for use to the appropriate community. Portico meets the rigor of these goals through a migration-based strategy; Portico will migrate or transform the preserved content from one file format to another as technology changes. Portico supplements and supports this migration policy by preserving the original source files along with the migrated versions. In addition, Portico has developed a technology and application-independent archive (for example, the Portico archive

A. Kirchhoff (✉) · S.M. Morrissey
ITHAKA, Portico 100 Campus Drive, Suite 100, Princeton, NJ 08540, USA
e-mail: amy.kirchhoff@ithaka.org

S.M. Morrissey
e-mail: sheila.morrissey@ithaka.org

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_7

can be exported onto a standard file system with all the information necessary to understand the contents of the archive in organized files).

The nature of e-journal and e-book content presents several challenges. One is that the content is delivered in a large variety of proprietary formats (on average, Portico receives e-journal and e-book content in 2.5 formats per publisher) and Portico's arrangements with publishers is such that we cannot require specific submission formats. Simultaneously, an e-journal and e-book preservation service requires robust descriptive metadata, as the preservation service needs to guarantee that it can deliver content within the framework of a future journal or book delivery service (whereas an archive of web pages may be willing to deliver content as-is today and in the future, relying on the then current technology of the Internet to resolve rendition issues). Another challenge in this space is that the published content is not as static as it appears. One benefit to the academy of electronic publication is that publishers can easily correct metadata errors or errors in the article PDFs and such. A preservation service, therefore, must be able to handle receipt of the same article or e-book multiple times.

PREMIS [2] provides one way to model content for preservation and to attach preservation metadata to it. PREMIS may also be used as a point of reference for organizations which need to design an information architecture to meet their specific needs. Standards are very useful in providing both a framework for thinking about a topic and as an interchange specification so that multiple organizations can exchange data without a tremendous amount of up front negotiation. Portico e-journal, e-book, and digitized collection preservation services leverage PREMIS as a framework for thinking about preservation metadata and content models. Although we do not explicitly use PREMIS internally, Portico has structured preservation metadata in such a way that it could be exported in PREMIS for delivery to a third party if need be.

7.2 Preserving E-Journals and E-Books

As of February 2016, over 60 million e-journal article DOIs[1] were registered with CrossRef across 43,000 journals,[2] 10 million e-book DOIs were also registered. The number of DOIs is a useful "stand-in" for the number of e-journal articles and e-books to be preserved. Most in the library and publishing community understand that there are a significant number of journal publications that have not registered articles for DOIs [3], thus that the actual number of published e-journal articles is greater than 60 million. By Portico's rough estimate, between 1 and 2 million articles will be published every year for the next decade. We are not able to estimate the growth of future scholarly e-book publications yet, due to the young age of the

[1]CrossRef Status, http://www.crossref.org/06members/53status.html.

[2]CrossRef Title List, http://www.crossref.org/titlelist/titleFile.csv.

market, but growth will likely be substantial. Thus, there are a very large number of items to be preserved within the e-journal and e-book space now and in the future.

E-journals are comprised of articles—articles which may or may not be published within a volume and issue hierarchy. While most journals are delivered to end users as volumes consisting of issues and issues consisting of articles, we are increasingly seeing journals with a single volume published each year, without issues, where the articles are made available to end users within the volume as they are ready. At the moment, scholarly e-books are published similar to scholarly journals, with the book delivered to users like an e-journal issue and the chapters delivered to users like articles. However, as e-book publishing increases and end users become more experienced with reading and interacting with complete e-books on dedicated reading devices in their non-scholarly lives, the scholarly e-book delivery mechanism may change such that the e-book content is independent of an overarching website. At their base, though, e-journals and e-books (and most content that has a traditional "publication" model) look very similar. There are a series of published objects (articles or chapters) that are grouped together into hierarchies—hierarchies of issues, volumes, and journals or hierarchies of book and series.

Portico has created a simple six-level container-based content model to preserve these types of scholarly objects. While it is specific to Portico, our content model was influenced by PREMIS [2], DIDL [4], OAIS [5], and our own experiences. It consists of the following:

- *Content Type*: A content type is a container that groups together content that would look similar in a delivery interface *and* belongs to the same preservation service. For example, Portico preserves historical collections of digitized books and while we preserve it in the same manner that we preserve current e-books; they are preserved as separate content types because the business model supporting each is different and therefore the manner in which Portico would deliver each differs (the content type gives us a way to identify a set of similar items at once). A Portico Content Type maps to a PREMIS *Intellectual Entity*. In regard to implementation, it could also be captured as an element of metadata on the Archival Unit, rather than encoding it as an *Intellectual Entity*. Portico would make that choice through discussion with the recipient.
- *Content Set*: Each content type contains one or more content sets. A content set is a grouping level. For books, the content set is the publisher, whereas for journals the content set is the journal title. A Portico Content Set maps to a PREMIS *Intellectual Entity*. In regard to implementation, it could also be captured as an element of metadata on the Archival Unit, rather than encoded as an *Intellectual Entity*. That choice would be made through discussion with the recipient of the content.
- *Archival Unit*: Each content set contains one or more archival units. The archival unit is the Portico unit of preservation. The archival unit is an abstraction of a publication unit (for e-journals, the archival unit is the article and for e-books, the archival unit is the book). The content type and content set are tracked as metadata elements on the archival unit. A Portico archival unit maps to a PREMIS *Intellectual Entity*.

- *Content Unit*: Each archival unit contains one or more content units, and each content unit is one complete version of the archival unit. For example, if the publisher were to deliver an article to Portico in January and then deliver another version in September with a corrected title, both versions would be preserved, each as its own content unit, within the single archival unit for the article. This addresses the update challenge in the preservation of e-journal and e-book content. A Portico content unit maps to a PREMIS *Intellectual Entity*.
- *Functional Unit*: Functional units provide a way to group together files that serve the same purpose or function in the content unit. For example, if a figure graphic were to be provided from the publisher in three files—a high resolution print-ready TIFF, a web-ready GIF, and a thumbnail GIF—all three would be preserved together in a single functional unit. A Portico functional unit maps to a PREMIS *Intellectual Entity*.
- *Storage Unit*: Storage units represent files (one storage unit per file). A Portico storage unit maps to a PREMIS *File*.

It is noteworthy that within the Portico content model, there is no concept that explicitly maps to the PREMIS *Representation* entity. Per PREMIS, a *Representation* is the set of *Files* required to display an *Intellectual Entity* to a human [2]. The Portico content model presumes that a delivery system will use the XML file for the book or journal article to create *Representations* from the files preserved (in some cases, as shown below, the *Representations* are driven by the Portico preservation metadata file, itself). In other terms, *Representations* are generated on-the-fly at the time of delivery. For example, here is the navigation provided for an article in the Portico audit interface (where librarians and publishers can confirm that Portico is preserving the content it claims):

Each tab in Fig. 7.1 is a "*Representation*" of the article.

Fig. 7.1 Portico audit interface-e-journal article

- The PDF tab is a list of all the PDF files in the book.
- The HTML tab is an HTML rendition of the full-text XML of this article.
- The Preservation Information tab provides a high-level overview of the preservation status of the item and is driven by the Portico preservation metadata file.
- The Tree View tab provides a tree-like view of the Portico content model and is driven by the Portico preservation metadata file.
- The PMD tab provides the actual Portico preservation metadata file.
- The XML tab provides the Portico normalized XML file for this e-book (to Portico staff and the publisher, this tab also provides the original XML file).

If a recipient of Portico content in PREMIS format needed explicit *Representations*, Portico would create them in the same manner we do on the delivery websites.

In terms of preservation metadata, descriptive, technical, event, and rights metadata can be attached at any point in the Portico content model—for example, a content type, content set, archival unit, content unit, functional unit, or storage unit could all have descriptive metadata attached. The structural metadata is represented by the content. Portico also uses this content model to support digitized newspapers, digitized books, digitized documents, the JSTOR archive collections content, and a number of archive management categories of content (publisher license agreements, DTDs, etc.). This model would be appropriate for any content that is published in the traditional sense of the word—any content that can be deconstructed into a set of files and a set of metadata.

7.3 Preservation Metadata File

The preservation metadata for each preserved object should be able to be exported to a single file that can be moved with the preserved object from system to system (while metadata may be stored as separate elements, it is extremely useful to present the entirety of the preservation metadata as a single file to aid in human comprehension of the data). This file should be both machine and human readable. Organizations may choose to cache some elements of the preservation metadata in a database for easy access. There are number of file format choices when it comes to preservation metadata, including a pure PREMIS-based format. At Portico, we developed an internal preservation metadata format called PMD (preservation metadata). PMD maps closely to the Portico content model and provides the structural metadata for content preserved by Portico. If needed, Portico can export the PMD to pure PREMIS or METS [6] with PREMIS or another format within the METS to capture the preservation metadata. Portico also caches the preservation metadata in an Oracle database to make it easily accessible for archive management purposes and to build websites to deliver the content to end users.

7.3.1 Structural Metadata

Often, the relationships between files in a complex preserved object are implicitly encoded in the structure of the preserved object. For example, at Portico the relationships between different content units or functional units or storage units are represented by the level these units play in the overall content model, and the implicit whole-part relationships they have with each other. In order to represent Portico structural metadata in PREMIS, these implicit relationships would be made explicit through relationship semantic units.

Figure 7.2 is a high-level XML summary of the PMD and content model structure of an e-journal article from Astronomy Education Review[3] (an open-access title that Portico has triggered).

7.3.2 Descriptive Metadata

Portico's preservation philosophy and delivery needs require that descriptive metadata is an integral element of our preservation metadata. Publishers, libraries, and the preservation organization itself will have need to find the preservation status of specific items where the only unit of identification for the preserved object is a bibliographic citation. At Portico, descriptive metadata is also required to determine whether or not we have preserved all of the content which has been committed. At Portico, we attach descriptive metadata to both the Portico archival unit and the content unit. The descriptive metadata lives in the preservation metadata file and we also cache a copy in an Oracle database to make it easier to manage the archive and find specific items. The descriptive metadata is also used in the Portico audit and access interfaces when making the archival units accessible on the web.

While descriptive metadata is required on at least the archival unit and content unit, it may be placed on any unit of the Portico content model. All Portico archival units (whether book, journal article, or d-collection item) have Dublin Core [7] descriptive metadata attached. This uniformity allows us to manage the archive and to display the content consistently in a web interface, regardless of the content type (Fig. 7.3).

At Portico we also put descriptive metadata on the content units using a content type-specific XML structure. For example, for journals we use the JATS [8] standard (Fig. 7.4).

This particular implementation choice allows Portico to make content available on a website very quickly using the general Dublin Core metadata, but still provides the opportunity to make the content available in a more specialized way as necessary, without reprocessing the XML files of each article.

[3]Article on Portico delivery site is available to all, http://portico.org/stable?au=pgg3ztfcvw0.

Fig. 7.2 Portico PMD structure

```
<DescriptiveMetadata objID="ark:/27927/pgg3ztfr4ss" created="2013-12-24T14:54:34.031-05:00"
 formatName="Ithaka Dublin Core Metadata:1.0:2009-11-17" mimeType="application/xml">
 <DC xmlns="http://www.ithaka.org/standards/Public/XML/schema/IthakaDublinCoreMetadata/1.0/"
  xsi:schemaLocation="http://www.ithaka.org/standards/Public/XML/schema/IthakaDublinCoreMetadata/1.0/
  <title xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/">Modeling the
   History of Astronomy: Ptolemy, Copernicus, and Tycho</title>
  <creator xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/">Todd K.
   Timberlake</creator>
  <date xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/">December
   2013</date>
  <identifier xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/"
   type="bibCitation">Todd K. Timberlake. "Modeling the History of Astronomy: Ptolemy, Copernicus,
   and Tycho," Astronomy Education Review 12 no. 1 (December 2013) 010201</identifier>
  <identifier xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/"
   type="bibCitationOpenUrl"
   >ctx_ver=Z39.88-2004&amp;rft_val_fmt=info%3Aofi%2Ffmt%3Akev%3Amtx%3Ajournal&amp;rft.au=Todd%20K.%2(
  <language xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/"
   >en</language>
  <type xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/">E-Journal
   Content</type>
  <identifier xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/"
   source-elem="journal-id" type="publisher-id">AER</identifier>
  <identifier xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/"
   source-elem="issn" type="issn">15391515</identifier>
  <identifier xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/"
   source-elem="article-id" type="publisher-id">006301AER</identifier>
  <identifier xmlns="http://www.ithaka.org/standards/Public/XML/schema/dc-elements/1.1/"
   source-elem="article-id" type="doi">10.3847/AER2013001</identifier>
 </DC>
</DescriptiveMetadata>
```

Fig. 7.3 Portico Dublin Core metadata on the archival unit for an e-journal article

7.3.3 Event Metadata

Actions can be taken on any element of a content model and each preservation organization must decide which actions should be recorded for posterity—per PREMIS, "it is up to the repository which actions to record as events" [2: p. 15]. It may be appropriate for some organizations to record every action taken and other organizations may choose to record only specific kinds of actions. As with descriptive metadata, a preservation organization could choose to tailor event metadata by content type. At Portico we track the same events on each archival unit no matter the content type, as we have found that our processing and preservation of e-books, e-journals, and d-collections is so similar, that our events are all generic. Portico's event model closely aligns with the PREMIS Event model.

At the Portico archival unit level we track two events:

- *Create file*: Portico has added a file to the content provided by the publisher (this could be the Portico XML file created or a file downloaded from the publisher's website and inserted into the files being preserved).
- *Generate descriptive metadata*: This event occurs when Portico pulls descriptive metadata out of the publisher supplied content and creates a descriptive metadata section on the archival unit (See example in Fig. 7.5).

```
<DescriptiveMetadata objID="ark:/27927/pgg3ztfq2wx" created="2013-12-24T14:54:17.091-05:00"
 formatName="Portico Article Metadata:1.3:2012-06-06" mimeType="application/xml">
 <PorticoArticleMetadata
  xmlns="http://www.portico.org/standards/Public/XML/schema/PorticoArticleMetadata/1.3/"
  display-label="v12 n1 010201 December 2013" metadata-type="curated" schema-version="1.1"
  sort-key="201312          12              1               010201"
  xsi:schemaLocation="http://www.portico.org/standards/Public/XML/schema/PorticoArticleMetadata/
  <article article-type="article" xml:lang="EN">
   <front>
    <journal-meta>
     <journal-id journal-id-type="publisher-id">AER</journal-id>
     <journal-title>Astronomy Education Review</journal-title>
     <issn>15391515</issn>
    </journal-meta>
    <article-meta>
     <article-id pub-id-type="coden">AERSCZ</article-id>
     <article-id pub-id-type="sisac">1539-1515()12:1L.010201;1</article-id>
     <article-id pub-id-type="publisher-id">006301AER</article-id>
     <article-id pub-id-type="doi">10.3847/AER2013001</article-id>
     <title-group>
      <article-title>Modeling the History of Astronomy: Ptolemy, Copernicus, and
       Tycho</article-title>
     </title-group>
     <contrib-group>
      <contrib>Todd K. Timberlake</contrib>
     </contrib-group>
     <pub-date pub-type="cover">
      <month>12</month>
      <year>2013</year>
      <string-date>December 2013</string-date>
     </pub-date>
     <volume>12</volume>
     <issue>1</issue>
     <elocation-id>010201</elocation-id>
     <copyright-year>2013</copyright-year>
     <copyright-holder>The American Astronomical Society</copyright-holder>
    </article-meta>
   </front>
  </article>
 </PorticoArticleMetadata>
</DescriptiveMetadata>
```

Fig. 7.4 Portico JATS descriptive metadata on the content unit for an e-journal article

At the Portico content unit level, four possible events are tracked:

- *Check descriptive metadata*: This is a double check on the descriptive metadata, to ensure we have all the elements we expect (See example in Fig. 7.5).
- *Edit descriptive metadata*: This occurs very rarely when we manually edit the descriptive metadata in the preservation metadata file.
- *Generate descriptive metadata*: This event occurs when Portico pulls descriptive metadata out of the publisher supplied content and creates a descriptive metadata section in the preservation metadata for the content unit.
- *Ingest into archive*: Portico tracks the moment when the content unit is ingested into the archive.

```
<EventSet objID="ark:/27927/pgg3ztfq2xg" processingRecordID="ark:/27927/pgg3ztfr65w"
 conformanceVersion="1">
 <Event objID="ark:/27927/pgg3ztfq2z1" eventType="Generate Descriptive Metadata">
  <Timestamp>2013-12-24T14:54:17.091-05:00</Timestamp>
  <InputList>
   <Input refID="ark:/27927/pgg3ztfm3q7"/>
  </InputList>
  <Output refID="ark:/27927/pgg3ztfq2wx"/>
  <ToolWrapper name="DmdExtractTool:1.0:2007-08-03"/>
  <ToolComponentList>
   <ToolComponent name="com.saxonica.config.ProfessionalTransformerFactory"/>
   <ToolComponent name="PorticoArticleMetadataExtract_2_6.xsl"/>
  </ToolComponentList>
 </Event>
 <Event objID="ark:/27927/pgg3ztfq54b" eventType="Check Descriptive Metadata">
  <Timestamp>2013-12-24T14:54:17.656-05:00</Timestamp>
  <InputList>
   <Input refID="ark:/27927/pgg3ztfq2wx"/>
  </InputList>
  <ToolWrapper name="DmdCurationTool:1.0:2005-04-26"/>
  <ToolComponentList>
   <ToolComponent
    name="org.portico.threadedtool.tool.dmd_curation_extract_1_0.Dmd_Curation_Extract_1_0"/>
  </ToolComponentList>
  <Outcome>Success</Outcome>
 </Event>
</EventSet>
```

Fig. 7.5 Two common Portico events

In all events, Portico tracks the same categories of information:

- *Event type*: The type of events are controlled by a delimited list. In PREMIS this would be modeled as the semantic unit *eventType* on the Event Entity.
- *Time stamp*: The time and date the event took place. In PREMIS this would be modeled as the semantic unit *eventDateTime* on the Event Entity.
- *Input list*: A list of files that served as an input to the action that caused this event (if there was such a list). Within the PREMIS Event Entity, these would be represented as *linkingObjectIdentifiers*. In addition, the PREMIS *File* object for each input file would have a relationship semantic unit with each output file and vice versa.
- *Output list*: A list of files created as an output to the action (if there was such a list). Within the PREMIS Event Entity, these would be represented as *linkingObjectIdentifiers*. In addition, the PREMIS *File* object for each output file would have a relationship semantic unit with each input file and vice versa.
- *Tool name*: The name of the tool used in the action. Within PREMIS, the tool would be modeled as an Agent, in which the tool name may be captured with semantic units such as *agentName*, *agentType*, and *agentVersion*.
- *Tool registry version*: All tools Portico uses in its workflow are captured in a tool registry, which is versioned. In order to track the name of a tool back to a specific piece of software, it is necessary to know which version of the registry was active when this action took place. Within PREMIS, this would be recorded in the Event entity with a *linkingAgentIdentifier*.

```
<ProcessingRecord objID="ark:/27927/pgg3ztfr65w" created="2013-12-24T14:54:34.305-05:00">
 <Timestamp>2013-12-24T14:54:34.305-05:00</Timestamp>
 <Rationale>Processing of New Content</Rationale>
 <SuppliedFiles objID="ark:/27927/pzd1h8cjtn"/>
 <System name="ConPrep" version="2.0"/>
 <Workflow name="E-Journal Workflow 2.1" version="1.0"/>
 <Batch objID="ark:/27927/pgg3ztbjwbr"/>
 <Profile name="AAS Current AIP Profile" version="1.0"/>
 <RuntimeEnv OS="Linux:amd64:2.6.18-194.el5" VM="Java:Sun Microsystems Inc.:1.6.0_22"/>
 <ToolRegistry date="2013-12-24-05:00" version="4.04"/>
</ProcessingRecord>
```

Fig. 7.6 Sample Portico processing record from initial processing of content

- *Components*: Any specific components used by the tool (for example, a specific DTD). Within PREMIS, this would be modeled as an *agentExtension* semantic unit on the Agent.
- *Runtime environment*: A very brief one-line description of the operating environment in which the event took place. Within PREMIS, this would be modeled as a linked environment.[4]
- *Processing record*: A reference back to a processing record (see below for more information on processing records).

Below are two common Portico events as encoded in PMD:

Portico originally employed a METS-based preservation metadata file. In 2010, Portico moved to a new preservation metadata format, the aforementioned PMD file. As seen above, the Portico PMD file is closely aligned to the Portico content model and this alignment is one substantial difference from our previous METS-based preservation metadata file. One other substantial change we made was to move environment information out of event records and into a new statement called a processing record—each event is tied to a single processing record (see Fig. 7.6). Portico's original model assumed that events might occur across multiple environments, however experience clarified that our processing was going to be centralized. Moving the environment information from being stamped on individual events to processing records removed a large amount of duplication from the Portico preservation metadata files. Were Portico to export PREMIS to a third party, we would decide together how best to represent the processing record information for the other party's purposes, whether to move it to each event or encode it as a linked environment.

[4]See Chap. 10 for more on Environments and how they are handled in PREMIS 3.

```
<File
 checksum="8A4AEDAE96A5E6ACF26A25B2BBA2EB0DDDAA7C037A214CD2D3C1229F8C2A26EAEBC177B5FD08D199C18C5
 checkSumType="SHA-512"
 format="Joint Photographic Experts Group File Interchange Format:1.01:1992"
 formatAssessment="Well Formed and Valid" fileSize="29738" mimeType="image/jpeg"
 originalFileName="AERSCZ-vol_12-iss_1.tar/./AERSCZ/vol_12/iss_1/webimages_010201_1.zip/4.jpg"
 archiveFileName="pgg3ztf5rzn.jpg" reidentified="false"/>
```

Fig. 7.7 Portico file entry from a PMD file

7.3.4 Technical Metadata

Technical metadata provides sufficient information to a preservation organization to allow it to manage its files for the long term. Each organization must decide what pieces of information are required. At Portico we have chosen to capture an abbreviated version of the JHOVE [9] output (we do not capture the entirety of the JHOVE output—should we need it in the future, we will run individual files through JHOVE again at that point). Portico also uses the BSD file utility [10] to identify the format of files. On the file proper, in addition to capturing a portion of the JHOVE output, we also track (these all synchronize well with PREMIS *File* Entity semantic units) (Fig. 7.7):

- The original file name and submission information package path.
- The mime type of the file.
- The formal file name of the file (a name that maps to an entry in the Portico file format registry—an XML file that describes all the formats found in the Portico archive).
- The size of the file.
- The file format assessment (does this file validate to its file format standard?).
- A checksum type and value for the file.

Portico does not currently permit encrypted files or DRM limited files. If our processing detects such content, the items will not be preserved until such time as the publisher delivers unencrypted and DRM free files as replacements. As such, we do not need to record this type of information in our PMD files for our current preservations services.

In many cases, it is important to know the exact format of the files in an archive (For a limited number of more esoteric file formats, an organization may choose to not to track these details). Portico preserves a tremendous amount of content—currently more than 53 million articles, 530,000 books, 3.2 million digitized historic items, and 1 billion files. Should we need to migrate files of specific formats in the future, we need to be able to identify files in just those formats in the archive without touching all the files we have preserved. This illustrates the practical necessity of capturing technical metadata around the file format.

```
<StorageUnit objID="ark:/27927/pgg3ztf5rzn" created="2013-12-24T14:53:08.646-05:00"
 origin="Provider" originRationale="Supplied Content" preservationLevel="Fully Supported"
 preservationLevelRationale="Format Assessment" status="Active" statusRationale="">
```

Fig. 7.8 Preservation metadata on the Portico storage unit

7.3.5 *Preservation Metadata for Archive Management*

In addition, on the storage unit that represents the file, we track (these synchronize well with PREMIS Entity semantic units) (Fig. 7.8):

- The status of the file (is this file active or inactive within its functional unit—for example, when me migrate the original publisher supplied XML file to JATS [8] or BITS [11], the newly migrated file is given an "active" status and the original supplied file is given an "inactive" status).
- The preservation level of the file (what level of commitment will Portico make to this particular file). Portico's preservation levels include: Fully Supported, Reasonable Effort, and Byte Preserve. The assignment of a preservation level to any given file is based both on business policy and whether or not the file validates against its format specification. The only file that must be valid before the item can be preserved at Portico is the XML file provided by the publisher.
- Information about the source of the file (did it come from the provider or did Portico create the file during the preservation process).

7.3.6 *Rights Metadata*

Rights metadata can be as simple as stating the content owner or as complicated as digital rights management terms and software that specify extensive terms of use for content. At Portico, our license agreements with the publishers specify the same terms of use for each preserved item. Portico preserves the license agreements in the Portico archive and places a reference to the license agreement in the preservation metadata of each archival unit preserved, and this fulfills our limited need for rights metadata (Fig. 7.9).

```
<ArchivalUnit objID="ark:/27927/pgg3ztfcvw0" created="2013-12-24T14:54:38.088-05:00"
 agreementID="ark:/27927/pt2vd6"
 agreementName="American Astronomical Society Agreement, Version 1.0 (April 4, 2008)"
 contentSetName="ISSN_15391515" contentType="E-Journal Content" contentUnitType="Article"
 preservationObjective="Fully Supported" preservationObjectiveRationale="Business Policy"
 status="Active" statusRationale="">
```

Fig. 7.9 Portico archival unit element with reference to license agreement

7.4 Lessons Learned

While e-journal and e-book content may be delivered to preservation agencies by publishers in many (many!) different formats, they are stable and well-understood scholarly constructs, with defined publication hierarchies. In addition, their delivery to end users is well-understood. Their challenges are tied to the tremendous variation in input packaging formats and the very large amount of content that exists.

There is no universally correct amount of preservation metadata to capture. Portico has scoped out an appropriate amount of preservation metadata to capture for our purposes of preserving a very large amount of e-journal, e-book, and digitized collection content for the long term and occasionally needing to provide access to this content to faculty, staff, students, and publishers. The scale of the archive requires that we capture enough metadata to find content again either by bibliographic citation or file format or preservation status. The fact that Portico must occasionally provide access to the content also drives decisions about what metadata to capture (for example, we create Dublin Core descriptive metadata for every archival unit preserved—including license agreements, DTDs, e-books, e-journal articles, etc.). A preservation organization with a different mandate and different content may reasonably make very different choices.

Most importantly: nothing is ever final. Technology costs, including the cost of disk and processing power, reduce year to year and thus it is reasonable to assume preserved content may be reprocessed in the future such that different metadata is captured. The most important factor is that the content be preserved by an organization committed to the content for the long term—with that commitment in place and the attention the commitment engenders, the content will be safe for the long term and available for users in the future.

References

1. ITHAKA (2016) Portico: your content. Preserved here. www.portico.org/. Accessed 21 Feb 2016
2. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 29 December 2015
3. Burnhill P, Pelle F (2013) Who is looking after your e-journals? Telling tales about the keepers registry & your digital shelves. Paper presented at the 79th IFLA conference (Singapore). http://library.ifla.org/121/1/098-burnhill-en.pdf. Accessed 21 Feb 2016
4. OASIS (2004) MPEG-21 part 2: Digital Item Declaration Language (DIDL), http://xml.coverpages.org/mpeg21-didl.html. Accessed 21 Feb 2016
5. CCSDS (2012). Reference model for an Open Archival Information System (OAIS). CCSDS 650.0-M-2. Magenta Book. Issue 2. June 2012. http://public.ccsds.org/publications/archive/650x0m2.pdf. Accessed 21 Feb 2016
6. Library of Congress, METS Editorial Board (2015) Metadata Encoding and Transmission Standard (METS): official web site. http://www.loc.gov/standards/mets. Accessed 05 Jan 2016

7. Dublin Core Metadata Initiative (2015) Web site. http://dublincore.org/. Accessed 10 Jan 2016
8. National Library of Medicine (2013) JATS: journal article tag suite. http://jats.nlm.nih.gov/. Accessed 21 Feb 2016
9. Open Preservation Foundation (2015) JHOVE [Software]. http://openpreservation.org/technology/products/jhove/. Accessed 28 Oct 2015
10. Wikimedia Foundation (2016) File (command). http://en.wikipedia.org/wiki/File_(command). Accessed 21 Feb 2016
11. National Library of Medicine (2016) Book interchange tag set: JATS extension. http://jats.nlm.nih.gov/extensions/bits/. Accessed 21 Feb

Chapter 8
Digital Preservation Metadata Practice for Disk Image Access

Alexandra Chassanoff, Kam Woods and Christopher A. Lee

8.1 Introduction

Many libraries, archives, and museums (hereafter referred to as LAMs) are now regularly acquiring, processing, and analyzing born-digital materials. In a 2010 OCLC Research Survey of Special Collections and Archives, nearly 80 % of respondents reported collecting born-digital materials in one or more formats [1]. Materials exist on a variety of source media, including flash drives, hard drives, floppy disks, and optical media.

Extracting disk images (i.e., sector-by-sector copies of digital media) is an increasingly common practice in LAMs. It can be essential to ensuring provenance, original order, and chain of custody [2]. Disk images allow users to explore and interact with the original data without the risk of permanent alteration. These replicas help institutions to safeguard against modifications to the underlying data that can occur when a file system contained on a storage medium is mounted, or a bootable medium is powered up.

Retention of disk images can substantially reduce preservation risks. Digital storage media become progressively difficult (or impossible) to read over time, due to "bit rot," obsolescence of media, and reduced availability of devices to read them. Simply copying the allocated files off a disk and discarding the storage carrier, however, can be problematic. The ability to access and render the content of files can depend upon the presence of other data that reside on the disk. These

A. Chassanoff (✉) · K. Woods · C.A. Lee
University of North Carolina, c/o Christopher A. Lee, 100 Manning Hall,
Chapel Hill, NC 27519, USA
e-mail: achass@email.unc.edu

K. Woods
e-mail: kamwoods@email.unc.edu

C.A. Lee
e-mail: callee@ils.unc.edu

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_8

dependencies are often not obvious upon first inspection and may only be discovered after the original medium is no longer readable or available. Disk images also enable a wide range of potential access approaches, including dynamic browsing of disk images [3] and emulation of earlier computing platforms.

Disk images often contain residual data, which may consist of previously hidden or deleted files [4]. Residual data can be valuable for scholars interested in learning about the context of creation. Traces of activities undertaken in the original environment—for example, identifying removable media connected to a host machine or finding contents of browser caches—can provide additional sources of information for researchers and facilitate the preservation of materials [5].

Digital forensic tools can be used to create disk images in a wide range of formats. These include raw files (such as those produced by the Unix tool dd), forensic formats such as the Encase image file format (E01) [6] and Advanced Forensic Format (AFF) [7], and platform-specific formats such as Apple Disk Image files (.dmg) [8].

8.2 Disk Images as Digital Objects

Disk images have an internal structure that is different from other kinds of digital objects typically found in repositories. Within a single disk image file, one finds one or more file systems comprised of many files, file system metadata such as timestamps and file access permissions, and various forms of trace data (discussed above). Software can access the contents of a disk image in the following ways:

- through mounting and navigating the file system(s), encountering the data as files and directories;
- by bypassing the file system and retrieving data based on an offset (number of bytes) into the disk image as a bitstream; or
- through using information recorded by the file system to identify contiguous or non-contiguous runs of bytes on the disk corresponding to individual files, and reconstructing the contents of those files.

There are various types of metadata that can be associated with disk images packaged in forensic formats. In this chapter, we emphasize three different types: (1) forensic tool metadata generated during acquisition, processing, and analysis of disk images; (2) source metadata, or characteristics relating to the original physical media; and (3) metadata about the content of the disk image (e.g., patterns of interest, file system attributes). Forensic tool output can capture all three types of metadata, documenting specific events (such as disk image creation) and persistent state information related to the original medium and its content. For example, the creation of a disk image using Guymager [9], an open source disk imaging tool, will result in the following output:

- name and version of the software used to create the disk image;
- cryptographic hash (MD5 and/or SHA256) of the raw disk image bitstream;

- time of acquisition; and
- technical characteristics of the original medium (e.g., whether it was removable or fixed, and exact size in bytes).

This metadata is automatically stored in a plaintext log file produced by Guymager. If the user generates a forensically packaged disk image (in the E01 or AFF format), the metadata will also be embedded in the image file. In the case of the forensic formats, optional user-supplied metadata may also be stored in the file, including a case number, an evidence number, a unique description, the name of the case examiner, and any additional notes.[1]

Information about file system contents on source media can also be extracted using forensic tools. Open source tools such as fiwalk [10] can provide significant insight about the file system(s) and files on the disk by analyzing and creating reports from file system metadata. Fiwalk captures information including

- user-level permissions[2]; access permissions;
- size of files and length of contents;
- timestamps that document when file contents were last modified, accessed, or changed;
- number of partitions on the disk;
- file system(s) on the partitions; and
- whether a file was accessible to users or marked as deleted.

When run against a disk or disk image, fiwalk writes the above metadata to a file in the form of Digital Forensics XML. DFXML is a container for file-level system metadata which was originally developed by Simson Garfinkel to enable exchange of forensic metadata [11].

8.3 Case Study: Using BitCurator to Meet Preservation and Access Goals for LAMS

Two central challenges for LAMs are preserving and providing long-term access to born-digital materials. During the past 5 years, a variety of workflows and strategies have emerged [12–14].

The BitCurator open source digital forensics software environment was developed during the course of a 3-year, grant-funded applied research project, which ran

[1]Terminology reflected in forensic tools is often based on the assumption that they are being used for legal investigations. LAMs can adopt their own conventions for use of these metadata elements, e.g., using acquisition number in the place of "evidence number" and processing archivist in the place of "case examiner".

[2]Permissions are printed as Unix-style permission values; file systems with more complex permissions models (such as NTFS) require additional information from the operating system in order to decode those values.

from 2011 to 2014 and was funded through the Andrew Mellon Foundation[3] [15]. The BitCurator environment can be used to acquire, process, and analyze disk images. Software development has focused on supporting curatorial tasks in LAM workflows related to long-term access and preservation of born-digital materials. One goal of the project was to ensure that metadata captured from and about disk images could be easily ingested and stored in a variety of repository environments. The BitCurator environment is currently maintained and supported by the BitCurator Consortium, which was officially launched in August 2014 [16]. A follow-on project, called BitCurator Access (2014–2016), also funded by the Andrew W. Mellon Foundation, is developing tools and methods to provide and mediate public access to data stored in and extracted from disk images [17].

Disk images can present intellectual challenges for repositories deciding how best to manage access to born-digital materials. Digital objects exist simultaneously at several different levels of representation, which have implications for future access [18]. A disk image can be understood as a representation of source media, documenting the file systems and files found on the original removable media. At the same time, a disk image can also be understood as a representation of processes related to disk image creation, recording information about the capture of the original bitstream. Repositories should consider which levels of representation they might want to document and potentially expose to users. For example, a researcher might want to navigate the file system of a disk, view a specific JPEG file in an application, and examine the EXIF metadata embedded in that file. Repositories must also decide what levels of representation they want to target for preservation, and consider the implications those decisions may have on providing access to materials. They may choose to preserve an entire disk image, extract individual files from the image, or a combination of both.

8.3.1 Using PREMIS to Capture Forensic Tool Output as Preservation Scenarios

The BitCurator environment supports four main functions related to archival workflow: forensic disk imaging, forensic processing and identification of potentially identifiable information (PII), data triage, and metadata export.[4] These actions usually take place prior to archival ingest (as the environment itself is not a preservation or collection management system). Our PREMIS [19] implementation emphasizes flexibility in export, providing simple XML structures that may be easily incorporated into, or transformed for, localized implementations. At the time of the writing (early 2016), BitCurator outputs PREMIS version 2.0 [20], and so are

[3]To learn more about the BitCurator project, see: http://www.bitcurator.net/bitcurator/.

[4]For an in-depth explanation of each area, please visit: http://wiki.bitcurator.net/index.php?title=BitCurator_and_Archival_Workflows.

Semantic component	Sample value(s)	Derived from
objectIdentifierType	UUID	Running libUUID
messageDigestAlgorithm	SHA256	Guymager output info.txt
messageDigest	F22c693ef1a40e390d0e71f	Guymager output info.txt
size	2000398934016 (2.0 TB)	Guymager output info.txt
formatName	Advanced Forensics Format v3	Guymager output info.txt

Fig. 8.1 A disk image represented as a PREMIS Object (UUIDs were chosen as identifiers as they are ubiquitous, independent, and easily replaceable from underlying software libraries part of the BitCurator environment)

the XML samples provided here. PREMIS 3.0 changes are highlighted whenever relevant.

In our implementation design, we treat the disk image as a PREMIS Object and use the PREMIS Event model to capture metadata about disk image capture, analysis, and report generation. Choosing to make the disk image the preservation target enabled us to design for capture and storage of metadata related to both the original media and the creation process. This design does not advocate for specific implementations of the PREMIS metadata schema or specific kinds of preservation strategies. Rather, the goal of our PREMIS encoding has been to document useful technical characteristics of the disk image regardless of disk image format.

In the BitCurator implementation, the disk image file is encoded as a PREMIS Object (see Fig. 8.1). PREMIS Events correspond to actions related to the newly created disk image (e.g., image capture via Guymager [9], analysis of file-level system metadata via fiwalk [10]). Our approach is similar in design to that of Archivematica, which employs PREMIS primarily as a means for logging events.[5]

To date, we have implemented four PREMIS Event types to record the capture and analysis of disk images (see Fig. 8.2). The initial four events were chosen to reflect what we believed to be common targets for preservation institutions. Each scenario corresponds to output generated from tools available in the BitCurator environment; however, they can be parameterized to accommodate other tool output as needed. The BitCurator environment tools do not generate Agent or Rights metadata, as these will vary significantly across repositories, and it is presumed that those metadata will be generated by other tools or processes.

The PREMIS output produced by the BitCurator environment is compliant with the PREMIS 3.0 standard. As the PREMIS standard remains in development, the following examples are shown only as discrete XML subtrees rather than as part of the XSD-compliant document produced by BitCurator.

An example of a PREMIS Object corresponding to a disk image is shown in Fig. 8.3. It includes the file system path and file name corresponding to the captured

[5]Archivematica is an open source digital preservation system developed and maintained by Artefactual Systems. A sample of Archivematica PREMIS Event encodings can be found at: https://www.archivematica.org/wiki/PREMIS_metadata:_events. See also Chap. 16 in this book.

PREMIS entity	Event Type	Action
Event	Image capture	Creation of a disk image [using Guymager]
Event	File system analysis	Extraction of information from the file system [using *fiwalk*]
Event	Stream-based feature analysis	Identification of features of interest [using bulk_extractor]
Event	Report generation	Collating intermediate forensic metadata into actionable, human reports

Fig. 8.2 PREMIS Events with associated scenarios and actions

```
<object>
  <objectIdentifier>
    <objectIdentifierType>UUID</objectIdentifierType>
    <objectIdentifierValue>0943f0c6-79ba-11e4-8143-
08002743597f</objectIdentifierValue>
  </objectIdentifier>
...
  <originalName>/home/bcadmin/Desktop/jo-work-usb-2009-12-
11.E01</originalName>
</object>
```

Fig. 8.3 Partial encoding of a disk image as a PREMIS Object

image in the BitCurator environment, along with a UUID generated by the reporting and PREMIS generation tool.

Additional object characteristics and fixity information are not encoded in this object, but may include size, cryptographic hashes, format name and type, format registry associations, and other relevant technical metadata. Depending on the degree of replication desired, these may be added to the PREMIS Object upon transfer to the preservation environment or stored separately in a database or static file. In our implementation, some of these values are preserved in the "Capture" event.

An example of a disk image capture Event in PREMIS is shown in Fig. 8.4. As with the associated object, a UUID is generated for the capture event. The *eventDetail* tag points to the location of the newly created disk image, while the *eventOutcome* and *eventOutcomeDetail* tags are employed to describe the type of disk image produced (in this case, an Encase image file format image) and the

```
  <event>
      <eventIdentifier>
        <eventIdentifierType>UUID</eventIdentifierType>
        <eventIdentifierValue>40fb1638-8703-11e4-bc63-
0800274cbb73</eventIdentifierValue>
      </eventIdentifier>
      <eventType>Capture</eventType>
      <eventDetail>/home/bcadmin/Desktop/jo-work-usb-
2009-12-11.E01
      </eventDetail>
      <eventDateTime>2011-01-19T12:09:31</eventDateTime>
      <eventOutcomeInformation>
        <eventOutcome>Completed</eventOutcome>
        <eventOutcomeDetail>Version: 20100226, Image
size: 118233120</eventOutcomeDetail>
      </eventOutcomeInformation>
  </event>
```

Fig. 8.4 Disk image capture as a PREMIS Event (This example is in PREMIS 2. In PREMIS 3, eventDetail is nested inside a repeatable *eventDetailInformation* container)

version of the software library used to produce that image. Whenever possible, event timestamps are recorded in an ISO 8601-compliant format to avoid ambiguity. Note that one could also establish a link between the PREMIS Object and the image capture event by employing the *linkingObjectIdentifier* tag.

An example of a PREMIS Event corresponding to file system analysis using the fiwalk tool is shown in Fig. 8.5. The fiwalk tool [10] produces as output an XML document describing the contents of the file system(s) on disk. Note that (as with the event associated with running bulk_extractor shown later in Fig. 8.6), an *eventDetail* is included that describes the command line sequence necessary to replicate the operation. This detail may be useful to both researchers and repository staff, as the tool itself has a wide range of optional settings.

Many events likely to occur within the digital curation lifecycle of an object have easily described outcomes (e.g., "do these checksums match?"). Others may produce a large amount of output that is not necessarily appropriate to record as metadata. For example, one of the tools incorporated into the BitCurator environment is Simson Garfinkel's bulk_extractor [21], which is a tool that scans the contents of disk images or directories of files to find a variety of potentially sensitive patterns of data (e.g., credit card numbers, social security numbers, EXIF metadata elements, email addresses). Figure 8.6 provides an example of a PREMIS Event corresponding to block-level analysis of a disk image using bulk_extractor. Comparison of two bulk_extractor scans executed at different times may be conducted to determine if the contents of a particular disk image object have changed.

```
    <event>
      <eventIdentifier>
        <eventIdentifierType>UUID</eventIdentifierType>
        <eventIdentifierValue>4839f20c-8703-11e4-a280-
0800274cbb73</eventIdentifierValue>
      </eventIdentifier>
      <eventType>File System Analysis</eventType>
      <eventDetail>fiwalk               -f               -X
/home/bcadmin/Desktop/bcrep7/fiwalk-output.xml
/home/bcadmin/Desktop/jo-work-usb-2009-12-
11.E01</eventDetail>
  <eventDateTime>2014-12-18T22:14:37Z</eventDateTime>
      <eventOutcomeInformation>
        <eventOutcome>Completed</eventOutcome>
        <eventOutcomeDetail>Produced      DFXML      file:
/home/bcadmin/Desktop/bcreports/fiwalk-
output.xml</eventOutcomeDetail>
      </eventOutcomeInformation>
  </event>
```

Fig. 8.5 File system analysis as a PREMIS Event (This example is in PREMIS 2. In PREMIS 3, eventDetail is nested inside a repeatable *eventDetailInformation* container)

```
    <event>
      <eventIdentifier>
        <eventIdentifierType>UUID</eventIdentifierType>
        <eventIdentifierValue>483b4774-8703-11e4-a280-
0800274cbb73</eventIdentifierValue>
      </eventIdentifier>
      <eventType>Feature Stream Analysis</eventType>
      <eventDetail>bulk_extractor                    -o
/home/bcadmin/Desktop/beout    /home/bcadmin/Desktop/jo-
work-usb-2009-12-11.E01</eventDetail>
      <eventDateTime>2014-12-
18T00:49:02Z</eventDateTime>
      <eventOutcomeInformation>
        <eventOutcome>Completed</eventOutcome>
        <eventOutcomeDetail>Produced bulk extractor fea-
ture files and XML report /home/bcadmin/Desktop/beout/
        </eventOutcomeDetail>
      </eventOutcomeInformation>
    </event>
```

Fig. 8.6 Feature analysis as a PREMIS Event

The changes are not recorded in the PREMIS record, but knowledge of the fact that the tool was executed at a previous point may be useful in determining when these changes may have occurred, particularly if the tool output is retained elsewhere in storage.

The PREMIS Objects and Events produced by the BitCurator environment are intended to simplify pre-ingest activities conducted by repositories, allowing them to make informed decisions about where and how to store disk image capture and processing metadata while providing flexibility with respect to the tools used to capture a disk image. As such, the encoding uses only the core PREMIS XML namespace and descriptive vocabulary.

8.3.2 Recommendations for Future Work

The BitCurator environment output generates PREMIS elements that we consider to be essential: an object with a corresponding UUID; an event corresponding to a demarcation of the contents of the file system(s); an event corresponding to block-level analysis of the disk (particularly to itemize sensitive data and items not extant within the file system); and an event describing an analysis of the disk image blocks for features of interest. PREMIS metadata captured for born-digital objects such as disk images could be stored within a repository (1) as XML (or other static encodings), (2) in a database (and exported as static encoding in special circumstances such as transfer between sites), or (3) both static encoding and within a database.[6]

Many additional events may be of interest to archives intending to preserve disk images, including virus scans—both of the disk image file itself and of the contents of file system(s) contained within the disk image—and regular checksum verifications. These types of events are not currently recorded in the BitCurator environment, as we expect those tasks to be performed within other systems. Options to record these events may be made available at the request of the user community in future releases.

Accurate and unambiguous timestamps are important records of provenance, and there are some actions for which the timestamp of an event and the timestamp corresponding to the creation of (or addition to) the PREMIS record may differ significantly. For example, the disk image capture timestamp recorded by the BitCurator environment corresponds to the time when the disk image file was created and is extracted from the disk image file itself (when working with forensic disk image formats) or the capture utility log file (when working with raw disk

[6]For example, in terms of the Reference Model for an Open Archival Information System (OAIS) [22], the static encoding could reside in Archival Storage, while the database encoding could be used for Data Management and Access. Such implementation options are further discussed in Chap. 18 on Conformance with PREMIS.

images). The creation timestamp of the PREMIS metadata is not recorded. Future implementations may record two distinct timestamps to remove any ambiguity.

Quantifying successes and failures for many tools can require judgment calls by qualified digital curation professionals. Verifying a checksum for a file is a simple case; the checksums either match or are different. In the events described in the previous sections, however, the conditions for success are fuzzier. For example, fiwalk will often "successfully" complete whether or not it is able to extract a meaningful record of the contents of file system(s) on a disk image. Likewise, bulk_extractor will simply report items of interest it has discovered. Knowing whether this output is useful (and whether it has changed between separate executions of a given tool) depends on comparison of the output between the two runs, information not currently recorded in the PREMIS document. In the BitCurator implementation, events are often recorded as having completed, rather than as having succeeded, to avoid ambiguity. Future iterations of the implementation may include more nuanced descriptions of event outcomes.

References

1. Dooley JM, Luce K (2010) Taking our pulse: the OCLC research survey of special collections and archives. OCLC Research, Dublin, Ohio
2. Lee CA (2012) Archival application of digital forensics methods for authenticity, description and access provision. Comma 2(4):133–139
3. Misra S, Lee CA, Woods K (2014) A web service for file-level access to disk images. Code4 Lib J 25. http://journal.code4lib.org/articles/9773. Accessed 23 Oct 2016
4. Redwine G, Barnard M, Donovan K, Farr E, Forstrom M, Hansen W, Leighton John J, Kuhl N, Shaw S, Thomas S (2013) Born digital: guidance for donors, dealers, and archival repositories. Council on Library and Information Resources, Washington, DC
5. Woods K, Lee CA, Garfinkel S (2011) Extending digital repository architectures to support disk image preservation and access. In: Proceedings of the 11th annual international ACM/IEEE joint conference on digital libraries (pp 57–66). New York, NY. Association for Computing Machinery. doi:10.1145/1998076.1998088. Accessed 15 Jan 2016
6. Encase image file format, http://www.forensicswiki.org/wiki/Encase_image_file_format. Accessed 12 Dec 2015
7. Advanced forensics format, http://www.forensicswiki.org/wiki/AFF. Accessed 12 Dec 2015
8. Wikipedia (2016) Apple Disk Image. https://en.wikipedia.org/w/index.php?title=Apple_Disk_Image&oldid=693887000. Accessed 15 Jan 2016
9. Guymager, http://guymager.sourceforge.net/. Accessed 12 Dec 2015
10. fiwalk, http://www.forensicswiki.org/wiki/Fiwalk. Accessed 12 Dec 2015
11. Garfinkel S (2012) Digital forensics XML and the DFXML toolset. Digital Investig 8:161–174
12. Lee CA, Woods K, Kirschenbaum M, Chassanoff A (2013) From bitstreams to heritage: putting digital forensics into practice in collecting institutions. http://www.bitcurator.net/docs/bitstreams-to-heritage.pdf. Accessed 15 Jan 2016
13. AIMS Work Group (2012) AIMS born-digital collections: an inter-institutional model for stewardship. http://www2.lib.virginia.edu/aims/whitepaper/AIMS_final.pdf. Accessed 15 Jan 2016

14. Gengenbach MJ (2012) 'The way we do it here': mapping digital forensics workflows in collecting institutions. A master's paper for the M.S. in L.S degree. http://digitalcurationexchange.org/system/files/gengenbach-forensic-workflows-2012.pdf. Accessed 15 Jan 2016
15. http://www.bitcurator.net/bitcurator/. Accessed 15 Jan 2016
16. http://www.bitcurator.net/bitcurator-consortium/. Accessed 15 Jan 2016
17. http://www.bitcurator.net/bitcurator-access/. Accessed 15 Jan 2016
18. Lee CA (2012) Digital curation as communication mediation. In: Mehler A, Romary L, Gibbon D (eds) Handbook of technical communication. Mouton De Gruyter, Berlin, pp 507–530
19. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 15 Jan 2016
20. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 2.0. http://www.loc.gov/standards/premis/v2/premis-dd-2-0.pdf. Accessed 15 Jan 2016
21. Forensics Wiki (2015) Bulk extractor. http://www.forensicswiki.org/wiki/Bulk_extractor. Accessed 15 Jan 2016
22. Consultative Committee for Space Data System (CCSDS) (2012) Reference model for an Open Archival Information System (OAIS). http://public.ccsds.org/publications/archive/650x0m2.pdf. Accessed 15 Jan 2016

Chapter 9
Digital Preservation Metadata Practice for Archives

Karin Bredenberg

9.1 Introduction

The International Council of Archives (ICA) supplies this definition of archives: "Archives are the documentary by-product of human activity retained for their long-term value. They are contemporary records created by individuals and organizations as they go about their business and therefore provide a direct window on past events. They can come in a wide range of formats including written, photographic, moving image, sound, digital and analogue. Archives are held by public and private institutions and individuals around the world" [1].

But for creating the archival information and building the archives, which will consist of both analog and digital material, we need to follow some principles, the most important being the principles of provenance and context. We need to know exactly who created the content and in what context. This is key in ensuring that the information kept in the archives for long-term preservation can be used as evidence for the long-term. But this also impacts how the information should be stored and what extra information needs to be stored for preserving the content information and its trustworthiness through format migrations, normalizations, and other activities that may affect the information.

Archives often use an Electronic Records Management System (ERMS), which is "a computer program or set of programs designed to track and store records" [2]. An institutional archive is similar to an ERMS in that it also stores information and metadata which describe context and provenance. However, as the context is different in each case, the metadata will refer to different things in an ERMS or in an institution archive and will be explained in detail in the part of this chapter called "Metadata for records management and archiving activities."

K. Bredenberg (✉)
Riksarkivet, Avdelningen för offentlig informationshantering/Public Information Management, P.O. Box 12541, 102 29 Stockholm, Sweden
e-mail: Karin.Bredenberg@riksarkivet.se

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_9

- In the ERMS, metadata is expressed with records management-specific terms.
- In the archives, metadata is expressed with the finding aid and its terms. The finding aid is "a tool that facilitates discovery of information within a collection of records." It consists of "a description of records that gives the archives physical and intellectual control over the materials and that assists users to gain access to and understand the materials" [3].

Two words are important to articulate for the clarity of this chapter: "preservation" and "archiving." "Archiving" is the act of selecting something for storage in the archives, while the former is the act of preserving the archived object so it can be used today or in the future regardless of the media on which the information is stored, whether analog or digital. Please note that an OAIS [4] Archive refers to an organization that intends to preserve information for access and use by a designated community. This function is related to preservation, and can concern any kind of actor (archives but also libraries, museums, private stakeholders…). Whenever this term is used with this meaning of preservation, we use "Archive" with a capital A.

9.2 Preservation for Archives: Specific Use Cases, Metadata, and Systems Requirements

Users of digital preservation systems have a lot of common issues in implementing the OAIS model [4]. Chap. 2 provides a methodology that can be applied in any context to make informed choices that match your institutional needs.

As far as archives are concerned, the requirements will differ, depending on the mandate of the archival institution—national archives versus local archives, public archives versus independent archives. Support for decisions made can be found by archivists online. Two projects are especially helpful in that light: "PLANETS"—Preservation and Long-term Access through Networked Services [5] which is focused on the whole landscape of preservation and access, and E-ARK—European Archival Records and Knowledge Preservation [6] which is only focused on preservation and access needs in the archival community.

Regardless of the choice of how and where we store the digital objects, recording preservation metadata is important and some decisions regarding capturing and storing them are needed.

9.2.1 Archival Metadata and Archival Systems

In archives, metadata comes in different flavors that can be matched to different use cases and their workflows:

- metadata for managing the information in the "living" system, often a records management system
- metadata for archiving the information
- metadata for the preservation of the archived objects.

The different types and uses of metadata, which may be called flavors, are an indication of the fact that they need to support different functions: there are different software programs, different activities carried out by the users and different processes for handling the digital objects in each one of them. In the living system you will likely find an ERMS created by a vendor who adds to the system their version of ERMS metadata based on, for example, ISO 15489-1:2001 standard for Records management [7]. The archival world also has specifically created standards like MoReq2010 [8] when dealing with information coming from an ERMS needed to be transferred to the archives.

9.2.1.1 Metadata for Records Management and Archiving Activities

The Swedish National Archives can serve as an example for records management and archival metadata: In our office we use an Electronic Records Management System (ERMS). For a given record, we may find the following key metadata in the system: identity and name of the sender, the receiver, classification numbers and names, and proofs of authenticity such as digital signatures. In the system the records and the current handling in a "living" environment are also defined. The same metadata elements for the sender, receiver, and the classification of the record are also used by the archives, but in a different context. The differences are highlighted in Table 9.1.

Table 9.1 Metadata elements in an ERMS and archival system

Term	In the ERMS	In the archives
Sender	The person depositing the record in the ERMS Example: Mr. Brown	The authority sending the record or the whole ERMS for archiving purposes Example: The Legal Department
Receiver	The receiver of the record on behalf of the ERMS Example: Case officer 1	The archival institution receiving the record or the whole ERMS Example: The state archives
Classification	Classification of the record in the ERMS Example: Research management	Classification of the record or ERMS in the archival environment Example: The ERMS contains personal information and therefore demands redaction before access

9.2.1.2 Metadata for Preservation Activities

The ERMS does not usually contain information regarding preservation activities, such as a completed migration of digital objects to alternative formats. Even though representations in alternative formats may be added to a record in the system, the information linking the related representations would not be recorded. For example, a document belonging to a record in the PDF/A format may be the result of a migration from an original document in Microsoft Word format; in such cases the ERMS does not usually store when and how this migration was performed. The information about this migration is useful evidence for the degree of authenticity of the record in the long-term and needs to be known when further preservation actions are preformed. Neither does an ERMS usually record technical information specific to a particular format. In the "living" system we might find a relationship from each record to a described item in the finding aid[1] (which serves as the descriptive metadata to support discovery) when the record is created or when it is archived in the archival system, depending upon the implementation. Alternatively, when consolidated archiving actions take place, all the objects in the ERMS may be saved in an exchange format and transferred to the archives. Such a bulk transfer may result in the items that exist as separate records in the ERMS being represented through one single entry in a finding aid, which depends on descriptive policies in the archive. In the archiving environment more metadata is added, for instance:

- information about the record creation or of the whole ERMS;
- other descriptive archival information like the physical location of the record in the archives, its position in the finding aid, restrictions on use.

Such information helps the user to find the records in an archival search and to provide context and evidence of their authenticity. These are two key missions of institutional archives.

However, from a preservation standpoint, finding aids, which mainly provide descriptive information about the items in archival collections, still lack metadata to support preservation activities—and when they do, it is mostly about the preservation and conservation of physical documents, not for electronically created records. This situation is changing, and systems are adding preservation metadata for digital objects, often developed with PREMIS [9] as the guide, so that preservation information is equally stored in the system for analog and digital objects. The often hybrid nature of archive documents is an important characteristic to highlight: the archive has both analog and digital objects described in the same finding aid.

[1]A finding aid is a tool that describes archival records in a collection and thus facilitates the discovery and understanding of those records.

9.2.2 Archival Systems

After delivery of the digital objects to the archives they are handled in an archiving system created by a commercial vendor or in a system created in-house. The electronic archival system is most likely working in cooperation with a system for finding aids and storage for the digital objects. Preservation can be handled in a separate software system, or be a part of the electronic archival system itself.

9.2.2.1 Setting up the Archival Storage

When setting up an electronic archive that includes preservation functionality, or when adding preservation functionality to an existing digital object in long-term storage, decisions need to be made and questions need to be answered. The questions include these ones:

- What do we store?
- Why do we store it?
- For whom do we store it?

What Do We Store?

Archival institutions receive and are mandated to preserve virtually any kind of digital object: text, image, sound, video, software, database, and so on. Sometimes it is not possible for the archives to have the producer provide the format that the archives has decided it will use as a preservation format; it may even be policy that the archives needs to take all possible formats. Therefore, system support for handling nonstandard kinds of formats might be needed.

Why Do We Store It?

Many decisions to collect digital objects are defined in the archives' strategy and based on the mandate the archives have to follow. Whether created in an analog or digital environment the records can be pieces of evidence and may be used as opposable documents in a trial. The digital objects might be transferred to the archives while they still have a legal value depending on the strategy and mandate of the archives, just as is the case with the analog objects. In some cases, the archival institutions may be mandated by their government to preserve objects that are considered cultural heritage, including digital objects. Sometimes the mandate implies that certain digital objects need to be handled differently from the ordinary digital collection.

For Whom Do We Store It?

It is sometimes difficult to know for whom we are preserving the digital objects and the accompanying metadata. The who may be stated in the archives' strategy together with the why. But they may also have to fulfill other user needs that have not been made explicit in the mandate and are not easy to define.

Here are some examples of archive users' possible needs and ways to fulfill them:

- Will it be for researchers visiting the archives?
- Will it be accessible online?
- Will the documents be transformed into another format and all personal information removed from them so they can be used by other parties?
- Will the information be sent elsewhere (e.g., to an archival portal) and, in that case, is the digital object duplicated in the portal or does the portal merely provide a link to the digital object on the archives' website?
- Will the digital object be used as evidence? In such a case, proofs of the authenticity of the object such as digital signatures or other provenance information will be required.
- If the digital object is in a complex format, such as a database, do we need to be able to recreate all of its functionality? In this case, we may need to have information about how the database was built.

These questions imply that different user needs and requirements for the digital objects should be defined considering the most appropriate capabilities available.

9.3 Information Packages in Archives: Specific Considerations

The OAIS model [4] defines three different types of information packages: Submission Information Package (SIP), the stored Archival Information Package (AIP), and the outgoing Dissemination Information Package (DIP). The package that the archives can define according to their own rules in the e-archiving solution and for which they make a long-term preservation commitment is the AIP. It is possible to store the incoming SIP as-is as an AIP, and then derive a normalized AIP where changes in a first step will be made. As a matter of fact this is usually how it is done in archives, in order to keep an untouched digital object in its original form. Thus there are two AIP representations upon ingestion to the e-archive: the original incoming version[2] and the normalized AIP that can be uniformly processed

[2]Please note that this chapter uses the term "version" to describe this approach, but the terms "version" and "generation" are used with different meanings depending on the e-archiving solution.

together with all other repository content. Creation of a new version of the AIP is made by decisions following the business rules the archive have set up. In addition to AIPs, the Archival Information Collection (AIC) package may be used for grouping AIPs containing the same kind of information. It will be necessary to have a way of relating different versions of an AIP, and AIPs with their AIC, if any. As with other types of objects, this can be done using the relationship section in PREMIS or by describing structural relations like the structural map defined in the standard "Metadata Encoding and Transmission Standard" (METS) [10] described in Chap. 14.

Digital signatures are one way of proving the authenticity of a digital object and are specifically important in institutions where authenticity and evidence belong to the core mandate, as is the case for archives. But for an archive the problem of maintaining the authenticity with the help of a digital signature may not be easily done with just saving the signature. Saving the signature for an indefinite period of time results in other long-term preservation challenges like how to authenticate the signature in the future. Sometimes the archive decides that having just a marker saying that it has been digitally signed is enough, while in other cases the archive actually wants to save the digital signature. Using a simple marker is currently the most common case due to the lack of a tested and available metadata standard for describing the digital signature. The marker is most often implemented as a Boolean stating true if the digital object carries a digital signature. How to best verify the authenticity of digital signatures and maintain their authenticity over the long-term is still an open question. This uncertainty is a common argument for not saving the digital signature. At the time of writing, in the EU several projects are considering defining or using a metadata format for digital signatures like the e-SENS project [11] and the earlier mentioned E-ARK [6] project, but for now there is no guarantee that those projects will result in a digital signature standard.

Using PREMIS for metadata about the digital signature is a solution, either using PREMIS semantic units or using the *digitalSignatureExtension* container. XML Dsig [12] from W3C or the result from the e-SENS project can be used as extensions. Please note that the PREMIS semantic units for digital signatures were designed to be compatible and/or replaced with the XML Dsig standard.

9.3.1 Describing Information Packages in Archives

In most cases, an archive uses one standard for describing the information package and another for expressing its digital preservation information. METS is a common standard for describing the package. Nonstandard metadata formats are often used when the e-archiving solution is an in-house built solution.

A digital object that has been placed in electronic archival storage will most certainly be described in an electronic finding aid. For creating a finding aid either a specific archival description system may be used or it may be created by hand with the help of the standards "Encoded Archival Description" (EAD) [13] and

"Encoded Archival Context-Corporate Bodies, Persons and Families" (EAC-CPF) [14]. It is important to differentiate between the finding aid and the information package. The finding aid allows the discovery of content and context.

- The content, which is carried by either analog or digital objects (or a mix of both), is described with descriptive information. Such information is not specific to digital documents and its core purpose is to allow the discovery of the object by the user.
- The context is the description of the corporate body, person or family that has created the information, which is key for archival records.

However, the finding aid itself does not contain in-depth information (such as preservation information) about the digital objects themselves. Such information is described and linked together in the information package.

9.3.2 *Implementing METS and PREMIS for Archives: An Example*

The AIP described below consists of a collection of images packaged together in a TAR file, and is described with METS, using a METS profile[3] used in archives in Sweden [16]. The package is referenced in a finding aid and can be accessed and rendered using a viewer showing the digital images for the user.

Figure 9.1 illustrates the METS AIP and Fig. 9.2 illustrates the PREMIS metadata for the content file A0072716. Note the reference in Fig. 9.1 to the PREMIS metadata in A0072716_PREMIS.xml illustrated in Fig. 9.2.

Finding Aid

Figure 9.3 shows the finding aid for the package described in METS and PREMIS in Figs. 9.1 and 9.2.

9.3.3 *Additional Metadata Commonly Used by Archival Institutions*

As described in Chap. 14, metadata that is not covered by PREMIS can be inserted in extension elements. The most commonly used extension metadata formats that are needed among archival institutions include format-specific technical metadata standards for still images, audio, video and databases. An additional format-specific

[3]A METS Profile is a document describing a class of METS documents with guidance on a specific implementation to facilitate consistency. See [15].

METS

```
<mets:mets
   PROFILE="http://xml.ra.se/e-
arkiv/METS/CommonSpecificationSwedenPackageProfile.xml"
   LABEL="Imaging AIP RA"
   ID="ID3496cc4e-b5a7-11e3-a7ff-001b2126ecbe"
   OBJID="RAID:A0072716">
   <mets:metsHdr CREATEDATE="2014-03-27T14:24:05+01:00">
      <!-- Information regarding agents involved in the handling of the IP including institu-
tion, persons and computer programs -->
      <mets:agent TYPE="ORGANIZATION" ROLE="ARCHIVIST">
         <mets:name>Riksarkivet</mets:name>
         <mets:note>ORG:2021001074</mets:note>
      </mets:agent>
      <mets:agent TYPE="OTHER" OTHERTYPE="SOFTWARE"
ROLE="ARCHIVIST">
         <mets:name>Digitala kedjan</mets:name>
      </mets:agent>
      <mets:agent TYPE="ORGANIZATION" ROLE="CREATOR">
         <mets:name>Riksarkivet</mets:name>
         <mets:note>ORG:2021001074</mets:note>
      </mets:agent>
      <mets:agent TYPE="INDIVIDUAL" ROLE="CREATOR">
         <mets:name>ESSArch_SVAR</mets:name>
      </mets:agent>
      <mets:agent TYPE="OTHER" OTHERTYPE="SOFTWARE" ROLE="CREATOR">
         <mets:name>ESSArch</mets:name>
         <mets:note>VERSION=2.2</mets:note>
      </mets:agent>
      <mets:agent TYPE="ORGANIZATION" ROLE="PRESERVATION">
         <mets:name>Riksarkivet</mets:name>
         <mets:note>ORG:2021001074</mets:note>
         </mets:agent>
         <mets:agent TYPE="OTHER" OTHERTYPE="SOFTWARE"
   ROLE="PRESERVATION">
            <mets:name>ESSArch</mets:name>
            <mets:note>VERSION=2.2</mets:note>
         </mets:agent>
         <!-- Different identificators of the IP -->
         <mets:altRecordID TYPE="SUBMISSIONAGREEMENT">Digitala kedjan, RA 13-
   2010/464</mets:altRecordID>
         <mets:metsDocumentID>A0072716_Content_METS.xml</mets:metsDocumentID>
      </mets:metsHdr>
      <mets:amdSec ID="amdSec001">
         <!-- Link to PREMIS information -->
         <mets:digiprovMD ID="digiprovMD001">
            <mets:mdRef MIMETYPE="text/xml" CHECKSUMTYPE="MD5"
   CHECKSUM="37af06b6b4d4bac12054a94f935ff2b3" MDTYPE="PREMIS"
   xlink:href="file:///m/A0072716_PREMIS.xml" LOCTYPE="URL" CREATED="2014-03-
   27T13:21:13+01:00" xlink:type="simple" ID="ID34965034-b5a7-11e3-a7ff-
   001b2126ecbe" SIZE="83476"/>
```

Fig. 9.1 METS-simplified sample for an AIP

```
      </mets:digiprovMD>
    </mets:amdSec>
    <mets:fileSec>
      <!-- All files in the AIP -->
      <mets:fileGrp ID="fgrp001" USE="FILES">
        <mets:file MIMETYPE="image/tiff" ADMID="digiprovMD001"
CHECKSUMTYPE="MD5" CREATED="2014-03-27T07:46:19+01:00"
CHECKSUM="e5e77fcd9712559ded6c98932bd75e77" ext:FILEFORMATNAME="TIFF"
ext:FILEFORMATVERSION="6.0" ext:FORMATREGISTRY="PRONOM"
ext:FORMATREGISTRYKEY="fmt/353" USE="IMAGE" ID="ID34966312-b5a7-11e3-
a7ff-001b2126ecbe" SIZE="12497490">
          <mets:FLocat xlink:href="file:///c/MOD-885_189_1112.tif" LOCTYPE="URL"
xlink:type="simple"/>
        </mets:file>
        <!-- 2 more files follow -->
      </mets:fileGrp>
    </mets:fileSec>
    <!-- Structural information regarding the AIP -->
    <mets:structMap>
      <mets:div LABEL="Package">
        <mets:div ADMID="amdSec001" LABEL="Content Description">
          <mets:fptr FILEID="ID34965034-b5a7-11e3-a7ff-001b2126ecbe"/>
        </mets:div>
        <mets:div ADMID="amdSec001" LABEL="Datafiles">
          <mets:fptr FILEID="ID34966312-b5a7-11e3-a7ff-001b2126ecbe"/>
          <mets:fptr FILEID="ID34967096-b5a7-11e3-a7ff-001b2126ecbe"/>
          <mets:fptr FILEID="ID3496c4ec-b5a7-11e3-a7ff-001b2126ecbe"/>
        </mets:div>
      </mets:div>
    </mets:structMap>
</mets:mets>
```

Fig. 9.1 (continued)

technical metadata scheme that may be needed is Technical Metadata for Text (TextMD) [17].

Many digital objects currently found in an archival institution are the result of conversion from analog to digital material, requiring metadata about the technical characteristics of still images and the digitization process. But we also find early deliveries of digital objects in the archives, for instance from the 1970s, most commonly databases, often delivered as one or more text files. Today archives also receive deliveries of born-digital objects of all kinds. Regardless of how the digital object is created, born-digital or converted, it still needs to have preservation metadata so that archivists are as informed as possible when carrying out their preservation tasks.

```
<premis version="3.0">
 <!-- First a description of the AIP in the form of a .TAR file -->
 <premis:object xsi:type="premis:file">
  <premis:objectIdentifier>
   <premis:objectIdentifierType>SE/RA</premis:objectIdentifierType>
   <premis:objectIdentifierValue>A0072716</premis:objectIdentifierValue>
  </premis:objectIdentifier>
  <premis:preservationLevel>
   <premis:preservationLevelValue>full</premis:preservationLevelValue>
  </premis:preservationLevel>
  <premis:objectCharacteristics>
   <premis:compositionLevel>0</premis:compositionLevel>
   <premis:format>
    <premis:formatDesignation>
     <premis:formatName>tar</premis:formatName>
    </premis:formatDesignation>
   </premis:format>
  </premis:objectCharacteristics>
  <premis:storage>
   <premis:storageMedium>bevarandesystemet</premis:storageMedium>
  </premis:storage>
 </premis:object>
 <!-- All the image files in the AIP are described with the help of PREMIS and MIX for
giving more specific image information -->
 <!-- All the following objects are the images -->
 <!-- The information includes object identifier, composition level, fixity, size, format, sto-
rage, deletion to the PREMIS object describing the tar file -->
 <premis:object xsi:type="premis:file">
  <premis:objectIdentifier>
   <premis:objectIdentifierType>SE/RA</premis:objectIdentifierType>
   <premis:objectIdentifierValue>A0072716/c/MOD-
885_189_1112.tif</premis:objectIdentifierValue>
  </premis:objectIdentifier>
  <premis:significantProperties>
   <premis:significantPropertiesType>PageName</premis:significantPropertiesType>
   <premis:significantPropertiesValue>SE/RA/83002/2013/01/A0072716/MOD-
885_189_1112.tif</premis:significantPropertiesValue>
  </premis:significantProperties>
  <premis:objectCharacteristics>
   <premis:compositionLevel>0</premis:compositionLevel>
   <premis:fixity>
    <premis:messageDigestAlgorithm>MD5</premis:messageDigestAlgorithm>
    <pre-
mis:messageDigest>e5e77fcd9712559ded6c98932bd75e77</premis:messageDigest>
    <premis:messageDigestOriginator>ESSArch</premis:messageDigestOriginator>
   </premis:fixity>
   <premis:size>12497490</premis:size>
   <premis:format>
    <premis:formatDesignation>
     <premis:formatName>TIFF 6.0</premis:formatName>
    </premis:formatDesignation>
```

Fig. 9.2 PREMIS metadata for A0072716_PREMIS.xml mentioned in Fig. 9.1

```
</premis:format>
<premis:objectCharacteristicsExtension>
 <mix:mix>
  <mix:BasicDigitalObjectInformation>
   <mix:Compression>
    <mix:compressionScheme>Uncompressed</mix:compressionScheme>
   </mix:Compression>
  </mix:BasicDigitalObjectInformation>
  <mix:BasicImageInformation>
   <mix:BasicImageCharacteristics>
    <mix:imageWidth>5510</mix:imageWidth>
    <mix:imageHeight>2268</mix:imageHeight>
    <mix:PhotometricInterpretation>
     <mix:colorSpace>BlackIsZero</mix:colorSpace>
    </mix:PhotometricInterpretation>
   </mix:BasicImageCharacteristics>
  </mix:BasicImageInformation>
  <mix:ImageCaptureMetadata>
   <mix:SourceInformation>
    <mix:SourceID>
     <mix:sourceIDType>DocumentName</mix:sourceIDType>
     <mix:sourceIDValue>SE/SSA/0031/06/G 1 BD/1074</mix:sourceIDValue>
    </mix:SourceID>
    <mix:SourceID>
     <mix:sourceIDType>ImageDescription</mix:sourceIDType>
     <mix:sourceIDValue> Överståthållarämbetet för uppbördärenden. 06/G 1
BD/1074 1911- </mix:sourceIDValue>
    </mix:SourceID>
   </mix:SourceInformation>
   <mix:GeneralCaptureInformation>
    <mix:dateTimeCreated>2014-01-14T08:19:03+01:00</mix:dateTimeCreated>
    <mix:imageProducer>Stockholms stadsarkiv</mix:imageProducer>
   </mix:GeneralCaptureInformation>
   <mix:ScannerCapture>
    <mix:scannerManufacturer>nextScan,Inc.</mix:scannerManufacturer>
    <mix:ScannerModel>
     <mix:scannerModelName>None, SN# 738008</mix:scannerModelName>
    </mix:ScannerModel>
    <mix:ScanningSystemSoftware>
     <mix:scanningSoftwareName>nextStar v2.7</mix:scanningSoftwareName>
    </mix:ScanningSystemSoftware>
   </mix:ScannerCapture>
   <mix:orientation>normal*</mix:orientation>
  </mix:ImageCaptureMetadata>
  <mix:ImageAssessmentMetadata>
   <mix:SpatialMetrics>
    <mix:samplingFrequencyUnit>in.</mix:samplingFrequencyUnit>
    <mix:xSamplingFrequency>
     <mix:numerator>300</mix:numerator>
    </mix:xSamplingFrequency>
    <mix:ySamplingFrequency>
     <mix:numerator>300</mix:numerator>
    </mix:ySamplingFrequency>
```

Fig. 9.2 (continued)

```
            </mix:SpatialMetrics>
            <mix:ImageColorEncoding>
              <mix:BitsPerSample>
                <mix:bitsPerSampleValue>8</mix:bitsPerSampleValue>
                <mix:bitsPerSampleUnit>integer</mix:bitsPerSampleUnit>
              </mix:BitsPerSample>
              <mix:samplesPerPixel>1</mix:samplesPerPixel>
            </mix:ImageColorEncoding>
          </mix:ImageAssessmentMetadata>
        </mix:mix>
      </premis:objectCharacteristicsExtension>
    </premis:objectCharacteristics>
    <premis:storage>
      <premis:contentLocation>
        <premis:contentLocationType>AIP</premis:contentLocationType>
        <premis:contentLocationValue>A0072716</premis:contentLocationValue>
      </premis:contentLocation>
    </premis:storage>
    <premis:relationship>
      <premis:relationshipType>structural</premis:relationshipType>
      <premis:relationshipSubType>is part of</premis:relationshipSubType>
      <premis:relatedObjectIdentifier>
        <premis:relatedObjectIdentifierType>SE/RA</premis:relatedObjectIdentifierType>
        <pre-
mis:relatedObjectIdentifierValue>A0072716</premis:relatedObjectIdentifierValue>
      </premis:relatedObjectIdentifier>
    </premis:relationship>
  </premis:object>
  <!-- A series of events describing check of the images that they are without problems. -->
  <premis:event>
    <premis:eventIdentifier>
      <premis:eventIdentifierType>SE/RA</premis:eventIdentifierType>
      <premis:eventIdentifierValue>31fe59de-b5a7-11e3-a7ff-
001b2126ecbe</premis:eventIdentifierValue>
    </premis:eventIdentifier>
    <premis:eventType>TIFF editing</premis:eventType>
    <premis:eventDateTime>2014-03-27T07:24:13+01:00</premis:eventDateTime>
    <premis:eventDetailInformation><premis:eventDetail>TIFF edite-
ring</premis:eventDetail></premis:eventDetailInformation>
    <premis:eventOutcomeInformation>
      <premis:eventOutcome>Status: OK</premis:eventOutcome>
      <premis:eventOutcomeDetail>
        <pre-
mis:eventOutcomeDetailNote>Profil:GREY;Gsuid:</premis:eventOutcomeDetailNote>
      </premis:eventOutcomeDetail>
    </premis:eventOutcomeInformation>
    <premis:linkingAgentIdentifier>
      <premis:linkingAgentIdentifierType>SE/RA</premis:linkingAgentIdentifierType>
      <pre-
mis:linkingAgentIdentifierValue>TIFFedit_ESSArch_SVAR</premis:linkingAgentIdentifi
erValue>
    </premis:linkingAgentIdentifier>
    <premis:linkingObjectIdentifier>
```

Fig. 9.2 (continued)

```
      <premis:linkingObjectIdentifierType>SE/RA</premis:linkingObjectIdentifierType>
      <premis:linkingObjectIdentifierValue>A0072716/c/MOD-
885_189_1112.tif</premis:linkingObjectIdentifierValue>
    </premis:linkingObjectIdentifier>
  </premis:event>
  <!-- Description of the agent creating the package and performing the events. -->
  <premis:agent>
    <premis:agentIdentifier>
      <premis:agentIdentifierType>SE/RA</premis:agentIdentifierType>
      <pre-
mis:agentIdentifierValue>TIFFedit_ESSArch_SVAR</premis:agentIdentifierValue>
    </premis:agentIdentifier>
    <premis:agentName>TIFFedit vid ESSArch_SVAR</premis:agentName>
    <premis:agentType>software</premis:agentType>
  </premis:agent>
</premis:premis>
```

Fig. 9.2 (continued)

Fig. 9.3 Finding aid for the AIP. In the header "Relaterade arkivenheter" field on the right we find the archival code ending with "A0072716" which is the same as the OBJID in the METS document and the objectIdentifierValue in the PREMIS document for the tar-file (The button "Bild" gives access to the image)

9.3.3.1 Digital Still Images

A typical use case for archives might be the digitization of their holdings of parish registers, creating master files of the records in the TIFF format that will be preserved in the electronic archival storage, and at the same time producing viewing copies in the JPEG format.

Format-specific technical metadata for a still image can be included, with the help of the standard Technical Metadata for Digital Still Images in XML (MIX) [18]. This includes metadata such as bit depth, color space, resolution, and so on.

9.3.3.2 Audio and Video Metadata

Let us take a real-world use case: the archive receives a collection of LP recordings of speeches made at the annual gathering. A video of the latest event is also included. At the delivery time the archive decides to create archival versions of the recordings and the video in the selected formats. They that will be preserved as master files in the electronic archival storage, while at the same time copies are made for access by users. The original delivered media is going to be stored in cold storage.

These types of metadata are covered in Chap. 5 on implementing PREMIS for audio–visual materials. As audio and video metadata standards have been slow to develop, information regarding use and examples of these standards is hard to find. One also needs to choose between simpler formats like audioMD and videoMD [19] hosted by the Library of Congress or finer grained formats like PBCore [20] or AES57-2011 [21] and AES60-2001 [22]. Nonetheless, a preservation plan for audio and video in the archival institution needs to be created in spite of these obstacles.

Worth mentioning is the Federal Agencies Digitization Guidelines Initiative (FAGDI) [23] a collaboration between the National Archives and Records Administration (NARA) and other federal agencies in the US. The work is based on developing standards for digitized audio and video, developing tools with Audio-Visual Preservation Solutions [24]. An XML-schema for video technical metadata called reVTMD [25] is under development, which includes not only technical metadata but also metadata about the original source. The initiative also gives guidance on the use of embedded metadata for images as well as audio and video.

9.3.3.3 Database Metadata

Let us take an illustrative use case: the archives receive a delivery of a database containing information about all the cats registered in the nation. The database is delivered as text files. The information about how the database was queried and used is missing at the initial delivery. The archive is fortunate enough to be able to get a new delivery following their instructions, including the queries, and also receiving metadata about how the database was structured.

The databases are among the most common digital objects that archives are tasked with preserving. A lot of work is and has been carried out regarding preservation of databases [26], notably by the Repository of Authentic Digital Objects (RODA) [27] and also the E-ARK project [6].

Many databases have the option of exporting the information as either text files or as XML documents; in either case the database itself needs to be described, but only a small number of standards are available for doing so. Consideration also needs to be given to the preservation of the database files themselves. Can the archive maintain the database as it is stored or should the information contained in it be exported and saved as files, together with accompanying information regarding

how the database was built and with Preservation Description Information? In such cases, structural information is essential in order to recreate the database so that a user (e.g., researcher) can query it, for example to create new information sets and perform data mining [28].

SIARD

Software Independent Archiving of Relational databases (SIARD) [29] is a format for long-term preservation of relational databases developed by the Swiss Federal Archives. The standard and the accompanying XML-schemas are free after a registration process. Update of the standard is being made in collaboration with the E-ARK project [6] which will make the format even more suitable for long-term preservation.

SIARD is based on XML, SQL, and Unicode standards, was developed in the earlier mentioned "PLANETS" project [5], and is used as the archiving format for relational databases in many countries of the world.

ADDML

Archival Data Description Markup Language (ADDML) [30] is a metadata standard originally developed by the Norwegian National Archives. It was designed to describe files and databases of all kinds when no other metadata standard like METS was available in the mid 1990s. Today the development is a joint work of the National Archives of Norway, Sweden, and Finland. Unlike SIARD, ADDML is not restricted to relational databases but also supports preservation of Microsoft Access, Filemaker or NoSQL databases for example.

9.3.3.4 Other Metadata Formats

An additional use case might be: the archive is receiving an electronic record management system for one of the agencies. A court case 10 years later mandates that all the documentation regarding one record handled by the agency needs to be found. Using the metadata information stored, it is possible to find the record. The court receives it together with the information regarding electronic signatures, proving the case.

Formats and metadata standards for describing, for example, records management, financial information, and HR information are currently being investigated in several different communities and projects like the E-ARK project and some efforts have resulted in a metadata format. These formats need to be evaluated and checked to see whether they will fulfill the needs of the archival institutions.

References

1. International Council on Archives. FAQs: All You have Ever Wanted to Know about Archives and ICA… http://www.ica.org/116/faqs/all-you-have-ever-wanted-to-know-about-archives-and-ica.html. Accessed 24 Jan 2016
2. The National Archives of Scotland (2013) Electronic records management systems. http://www.nas.gov.uk/recordKeeping/ERGuidance/ERMSystems.asp. Accessed 24 Jan 2016
3. International Council on Archives (2016) Glossary: finding aid. http://www2.archivists.org/glossary/terms/f/finding-aid. Accessed 24 Jan 2016
4. Consultative Committee for Space Data Systems, International Organization for Standardization (2012) Reference model for an Open Archival Information System (OAIS): issue 2. http://public.ccsds.org/publications/archive/650x0m2.pdf. Accessed 06 Jan 2016
5. PLANETS—Preservation and Long-term Access through NETworked Services (2010) Official Website. http://www.planets-project.eu/. Accessed 24 Jan 2016
6. E-ARK Project (2016) Official Website. http://www.eark-project.com/. Accessed 24 Jan 2016
7. International Organization for Standardization (2001) ISO 15489-1:2001: Information and documentation—records management—part 1: general. http://www.iso.org/iso/catalogue_detail.htm?csnumber=31908. Accessed 24 Jan 2014
8. DLM Forum (2016) MoReq2010: Modular requirements for records systems. http://moreq.info/. Accessed 24 Jan 2016
9. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 05 Jan 2016
10. Library of Congress, METS Editorial Board (2015) Metadata Encoding and Transmission Standard (METS): official web site. http://www.loc.gov/standards/mets. Accessed 05 Jan 2016
11. Electronic Simple European Networked Services (e-SENS) Project (2015) Official website. http://www.esens.eu/. Accessed 24 Jan 2016
12. W3C (2008) XML signature syntax and processing (second edition): W3C recommendation 10 June 2008. https://www.w3.org/TR/xmldsig-core/. Accessed 24 Jan 2016
13. Library of Congress, Society of American Archivists (2015) Encoded archival description (EAD): official web site. http://www.loc.gov/ead/. Accessed 24 Jan 2016
14. Staatsbibliothek zu Berlin, Society of American Archivists. Encoded Archival Context: Corporate Bodies, Persons, and Families (EAC-CPF). http://eac.staatsbibliothek-berlin.de/. Accessed 24 Jan 2016
15. Library of Congress (2015) METS profiles. http://www.loc.gov/standards/mets/mets-profiles.html. Accessed 24 Jan 2016
16. The National Archives of Sweden (2015) METS profile for all different IP in Sweden. View-source: http://xml.ra.se/e-arkiv/METS/CommonSpecificationSwedenPackageProfile.xml
17. Library of Congress (2009). Technical metadata for text (textMD). http://www.loc.gov/standards/textMD. Accessed 24 Jan 2016
18. Library of Congress (2015) NISO Metadata for images in XML schema (MIX). http://www.loc.gov/standards/mix. Accessed 24 Jan 2016
19. Library of Congress (2011) AudioMD and videoMD—technical metadata for audio and video: official web site. http://www.loc.gov/standards/amdvmd/. Accessed 24 Jan 2016
20. Public Broadcasting Metadata Dictionary Project (2015) PBCore Web Site. http://www.pbcore.org. Accessed 24 Jan 2016
21. Audio Engineering Society (2011) AES57-2011: AES standard for audio metadata—audio object structures for preservation and restoration. http://www.aes.org/publications/standards/search.cfm?docID=84. Accessed 24 Jan 2016

22. Audio Engineering Society (2011) AES60-2011: AES standard for audio metadata—core audio metadata. http://www.aes.org/publications/standards/search.cfm?docID=85. Accessed 24 Jan 2016
23. Federal Agencies Digitization Guidelines Initiative (FADGI) (2015) Website. http://www.digitizationguidelines.gov/. Accessed 24 Jan 2016
24. AudioVisual Preservation Solutions, Inc. (2016) AVPreserve website. https://www.avpreserve.com. Accessed 24 Jan 2016
25. The U.S. National Archives and Records Administration. reVTMD XML Schema. https://www.archives.gov/preservation/products/reVTMD.xsd. Accessed 24 Jan 2016
26. Wikimedia Foundation (2016) Database preservation: from wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Database_preservation. Accessed 24 Jan 2016
27. RODA Community (2014) Repository of authentic digital objects. http://www.roda-community.org/. Accessed 24 Jan 2016
28. Wikimedia Foundation (2016) Data mining: from Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/Data_mining. Accessed 24 Jan 2016
29. Swiss Federal Archives (2016) SIARD suite. https://www.bar.admin.ch/bar/en/home/archiving/tools/siard-suite.html. Accessed 24 Jan 2016
30. The National Archives of Norway (2016) Archival data description markup language (ADDML). http://www.arkivverket.no/arkivverket/Arkivbevaring/Elektronisk-arkivmateriale/Standarder/ADDML. Accessed 24 Jan 2016

Chapter 10
Digital Preservation Metadata Practice for Computing Environments

Angela Dappert and Adam Farquhar

10.1 Introduction

A digital object does not stand alone. We require a computing environment in order to render, interact with, or understand it. Over the long term, the computing environments that we use change dramatically so that the software, hardware, and formats that we once used are no longer widely available or even understood. Therefore, if we want to ensure the long-term usability of digital objects, it is necessary to either preserve their computing environments or at least bring together enough information so that the environment can be reconstructed or adapted to a changed world. Information that describes the components of a digital object's computing environment is a key part of its preservation metadata. The need becomes even more acute as we strive to archive audiovisual files, web pages with JavaScript and Flash, office documents and spreadsheets that embed complex calculations, or research outputs with data and software. Fortunately, widespread use of emulators and virtual machines and improved focus on managing software dependencies give us options that we have not had in the past.

Prompted by this growing demand, PREMIS version 3.0 [1] has changed the way computing environment information is recorded. The new approach greatly improves expressiveness and consistency. This chapter describes the basic concepts.

Computing environments are a key part of Representation Information in the OAIS information model [2]. Representation Information is "the information that

A. Dappert (✉) · A. Farquhar
The British Library, 96 Euston Rd, London NW1 2DB, UK
e-mail: angela.dappert@bl.uk

A. Farquhar
e-mail: adam.farquhar@bl.uk

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_10

maps a Data Object into more meaningful concepts" [2].[1] This is a very rich concept. Consider a simple .docx file. Its Representation Information includes the format specification that defines how to interpret the bit sequences, a list of software tools that can render it, and hardware requirements for that software. Representation Information for software may include documentation, manuals, cheat sheets, user behavior studies, and other aids for interpretation. Representation Information may go even further to describe the language a document is written in, the business context, underlying policy documents, guidelines, the author, the purpose and time of writing, the intended audience, and more. Different components of Representation Information link to each other and together create a Representation Information Network.

In the following, we use the term environment to refer to a component in the Representation Information Network. We distinguish the subset of computing environments as an essential part of Representation Information. They describe the rendering or executing components of the computing platform. Examples are operating systems, software applications, hardware or computing resources. They link to other forms of Representation Information, which are usually captured as documents. Together with the content objects that are preserved in a repository, they form a single renderable, preserved unit.

PREMIS version 3.0 can describe the whole rich landscape of Representation Information (a) through the reuse of the '*object'* entity for describing all types of environments, computing environments and others, (b) by offering an enriched set of relationships for expressing dependencies and (c) by incorporating *Intellectual Entities* that describe the wider context in the core data model, so that interesting relationships between *Intellectual Entities* can be described.

Environments are described in the established high-level PREMIS data model. In line with its familiar scoping principles, PREMIS describes high-level core preservation metadata. Detailed technical metadata may be specific to an environment type, such as operating systems software, a certain chip design, or details of an application software package. This additional metadata can be implemented as extension through external metadata frameworks.

[1]Computing environments are explicitly classified by OAIS under the "Other Representation Information" subcategory, along with Structural Information (e.g., formats) or Semantic information (e.g., documentation). "Software, algorithms, encryption, written instructions and many other things may be needed to understand the Content Data Object, all of which therefore would be, by definition, Representation Information, yet would not obviously be either Structure or Semantics. Information defining how the Structure and the Semantic Information relate to each other, or software needed to process a database file would also be regarded as Other Representation Information" [2: pp. 1–13].

10.2 Environments

To better understand computing environments, consider the statements that people make when asked what is required to use a software program or render a file. The following are typical examples. They give a sense of the richness and variety that environments must model.

- "This operating system only runs on a 64-bit environment." The environment describes a hardware environment, but it is a broad category comprising several hardware architectures.
- "This data object can be read on a European NES Games Console environment." Here, the environment is defined precisely and integrates hardware (including cartridge and controllers) and software (notably the BIOS) at the same time.
- "This ePUBReader plugin requires Firefox 3.0 or later as an execution environment." Here, the environment references software, without pointing to a precise version (all Firefox versions above 3.0 are supposed to work).

Environments and their descriptions have important properties:

- Environments apply to classes of digital objects.
- Environments can link to or depend on others.
- Environments can be composed of others at lower levels of granularity.
- Environments can themselves be content objects (for example, software source code) and be preserved as a first-class entity in their own right.
- Environments have a purpose (e.g., they allow objects to be rendered, edited, or executed)
- Environment descriptions may be incomplete with only some aspects specified, covering a wide range of real-world instances.
- Environment descriptions may be (overly) specific describing a specific real-world instance, out of the set of applicable ones.
- The relevant environment description depends on context and business requirements.

In the example in Fig. 10.1, the content file, *File 1,* requires a certain software application that in turn requires a software library. Both of them run on an operating system that requires a hardware architecture. The software also requires user input from a hardware peripheral, a keyboard or mouse, which in turn requires the same hardware architecture, and whose software driver requires the same operating system. This network specifies the rendering or execution stack (or platform) that is needed for long-term access to *File 1*. If parts of the rendering stack become obsolete, one can use this information to determine what newer platform they can be migrated to or how they can be replaced. Specifying environments in this way also enables reuse. For example, a second content file, *File 2*, requires the same software application. The existing description of the software application and all its dependencies can simply be reused.

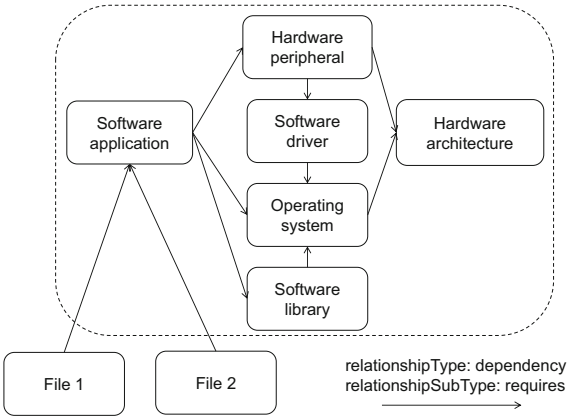


Fig. 10.1 Example: Environment stack and dependency relationships

Environment information can be stored in repositories or in registries and can be reused across institutional boundaries and across communities. For example, *File 1*, the software application and the software library might be local to the repository where they are stored, preserved, and described. But the operating system and the hardware architecture might be described and preserved in a shared external regisùtry.

10.3 Implementing Environments in PREMIS

10.3.1 The Data Model

PREMIS version 2 modeled computing environments as semantic units associated with content objects. This choice limited the description of computing environments and introduced some inconsistencies. It also suggested that information about computing environments should be repeated for each object, leading to inflated metadata files, duplication of information, and redundancy—although this was never actually required. Version 2 did not explicitly address how to handle other forms of document-based environments.

PREMIS version 3.0 models all types of environments using the Object entity. The new approach reuses the pre-existing PREMIS capabilities and supports rich descriptions of entities and connections among them. By treating environments as modular entities, the model easily supports their connected and inter-dependent nature. Environments can be defined by logical or structural relationships to other environments. They can be flexibly reused in order to create new environments to meet changing business needs.

Environments can now be described as *Intellectual Entities*, *Representations*, *Files* and *Bitstreams*—just like any other object. Furthermore, they can reuse the relationships to other PREMIS entities as depicted in Fig. 10.4 (more on

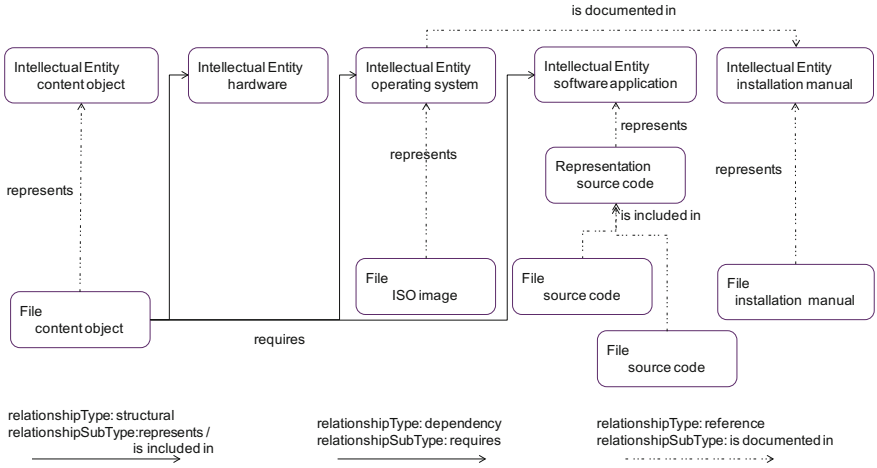


Fig. 10.2 Example: An object and its rendering environment

relationships is discussed below). This powerful model makes it straightforward to handle all of the use-cases that we have seen and encourages reuse of environment information.

Figure 10.2 shows an example of how the model can be applied. A content *File* object is related to its *Intellectual Entity* object through a *represents* structural relationship. Its *Intellectual Entity* object captures the descriptive metadata for the content file. The *File* object holds technical and other preservation metadata for the file. The *File* object also requires hardware, an operating system and software. They are each described as *Intellectual Entity* objects.

Assume that the operating system exists in the repository as an ISO image, which is a file. The *File* object holds associated technical and other preservation metadata for the ISO image. This ISO image *File* object is in a *represents* relationship with the *Intellectual Entity* that holds the descriptive metadata for the operating system. This reuses the same layered object category structure available for content objects: *Intellectual Entity*, *Representation*, *File*, or *Bitstream*. The content *File* object is in a *requires* relationship with the *Intellectual Entity* for the operating system.

The software application required by the file is modeled similar to the operating system. In contrast, the hardware platform is solely described through an *Intellectual Entity*, since it is a physical object and therefore cannot use the technical metadata for a *File* object. This means that, in this example, we only have descriptive metadata for the hardware, and do not link to an actual physical hardware object.

Figure 10.2 shows an additional relationship type: *is documented in*. It illustrates how a relationship can be used to link a computing environment object (the *Intellectual Entity* object describing the operating system) to an *Intellectual Entity* object that documents it (its installation manual). The installation manual is part of

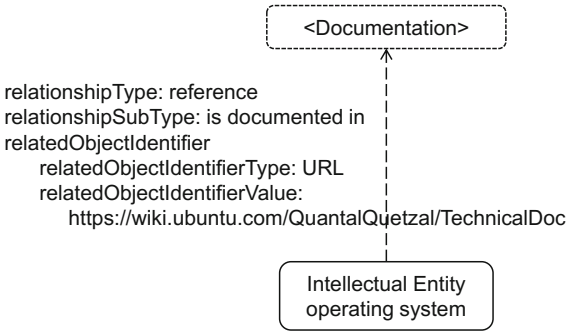


Fig. 10.3 Example: Relating an environment object and its documentation

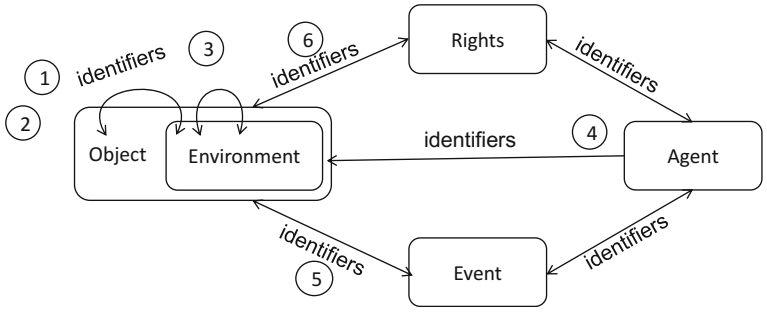


Fig. 10.4 PREMIS data model and relationships involving environments. The *numbers* distinguish relationship variants that are explained in the list above

the content object's Representation Information Network. In the example, the installation manual is preserved in the repository as a file. Alternatively, it might be held externally to the repository. In this latter case, the relationship could be modeled as shown in Fig. 10.3. In this example, the content object is identified through a link to an external location using a URL identifier instead of a repository internal identifier.

10.3.2 Relationships

The examples in Figs. 10.1, 10.2, and 10.3 illustrate a number of relationships involving environments. Many others are useful in the digital preservation domain as depicted in Fig. 10.4. Here is a non-exhaustive list of relationship variants among environments and between environments and other entities:

1. A content object is related to an environment in order to specify its Representation Information context (for example a *File* that relates to the software application that renders it. See Fig. 10.2).
2. A computing environment is related to a non-computing environment object (for example, to the documentation that supports its use, such as a user manual; installation, build and configuration instructions; or test suites. See Fig. 10.3)
3. A computing environment is related to another computing environment through inclusion, dependency, derivation or other relationships (for example, environments that interact within the rendering stack, such as applications, software plug-ins, operating systems, computer hardware and peripherals, or software drivers. See Fig. 10.1).
4. An environment takes the role of an agent (for example, a format migration software agent involved in a preservation action event is described and preserved).
5. An environment has an event associated with it that records its lifecycle (for example, the creation of an environment, adding computer memory/RAM, or a software versioning event).
6. An environment has a rights statement associated with it (for example, a software license or a policy rights statement).

This model allows for a consistent and powerful representation of environments. For example, rather than just specifying a name and version of software used in an event, one can create the complete description of the software agent in an environment *Intellectual Entity* object. One can then link from the software agent directly to its environment description. Using it in this way provides essential Provenance Information about the tools that were applied to a digital content object. Another use example is the ability to link research software and the data that it created with each other so that they can be stored, preserved and accessed together. This enables others to validate the data and to build on this research.

A rich relationship vocabulary may be appropriate for environment objects. Content objects that link to each other usually relate via *structural* or *derivation relationshipTypes* only. Examples for *structural relationshipSubtypes* may be *represents/is represented by*; *includes/is included in*; *has part/is part of*; *has root*; *has sibling*. An example for *derivation* relationship is *has source/is source of*. When environment objects are involved richer relationships need to be described. There may be *reference* relationships between an environment and its documentation, such as *documents/is documented in*. There may be *replacement* relationships between environment objects, such as *supersedes/is superseded by*. There may be a *logical* relationship of *generalizes/specializes* if one environment is a more generalized description comprising several more specific variants. And, finally, there may be *dependency* relationships that capture relationship subtypes such as *requires/is required by*, *deploys/is deployed on*, and *emulates/is emulated by*.

The actual vocabulary used to define relationship types can be taken from a controlled vocabulary, such as id.loc.gov, or can be defined local to a repository, and therefore vary between repositories. The example relationship types here are intended to illustrate meaningful relationship types in the digital preservation domain.

10.3.3 Describing Environments

PREMIS does not define any descriptive metadata for content objects, but relies on implementers to choose from a rich set of existing frameworks. However, there are no existing frameworks for core descriptive metadata that describe computing environments. Because of this, a core set of descriptive metadata semantic units have been added to the PREMIS Data Dictionary that are specific for computing environments. They support the description of the environment's function, designation, and links to external descriptions in registries. In addition, they support the description of the purpose and characteristics of a relationship with an environment. It is worth noting that PREMIS only applies descriptive metadata to *Intellectual Entity* objects. Therefore, the descriptive semantic units for environments are limited to *Intellectual Entity* objects that describe computing environments.

The example in Fig. 10.5 shows a description for the XP Professional Service Pack 3 operating system in the form of an environment *Intellectual Entity* object. Like all PREMIS Objects, it has (at least one) *objectIdentifier*. In addition, it defines the environment's function with *environmentFunction*. This can be iteratively refined through levels of increasing granularity. In the example, on the highest level, it is specified as software. On the next level, it is specified as an operating system.

In addition, it lists the environment's designation information—possibly recording names, versions, origin information, notes and offering an opportunity for embedding more detailed extension metadata.

The environment description can refer to external registries so that it can reuse descriptions prepared and shared in the community. Such registries need to be intended for long term, persistent recording of this information. And they need to be supported by sustainable governance structures in order to be part of a trusted

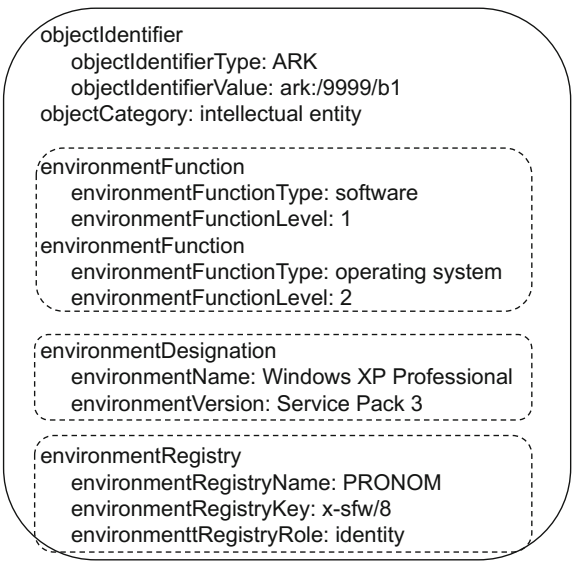


Fig. 10.5 Example: Descriptive metadata for an environment object

long-term infrastructure. In the example, instead of (or in addition to) giving a full description of the environment in the repository, it refers to the PRONOM registry with the environment information there. In the *environmentRegistry* semantic unit, one can record the registry's name, key and role.

Within an object's relationship with an environment one can also record the *purpose* and *characteristics* that the environment has for this object. Examples for purpose are *create*, *render*, or *edit*. Examples for characteristics are *minimal*, *recommended*, or *known to work*. They are an assessment of the extent to which the described environment supports the object. The two semantic units are associated with a relationship between an environment and another object that it supports, rather than with an environment object itself. This is because the same environment object can have different purposes and characteristics in different contexts.

In the example in Fig. 10.6, the object on the top represents an *Intellectual Entity* describing the Firefox software application. The content object, an HTML file, records a requirement dependency to Firefox. The *purpose* and *characteristic* semantic units distinguish why that is. Firefox can be used to render the file and the characteristic states that this is *known to work*. Alternatively, it could record relationships to *compulsory* or *recommended* environments. The underlying preservation policy would determine which types of environment specifications should be preserved for an object.

The PREMIS approach to describing environment information is not limited to digital objects but also includes physical ones.

Objects are not limited to those held within the repository. Objects may relate to things that are outside the repository, including physical objects. Physical objects can be environment objects such as physical hardware devices, but they can also be content objects, such as manuscripts or printed documents. Physical, like digital, objects can be described through an *Intellectual Entity* object and can be represented through a *Representation* object. Obviously *File* and *Bitstream* object categories do not apply to them. This approach means that digital and non-digital objects can be

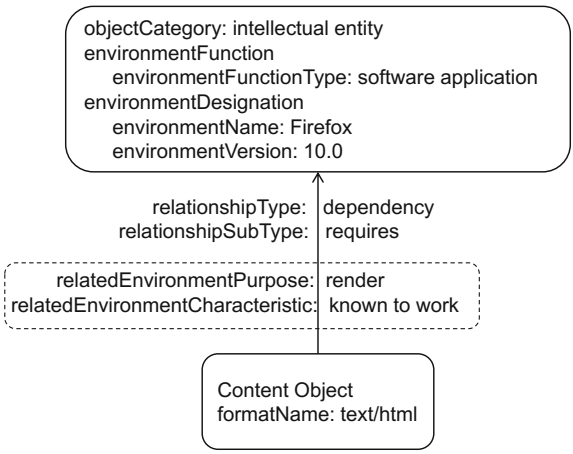


Fig. 10.6 Example: Describing the nature of the relationship

captured uniformly; they can be related to each other; and they can have storage information recorded for them. This feature supports seamless description of the objects, including environments, without artificial discontinuities created by the scope of the repository or the digital nature of the objects that can be described.

10.4 Conclusion

Over the past decade, repositories have focused on gathering digital objects, ingesting them, providing access, and ensuring that core digital preservation metadata is in place. This makes sense for the start-up phase and, even today, for repositories that preserve large volumes of objects in widely used and readily supported file formats. As repositories mature, it is increasingly apparent that recording computing environments is essential. This may be for audiovisual materials where there is a large variation of platform parameters that are essential to know for reuse of the objects. It may be for digital art, where slight changes in the rendering or execution can substantially alter the essence of the performance. It may be for original research, where often neither the file formats nor the software needed to use the files is publicly available. It may be for research data and the software required for their processing and analysis. And finally, detailed computing environment information is important to support emulation or reimplementation of obsolete computing platforms.

In the coming years, the core of digital preservation metadata must expand to embrace these new areas. PREMIS version 3.0 provides the extensible framework that we need. Dappert et al. [3] discusses initial requirements, options, and considerations that informed the definition of the PREMIS version 3.0 computing environment approach. It has since evolved further and been released as v3.0. In the next few years, the community will establish consensus on how the framework can best be used to describe computing environments. This will require practical implementation experience as well as innovation in practice as repositories expand their scope. It will also require innovative research, as well as reaching out to communities, such as those developing new approaches to virtual machines or reproducible research.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata (full document) version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 06 Mar 2016
2. Consultative Committee for Space Data Systems, International Organization for Standardization (2012) Reference model for an Open Archival Information System (OAIS): issue 2. CCSDS 650.0-M-2, Magenta Book, June 2012. http://public.ccsds.org/publications/archive/650x0m2.pdf. Accessed 06 Mar 2016
3. Dappert A, Peyrard S, Chou C, Delve J (2013) Describing and preserving digital object environments. New Rev Inf Netw 18(2):106–173. doi:10.1080/13614576.2013.842494

Chapter 11
Implementing Event and Agent Metadata for Digital Preservation

Euan Cochrane

11.1 Introduction

Event metadata is structured (human and machine readable) information that documents actions or activities that have happened and which relate to one or more objects that an organization is tasked with preserving.

Event metadata is crucial to enabling the people, processes, and technologies involved in preserving digital objects to successfully preserve those objects. Event metadata is the glue that joins metadata about objects that are managed by an organization, to metadata about the people, systems, or software that interact with those objects while they are being managed. Events are defined in PREMIS as follows:

> [An event is] an action that involves or impacts at least one Object or Agent associated with or known by the preservation repository. [1]

The relationship between Events, Agents and Objects in the PREMIS standard version 3 is highlighted in Fig. 11.1.

11.2 Key Metadata for Events

11.2.1 Events and Time

The PREMIS standard specifies only three semantic units that are mandatory for organizations implementing event metadata. *eventDateTime* is one of those mandatory semantic units and is an essential component of event metadata. (Note

E. Cochrane (✉)
Yale University Library, PO Box 208330, New Haven, CT 06520-8330, USA
e-mail: euan.cochrane@yale.edu

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_11

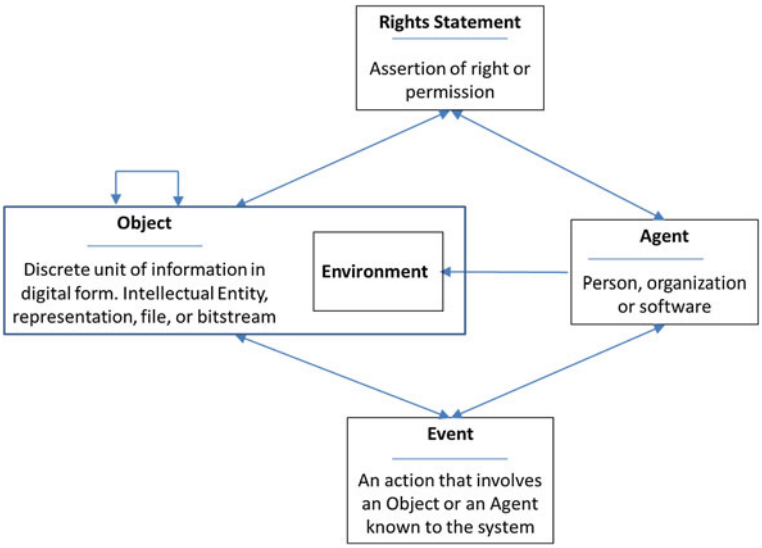


Fig. 11.1 The PREMIS data model (version 3.0) with event relationships highlighted

that the other two mandatory semantic units are *eventIdentifier* and *eventType*.) Capturing information about the time during which events occur is mandatory in the PREMIS standard, because events are always time bound. It is important to know when an action occurred on an object in order to track changes to the object over time. This means that events always occur at a point in time or during a period in time. The impact of this is that event metadata is always associated with information about the time that the event related to.

Event time periods need to be as explicit as possible in order to minimize ambiguity. In some cases, conventions are needed to further qualify the period, e.g., with date/time ranges, open-ended or questionable dates. Event time periods may be formatted using any form of time or date formatting provided it is standardized and understood within the context in which it will be used. When implementing PREMIS Events using the PREMIS XML schema, organizations will need to comply with the requirements of the xsd:date and xsd:dateTime datatypes. Where XML or W3C standards do not cover a specific type of date, *Extended Date Time Format* (EDTF [2] may be used (which extends xsd:dateTime to accommodate open-ended or questionable dates).

Event times and/or time periods may be unknown or uncertain. In such situations it may be necessary to specify this uncertainty by, for example, only specifying the day, month or year of the event, specifying a date range rather than a specific date, or using the conventions in EDTF.

11.2.2 Events and Processes

When implementing PREMIS it can be easy to confuse event metadata with process documentation. The Merriam-Webster dictionary defines a "process" as:

> a series of actions that produce something or that lead to a particular result. [3]

This definition highlights the difference between events and processes: Processes are linked series of actions intended to achieve one or more desired outcomes. Events themselves are single actions or activities that happen over a specific period of time.

An event can be considered to be an instance of a process, and as such it needs to be separately understood. For example, an organization might define a process that was used to migrate content in one or more source digital files of a particular format to one or more new target files in a different format. This process might involve multiple steps including:

1. Specifying the file(s) to migrate.
2. Initiating the migration software.
3. Running the migration software.
4. Creating the new file(s).
5. Verifying the migration.
6. Capturing metadata.
7. Ingesting the new files into the repository.
8. Notifying the content-owner that the content has been successfully migrated.

This process is documented in a basic form in the steps outlined above, and that documentation can be considered *process documentation*. Process documentation can be captured in various ways such as in a formal description in an office document, in structured XML files, or in Business Process Model and Notation (BPMN) [4]. In contrast, if a particular file or set of files is migrated using this process, then that instance of the application of that process could be documented as *event metadata*. E.g.:

eventidentifier
    eventIdentifierType: FDA-E
    eventIdentifierValue: E-2005-9963733
eventType: migration
eventDateTime: 20050705T077655-0500
eventOutcomeInformation
    eventOutcome : 00 [code meaning "successfully completed"]
linkingObjectIdentifier
    lnkingObjectIdentifierType: FDA-DF
    linkingObjectIdentifierValue: DF-2005-00 1013

11.3 Why Should Event Metadata Be Captured?

Event metadata is primarily captured to provide evidence of activities that affect one or more objects that the organization manages. Normally these events are undertaken by or involve at least one agent (a person, system, or software application). Once captured, event metadata can be used for many purposes:

1. *To prove the authenticity of an object, or set of objects*

   For example, to support the use of a digital document as evidence in a legal context.
   Event metadata helps to prove the authenticity of an object by documenting activities that may impact that authenticity, i.e., activities that may have altered the integrity of the object. By maintaining documentation of relevant activities organizations that manage digital objects can show that the objects that they provide to users are the same objects that they were tasked with preserving. Alternatively, this documentation can be used to show to what extent non-significant content has been lost as a result of undertaking essential preservation actions.

2. *To document the provenance of an object while it is under the management of an organization.*

   For example, to show that what a user has requested is what was originally received by the organization.
   Documenting the chain of custody of content allows its provenance to be proven, i.e., it enables an organization to show that no unauthorized users were able to interact with the content in ways that may have affected it. This is particularly important for content stored and provided digitally as it can need to be altered over time in order to maintain its accessibility. Event metadata can be used to prove the provenance of managed content by providing evidence of who controlled the content at any point in time, where the content was stored and when it was moved, and evidence of any alterations that were made to it during that period.

3. *To troubleshoot issues with the systems and actions used by an organization to manage its digital objects.*

   For example, to identify why an object does not render correctly in a specific rendering application (which may be identified by discovering that a migration process had introduced compatibility issues).
   By reviewing event metadata for anomalies it can be possible to discover how, when, and why an issue may have occurred in the systems managing the actions.

4. *To facilitate a security or trustworthy repositories certification audit* (*like* [5])

   For example, to provide evidence of trustworthiness or for compliance purposes.

In order to pass a security audit (e.g., against ISO/IEC 27001:2005 [6]) it is required that the repository provide evidence of interactions between agents, content, and the systems that manage that content that affect the objects' authenticity and provenance. Event metadata provides that evidence and is required for a successful security audit.
Event metadata is particularly useful for providing evidence that a content-managing organization actually followed their documented preservation policies and procedures. This evidence is used when such an organization undergoes a Trustworthy Repositories audit (e.g., against ISO 16363:2012 [5]).

5. *To aid in reversing or understanding a preservation action.*

For example, in order to identify which software environment is compatible with a migrated digital object it may be necessary to understand what software was used to undertake the migration (and therefore the precise form that the resulting object is stored in).
Preservation actions often need to be undertaken in circumstances of limited knowledge of the content or the future use of the content. This can sometimes lead to future practitioners needing to reverse the preservation actions or to gain a better understanding of how the content that was the subject of an action, came to be in the form that it eventually resided in. Event metadata can greatly aid in this process by providing documentation of actions that were undertaken on objects and documentation of the outcome of those actions. Event metadata can also document the links between source and target objects that may be interacted with or created when actions have been undertaken. This can help practitioners to better understand why the actions were undertaken, how to reverse them and/or how to get the best use out of the content in the form it came to be in. Such links can be implemented through PREMIS object-to-object derivative relationships. The event that resulted in the creation of the target object can be documented within the relationship to specify how such a derivation was made.

11.4 Where and How Should Event Metadata Be Stored?

Event metadata, like all PREMIS metadata, can be captured anywhere that is practical for the preserving organization to manage. In practice event metadata is often captured in many different systems and locations including:

- System logs.
- Stand-alone XML files.
- Databases.
- Within digital object files.

While event metadata can be and is captured in many diverse locations and systems, there are some options for storing event metadata that are more

advantageous than others. These options can be considered best practice recommendations.

To decide how to store event metadata in a particular organizational context it is worth reviewing the following considerations:

1. How frequently does the metadata need to be accessed?
   In order to minimize costs, metadata that needs to be frequently accessed may need to be stored in a system for which the cost of retrieval is not significant.
2. How timely does the access to the metadata need to be?
   In order to meet the demands of the designated community event metadata may need to be stored in systems that have low latency (time before the file is made available on request). Alternatively an object may not need to be accessed very quickly when requested, even though it may be requested very often.[1]
3. Can the metadata be stored in a system with at least the same level of risk of loss as the storage system of the objects the metadata pertains to? When a object and its metadata are stored separately this increases the risk that the link between the two will be broken (though a robust identifier assignment and management approach can mitigate this risk).

Event metadata is critical to proving the authenticity of digital objects. As such, it is best practice to ensure that the risk of loss of the event metadata is no greater than the risk of loss of the digital objects that it pertains to. Often this is achieved by storing at least one copy of the event metadata in a way that closely associates it with the digital files that make up the objects. Doing so ensures that if there is a system failure all the files that make up a digital object, including its event metadata, can be found in a single location. This approach will aid in a more rapid and successful recovery of the digital objects and will ensure that the provenance and authenticity of the objects can continue to be proven and maintained.

The implementation of these best practice recommendations may differ depending on the local requirements of the implementing organization. For example, organizations that have robust bit-preservation processes and systems in place for ensuring that their metadata management systems have a low risk of loss may find that there is no benefit to be gained from creating and maintaining a copy of the event metadata alongside the primary files that make up the digital objects that they are tasked with preserving. On the other hand, organizations that rarely need to access their event metadata may find that the best option for them is to keep a static copy of the event metadata alongside the primary data files that it pertains to (in order to ensure that they are equally well backed up, but to also minimize costs by not maintaining a separate metadata database). However, these scenarios are not mutually exclusive. Organizations may choose to store their PREMIS metadata in a database that is separate from the primary data files and keep a static copy alongside those files. Each organization looking to implement event metadata for preservation

[1]For example, full audit trails related to certain types of digital objects may not need to be able to be delivered to users as quickly as simple identifier or location information.

purposes should evaluate their environmental context and decide which options best fit their organization.

11.5 What Event Metadata Should Be Captured?

As with all metadata creation the decision as to what metadata should be captured will depend on what functionality the metadata is expected to support. The goal of capturing preservation metadata is to ensure that digital objects can be adequately preserved over time. More specifically, the goal of capturing event metadata is to ensure that evidence of activities that involved digital objects can be made available when needed, primarily for proving the authenticity of the objects and for troubleshooting issues. For this reason, event metadata should be captured whenever a significant activity is undertaken relating to the preservation of digital objects. The following discusses factors to consider in order to achieve the right balance between having too much metadata to manage and too little to satisfy preservation goals.

11.5.1 Metadata Bloat

"Metadata bloat" is the term used to describe the situation that occurs when an excessive volume of metadata is captured by an organization. It can cause headaches for organizations in both storage and processing:

- Continual growth in preservation metadata can eventually develop into a significant storage burden.
- Processing metadata can become resource intensive or impossible due to the sheer volume of transactions that need to be processed to understand the contents of the metadata.

In practice, organizations tasked with preserving large volumes of digital content often have to make difficult decisions about how best to achieve their ideal outcomes with their less-than-ideal budgets. This can manifest itself in relation to event metadata by forcing organizations to be parsimonious in respect to how many events they capture metadata about in order to ensure they do not succumb to metadata bloat.

11.5.2 Preservation Significant Events

As outlined, there are two competing considerations that are particularly important when implementing the capturing of event metadata:

- Ensuring authenticity and integrity.
- Minimizing metadata bloat.

Practitioners disagree as to how important point 2 is [7]. With the continual progress in storage cost/GB and transistors on a chip (Kryder's and Moore's laws respectively)[2] it can seem feasible to just capture everything that might be needed just in case it is eventually needed. However, unless an organization has an endless supply of funding, it is likely that it will need to make some practical decisions about the minimum requirements for capturing metadata about events. This process involves identifying which types of events are deemed significant enough to need to be documented and for the documentation to be preserved to the same risk-level as the primary content files.

Preservation significant events will differ depending on the organizational context in which event metadata capture is being implemented. Such events may include (but are not limited to):

- Integrity validation.
- Data replication.
- Content migration.
- Data movement.
- Data access.
- Security audits.
- Repository audits.
- Data ingest.
- Data compression.
- Data decompression.
- Data encryption.
- Data decryption.
- Deletion.
- Format normalization.
- Data replication.
- Virus checking.
- Data creation.
- Signature validation.

Some of these examples may be counterintuitive in any particular organizational context, but may be essential in others. For example, when a request for deletion of a digital object is made, an organization may be required to keep a record of that deletion for traceability and accountability reasons. Without keeping such a record the trustworthiness of the organization may be affected. Alternatively, in other organizations it may be essential to remove any trace of information about a deleted object from any and all systems that ever interacted with it.

[2]Though there is considerable concern regarding the continuation of both Moore's and Kryder's laws (see [8]).

When implementing event metadata capture, organizations should review their environmental context, and evaluate which events are significant in relation to their preservation goals. Provided the preservation goals are clear, this process should be reasonably straightforward, and should allow the organization to configure their systems and processes to best meet their preservation goals and outcomes.

11.5.3 Event Agent Metadata

Events are always undertaken by and/or involve at least one agent. These agents may be individuals or organizations, but may also be systems including software applications or programs or hardware products. Documenting the agents involved in an event is essential to ensuring accountability and enabling the provenance of the objects affected to be proven. The name (and version, in case of a software agent) recorded for any PREMIS agent should be applied consistently throughout a particular institution.

Agents involved in preservation events can be documented using the *linkingAgentIdentifier* semantic units within the event section of PREMIS.

The most valuable semantic unit to include when implementing this section in an organization can be the *linkingAgentRole* semantic unit. This piece of metadata captures how the agent was involved in the event. In addition to specifying the role it is also essential to document the standard identification information relating to the agent (*linkingAgentIdentifier*) in order to ensure the role information is useful. *agentName* and *agentVersion* (in the case of a software agent) may be recorded for any PREMIS Agent and should be applied consistently within a particular institution.

11.5.4 Pre-ingest Activities and Event Metadata

Many organizations tasked with undertaking the preservation of digital objects apply processes to their digital objects before they are ingested into their repository. Such processes can include:

1. Creation of digital objects (through digitization, normalization, web harvest, or other content creation processes).
2. Migration of content that is already inaccessible in current software, or that has been carried out as a result of risk assessment undertaken on the digital object.
3. Configuration of disk images for use in interacting with the content contained in digital objects using emulation tools and services.
4. Imaging disks to capture content into disk image files.
5. Documentation of digital objects.
6. Arrangement and description of digital objects.

7. Cataloging of digital objects.
8. Appraisal of digital objects.
9. Selection of digital objects.

All of these activities are important. However not all of them relate to the preservation of digital objects.

When an organization evaluates which events are significant to its preservation goals in order to assess which event types to capture, events that do not relate to the preservation of digital objects will normally not be included in the resulting requirements. This should not be seen as a reason not to capture information about these events: there may be many other reasons for capturing information about events within an organization that do not relate to the preservation of the objects. But it can provide a means for deciding how long each piece of event metadata needs to be preserved.

Preservation significant event metadata generally should be preserved for the life time of the object(s) that it pertains to (or even longer in the case of deleted objects for which a piece of evidence for their deletion is required), whereas other metadata may be able to be destroyed after a much shorter period of time. Disposing of event metadata can be advantageous for many reasons, for example, it can reduce storage, processing, and management costs. For these reasons it is important that organizations distinguish between preservation significant event metadata, and non-preservation significant event metadata, and ensure that all event metadata is preserved for the length of time appropriate to its purpose.

11.5.5 Event Outcome Metadata

In many contexts it may be useful to document the outcome of an event. This can be useful when an event has multiple acceptable outcomes (and one actual outcome) and/or in order to track when a particular preservation action was a failure (which is useful when planning future preservation actions). This information can be captured within the *eventOutcomeInformation* semantic unit in PREMIS.

The *eventOutcomeInformation* semantic unit has two subunits:

1. *eventOutcome*
2. *eventOutcomeDetail*

These units can be used to capture either a code that represents the outcome of the event, or a description of the outcome of the event. There are also two further semantic units which can be used to specify more detail:

1. *eventOutcomeDetailNote*
2. *eventOutcomeDetailExtension*

Where useful these can be used to add additional information about the outcome of the event.

In practice event outcomes are often quite easily described as either a success or failure. However by including these additional semantic units the PREMIS standard enables the documentation of events whose outcomes may be more complex, such as migration actions that create multiple files from a single source file. In any case, it is important to record the software agent that performed the action, as the *eventOutcome* could not be easily interpreted without knowing which agent did the assessment. This is particularly true for file analysis software: the same PDF files would appear to be valid against a particular version of JHOVE, and invalid against other versions.

11.6 Conclusion

Event metadata is necessary for ensuring that there is evidence of interactions between digital objects and agents within digital preservation systems. This evidence can be used for many purposes including ensuring success in security and trustworthiness audits and in proving the provenance and authenticity of digital content preserved by an organization.

Decisions about where and how to store event metadata are often dependent on the environment in which the preservation is being undertaken. While it can be important to store at least one copy of any event metadata alongside the data it pertains to, this can be avoided if the metadata storage systems have equally rigorous bit-preservation processes governing them.

Tough decisions often need to be made by organizations implementing the capture of event metadata in order to ensure they do not succumb to costly "metadata bloat." Therefore, organizations need to consider what metadata is important to the long-term preservation of their content before they begin capturing and preserving unnecessary metadata.

An illustration of a tool implementing PREMIS Events can be found in Chap. 17.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 3.0, p 7. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 6 Jan 2016
2. Library of Congress. Extended date/time format (EDTF) (2015) https://www.loc.gov/standards/datetime/. Accessed 6 Jan 2016
3. Merriam-Webster Incorporated (2014) Process definition. http://www.merriam-webster.com/dictionary/process. Accessed 6 Jan 2016
4. Object Management Group. Business process model and notation. http://www.bpmn.org/. Accessed 3 Jan 2016
5. International Organization for Standardization (2012) ISO 16363:2012 Space data and information transfer systems—audit and certification of trustworthy digital repositories. http://www.iso.org/iso/catalogue_detail.htm?csnumber=56510. Accessed 6 Mar 2016

6. International Organization for Standardization (2013) SO/IEC 27001:2013 Information technology—security techniques—information security management systems—Requirements. http://www.iso.org/iso/home/standards/management-standards/iso27001.htm. Accessed 6 Mar 2016
7. Gollins T (2009) Parsimonious preservation: preventing pointless processes! Online Information 2009 Proceedings. http://www.nationalarchives.gov.uk/documents/information-management/parsimonious-preservation.pdf. Accessed 6 Mar 2016
8. Rosenthal DSH, Rosenthal DC, Miller E, Adams I, Storer M, Zadok E (2012) The economics of long-term digital storage. http://www.lockss.org/locksswp/wp-content/uploads/2012/09/unesco2012.pdf. Accessed 08 Mar 2016

Chapter 12
Implementing Rights Metadata for Digital Preservation

Evelyn McLellan

12.1 Introduction

When repositories acquire and preserve digital objects, certain types of metadata are automatically generated by systems and software while others are added by users. Technical information about file formats or standard outputs resulting from preservation actions are typically machine-generated; descriptive or cataloging information, information about archival processes such as accessioning and appraisal, and information about intellectual property and other types of rights must usually be created at some point by the user and entered into software tools via data entry templates or other means. Repositories use different types of metadata for different purposes: for example, file format metadata can be used to assess format obsolescence risk and select preservation plans; descriptive information can be exposed in online access systems for discovery and citation purposes; and information about rights can be used as the basis for understanding the range of actions that can be taken by repositories with respect to the digital objects they have acquired.

Rights can be a complex area for preservation repositories. Copyright and other statute-based restrictions, restrictions imposed by licenses or donors, and restrictions derived from institutional policies can sometimes overlap and compete with one another. Metadata standards for capturing rights information need to be flexible enough for preservation repositories to record rights data in ways that best meet their needs, and software tools implementing those standards need to support this type of flexibility. However, the data still need to be standardized enough to support a common and consistent understanding of what the information means. Moreover,

E. McLellan (✉)
Artefactual Systems Inc., Suite 201–301 6th Street, New Westminster, BC, Canada
e-mail: evelyn@artefactual.com

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_12

rights information should be understandable to both human readers and software systems, which may be called upon to automate certain processes based on restrictions or permissions associated with digital objects. This chapter provides an overview of the PREMIS rights entity and how it is implemented in Archivematica, a digital preservation software system, in a way that attempts to allow repositories to address these complex requirements.

12.2 Rights and Permissions in PREMIS

The Rights entity in the *PREMIS Data Dictionary* provides a highly flexible framework for recording information about the rights and permissions pertaining to digital objects. The distinction between rights and permissions is important: rights, according the *Dictionary*, "are entitlements allowed to agents by copyright or other intellectual property law." Permissions flow from these rights, being "powers or privileges granted by agreement between a rightsholder and another party or parties" [1]. Rights captured in PREMIS can be abstract and pertain to objects not held by the repository; however, "[t]he minimum core rights information that a preservation repository must know…is what rights or permissions a repository has to carry out actions related to objects within the repository" [2].

The *Dictionary* defines three specific types of rights, or rights bases, and provides a means for the implementer to define permissions linked to these bases. The three defined rights are *Copyright*, *License,* and *Statute*; the *Dictionary* also provides an *Other* basis which can be used for rights not based on any of these. Once a rights basis is determined, permissions and restrictions can be recorded and associated with that basis. PREMIS does this by allowing one or more acts to be associated with a rights basis, and for each act to be refined by restrictions if there are any. Thus, a simple rights statement in a PREMIS implementation can consist of a *rightsBasis* (such as Copyright); an *act* (such as replicate), and information about any *restriction* on the *act* (for example, replication permitted only for the purpose of making preservation copies).

A given digital object in a preservation repository may be subject to multiple rights bases and accompanying acts and restrictions. For example, an object may be subject to both Copyright and freedom of information legislation, or it may be governed by a license agreement that places restrictions on modification, dissemination or other acts. Figure 12.1 shows how one digital object may have a number of associated rights bases, each of which may be linked to one or more *act*.

In Fig. 12.1, permissions to disseminate the object are governed by both a *Copyright* and a *Statute* rights basis. This may reflect a situation in which (as an example) there is no restriction on dissemination relating to *Copyright* but there is

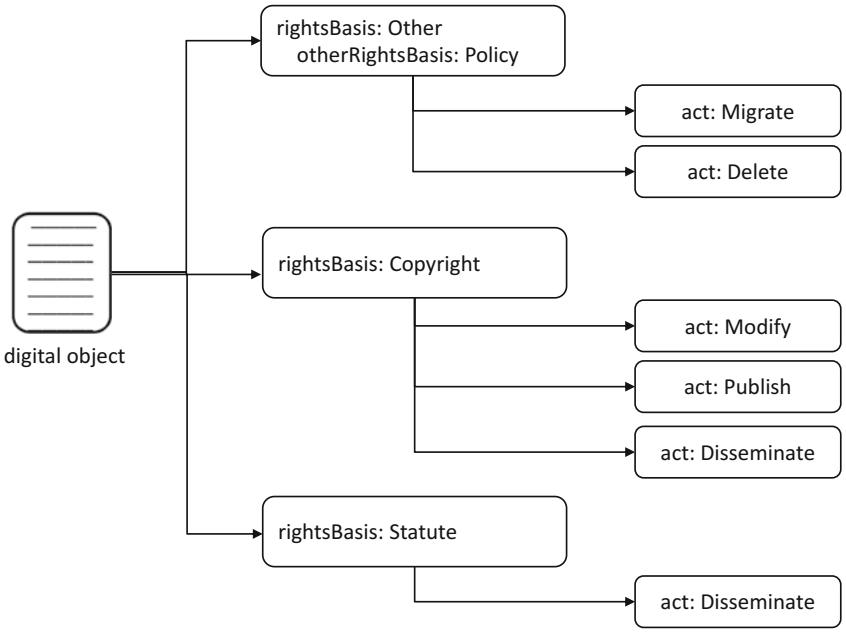


Fig. 12.1 Multiple rights bases and acts can apply to a single digital object

one relating to *Statute*, or vice versa. The ability to record multiple rights bases and related acts and restrictions makes the PREMIS Rights entity a powerful model for capturing these complex distinctions in a highly granular and consistent way. The *Dictionary* also provides several discrete elements for recording information about start and end dates of rights bases and permissions. Note that these may be different: for example, a donor agreement may apply to an object for as long as it is held by the repository, but a restriction on modifying the object may apply for only 10 years under the terms of the agreement. This type of precision and granularity distinguishes the PREMIS Rights entity from the types of rights statements found in descriptive metadata schemas, where rights information is often captured in only two or three fields. To take a few examples, Dublin Core has three elements: *rights*, *rightsHolder* and *license* (http://dublincore.org/documents/dcmi-terms); ISAD(G) has two elements, *conditions governing access* and *conditions governing use*; and MODS has one element, *accessCondition*.

For digital curators and developers of digital curation software, these aspects of the PREMIS Rights entity have a number of workflow and design implications. The first of these is that for any given object, one or more rights bases may be assigned, and one or more permissions may be granted (or restriction placed) on the object for each rights basis. Thus any data model for implementing rights in a digital repository should provide for one-to-many relationships between rights bases on the one hand and permissions on the other. Another implication is that any PREMIS rights-based metadata in a repository should support automating processes undertaken by digital repository and access software.[1] This means that there need to be strict rules for capturing information about rights, and a clear distinction must be made between actionable metadata fields and narrative fields which may provide valuable contextual information for curators but which cannot necessarily be parsed and acted upon by software.

12.3 The Archivematica Rights Module

Archivematica is an open-source suite of tools and workflows for preserving digital objects [3]. It is used to ingest bodies of digital objects, perform a series of preservation services on them and package them into Archival Information Packages (AIPs) for long-term archival storage. The stored AIPs consist of the original objects, any preservation copies of the objects generated during processing, and technical, preservation and descriptive metadata associated with the objects. Archivematica allows the user to add rights metadata in accordance with the PREMIS 2.2 Rights entity, both upon ingest and during processing. The system presents the user with a metadata entry template and applies the inputs in the template to all the digital objects within a Submission Information Package (SIP). Archivematica also automatically generates a wide range of technical and preservation metadata during processing, which correspond to the PREMIS Object, Event and Agent entities. During AIP preparation, the metadata are serialized as PREMIS XML and written to a METS (Metadata Encoding and Transmission Standard) file, along with the rights statements. This METS file is designed as a wrapper around the PREMIS, and is used to link all of the digital objects in the AIP to their PREMIS metadata. The METS elements also capture structural and descriptive information about the AIP as a whole, and may provide linkages between related AIPs, while the PREMIS metadata is designed to focus on the digital objects within

[1]Support for automated processing was built into the design of the PREMIS Rights entity. "[T]he Rights entity…is intended to support an automated process that determines if a particular preservation-related action is permissible in regard to an Object or set of Objects within the repository, as well as to record important information about the permission." "PREMIS with a fresh coat of paint: Highlights from the Revision of the PREMIS Data Dictionary for Preservation Metadata", Brian F. Lavoie, D-Lib Magazine, May/June 2008, http://www.dlib.org/dlib/may08/lavoie/05lavoie.html. See also "The automation of rights", Karen Coyle, The Journal of Academic Librarianship, v. 32, n. 2, May, 2006, pp 326–329 http://www.kcoyle.net/jal-32-3.html.

the AIP. The METS file forms part of the AIP when it is sent to archival storage, and is transferred along with other AIP contents when the AIP is moved between storage systems or repositories. The METS file is also copied to any Dissemination Information Packages (DIPs) which are generated from the SIP or AIP. When ingested, the user may choose to generate a DIP at the same time that the AIP is generated and thereby create an access copy.

From a rights metadata standpoint, this workflow has two important results. The first is that the rights information entered by the user is stored permanently with the preserved digital objects and associated with those objects through the METS file. This means that the rights information can later be acted upon by Archivematica, or by another system if the AIP has been transferred to another digital repository. The second is that the rights metadata can be passed along with access copies of the preserved digital objects to an access system, where they can be acted upon by any software capable of parsing the METS file.

Archivematica implements the one-to-many relationship between a rights basis and associated permissions described earlier. When the user opens the metadata entry template, the first step is to select a basis from a drop-down picklist. The system provides the three bases defined by the *Data Dictionary*, *Copyright*, *Statute,* and *License*, and additionally allows the user to select *Donor* or *Policy*. These additional choices reflect restrictions often placed on archival holdings by donor agreements and institutional policies, and speak to Archivematica's origins as a tool designed for archival repositories.

The rights basis selection affects the fields that are then presented for input. For example, if the selected rights basis is *Copyright*, fields for metadata entry include "Copyright status", "Copyright jurisdiction" and all the other PREMIS semantic units associated with the *Copyright* rights basis; if the selected basis is *Statute*, the fields change to "Statute jurisdiction", "Statute citation" and so forth. If *Donor* or *Policy* is selected as the basis, the field labels indicate this basis; in the PREMIS output these choices are expressed as *rightsBasis*: "Other" and *otherRightsBasis*: "Donor" or "Policy".

Figure 12.2, shows the metadata entry template and sample data entry in Archivematica where the selected rights basis is *Copyright*.

The data entry shown in Fig. 12.2 populates semantic units in the PREMIS Rights entity as follows:

rightsBasis: Copyright

copyrightInformation

copyrightStatus: Copyrighted

copyrightJurisdiction: CA

copyrightStatusDeterminationDate: 2009-01-03

copyrightNote: Copyright held by Caledonia Foundation

copyrightDocumentationIdentifier

copyrightDocumentationIdentifierType: Records transfer form

copyrightDocumentationIdentifierValue: CFA 2009-13

copyrightDocumentationRole: Copyright statement

copyrightApplicableDates

startDate: 2006-08-11

endDate: OPEN

Once a rights basis has been created and saved, the user may add one or more permissions associated with this basis. As discussed earlier in this chapter, PREMIS parses the concept of permission into acts and associated grants and restrictions, and Archivematica replicates this structure in its metadata entry template. Thus the user names an act, indicates whether there are any grants or restrictions pertaining to the act, and supplies further details in date fields and a free-text note field.

The data entry shown in Fig. 12.3 populates semantic units in the PREMIS Rights entity as follows:

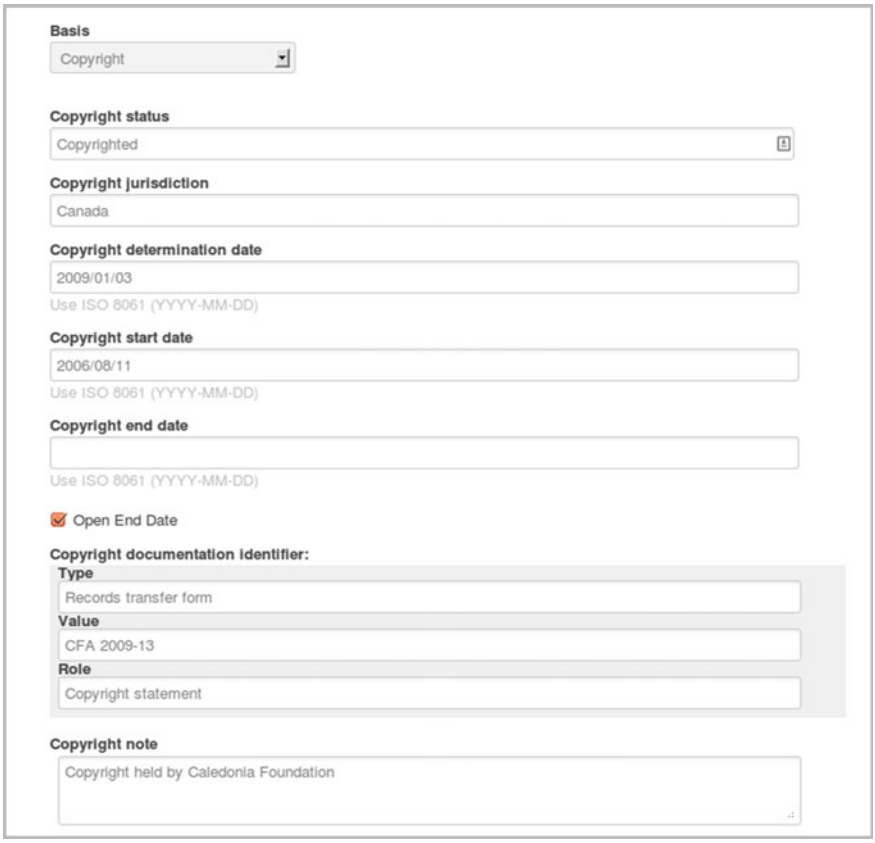


Fig. 12.2 Entering rights basis information using the metadata entry template

rightsGranted

act: Disseminate

restriction: Disallow

termOfRestriction

startDate: 2009-01-03

endDate: 2019-01-03

rightsGrantedNote: No public access until 10 years after transfer to Archives

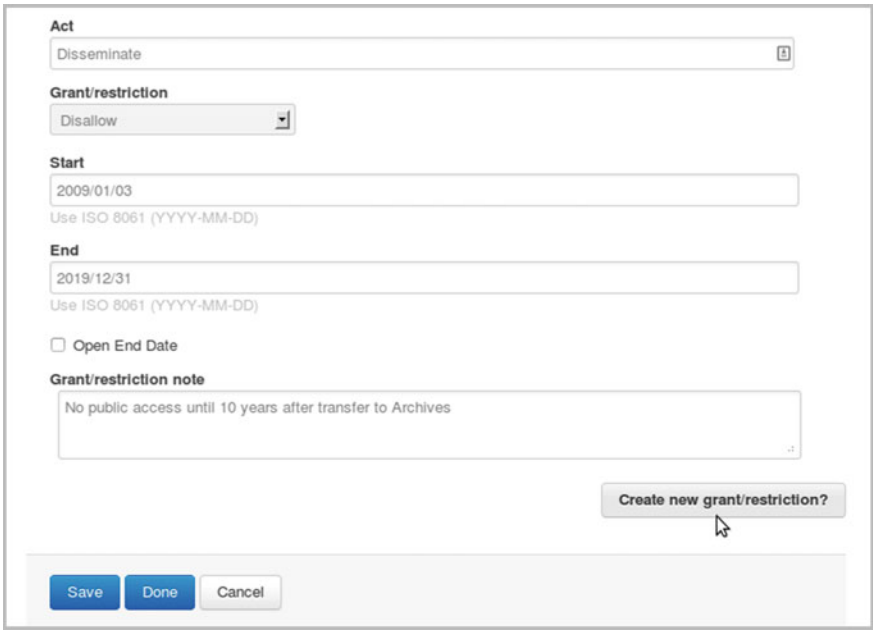


Fig. 12.3 Adding an *act* and associated *restriction* to the rights basis. The user has finished entering information about the act disseminate and now has the option of adding another act

Of special note is the highly structured implementation of the semantic unit *restriction* ("Grant/restriction" in the metadata entry template). The *Data Dictionary* defines this unit to mean "A condition or limitation on the act", and supplies examples of narrative text that could be used, such as "Allowed only after one year of archival retention has elapsed" or "Rightsholder must be notified after completion of act." However, Archivematica's designers decided early on to make this a more structured and limited field in order both to simplify the metadata entry process and to make the value machine-actionable to inform automated preservation or access decisions.

Thus the metadata entry templates allow only one of three possible values to be selected: "Allow", "Disallow," and "Conditional", which enforces specific combinations and makes the information machine-actionable by the system. If the user selects "Allow", Archivematica automatically creates a *termOfGrant* semantic unit to capture information about the start and end dates of the grant; if either "Disallow" or "Conditional" is selected, Archivematica creates a *termOfRestriction* semantic unit instead. More importantly, specific combinations of values in *act* and *restriction* can be used to automate rules relating to processing and access. For example, if *act* is "Migrate" and *restriction* is "Disallow", Archivematica or other preservation systems could be programmed to skip migration on the associated digital objects during a bulk migration action in the future; if *act* is "Disseminate", an access system can be programmed to check whether *restriction* is "Allow",

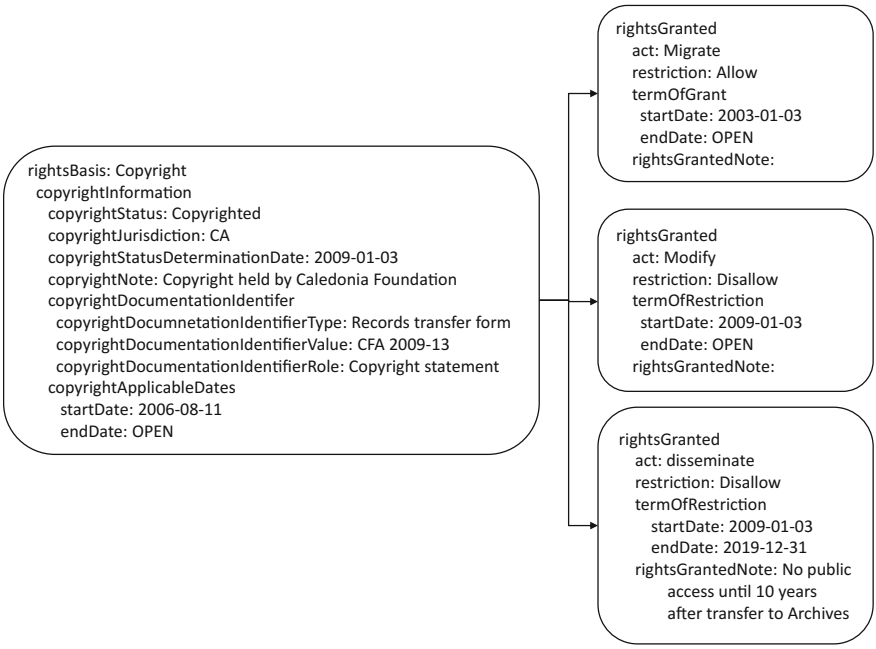


Fig. 12.4 A rights basis and related permissions

"Disallow," or "Conditional" and display or hide the digital object based on this information. The structured and granular nature of the restriction information also simplifies searching for digital objects in the preservation repository. For example, the user might want to search for all AIPs with specified *restrictions* that expire on a given date, by naming the act, selecting the restriction type and entering the restriction end date.

Note that a user can select "Conditional" as a value for *restriction*. This is to capture information about acts that can be carried out under certain circumstances. These circumstances should be specified in the *rightsGranted* semantic unit ("Grant/restriction note" in the metadata entry template). For example, the user might select "Publication" as the *act* and "Conditional" as the *restriction*, and add a note that permission to publish must be obtained from the copyright holder. As for "Disallow", if one selects "Conditional" in the *restriction*, this causes the *termOfRestriction* semantic unit to be used in the PREMIS output, rather than the *termOfGrant*.

Figure 12.4 shows sample outputs reflecting permissions linked to the *Copyright* rights basis.

12.4 Automating Decisions and Processes

The PREMIS Rights entity provides enough structure, granularity, and flexibility to have considerable potential for automation. In Archivematica, this potential is only just beginning to be exploited. One key area of development has been in the preparation and transmission of metadata to other systems, such as curation tools which use digital preservation software such as Archivematica as a backend. For example, Archivematica currently has the ability to write digital object metadata to Archivists' Toolkit (AT) [4]. When this function is used, certain PREMIS *acts* and *restrictions* are used to determine whether the <restrictionsApply> field in AT's database is set to TRUE or FALSE, and ultimately whether the digital object is displayed to end users online. The type of act in the PREMIS Rights entity also affects how certain AT note fields are populated. Work is currently underway to exchange rights metadata between Archivematica and ArchivesSpace [5]. Unlike Archivists' Toolkit, ArchivesSpace implements PREMIS rights using templates and outputs similar to Archivematica's, which opens up interesting possibilities for deeper integration.

Other uses for the PREMIS rights information captured in Archivematica have yet to be explored, and will depend on the priorities of those who use the system and who fund future development. Enhancements such as the ability to automate preservation processes based on specified permissions and restrictions or to invoke new processes when certain types of restrictions reach their expiry date, for example, await development. However, the structured nature of both the PREMIS Rights entity and the METS file in which the information is embedded provides a firm foundation for building on what has already been accomplished.

References

1. PREMIS data dictionary for preservation metadata, version 3.0, p 178
2. Ibid
3. https://www.archivematica.org
4. Archivists' toolkit. http://www.archiviststoolkit.org/. Accessed 14 Feb 2016
5. ArchivesSpace. http://archivesspace.org/. Accessed 14 Feb 2016

Chapter 13
Serialization of PREMIS

Thomas Habing

13.1 Introduction

Serialization typically describes how a data structure or data model is converted into formatted bits that can be stored in some physical medium, such as disk, tape, or computer memory, or transmitted across a network. The goal is to be able to recreate a semantically equivalent data structure or data model by reading the serialized, formatted bits from the storage media or from the network. This chapter discusses the common serialization options XML [1], Linked data [2], and relational databases [3] and applies them to possible implementations of the PREMIS Data Dictionary [4].

In some situations, a serialization process may include transformations, possibly, for example serialization into a similar but not equivalent data model. However, formally this would be considered as two different processes, a transformation or mapping and then a serialization or vice versa, but often the two operations can become conflated in actual data management systems. This chapter uses a somewhat less formal definition of serialization which may include transformations as well as marshaling to or from some storage medium or network.

13.2 Implementation Options

There are a number of factors to consider when weighing serialization options. Historically the compactness of the serialization format was an important consideration, reflecting the need to optimize scarce storage and network bandwidth.

T. Habing (✉)
Library, University of Illinois at Urbana-Champaign, 1408 W Gregory Dr, Urbana, IL 61801, USA
e-mail: thabing@illinois.edu

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_13

However, with current storage and network bandwidth capacities at magnitudes greater than they were even a few years ago this is generally not a major concern for the types of systems that would be dealing with PREMIS or similar metadata. However, in some situations this might still be a factor, such as the need to transmit millions of PREMIS Events over the network on a regular basis, for example in a continuous checksum validation scenario. In this kind of scenario using the most compact serialization format could be very important. An example of a compact serialization format for PREMIS is serializing PREMIS XML using the Efficient XML Interchange (EXI) Format 1.0 [5].

Another factor to consider is processing efficiency. This can include not just efficiency of the serialization process itself, but also how efficiently the serialized data may be queried, accessed, and manipulated in the underlying storage medium. Performance considerations can influence decisions such as what type of database to choose, such as a traditional SQL relational database, a SPARQL RDF database, a native XML database, or some hybrid approach, or in some cases just storing PREMIS XML files to a disk using a standardized file naming scheme.

There are also human factors to be considered. For example, the original design goals for XML [1] included "human legible and reasonably clear," "easy to create," and "easy to write programs which process XML." These are among the reasons that XML has become a popular serialization format for various metadata schemas. (It is interesting to note that compactness was explicitly not a goal for XML.) However, the human factor should extend not just to the creators and maintainers of the data themselves, but also to the developers and maintainers of systems that must manipulate the serialized data. XML strikes a good balance between human readability and editing and computer processing, especially with its large suite of developer and editing tools. However, serialization formats such as JSON [6] and YAML [7] have become popular with software developers because they make it easier to manipulate the data in various programming languages. There are numerous conventions and tools that can convert between these formats, for example BadgerFish [8] which has some support in different programming APIs and tools [9] as a convention for translating XML to JSON. There are also other JSON/XML conversion tools with different conventions, both open source and commercial [10–12].

Because of their standard query languages, APIs and performance characteristics, various databases are also used to serialize PREMIS data. This could include not just relational databases, but also RDF SPARQL triple stores, or index and search engines such as Apache SolR/Lucene [13]. The Resource Description Framework (RDF) model and one of its serialization formats, such as Turtle, N3, JSON-LD, or RDF/XML [2], along with an RDF SPARQL [14] triple store, would be a good choice when there is a requirement to easily merge different datasets even if they do not share a common underlying schema.

Finally, the structures inherent in the data model itself can often lead toward a specific serialization format. For example, hierarchical data models are well suited to serialization as XML. Well-defined relational models lend themselves to SQL databases. Models consisting of nodes connected by directed edges (directed

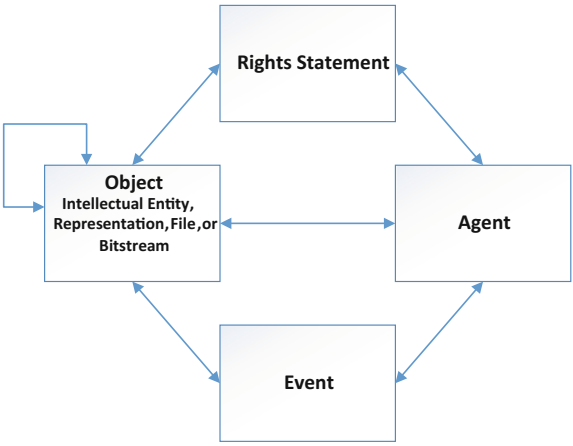


Fig. 13.1 Simplified PREMIS data model

```
1.5 objectCharacteristics
    1.5.1 compositionLevel
    1.5.2 fixity
        1.5.2.1 messageDigestAlgorithm
        1.5.2.2 messageDigest
        1.5.2.3 messageDigestOriginator
    1.5.3 size
    1.5.4 format
        1.5.4.1 formatDesignation
            1.5.4.1.1 formatName
            1.5.4.1.2 formatVersion
        1.5.4.2 formatRegistry
            1.5.4.2.1 formatRegistryName
            1.5.4.2.2 formatRegistryKey
            1.5.4.2.3 formatRegistryRole
        v1.5.4.3 formatNote
```

Fig. 13.2 Fragment of *objectCharacteristics* hierarchy

graphs) lend themselves to RDF. However, this does not preclude serializing a hierarchical data model into a relational database, for example, or serializing a relational model into XML, among other combinations. It is also becoming common to serialize both relational and hierarchical models as RDF, including standardized mappings and tools to support the conversions, such as the W3C recommendation "A Direct Mapping of Relational Data to RDF" [15] or the Ontmalizer [16] which can convert an XML Schema into an RDF ontology.

Interestingly, the PREMIS data model includes relational components, such as the core entities of the model and their relationships, as shown in Fig. 13.1.

However, the semantic units included within the individual core entities are often modeled as hierarchies, for example Fig. 13.2. This may be done simply because the parent in the hierarchy provides a convenient grouping or container for related elements, but in other cases the hierarchy provides semantics critical for

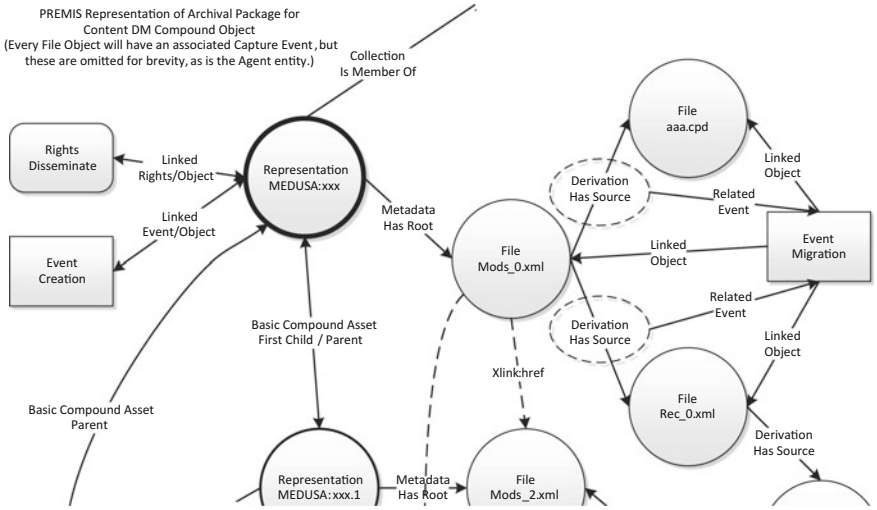


Fig. 13.3 Example PREMIS Object relationships

determining the meaning, such as tying a fixity value to the algorithm used to generate it.

In addition, the PREMIS model also has some features of a directed graph, especially in the way that objects can be related to each other via the *relationshipType* and *relationshipSubType* semantic units, as shown in Fig. 13.3. For example, the root *Representation* object is linked to a metadata *File* object via the *relationshipType*/*SubType* of *metadata*/*has root*, and this metadata file is in turn related to two other metadata *File* objects via the *derivation*/*has source* relationships.

Because PREMIS has aspects of relational, hierarchical, and directed graph models there is no single clear-cut serialization that is optimal for all use cases. The remaining parts of this chapter will explore the issues associated with serializing PREMIS as XML, RDF, and relational databases.

13.3 XML Implementation

In addition to being a text-based format for creating structured documents, XML also includes a suite of related standards [17], including XML Namespaces, XML Schema, XSLT, XQuery, and numerous others. This chapter assumes a basic understanding of XML, Namespaces, and to a lesser extent XML Schema. However, there are a few somewhat more advanced concepts that will be introduced

as needed. All of the examples used in this chapter assume the following namespaces are declared[1]:

```
xmlns="info:lc/xmlns/premis-v2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:txt="info:lc/xmlns/textMD-v3"
xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#
```

These namespace declarations will be omitted for brevity. As mentioned in the preceding part of this chapter, because of its original design goals of human legibility, ease of creation, ease of machine processing, and the widespread availability of tools, XML has become the default serialization format for numerous metadata schemas, including PREMIS. The official XML Schema for PREMIS is maintained by the PREMIS Editorial Committee, and it is hosted at the Library of Congress' website. As of this writing the PREMIS Editorial Committee has released version 3.0 of the Data Dictionary, and the corresponding XML schema was finalized in January, 2016. Because the 3.0 XML schema was not yet finalized during the writing of this chapter, the version 2.3 schema, dated August 4, 2014, will be used for all examples throughout this chapter. The schema can be found at this URL, http://www.loc.gov/standards/premis/v2/premis-v2-3.xsd.[2] Differences between the 2.3 and 3.0 versions of PREMIS which might impact this chapter will be noted in footnotes. Finally, previous versions of the PREMIS XML schema are available from this URL, http://www.loc.gov/standards/premis/schemas.html.

13.3.1 Organizing PREMIS XML Documents

The top-level structure of the PREMIS XML schema maps very closely to the top-level data model expressed in the PREMIS Data Dictionary. There are four top-level XML elements that map directly to the Data Dictionary entities[3]:

```
<object xsi:type="bitstream">…</object>
<object xsi:type="file"> …</object>
<object xsi:type="representation"> …</object>

<event>…</event>
<agent>…</agent>
<rights>…</rights>
```

[1]For the 3.0 XML schema the namespace will be http://www.loc.gov/premis/v3.

[2]The URL, http://www.loc.gov/standards/premis/premis.xsd, always points to the latest XML schema version.

[3]Version 3.0 will add <object xsi:type="intellectualEntity">…</object>.

Notice the three types of objects; in the PREMIS XML Schema the <object> element is defined as abstract. This means that all <object> elements must explicitly declare their subtype: *Bitstream*, *File*, or *Representation*. This is done using the special xsi:type attribute as shown above. These subtypes map directly to the same concepts in the PREMIS Data Dictionary, allowing the XML schema to validate that the different object types conform to their corresponding data models in the dictionary.

Finally, there is also a top-level container XML element, <premis>, which is used for grouping any combination or number of the above elements into a single XML document, if desired:

```
<premis>
  <object xsi:type="file"> …</object>
  <event>…</event>
  <agent>…</agent>
  <rights>…</rights>
</premis>
```

These top-level XML elements allow some flexibility in how PREMIS XML documents can be organized and stored. The <premis> container element allows all object, event, agent, and rights entities comprising an *Intellectual Entity* to be contained in a single XML document. This single document makes it easier to move or share the entity as a self-contained whole, for example moving the entity between repositories or sharing the entity with external partners. Using terms from the OAIS Reference Model [18], a single self-contained PREMIS XML document might be a good choice for a Submission (SIP) or Dissemination Information Package (DIP).

However, the single self-contained document approach can also introduce inefficiencies and management overhead, especially if the data are being actively updated or there are significant redundancies. For example, if all the objects in a repository are subject to one of three possible PREMIS rights statements, it would be inefficient to repeat one of the rights statements in every <premis> container document. Instead, a single XML <rights> document would be created for each rights statement and then the appropriate statement would be linked to from the <object> that utilizes that statement (see Fig. 13.4). This also makes it easier if a rights statement ever needs to be modified; it can be modified in one place instead of needing to find and modify it in multiple places.

Similarly, if all *File* objects are subject to regular checksum validation that requires the creation of a new PREMIS Event, it can be easier to create all the events as stand-alone <event> XML documents which link back to their associated objects, as opposed to needing to continually open each XML document to add a new <event> element. Creating stand-alone entities also avoids the creation of large, unwieldy XML documents. Therefore, the alternative to a single, self-contained <premis> container XML document is to have each top-level PREMIS entity contained in its own entity-specific XML document.

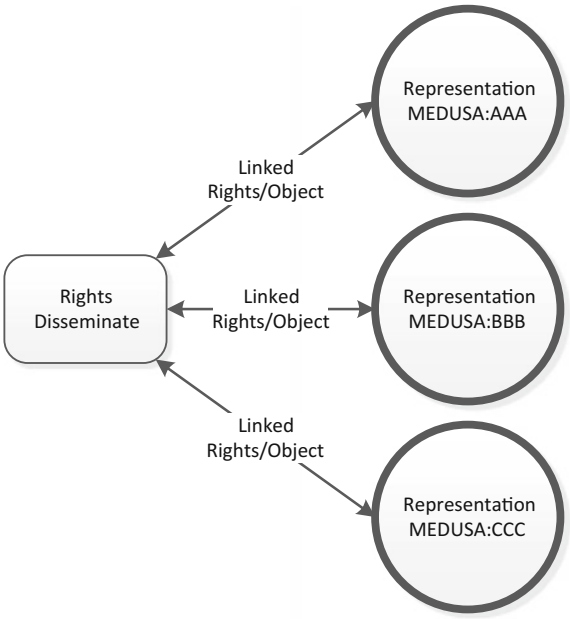


Fig. 13.4 Link to a single rights statement instead of repeating it for each object

13.3.1.1 Linking Between Entities Within PREMIS Documents

The above two organizational options lead naturally to a discussion of how to link between the various PREMIS entities in the context of an XML document, such as between an object and its related event.

13.3.1.2 Linking with XML Elements

The preferred linking mechanism is via the mandatory <[entity]Identifier> elements associated with each type of PREMIS entity. Any <linking[Entity]Identifier> or <related[Entity]Identifier> element must then reference one of these identifiers, as shown in Fig. 13.5.

All PREMIS identifiers consist of a type and value. The <object> and <agent> entities allow for multiple alternate identifiers, any one of which can be used for linking. Allowing multiple alternate identifiers can make the location or identification of these entities more robust over time. Because events and rights are usually specific to a given repository or archive system, these entities are assigned only a single identity which is usually local to the repository.

The advantage of using the above identifiers and linking elements is that the identifiers used can be of any type, such as URIs, Handles, or local identifiers. Using this type of identification schema also allows the various PREMIS entities to be split into separate XML documents without worrying that ID and IDREF links

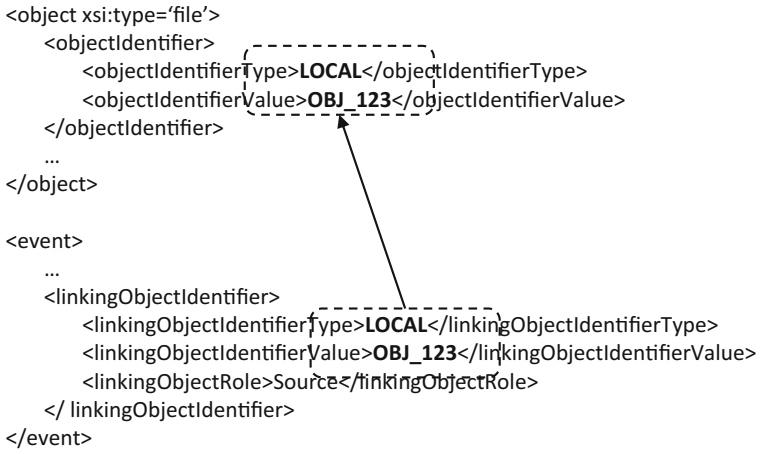


Fig. 13.5 XML fragments illustrating how an event is linked to an associated object

will be broken causing XML validation to fail; see the next part for details. For example, if the same agent participates in multiple events across multiple PREMIS Object entities, the agent only needs to be described once in an <agent> XML document. It can then be referenced from multiple places without the need to copy the entire agent description into each *File* that references it. Of course this requires that all identifiers across the entire corpus must be unique, i.e., the identifier value is unique within its identifier type scheme. Ideally the identifiers would be persistent and globally unambiguous which is why URIs are popular, especially for object and agent entities. Event and rights entities are typically assigned as local identifiers; although, they still need to be unique within the system. The disadvantage of this approach is the overhead required to manage all the identifiers and ensure that links are not broken over time. However, identifier and link management is a critical component of almost any system required to manage digital objects for long-term preservation, in any case.

13.3.1.3 Linking with ID and IDREF Attributes

The PREMIS XML schema also optionally supports internal linking between entities using attributes of the XML ID and IDREF data types. Each of the four top-level PREMIS elements has an ID attribute named xmlID, and all of the <linking[Entity]Identifier> or <related[Entity]Identifier> element types have a corresponding 'Link[Entity]XmlID' or 'Rel[Entity]XmlID' attribute which is an IDREF. These IDREF values must point to one of the xmlID attributes, as shown in Fig. 13.6.

It is worth noting that even if the ID and IDREF attributes are used to create links between entities, the complete identifier type and value elements are still required both in the source and the target entities; although, their values can be different than the ID or IDREF values.

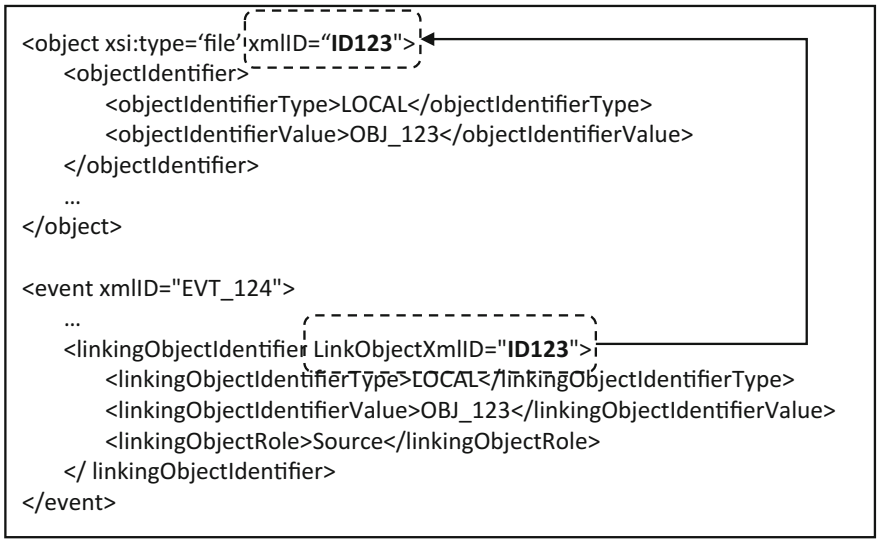


Fig. 13.6 XML fragments illustrating how an event is linked to an associated object using ID and IDREF attributes

The primary advantage of using ID and IDREF attributes is that most XML parsers can automatically validate the linkages, ensuring that there are no duplicate ID values within a single document and ensuring that each IDREF value refers to a corresponding ID value in the same document. There is also some support for this style of identifier in the general Web architecture, for example using fragment identifiers appended to URIs to identify a specific subsection of a document by its ID value. For example, the URL http://example.org/premis.xml#EVT_001 could identify the <event> with xmlID='EVT_001' in the premis.xml document. This means that if a <premis> container document is published to a specific URL, it is easy for external applications or documents to refer to specific entities in the PREMIS document so long as they are assigned a unique xmlID attribute.

There are some disadvantages to the ID/IDREF approach, one being that if IDREF attributes are used, the PREMIS entities with the corresponding ID values must be in the same XML document as the IDREF attributes; otherwise, XML validation will fail. This generally requires bundling all the linked or related PREMIS entities into a single <premis> container XML document. Another disadvantage is that the XML ID data type is restricted to a specific syntax and set of characters. Generally, it must start with an underscore or letter followed by letters, digits, or some other special characters, such as periods or dashes. For example, this means that most URIs are not valid XML IDs.

13.3.1.4 Linking with XLink

Another linking mechanism allowed by the PREMIS XML schema is the use of XML Linking Language (XLink) Simple Links [19]. Simple XLinks allow a small

Fig. 13.7 Example showing XLink pointing from an event to the associated external object

number of specialized attributes to be attached to an element. These attributes are used to specify the meaning and behavior of a single outbound link from the element to some external resource, similar to an HTML hyperlink using the <a> tag. The PREMIS <linking[Entity]Identifier> and <related[Entity]Identifier> element types allow for XLink Simple Link attributes which could be used as an alternate method for pointing to linked entities or related objects. It has an advantage over the IDREF in that it must be a URI, allowing it to point to external files, and it could be a URI with a fragment identifier pointing to a specific entity's xmlID in a different file. Figure 13.7 shows an event that points back to an associated object through the use of Simple XLink attributes. Notice that the event and object entities can be in separate files or even in separate servers at different institutions as long as they have a reliable URL locator.

Note that PREMIS version 3.0 no longer supports XLink. This eliminates the reliance on the XLink namespace and external schema, which has had limited use and has caused unnecessary complexity in other XML schemas (e.g., EAD, MODS). The linking functionality is now built-in using a local simpleLink attribute.

13.3.1.5 Linking Conclusions

As the above options illustrate, there are multiple ways to link the various PREMIS entities together. All entities must have a mandatory <[entity]Identifier> element, and the <linking[Entity]Identifier> and <related[Entity]Identifier> elements are also mandatory if you want to establish links between entities. In addition to these methods, you can also employ the ID/IDREF attributes if you want to establish some additional rigor to your linking and all related entities are contained in a single XML document. Similarly, the Simple XLink attributes can be employed if you have software tools that can process XLinks, or possibly if you have a need to link between entities that cross XML documents or even institutional boundaries.

Another issue associated with linking between PREMIS entities is directionality. The PREMIS Data Dictionary and the XML Schema allow for two-way linkages, but they are not mandatory. For example, an object could link to all of its events and all of the events could link back to the object, or the object may not have any links to the events, but the events link back to the associated object. There are factors to consider when deciding the directionality of links. One factor is the cardinality of the linkages and efficiency of updating entities over time. A common example is frequently generated events such that an object accumulates a large number of events over time, for example in a continuous checksum validation scenario. Continually updating the object to link back to these events can be inefficient and will lead to XML bloat as the object's number of associated events continues to grow over time; instead it would be more efficient to generate the events as needed and have them point back to the associated object. In general, most information retrieval systems, such as relational databases, would not require that all entities support two-way linkages; a database would just be organized in the typical normalized fashion for one-to-many and many-to-many relationships. However, in some cases you might improve performance if two-way linkages are supported, especially in the case of RDF Linked data where two-way linkages might substantially improve the traversal of the entity graph for certain retrieval operations. Implementations will need to weigh their own requirements against the characteristics of their technology stacks when making these decisions.

Finally, another factor that could affect which linking mechanisms to utilize would be the use of container formats such as METS.[4] PREMIS in METS is a common enough combination that there are guidelines for its use, "Guidelines for using PREMIS with METS for exchange" [20]. Generally, the advice is to utilize METS ID and IDREFs attributes (not PREMIS ID or IDREFs) to establish links between METS sections that contain PREMIS entities, but to use the PREMIS <[entity]Identifier> elements and <linking[Entity]Identifier> or <related[Entity]Identifier> elements to establish links between PREMIS entities.

[4]The use of container formats is further discussed in Chap. 14.

13.3.2 External Content in PREMIS

PREMIS supports numerous elements that allow arbitrary XML content from other schema or namespaces to be embedded within a PREMIS XML file. The following elements allow for externally defined extensions[5]:

<agentExtension>	*<eventDetailExtension>*
<rightsExtension>	*<eventOutcomeDetailExtension>*
<creatingApplicationExtension>	*<objectCharacteristicsExtension>*
<environmentExtension>	*<signatureInformationExtension>*
<environmentDesignationExtension>	*<significantPropertiesExtension>*

All of the above are defined in the XML schema using the extensionComplexType which allows these elements to contain a sequence of any number of arbitrary XML elements from any namespace. The elements are also declared as "lax" meaning that if an XML schema is available for the embedded elements it will be used to attempt to validate them, but it is acceptable if there is no schema available. An example might be using the TextMD [21], Technical Metadata for Text, schema to provide detailed technical metadata to the *objectCharacteristics* container.

```
<objectCharacteristics>
<compositionLevel>0</compositionLevel>
<format>
    <formatDesignation>
        <formatName>text/plain</formatName>
    </ formatDesignation >
</format>
<objectCharacteristicsExtension>
    <txt:textMD>
        …
    </txt:textMD>
</objectCharacteristicsExtension>
</objectCharacteristics >
```

In addition, the PREMIS 2.2 XML schema introduced a new element borrowed from the METS XML schema; the <mdSec> element may be used in place of any of the above extension elements.[6] This element has several advantages over the original extension elements. First it has two child elements, <mdRef> and <mdWrap>, which allow the non-PREMIS content to either be referenced as an external resource or wrapped within the PREMIS XML file. In addition, the <mdWrap> has two children, <binData> and <xmlData>, which allow either binary

[5]PREMIS version 3.0 also adds: *environmentDesignationExtension* and eventDetailExtension.

[6]In version 3.0, the <mdSec> element is discontinued in the XML schema and is now a stand-alone schema that may be referenced as another extension: http://www.loc.gov/standards/premis/v3/mdSec.xsd.

data or XML data to be wrapped. Finally, these elements support many attributes that allow a richer description of the external content that is being referenced or wrapped, such as MDTYPE for the type of metadata, MIMETYPE for the Internet media type, or CREATED for the date the external resource was created, among many others. Following is the same example as shown above, but using <mdSec>:

```
<objectCharacteristics>
<compositionLevel>0</compositionLevel>
<format>
   <formatDesignation>
      <formatName>text/plain</formatName>
   </formatDesignation >
</format>
<mdSec ID="TEXTMD_12">
     <mdWrap MDTYPE="TEXTMD" CREATED="2015-12-14"
      MIMETYPE="application/ xml" >
       <xmlData>
          <txt:textMD>
            …
          </txt:textMD>
       </xmlData>
   </mdWrap>
</mdSec>
</objectCharacteristics>
```

In addition to the <mdRef>, there are also several other PREMIS elements that can be used to link to external resources. Each PREMIS entity <[entity]Identifier> element allows Simple XLink attributes. These XLinks could point to an external representation of the entity, as shown in Fig. 13.8. In this example, the HTML Wikipedia article about the Mona Lisa has been captured and stored in the archive as OBJ_123; the xlink points back to the original Wikipedia URL.

The entity <[entity]Identifier> elements are also often used to point to external resources, for example if the <[entity]IdentifierType> is a URL or DOI, the <[entity]IdentifierValue> can serve double duty as both an identifier and as a pointer to external content. This may be used with the object and agent entities that allow for the use of global identifiers and may have multiple identifiers with some of the identifiers doubling as pointers to external resources.

Finally, there are a few other elements used for pointers to external content. The <originalName> element can point to the original external location of an object. The location of the object as stored in the controlling repository may be indicated by the <contentLocation>. A *Representation* or surrogate of the *Intellectual Entity* whose representation is being preserved may also be identified or located using the <linkingIntellectualEntityIdentifier> element.[7] Finally, external format registries

[7]In PREMIS 3 the Intellectual Entity is a category of Object with generally the same semantic units as Representation. Therefore, <linkingIntellectualEntityIdentifier> is not available in PREMIS 3, instead Intellection Entity is treated as another type of Object which may be fully described and linked to related *Files*, *Bitstreams*, or *Representations* using the usual mechanisms.

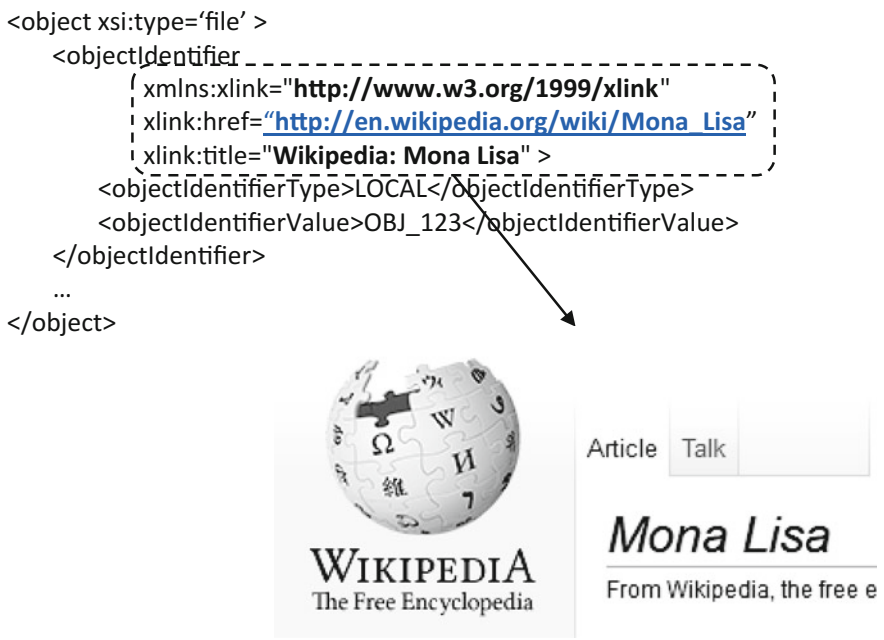


Fig. 13.8 Using *objectIdentifier* XLink to point to an external resource

may be identified or located using the <formatRegistry> element, and registry entries using the <formatRegistryKey> element. All of these elements support either textual content for the link or Simple XLink attributes.

13.3.3 Storing the XML

The XML can be created as either one large XML document with the root <premis> wrapper element, as multiple smaller XML documents for each entity, or using an intermediate solution in which several XML documents are created, with each document containing all metadata that corresponds to a logical unit, such as an OAIS [18] Archival Information Package (AIP).

There are several options for storage. The most obvious is simply storing the XML documents in a file system using some sort of standardized folder and file naming conventions. The PREMIS documents are often stored alongside the actual digital objects that they represent. This is often combined with some sort of database or metadata management system that keeps track of key metadata elements for search, retrieval, and reporting, along with identifiers, and pointers to the entity locations in the file system. This approach is taken by many bespoke preservation systems which utilize PREMIS. However, there are also turnkey systems, such as the Fedora Commons [22] Repository software, that can be used to implement systems similar to the above.

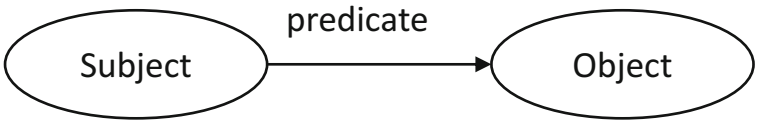


Fig. 13.9 RDF data structure

There are also databases that will support the direct storage of XML, which include native XML databases as well as mainstream relational databases, such as Microsoft SQL Server, Oracle, and others [23, 24]. With the native XML databases, you can simply insert the PREMIS XML directly into the database, and then use standard query languages, such as XQuery [25] to search and retrieve data. In the standard Relational databases, XML is usually supported as an extension to the standard SQL functionality. For example, a table column can be defined as a special XML data type. XML can then be inserted into a cell of this type as a blob of text, and then queried using extensions to the standard SQL query language, often XQuery, XPath, or some subset. Additional details about PREMIS and databases will be presented in the following parts of this chapter.

13.4 RDF Implementations

RDF, the Resource Description Framework [26], and the RDF Schema (RDFS) [27] extension are among the core standards for the W3C's semantic web and Linked data technologies. RDF provides the framework for a simple data structure consisting of triples as illustrated in Fig. 13.9.

Triples are used to represent a single fact. The subject and predicate are always URIs, but the object can be either a URI or a literal value, optionally with a data type or language. RDF and RDFS also provide a number of core properties and classes, such as rdf:type, rdfs:Class, rdfs:subClassOf, and others, which are used to describe basic ontologies. For example, the rdf:type property specifies the class or type of a resource. RDF triples are typically combined in a triple store which can represent large and complex data models. Below is a minimal PREMIS *File* Object modeled as RDF and serialized as Turtle [28]; namespace prefixes are omitted, indentation is not significant and is used only for clarity, and qualified names (QNames) are used in place of full URI references[8]:

[8]For example, the QName, premis: File, is shorthand for the full URI, <https://www.loc.gov/premis/rdf/v1#File>.

```
    _:File  a  premis:File  .
_:File  premis:hasIdentifier  _:FileID  .
        _:FileID  a  premis:Identifier  .
        _:FileID  premis:hasIdentifierType  "DOI"  .
        _:FileID  premis:hasIdentifierValue  "10.12345/PID12"  .
_:File  premis:hasObjectCharacteristics  _:ObjChars  .
        _:ObjChars  premis:hasCompositionLevel  "0"  .
        _:ObjChars  premis:hasFormat  _:ObjFormat  .
                _:ObjFormat  premis:hasFormatDesignation  _:ObjFormatDes  .
                        _:ObjFormatDes  premis:hasFormatName  "image/tiff"  .
```

The values above which are prefixed by _: are blank nodes which are used to denote a resource without explicitly assigning it a URI, which is often convenient for intermediate nodes in a complex graph. For example, the *objectCharacteristics* entity, identified by the_:ObjChars blank node, can have multiple properties such as *compositionLevel* and *format* without the *objectCharacteristics* entity itself being explicitly identified. Also, the single word 'a' is shorthand notation for the predicate <rdf:type>. For comparison purposes the above PREMIS RDF is semantically equivalent to the following PREMIS XML:

```
<object xsi:type="file">
  <objectIdentifier>
    <objectIdentifierType>DOI</objectIdentifierType>
    <objectIdentifierValue>10.12345/PID12</objectIdentifierValue>
  </objectIdentifier>
  <objectCharacteristics>
    <compositionLevel>0</compositionLevel>
    <format>
      <formatDesignation>
        <formatName>image/tiff</formatName>
      </formatDesignation>
    </format>
  </objectCharacteristics>
</object>
```

In addition to RDF and RDFS there is also the Web Ontology Language (OWL) [29] which adds properties and classes which afford the creation of more complex ontologies beyond what is possible with just RDF and RDFS. For example, with OWL you can explicitly declare that two resources are different or the same as each other:

```
<http://dx.doi.org/10.12345/PID12> owl:differentFrom
<http://example.org/PID12> .
<http://dx.doi.org/10.12345/PID34> owl:sameAs
<http://example.org/PID34> .
```

This would be impossible with just RDF and RDFS alone, but it is extremely useful in the common situation where different organizations are coining new

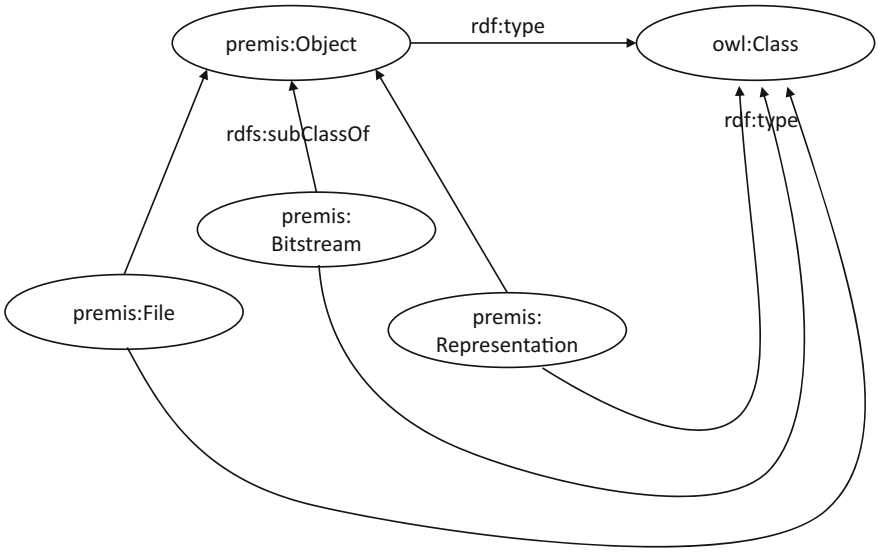


Fig. 13.10 Example OWL representation of core PREMIS entities and object categories

identifiers for the same thing. Using owl:sameAs it becomes almost trivial to merge disparate triple stores that would otherwise be disjoint.

A draft RDF OWL ontology has been available for PREMIS since late 2011 [30] and was superseded by a stable ontology which is now officially part of the loc.gov namespace: http://www.loc.gov/premis/rdf/v1#. This part is based on this version of the PREMIS ontology. An HTML browsable version of the ontology is here: http://id.loc.gov/ontologies/premis.html [31]. The RDF/XML can be downloaded from here: http://id.loc.gov/ontologies/premis.rdf. The remainder of this part will explore several specific aspects of the PREMIS OWL ontology.[9]

13.4.1 PREMIS OWL Basics

The core PREMIS entities have all been defined as OWL classes, and where appropriate these classes have subclasses as illustrated in Fig. 13.10.

The links between the entities have all been represented as OWL object properties along with the appropriate domain and range properties (Fig. 13.11).

[9]A PREMIS Ontology working group is in the process of revising the current PREMIS OWL ontology using version 3.0. As part of this effort, significant modeling changes will be introduced that are more consistent with current Linked data best practices and with less reliance on a transformation from the hierarchical XML.

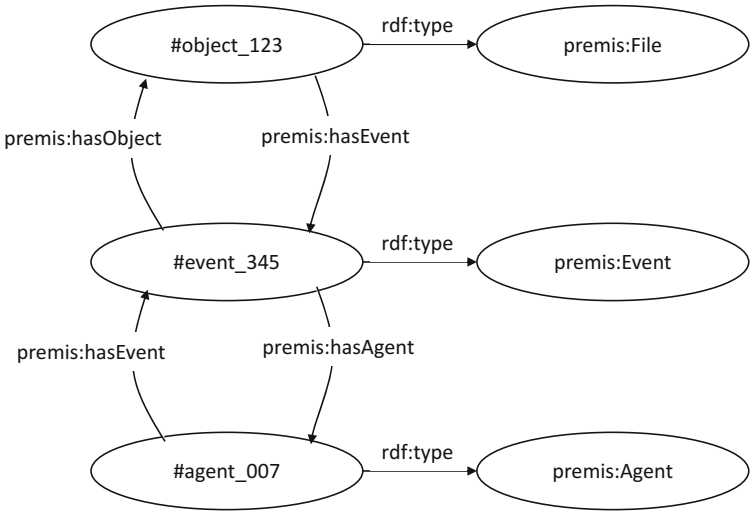


Fig. 13.11 Example OWL representation of core PREMIS links

For example, the PREMIS OWL ontology contains statements similar to the following which specify that hasAgent and hasEvent are OWL object properties and which types of resources can be the subjects (domain) or objects (range) of triple statements using the given property as a predicate.

```
premis:hasAgent  a  owl:ObjectProperty  ;
                    rdfs:range  premis:Agent  ;
                    rdfs:domain  premis:Event  ;
                    rdfs:domain  premis:RightsStatement  .
premis:hasEvent  a  owl:ObjectProperty  ;
                    rdfs:range  premis:Event  ;
                    rdfs:domain  premis:Object  ;
                    rdfs:domain  premis:Agent  .
```

All the elements of the PREMIS Data Dictionary are defined and constrained in a similar fashion. For example, there is an *ObjectCharacteristics* class and a *Format* class with a *hasFormat* property which can be used to link the two. In addition, the *hasFormat* property is constrained by an OWL minCardinality of 1, meaning that there has to be at least one format.

Note that a common convention in RDF is to capitalize the names of classes or types and to start the names of properties or predicates as lowercase and express them as verbs. This can help when interpreting triple statements.

13.4.2 Identifiers

RDF provides an alternative mechanism for dealing with identifiers. This is because in RDF a URI is the object. Instead of qualifying an object with an identifier as shown in the first example in this part:

```
    _:File  a  premis:File  .
_:File  premis:hasIdentifier  _:FileID  .
        _:FileID  a  premis:Identifier  .
        _:FileID  premis:hasIdentifierType  "DOI"  .
        _:FileID  premis:hasIdentifierValue  "10.12345/PID12"  .
```

The URI identifier can be used directly as the subject of a triple. For example, this single triple is equivalent to the above

```
<http://dx.doi.org/10.12345/PID12> a premis:File .
```

Similarly, the URI identifier can be used as the object of a triple, as shown below with an event linking to its associated object:

```
_:EVENT5  premis:hasObject  <http://dx.doi.org/10.12345/PID12> .
```

Another example of this kind of linking directly to an object through its URI identifier is format registries. Instead of the more complex premis: FormatDesignation, as described in the Data Dictionary and shown below:

```
_:ObjChars  premis:hasFormat  _:ObjFormat  .
    _:ObjFormat  premis:hasFormatDesignation  _:ObjFormatDes  .
            _:ObjFormatDes  premis:hasFormatName  "image/tiff"  .
```

You can link directly to a format registry

```
_:ObjChars  premis:hasFormat  <http://UDFR.org/UDFR/u1f328>  .
```

In addition, you can define your own kinds of identifiers by creating subproperties of premis:hasIdentifier, for example:

```
<http://some.edu/ontology#hasGUID> rdfs:subPropertyOf
premis:hasIdentifier .
```

Now when you use the hasGUID predicate it is understood that you are indicating that an object has a specific kind of identifier

```
_:File  <http://some.edu/ontology#hasGUID>
      "21EC2020-3AEA-4069-A2DD-08002B30309D"  .
```

13.4.3 Controlled Vocabularies

Controlled vocabularies can also be used with the PREMIS RDF OWL ontology (as well as with the PREMIS XML schema). Most of the common preservation controlled vocabularies are available as RDF ontologies from the Library of Congress' Preservation Schemes list, http://id.loc.gov/vocabulary/preservation.html. These are typically described using some combination of the Metadata Authority Description Schema (MADS) [32] and its corresponding RDF OWL ontology [33] and the Simple Knowledge Organization System (SKOS) [34]. These ontologies describe a concept's relationships to other concepts, such as that it is a broader or narrower concept, its common human-readable labels, and its revision history. The controlled vocabulary values have canonical URIs which can be used as the object in RDF triples. In the examples below, the namespace prefix 'loc_pres' stands for 'http://id.loc.gov/vocabulary/preservation/'.

An example of a preservation controlled vocabulary is the list of PREMIS Event types, http://id.loc.gov/vocabulary/preservation/eventType.html.

_:EVENT5 premis:hasEventType loc_pres:*eventType*/cap .

From the MADS ontology you can see that this URI represents the 'capture' event:

loc_pres:*eventType*/cap skos:prefLabel "capture" .

It is also possible to extend these vocabularies to include your own local values if needed. This can be accomplished by adding your local controlled vocabulary term to the appropriate SKOS scheme

<http://some.edu/vocab#copyrightReview> skos:inScheme
loc_pres:*eventType* .

You can now use copyrightReview as the object of a hasEventType triple statement

_:EVENT5 premis:hasEventType
<http://some.edu/vocab#copyrightReview> .

In the XML schema, use of URIs for controlled values is available with the authority attributes that were introduced in version 2.3. This includes authorityURI to identify the vocabulary scheme and valueURI for the identifier of a controlled value from that scheme.

13.4.4 Extensions

Extensibility is one of the strengths of RDF. In the above examples, you have already seen how to extend the ontology itself through mechanisms such as owl: sameAs, rdfs:subPropertyOf, and skos:inScheme. You can also easily intermix subjects, predicates, and objects from different namespaces. Below is an example of using the Friend of a Friend (FOAF) ontology to enhance a PREMIS agent entity:

```
<mailto:thabing@illinois.edu> a premis:Agent ;
    a  foaf:Person  ;
    foaf:name  "Tom Habing"  ;
    premis:agentNote  "Tom is director of IT at UIUC"  ;
    foaf:page  <https://www.linkedin.com/in/thabing>  .
```

Note that the <premis:agentExtension> element does not translate in the ontology, as RDF is already natively extensible. This applies to all PREMIS extension elements.

13.4.5 Linking Roles

There are several linking role elements in PREMIS, such as *linkingAgentRole* and *linkingObjectRole*. These elements are not properties of an entity per se, but instead are properties of a relationship between two entities. In RDF these roles which are associated with links between entities can be modeled as subproperties of the linking properties.

The PREMIS ontology includes generic properties hasAgent, hasEvent, hasObject, and hasRightsStatement for linking between the PREMIS entities.

```
_:EVENT99  premis:hasAgent  _:AGENT007  .
```

There are also specific subproperties whose domains are restricted to the exact class of entity which is being linked to. For example, premis:hasEventRelatedAgent is used to establish links between an agent and an event:

```
premis:hasEventRelatedAgent
    rdfs:  subPropertyOf  premis:hasAgent  ;
    rdfs:domain  premis:Event  .

_:EVENT99  premis:hasEventRelatedAgent  _:AGENT007  .
```

In addition, these properties can also have subproperties which identify the specific role of the linkage

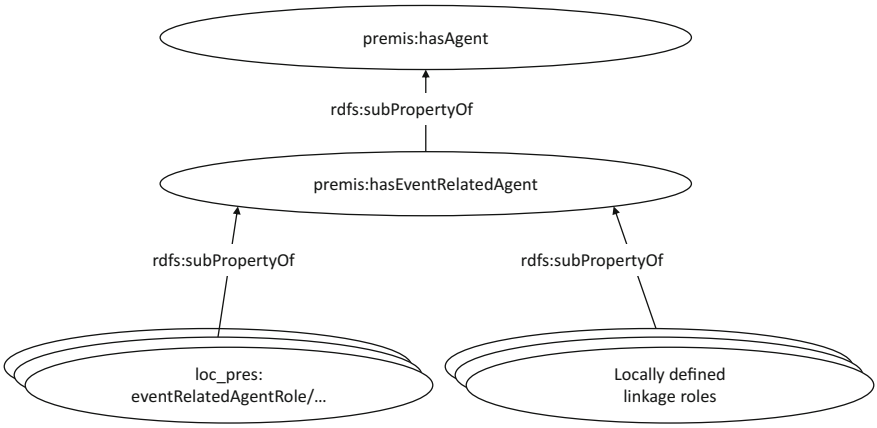


Fig. 13.12 Property hierarchy for hasAgent links with roles

```
loc_pres:eventRelatedAgentRole/aut
    rdfs:  subPropertyOf  premis:hasEventRelatedAgent  .

_:EVENT99  loc_pres:eventRelatedAgentRole/aut  _:AGENT007  .
```

The above establishes a link between the event and the agent with a role of authorizer (aut) (Fig. 13.12).

The PREMIS Editorial Committee in conjunction with the Library of Congress, as the managing agency for PREMIS has published controlled vocabularies in the Linked Data Service for Authorities and Vocabularies (id.loc.gov) for common roles based on examples in the PREMIS Data Dictionary

- http://id.loc.gov/vocabulary/preservation/eventRelatedAgentRole.html
- http://id.loc.gov/vocabulary/preservation/eventRelatedObjectRole.html
- http://id.loc.gov/vocabulary/preservation/rightsRelatedAgentRole.html

You can also easily create new roles by subclassing the appropriate domain-specific linking property. For example, this creates a custom role of rights reviewer:

```
<http://some.edu/ontology#rightsReview>  rdfs:subPropertyOf
    loc_pres:hasRightsRelatedAgent  .

_:RIGHTS27  <http://some.edu/ontology#rightsReview>  _:AGENT007
```

13.4.6 Relationship Types and Sequence

Similar to linking roles, the type and subtype of object relationships are properties of the relationship and not properties of the entity. This is handled the same as it was for linking roles, by creating subproperties of the premis:hasRelationship

property. In the XML schema both the <relationshipType> and the <relationshipSubType> are mandatory elements. However, expressing these in RDF is somewhat simpler because only the relationship subtype is required. This is because the parent relationship type for all the subtypes is defined in the PREMIS ontology and can be derived.

The Linked Data Service for Authorities and Vocabularies also provides ontologies for common relationship types and subtypes.[10]

- http://id.loc.gov/vocabulary/preservation/relationshipType.html
- http://id.loc.gov/vocabulary/preservation/relationshipSubType.html

Local relationship types can be also defined similarly to how local link roles were defined.

Another factor that complicates object relationships is that they have an optional *relatedObjectSequence* property to indicate the order of the related objects. This is dealt with using the RDF technique of introducing an intermediate class which acts as the subject for additional properties, such as sequence. The PREMIS ontology uses the premis:RelatedObjectIdentification class for this. This class supports two properties, premis:hasRelatedObjectSequence and premis:hasRelatedObject.[11] The following example shows an object with a relationship to two parts (hsp = has Part):

```
_:OBJ1  loc_pres:relationshipSubType/hsp  _:OBJ_A  .
_:OBJ1  loc_pres:relationshipSubType/hsp  _:OBJ_B  .
```

The next example shows how to represent the same relationships, except with a defined sequence.

```
_:OBJ1  loc_pres:relationshipSubType/hsp  _:RELATED_OBJ_A  .
    _:RELATED_OBJ_A
            a  premis:RelatedObjectIdentification  ;
            premis:hasRelatedObjectSequence  "1"  ;
            premis:hasRelatedObject  _:OBJ_A  .
_:OBJ1  loc_pres:relationshipSubType/hsp  _:RELATED_OBJ_B  .
    _:RELATED_OBJ_B
            a  premis:RelatedObjectIdentification  ;
            premis:hasRelatedObjectSequence  "2"  ;
            premis:hasRelatedObject  _:OBJ_B  .
```

[10]As of this writing these ontologies are being revised to include additional relationships from version 3 and to explicitly assert subproperty relationships between relationshipType and relationshipSubType ontologies.

[11]The working group revising the PREMIS ontology is considering other practices to assert sequencing.

13.5 Relational Databases

Relational databases (RDBs) [3] have been mentioned earlier in this chapter; this part will summarize some of the key points related to implementing PREMIS using a Relational database.

It is possible to design a Relational database which provides a complete implementation of the PREMIS Data Dictionary even including a reversible mapping to and from PREMIS XML. However, given the nested and repeatable nature of many PREMIS entities, a robust design would necessitate a rather complex database schema. If it were to represent the PREMIS container semantic units and subunits explicitly as hierarchies, it would contain a large number of tables joined by relationship keys with a deep nesting structure. In some cases this would require rather complex join operations to retrieve specific properties, not to mention the complex business logic to update information while maintaining the relational integrity of the database. That being said, databases with this level of complexity are not uncommon, and there may be good reasons to develop a complete PREMIS implementation using a Relational database, such as the need to support a rich preservation environment or flexibility to support potential future requirements.

Probably a more typical Relational database approach would be to only implement those aspects of the PREMIS Data Dictionary which are actually applicable to a specific preservation system. For example, if a system did not need to track rights, those 15 or so tables could be excluded from the database design. Similarly, if a local preservation system only required very limited technical metadata, the *objectCharacteristics* table and its child tables could be greatly simplified. Also to be considered is implementing a flatter structure than the full hierarchy of container units and subunits represented in the PREMIS Data Dictionary. Implementers would need to weigh the simplicity of the immediate solution against possible future expansion. Finally, it is also possible to combine Relational databases with other information retrieval technologies, for example a simplified Relational database may be used for data management purposes, but a full-text indexing system, such as SolR/Lucene [13] could be used for end user search and discovery, with both systems using common object identifiers, but having very different underlying schema to best support their intended functionality, possibly only loosely based on the PREMIS schema.

Most of the advantages of using a Relational database approach stem from the properties of Relational databases. These include that RDBs are a very stable standardized technology with many vendor and open source solutions [35]. RDBs offer a standardized and flexible query and update language called SQL [24]. Nearly all programming languages have robust libraries which support SQL-based access to common Relational database systems. With properly tuned indexes, retrieval performance can be very good. RDBs can enforce certain kinds of business logic, such as mandatory fields or ensuring that a field value is restricted to a value taken from an enumerated list. In addition, most RDBs support transactions, meaning that multiple inserts, updates, or deletions across the database can be treated as a single

atomic unit; if one of the operations fails, the entire transaction fails and the database is returned to its initial state as if nothing had happened. This can be vitally important in a complex system such as a PREMIS-based preservation system where a single failed update could leave a preserved object in an indeterminate state; it can be difficult to guarantee this level of consistency outside of a relational database.

A common question related to relational database solutions is how to deal with the actual files which are being preserved. Most relational databases support a data type called a blob (Binary Large Object). This makes it possible to actually store the files in the database itself, usually up to some file size limit such as 2 GB. Although this affords the benefits of a database, such as transaction management, to file management, generally, this is not a good approach. The primary reason not to implement this way is that it will cause the size of the database to grow very large, which could affect performance, backups, and the ability to easily manage the database. It could also make it more difficult to manage the files themselves, such as performing file migrations or providing access to them via a web server. A more common approach is to provide a link to the files from the database. PREMIS actually provides the *storage contentLocation* element for just this purpose. The *contentLocation* could be the path to the file on a local or attached storage system, or it could be the URL to the file in a cloud storage system. Also, some database systems have a special datatype which combines the best of both worlds. It acts like a blob, but the data are stored on the file system and the database only contains a pointer; an example is the Microsoft SQL Server FILESTREAM [36]. When evaluating relational databases for a PREMIS implementation these sorts of features should be considered.

References

1. Bray T, Paoli J, Sperberg-McQueen CM, Maler E, Yergeau F (2008) Extensible markup language (XML) 1.0, 5th edn. W3C. http://www.w3.org/TR/2008/REC-xml-20081126/. Accessed 10 Mar 2014
2. Heath T, Bizer C (2011) Linked data: evolving the web into a global data space. Morgan & Claypool, San Rafael. doi:10.2200/S00334ED1V01Y201102WBE001
3. Halpin T, Morgan T (2008) Information modeling and relational databases, 2nd edn. Morgan Kaufmann Publishers Inc, San Francisco
4. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 29 Dec 2015
5. Peintner J, Kyusakov R (2014) Efficient XML interchange (EXI) format 1.0, 2nd edn. W3C. http://www.w3.org/TR/2014/REC-exi-20140211/. Accessed 10 Mar 2014
6. ECMA International (2013) The JSON data interchange format. ECMA International. http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf. Accessed 10 Mar 2014
7. Ben-Kiki O, Evans C, Net I (2009) YAML ain't markup language (YAML™) version 1.2. YAML. http://www.yaml.org/spec/1.2/spec.html. Accessed 10 Mar 2014
8. Sklar D (nd) What is BadgerFish? http://www.sklar.com/badgerfish/. Accessed 3 Apr 2014

9. TIBCO Software Inc (2014) Badgerfish conversion rule. TIBCO Product Documentation. https://docs.tibco.com/pub/activematrix-businessworks-plugin-for-rest-and-json/1.0.0-november-2012/doc/html/tib_bwpluginrestjson_users_guide/wwhelp/wwhimpl/common/html/wwhelp.htm#href=bwpluginrestjson_users_guide.5.06.htm&single=true. Accessed 31 Mar 2014
10. Goessner S (2006) Converting between XML and JSON. XML.com. http://www.xml.com/lpt/a/1658. Accessed 7 Dec 2015
11. JSON (nd) http://www.json.org/. Accessed 18 Feb 2016
12. Newtonsoft (nd) Converting between JSON and XML. Json.NET Documentation. http://www.newtonsoft.com/json/help/html/convertingjsonandxml.htm. Accessed 18 Feb 2016
13. Apache Software Foundation (2014) Apache Solr. http://lucene.apache.org/solr/. Accessed 7 Dec 2015
14. W3C (2013) SPARQL 1.1 overview. World Wide Web Consortium. The W3C SPARQL working group (ed). http://www.w3.org/TR/sparql11-overview/. Accessed 9 Dec 2015
15. W3C (2012) A direct mapping of relational data to RDF. World Wide Web Consortium. Edited by Marcelo Arenas, Alexandre Bertails, Eric Prud'hommeaux and Juan Sequeda. http://www.w3.org/TR/rdb-direct-mapping/. Accessed 9 Dec 2015
16. SALUS (2013) Ontmalizer: transformation of XML schemas (XSD) and XML data to RDF/OWL. SALUS Blog. http://www.srdc.com.tr/projects/salus/blog/?p=189. Accessed 9 Dec 2015
17. W3C (2013) Extensible markup language (XML). http://www.w3.org/XML/. Accessed 16 Mar 2014
18. Consultative Committee for Space Data System (CCSDS) (2012) Reference model for an Open Archival Information System (OAIS). In: Recommended Practice, CCSDS Secretariat, Washington, DC
19. W3C (2010) XML linking language (XLink) version 1.1. W3C. DeRose S, Maler E, Orchard D, Walsh N (eds). http://www.w3.org/TR/xlink11/. Accessed 31 Mar 2014
20. Library of Congress (2008) Guidelines for using PREMIS with METS for exchange. PREMIS: Preservation Metadata Maintenance Activity (Library of Congress). http://www.loc.gov/standards/premis/guidelines-premismets.pdf. Accessed 9 Dec 2015
21. Library of Congress (nd) Technical metadata for text (textMD). http://www.loc.gov/standards/textMD/. Accessed 18 Feb 2016
22. Duraspace (nd) About Fedora. Fedora Repository. http://fedora-commons.org/about. Accessed 18 Feb 2016
23. Bourret R (2010) XML database products. XML and Databases. http://www.rpbourret.com/xml/XMLDatabaseProds.htm. Accessed 3 Apr 2014
24. International Organization for Standardization (2011) Information technology—database languages—SQL. International Organization for Standardization, Standard, Geneva, Switzerland
25. W3C (2010) XQuery 1.0: an XML query language, 2nd edn. W3C. Boag S, Chamberlin D, Fernández MF, Florescu D, Robie J, Siméon J (eds). http://www.w3.org/TR/xquery/. Accessed 3 Apr 2014
26. W3C (2014) RDF 1.1 primer. W3C. Schreiber G, Raimond T (eds). http://www.w3.org/TR/rdf11-primer/. Accessed 16 Apr 2014
27. W3C (2014) RDF schema 1.1. W3C. https://www.w3.org/TR/rdf-schema/. Accessed 18 Feb 2016
28. Becket D, Berners-Lee T, Prud'hommeaux E, Carothers G (2014) RDF 1.1 turtle. W3C. http://www.w3.org/TR/Turtle/. Accessed 16 Apr 2014
29. W3C (2012) OWL 2 web ontology language primer, 2nd edn. W3C. Hitzler P, Krötzsch M, Parsia B, Patel-Schneider PF, Rudolph S. (eds) http://www.w3.org/TR/owl2-primer/. Accessed 16 Apr 2014
30. PREMIS Ontology Working Group (2011) PREMIS OWL ontology now available. Standards at the Library of Congress. http://www.loc.gov/standards/premis/owlOntology-announcement.html. Accessed 11 Apr 2014

31. PREMIS Editorial Committee (2013) PREMIS OWL ontology 2.2 now available. Standards at the Library of Congress. http://www.loc.gov/standards/premis/ontology-announcement.html. Accessed 11 Apr 2014
32. Library of Congress (2013) Metadata authority description schema (MADS). http://www.loc.gov/standards/mads/. Accessed 27 Apr 2014
33. Library of Congress (2012) MADS/RDF primer. Metadata Authority Description Schema (MADS). http://www.loc.gov/standards/mads/rdf/. Accessed 27 Apr 2014
34. W3C (2012) SKOS simple knowledge organization system. http://www.w3.org/2004/02/skos/. Accessed 27 Apr 2014
35. Wikimedia Foundation, Inc (2015) Comparison of relational database management systems. Wikipedia, the free encyclopaedia. https://en.wikipedia.org/wiki/Comparison_of_relational_database_management_systems. Accessed 16 Dec 2015
36. Microsoft (2015) Designing and implementing FILESTREAM storage. TechNet. https://technet.microsoft.com/en-us/library/bb895234%28v=sql.105%29.aspx?f=255&MSPPError=-2147217396. Accessed 16 Dec 2015

Chapter 14
Digital Preservation Metadata in a Metadata Ecosystem

Eld Zierau and Sébastien Peyrard

14.1 Introduction

In most contexts, PREMIS [1] will not be the only metadata standard implemented in an institution. Different standards aim at covering complementary functionality and can be combined in a modular fashion. This section intends to show how PREMIS metadata can be articulated with other metadata formats or container formats.

From the beginning PREMIS has been designed with a very clear purpose, summarized by a set of principles:

- It is geared toward *digital preservation*
- It is meant to capture *core* preservation metadata
- It is *technically neutral*
- It is *domain-agnostic*.

PREMIS was developed at the same time as other metadata formats specific to digital documents, among which are:

- METS [2], an XML schema, which provides a container for providing metadata together with digital content
- MIX [3] and TextMD [4], which, respectively, define a set of technical metadata for still images and text files
- MODS [5] and EAD [6], which define a set of largely descriptive metadata elements to describe, respectively, library content and archive records.

E. Zierau (✉)
The Royal Library of Denmark, P.O. Box 2149, 1016 Copenhagen K, Denmark
e-mail: elzi@kb.dk

S. Peyrard
Bibliothèque nationale de France, Site François-Mitterrand, Quai François-Mauriac, 75706 Paris Cedex 13, France
e-mail: sebastien.peyrard@bnf.fr

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_14

This means that most of those schemas have some overlaps with PREMIS.

What is more, PREMIS has been widely adopted over the last decade and has become in many cases a de facto standard for describing digital documents. PREMIS may even sometimes be used outside its original preservation scope in the case of rights metadata, nonpreservation events and with the ability in version 3.0 to provide PREMIS-defined metadata for *Intellectual Entities* [1].

Therefore, though the initial set of rules made it very clear what PREMIS was and what it was not, from the implementation standpoint, there is no straight answer on how those different schemas should be combined in a particular implementation. You will most likely use other metadata schemas in conjunction with PREMIS because:

- They achieve features that PREMIS does not; PREMIS does not provide all information needed for functionality in a digital repository.
- They are already used in your organization and up-and-running workflows rely on them to function.
- They fill in the gaps where PREMIS considers the information out of scope.

This chapter intends to provide guidelines and help whenever an implementer wants to articulate PREMIS with other metadata schemas. It also provides a high-level view of the metadata formats with which PREMIS is commonly used.

This chapter is focused on metadata schemas in the form of XML schemas. It is also possible to use RDF [7] to express metadata according to Linked data [8] principles by mixing and matching vocabularies from different sources. The PREMIS ontology [9] and the id.loc.gov preservation vocabularies [10] are core assets for doing this for preservation metadata. Hints about how to express PREMIS metadata in RDF with the PREMIS ontology are provided in Chap. 13 of this book.

14.2 Extending PREMIS with File Format-Specific Technical Metadata

14.2.1 PREMIS-Independent Standalone Schemas

There are a number of XML schemas, which are designed to provide file format-specific technical information that have been designed independently of PREMIS. Examples of PREMIS-independent standalone technical information schemas are:

- Metadata for images, e.g., in the XML-based image metadata format Images in XML Schema (MIX) [3], based on the ANSI/NISO Data Dictionary—Technical Metadata for Digital Still Images [11]
- Metadata for texts, e.g., in the XML-based text metadata format Technical Metadata for Text TextMD [4]
- Metadata for sounds and moving images, e.g., in the XML-based metadata format *Public Broadcasting Metadata Dictionary Project* (PBCore) [12]

This list is a sample and is in no way exhaustive, and there will probably be a continuous number of formats appearing as different use cases are developed and metadata needs articulated. Common for all these XML-based technical information schemas is that they can be specified either within PREMIS or outside PREMIS.

An example where technical metadata are specified outside PREMIS is when such technical metadata is placed in the technical metadata part of a METS [2] <techMD> section independently of PREMIS metadata. The METS <techMD> allows for extensibility within METS for technical metadata specified by other schemas. This is, for example, the case in the following extract of METS xml (Fig. 14.1).

An example where such technical metadata are specified within a PREMIS description is the use of the <objectCharacteristicsExtension>. This element allows for extensibility of PREMIS by embedding technical metadata from other schemas using a similar mechanism to the METS techMD container. This is, for example, the case for the following portion of PREMIS XML (Fig. 14.2).

The PREMIS objectCharacteristicExtension allows keeping all technical preservation-related metadata together. This includes technical metadata that does not apply to all or most file format types and is specific for a particular type of format. It also decreases the need for expressing element values redundantly. In all cases, it is the implementers who must decide whether to include those elements that exist in both schemas embedded in PREMIS, or directly in METS, or both. For other format-specific metadata schemas similar issues concerning redundancy with defined PREMIS elements arise. To help the implementer, the PREMIS Data Dictionary specifies principles for using extensions with PREMIS-defined metadata in the Introduction under Extensibility [1: pp. 27–29].

14.2.2 Technical Metadata Formats Designed to Be Combined with PREMIS

Like MIX for images or TextMD for text, documentMD [13] and containerMD [14] are technical schemas that provide information specific to certain format types. documentMD has been designed to record technical features specific to document-oriented files (office documents, PDF files [15] or e-books), while the target of containerMD is technical metadata for files that aggregate other files into a container (e.g., ZIP [16], TAR [17], ARC [18] or WARC [19] files).

The principle is the same as before: extending PREMIS with a set of technical metadata specific to a particular file type. The main difference is that these schemas have taken PREMIS into account from their inception, and therefore, explicitly address embedding technical metadata within the PREMIS *objectCharacteristics Extension* semantic container.

```
<?xml version="1.0" encoding="UTF -8"?>
<mets:mets xmlns:mets="http://www.loc.gov/METS/" ...>
 <mets:metsHdr CREATEDATE="2013 -01-18T19:28:01.025+01:00">
  ...
 </mets:metsHdr>
 <mets:dmdSec CREATED="2013 -01-18T19:28:01.035+01:00" ID="Mods1">
  ...
 </mets:dmdSec>
 <mets:amdSec>
  <mets:techMD CREATED="2013-01-18T19:28:01.426+01:00" ID="PrOb1">
   <mets:mdWrap MDTYPE="MIX">
    <mets:xmlData>
          <mix:mix xsi:schemaLocation="http://www.loc.gov/mix/v20 ... ">
           <mix:BasicDigitalObjectInformation>
            ...
           <mix:fileSize>137362594</mix:fileSize>
            <mix:FormatDesignation>
             <mix:formatName>TIFF Image</mix:formatName>
             <mix:formatVersion>6.0</mix:formatVersion>
            </mix:FormatDesignation>
            <mix:Fixity>
             <mix:messageDigestAlgorithm>MD5</mix:messageDigestAlgorithm>
             <mix:messageDigest>3f349a40b0c47bb070ea6bdd2759a731
             </mix:messageDigest>
            </mix:Fixity>
           </mix:BasicDigitalObjectInformation>
           <mix:BasicImageInformation>
            <mix:BasicImageCharacteristics>
             <mix:imageWidth>5894</mix:imageWidth>
             <mix:imageHeight>7768</mix:imageHeight>
             ...
            </mix:BasicImageCharacteristics>
           </mix:BasicImageInformation>
            ...
          </mix:mix>
    </mets:xmlData>
   </mets:mdWrap>
  </mets:techMD>
  <mets:rightsMD CREATED="2013 -01-18T19:28:01.455+01:00" ID="MoRi1">
  ...
  </mets:rightsMD>
  <mets:digiprovMD CREATED="2013 -01-18T19:28:01.456+01:00" ID="Premis1">
   <mets:mdWrap MDTYPE="PREMIS">
    <mets:xmlData>
     <premis:premis xmlns:xlink="http://www.w3.org/1999/xlink" … >
      <premis:premis>
       <premis:object xsi:type="premis:file">
       …
       </premis:object>
       <premis:agent>
       …
       </premis:agent>
       <premis:rights>
       …
       </premis:rights>
       <premis:event>

        …
       </premis:event>
      </premis:premis>
     </mets:xmlData>
    </mets:mdWrap>
   </mets:digiprovMD>
  </mets:amdSec>
  <mets:fileSec>
  …
  </mets:fileSec>
  <mets:structMap TYPE="logical">
  …
  </mets:structMap>
 </mets:mets>
```

Fig. 14.1 Example of a METS document holding MIX technical metadata in techMD, separately from PREMIS metadata expressed in digiprovMD as provenance metadata

```
<premis xmlns="info:lc/xmlns/premis-v2" version="2.2">
  <object xsi:type="file">
    <objectIdentifier>
      <objectIdentifierType>UUID</objectIdentifierType>
      <objectIdentifierValue>41d153d0-0099-11e2-9397-005056887b67
      </objectIdentifierValue>
    </objectIdentifier>
    <objectCharacteristics>
      <objectCharacteristicsExtension>
        <mix:mix xsi:schemaLocation="http://www.loc.gov/mix/v20 ... ">
          <mix:BasicDigitalObjectInformation>
            ...
          <mix:fileSize>137362594</mix:fileSize>
           <mix:FormatDesignation>
            <mix:formatName>TIFF Image</mix:formatName>
            <mix:formatVersion>6.0</mix:formatVersion>
           </mix:FormatDesignation>
           <mix:Fixity>
            <mix:messageDigestAlgorithm>MD5</mix:messageDigestAlgorithm>
            <mix:messageDigest>3f349a40b0c47bb070ea6bdd2759a731
            </mix:messageDigest>
           </mix:Fixity>
          </mix:BasicDigitalObjectInformation>
          <mix:BasicImageInformation>
          ...
          </mix:BasicImageInformation>
          ...
         </mix:mix>
        </objectCharacteristicsExtension >
       </objectCharacteristics>
     </object>
   </premis>
```

Fig. 14.2 Example of a PREMIS Object extended with MIX image-specific metadata

If specific extension containers are available they should be used in preference to the more generic ones. For example, if one needs to record more granular information about signatures, one must use the dedicated *signatureInformationExtension* semantic container rather than the objectCharacteristicExtension. This allows extending PREMIS as needed, while keeping as close as possible to core semantics and therefore, interoperability.

The two schemas adopted a different strategy for achieving this goal.

14.2.2.1 documentMD: A Technical Metadata Schema for Office Documents and E-Books

documentMD [13] was designed as authoritative XML schema for digital preservation of office documents and books in electronic form (PDF files that are essentially textual or electronic book formats, e.g., ePUB [20] files).

XMP (eXtensible Metadata Platform) [21], maintained by Adobe, is another metadata format for office and PDF files. It encompasses descriptive, technical, and provenance metadata, but it misses font definitions, which are necessary for preservation.

documentMD, from the start, drew a clear distinction to PREMIS semantic units:

- any type of information that can be recorded in a PREMIS semantic unit is not expressed in documentMD
- any type of information that can or should not be recorded in a PREMIS semantic unit, or that requires additional granularity, is expressed in a documentMD field that is contained in the *objectCharacteristicsExtension* section of the corresponding *File*.

An XML example of documentMD elements embedded in a PREMIS extension container is shown in Fig. 14.3.

The strength of this choice is its understandability, and the clear separation between the respective PREMIS and documentMD definitions. This leads to a schema that is easy to use in conjunction with PREMIS, easy to maintain, prevents the aforementioned *overlap problems*, and improves understandability and interoperability between systems.

As can be seen in the example of Fig. 14.3, there is a number of document-specific information important for preservation, such as

- number of characters (with or without spaces), pages, paragraphs, and lines
- number of graphics and tables
- language
- fonts used: font name, embedded font or not
- some special features: layers, transparency, thumbnails, attachments, annotations, fixed layout (relevant for e-books).

General information not specific to a particular format, such as file size, file format, signature, or encodings, is handled by regular PREMIS semantic units.

documentMD examples that extend PREMIS are provided by the online FLV description service (http://description.fcla.edu); for any document file that you upload, it provides an XML PREMIS output with documentMD elements wrapped in their corresponding *objectCharacteristicsExtension* container.

14.2.2.2 A Technical Metadata Schema for Container Files: ContainerMD

The Bibliothèque nationale de France (BnF, or National Library of France) designed containerMD [14] to express the technical characteristics of their web archive's ARC container files.[1] In a preservation repository the purpose of containerMD is to support a technical file analysis of content files that are

[1]See Chap. 6 about web archives for further information.

```
<?xml version="1.0" encoding="UTF-8"?>
<premis xmlns="info:lc/xmlns/premis-v2" version="2.2">
  <object xsi:type="file">
    <objectIdentifier>
      <objectIdentifierType>ARK</objectIdentifierType>
      <objectIdentifierValue>ark:/12345/bc6gh</objectIdentifierValue>
    </objectIdentifier>
    <objectCharacteristics>
      <compositionLevel>0</compositionLevel>
      <size>2194735</size>
      <format>
        <formatDesignation>
          <formatName>PDF</formatName>
          <formatVersion>1.7</formatVersion>
        </formatDesignation>
        <formatRegistry>
          <formatRegistryName>PRONOM</formatRegistryName>
          <formatRegistryKey>fmt/276</formatRegistryKey>
        </formatRegistry>
      </format>
      <creatingApplication>
        <creatingApplicationName>Springer-book-section.doc</creatingApplicationName>
          <dateCreatedByApplication>2014-02-25T16:22:40Z</dateCreatedByApplication>
      </creatingApplication>
      <fixity>
        <messageDigestAlgorithm>MD5</messageDigestAlgorithm>
        <messageDigest>d6aa97d33d459ea3670056e737c99a3d </messageDigest>
      </fixity>
      <objectCharacteristicsExtension>
        <doc xmlns:docmd="http://www.fcla.edu/dls/md/docmd.xsd">
          <docmd:document>
            <docmd:PageCount>58</docmd:PageCount>
            <docmd:Font FontName="Arial" isEmbedded="false" />
            <docmd:Font FontName="TimesNewRoman,BoldItalic" isEmbedded="false" />
            <docmd:Font FontName="Arial,Bold" isEmbedded="false" />
            <docmd:Font FontName="TimesNewRoman,Italic" isEmbedded="false" />
            <docmd:Feature>hasThumbnails</docmd:Feature>
          </docmd:document>
        </docmd:doc>
      </objectCharacteristicsExtension>
    </objectCharacteristics>
  </object>
</premis>
```

Fig. 14.3 Example of PREMIS extended with documentMD

implemented as container files, such as TAR [17]. It records metadata, such as the number of files, formats represented, and compression used. While a repository may choose to implement the Archival Information Package (AIP) [22: pp. 1–7] as a TAR file, you would not use containerMD to describe the AIP container but rather the content files that happen to be containers, because preservation metadata is defined at the level of content objects.

Though initially designed for ARC files [18], this schema was conceived as a generic way to express information about any kind of container; containerMD was established to address the following needs:

- Provide scalable (concise) descriptions, without having to describe each file individually;
- Express container-specific information, including information that is very specific to web archives and WARC [19] and ARC [18] container files
- Have a self-contained description template that can be used as an output for a characterization tool.[2]

Those different requirements were met with the following structure:

- a description of the container file itself in a <container> section
- for the content files, two sections that correspond to different levels of granularity:
  - a concise, nonverbose mode aggregating information about the entries in an <entriesInformation> section;
  - a verbose mode providing a detailed description of each contained file, in separate <entry> child elements.
- At any of those levels, extension sections that provide fields that are specific to some container formats, e.g., WARC or ARC container files.
- To achieve the ability to output a standalone description, it, unlike documentMD, duplicates generic information that can be expressed in PREMIS.

Within PREMIS containerMD is expressed as follows:

- PREMIS already provides dedicated generic semantic units to describe the container and the content file: a file containing another file is modeled as two PREMIS *Files* that are related with a structural whole/part relationship. If one wants to describe container and content files with discrete descriptions, one should use PREMIS for those generic fields.
- Some information important for preservation, but specific to the web archiving process is available in ARC and WARC specific fields and can only be expressed in containerMD. Such information can be embedded in a parent PREMIS *objectCharacteristicsExtension* semantic container.
- containerMD allows a concise serialization of the metadata of the contained files in the <containerMD:entriesInformation> field. If one wants to leverage containerMD to reduce granularity, one can embed the <entriesInformation> element in the PREMIS *File* description describing the parent container file.

A sample *File* can be seen here in Fig. 14.4.

[2]ContainerMD in nonverbose mode is an optional output for the JHOVE2 ARC file analysis module, used in the JhoNAS project. See [23].

Fig. 14.4 Fig. 14.4 ContainerMD sample File (*Note* The <responses> section records the information exchanged between the harvesting robot and the web server. Here, 14 files are content files (HTTP 200 response code: content found) and 4 files are error messages from the server (HTTP 500 response code).)

```
<premis:premis version="2.2" xmlns:premis="info:lc /xmlns/premis-v2">
  <premis:object xsi:type="premis:file">
    <premis:objectIdentifier>
      <premis:objectIdentifierType>ARK</premis:objectIdentifierType>
      <premis:objectIdentifierValue>ark:/12345/b123456
      </premis:objectIdentifierValue>
    </premis:objectIdentifier>
    <premis:objectCharacteristics>
    <!--First compositionLevel = the file ready-to-access. It is the gunzipped ARC file--
>
      <premis:compositionLevel>0</premis:compositionLevel>
      <premis:fixity>
        <premis:messageDigestAlgorithm>MD5</premis:messageDigestAlgorithm>
        <premis:messageDigest>954b95c2ab7865da1262de99a9f62060
        </premis:messageDigest>
      </premis:fixity>
       <premis:size>996991</premis:size>
       <premis:format>
         <premis:formatDesignation>
           <premis:formatName>application/x-ia-arc</premis:formatName>
           <premis:formatVersion>1</premis:formatVersion>
         </premis:formatDesignation>
         <premis:formatRegistry>
           <premis:formatRegistryName>PRONOM</premis:formatRegistryName>
           <premis:formatRegistryKey>x-fmt/219</premis:formatRegistryKey>
         </premis:formatRegistry>
       </premis:format>
       <!--The objectCharacteristicsExtension, describing the ARC file.-->
       <premis:objectCharacteristicsExtension ID="container.1">
         <premis:mdWrap MDTYPE="OTHER" OTHERMDTYPE=" containerMD">
           <premis:xmlData>
             <containerMD:containerMD
xmlns:containerMD="http://bibnum.bnf.fr/ns/containerMD-v1">
               <containerMD:entries>
                 <!--Aggregated information about the harvested files inside the ARC
container-->
                 <containerMD:entriesInformation number="18" globalSize="996991"
                   firstDateTime="2010-06-27T19:05:26Z"
                   lastDateTime="2010-06-29T07:55:27Z" minimumSize="55"
                   maximumSize="433989">
                   <containerMD:formats>
                     <containerMD:format name="text/html" number="15"
                       globalSize="563002" />
                     <containerMD:format name="application/x-shockwave-flash"
                       number="3" globalSize="433989" />
                   </containerMD:formats>
                   <containerMD:entriesExtension>
                     <containerMD:ARCEntries>
                       <containerMD:declaredMimeTypes>
                         <containerMD:declaredMimeType number="14"
                           globalSize="303160"
                           >text/html</containerMD:declaredMimeType>
                         <containerMD:declaredMimeType number="3"
                           globalSize="259842"
                           >image/gif</containerMD:declaredMimeType>
                         <containerMD:declaredMimeType number="1"
                           globalSize="433989">application/x-shockwave-
                           flash</containerMD:declaredMimeType>
                       </containerMD:declaredMimeTypes>
                       <containerMD:hosts>
                         <containerMD:host number="16" globalSize="702435">
                         kavo.fr</containerMD:host>
                         <containerMD:host number="2" globalSize="294556">
                         kavo-everest.fr</containerMD:host>
                       </containerMD:hosts>
                       <containerMD:responses>
                         <containerMD:response number="14"
                           protocolName="http" protocolVersion="http /1.1"
                           globalSize="965575">200</containerMD:response>
                         <containerMD:response number="4" protocolName="http"
                           protocolVersion="http/1.1" globalSize="31416"
                           >500</containerMD:response>[62]
                       </containerMD:responses>
                     </containerMD:ARCEntries>
                   </containerMD:entriesExtension>
                 </containerMD:entriesInformation>
               </containerMD:entries>
             </containerMD:containerMD>
           </premis:xmlData>
         </premis:mdWrap>
       </premis:objectCharacteristicsExtension>
    </premis:objectCharacteristics>
    <premis:originalName>557-2-20101227190928-0145.arc</premis:originalName>
  </premis:object>
  </premis:premis>
```

14.3 Container Formats

There are a number of different container formats that have different purposes and can thus relate to PREMIS metadata in different ways.

In most cases container formats are metadata formats that can include metadata, such as PREMIS. We refer to these formats here as *metadata container formats*. For instance, this is the case for METS [2] and XFDU [24] as described below. It is seen as a metadata container rather than a general packaging format if the format is primarily concerned with packaging metadata rather than content files and is used to structure the metadata.

Other container formats mainly have the purpose of packaging PREMIS or other metadata along with the actual data. These formats are here called *packaging container formats*. An example of such a format is the WARC [19] format described below.

The distinction between *metadata container formats and packaging container formats* is helpful in order to distinguish between the challenges that the format intends to address. The distinction is not always clear-cut. For instance, METS is designed as a transmission standard, allowing embedding files directly in the XML stream using <bindata>. However today, METS is only used as a *metadata container format* in the majority of use cases with links to the content, while other formats like WARC [19] and BagIt [25] are used for the purpose of packaging content and metadata.

Different container formats may include PREMIS metadata in different ways, and the way PREMIS is included may be just as important as the choice of the actual metadata container format. Therefore, this topic is described here including how to embed PREMIS in the METS format.

14.3.1 Metadata Container Formats

A metadata container format is needed where there is a need to specify the PREMIS metadata in connection with other metadata, for example, with descriptive metadata or metadata describing the structure of a complex object.

There are many different metadata container formats, but there is no one simple way to choose between metadata container formats. In each case the choice of metadata container format is related to requirements and policies regarding metadata.

Below is a nonexhaustive list of metadata container formats. It includes formats with varying original purposes and forms that are found in use with PREMIS in the literature. The list includes a short description of the selected metadata container formats given in alphabetic order.

DIDL: Digital Item Declaration Language
The DIDL format is defined in relation with the MPEG-21 standard [26], which was developed by the Moving Picture Experts Group. DIDL can be used for description of complex digital objects at a very abstract level. DIDL, unlike METS, has an underlying model that has XML as just one possible serialization choice. It is mainly concerned with structure and representation as an XML byte stream, with only a few attributes to describe descriptive and technical metadata [27, 28].

METS: The Metadata Encoding and Transmission Standard
The METS format was originally designed for transmission of information packages [29]; METS is hosted by the Library of Congress [2] and maintained by an Editorial Board [30]. METS is a flexible XML-based container format that can include a range of other XML-based metadata schemas. Today METS is the most widely used container format for storing preservation metadata and for linking it to other types of metadata for digital material [31].

TIPR: Towards Interoperable Preservation Repositories
The TIPR initiative has defined a standards-based package of metadata files that can act as an intermediary information package: Repository eXchange Package (RXP) [32, 33]. The original purpose was to make it possible to transfer complex digital objects between dissimilar OAIS-based [22] preservation repositories. This format is based on elements expressed in PREMIS and METS, and there are requirements about which parts of PREMIS and METS are to be included in different parts/files of the RXP packages [32, 33].

XFDU: XML Formatted Data Unit
The XFDU format is a format described in an ISO standard [24]. The format is developed by the Consultative Committee for Space Data Systems (CCSDS), which also is the organization that developed the OAIS reference model [22]. XFDU is similar to METS, but is designed to reflect more closely the OAIS Information Model.

Choosing one of these formats could be based on functionality requirements such as the most frequently used formats in a given community (e.g., METS for libraries), the format closest to OAIS (e.g., XFDU), or the format that uses the same serialization choice (XML, RDF) as other chosen metadata standards.

14.3.2 Packaging Container Formats

A packaging container format is needed where there is a need to specify the PREMIS metadata (or wrapped PREMIS metadata) in connection with other metadata and/or in connection with the content object.

There are many different packaging container formats. A discussion on which package formats that can best support requirements for digital preservation can be found in [34, 35].

Below is a nonexhaustive list of packaging container formats that have been considered suitable for digital preservation in [34, 35]. The list includes a short description of the selected packaging formats given in alphabetic order.

AFF: Advanced Forensic Formats
AFF are formats specifically designed to contain metadata for forensics [36], i.e., for information extraction from, often obsolete, data carrier formats. The format can include files and folders. In order to use it for packaging metadata and contents, it is necessary to define a structure separately, as AFF are not designed to address this. One disadvantage of AFF is that it is limited to describing disk images (e.g., ISO images) as opposed to describing a general collection of files or folders [35].

BagIt
The BagIt [25] format is intended for packing and unpacking of files into and from folders. It was originally designed for exchange of information rather than as a storage format. The BagIt format provides a way to specify metadata on the bag itself (e.g., external identifier and contact information) as well as specifying the file structure. BagIt is only concerned with this structure and bag metadata, while it leaves inclusion of bundling of package pieces to other formats such as ZIP [16] or TAR [17].

TAR: Tape ARchive format
The TAR format is a standardized (POSIX.1-2001) packaging format, originally designed for archiving on tapes. The TAR format is therefore file-oriented, but also byte-oriented (i.e., relative position of files in the container information package, handled by offsets) [17]. The TAR format is designed to handle collections of many files for backup or transfer purposes and is especially suitable for streaming very large packages. However, for metadata, the TAR format has no centralized location for the information about the content files, i.e., it is not easy to make relations between identifiers and files. This means that structure for packaging metadata and contents must be defined separately.

WARC: Web ARChive format
The WARC format is a standardized packaging format, originally designed for web-archived material [19]. The WARC format refers to byte streams by pointing to their byte position and WARC provides the capability for adding metadata for the byte streams it includes, for example, identifier, type of resource (e.g., content or metadata), mime type, and linking information. Some of this is web-specific, but it is a general design which can be and has been used for packaging objects for preservation, leaving out web-specific information [19, 34].

The choice of a packaging format is determined by the requirements related to the purpose of the packaging. This can, for instance, be a preservation-related requirement of assigning identifiers to resources without changing the content (e.g., in WARC [34]) or specific requirements related to forensic packages (e.g., in a AFF format [35]). Requirements can also relate to the use of the package, e.g., for exchange of packages across institutions, for instance, in a BagIt file structure wrapped in TAR.

14.3.3 Inclusion of PREMIS Metadata in Container Formats

There are many ways to include PREMIS metadata in container formats. Packaging container formats do not distinguish different types of metadata or metadata from content, thus these formats can include PREMIS directly or as part of a metadata container format. Including PREMIS in metadata container formats requires more planning. This section describes inclusion of PREMIS metadata in METS, because METS is the best-known and best-documented container format that has strong metadata components and is often used for associating preservation metadata with content [31]. But many of the considerations concerning the relationship between PREMIS and METS have parallels with other similar formats.

One example is that all PREMIS metadata could be placed as digital provenance information in the METS <digiProvMD> element. This is illustrated in Fig. 14.5, where the white elements are the METS elements, the dark gray elements are types of metadata other than PREMIS that may be included in the METS container, and the light gray elements are the PREMIS elements.

Another example is when information captured in PREMIS is spread across several METS elements, following the intended function of the PREMIS entity. For instance, the part of the PREMIS Object with technical information (possibly along with associated PREMIS Agent information) is placed in the METS technical metadata <techMD> element. Similarly, the PREMIS Rights part (possibly along with associated PREMIS Agent information) is placed in the METS rights metadata <rightsMD> element. This is illustrated in Fig. 14.6.

The alternatives expressed in the above two examples are discussed in the set of guidelines [37, 38] giving pragmatic recommendations for using PREMIS with

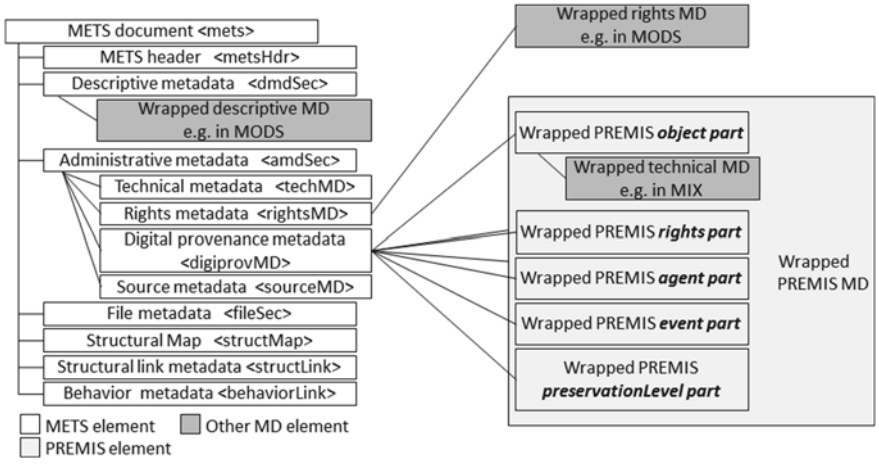


Fig. 14.5 Inclusions of PREMIS in METS digital provenance metadata element

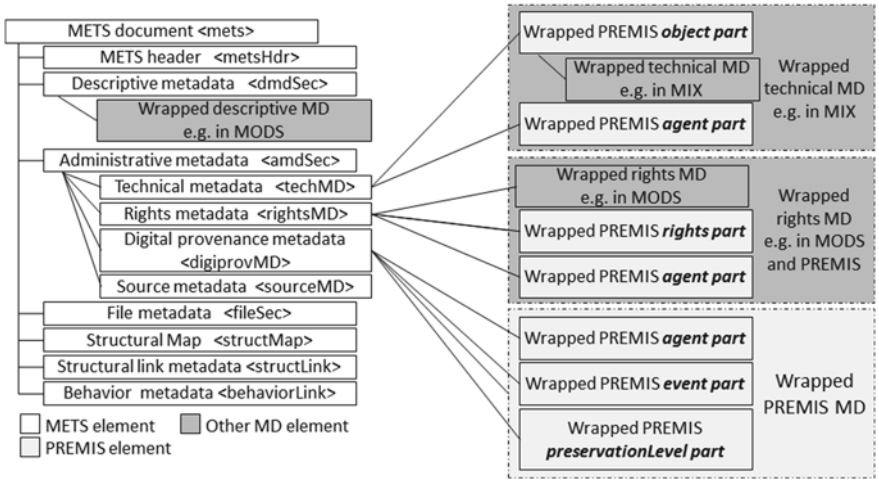


Fig. 14.6 Inclusions of PREMIS spread across several METS elements

METS. A checklist for making implementation decisions about how to include PREMIS in a METS profile is also available [39]. A variation of spreading PREMIS information across several METS elements is described in an Australian case study [40]. A detail that is not explicitly mentioned in these papers is the placement of the PREMIS semantic unit *preservationLevel*. The Royal Library of Denmark regards PREMIS *preservationLevel* as part of the digital provenance [41].

An example of the package container format WARC, which wraps the content files together with their PREMIS-in-METS metadata is given in abbreviated form (Fig. 14.7). It is based on version 2.2 of the PREMIS Data Dictionary [42], thus changes in version 3 are explained in footnotes when the case occurs. "…" indicates omitted information; a full example can be found in [43].

14.4 Descriptive Metadata Formats

14.4.1 Outline of Descriptive Metadata Formats

Descriptive metadata is metadata that is used for discovery of and access to digital content. It often describes the intellectual content of digital objects and is therefore associated with PREMIS *Intellectual Entities*. It is used in the overall lifecycle and curation of the object. While it is not a piece of information that is used to trigger a particular preservation action specific to the *form* of the digital object it is essential for understanding the *content* over the long-term and therefore constitutes a form of preservation metadata. However, PREMIS is designed to remain domain-agnostic and descriptive metadata is often domain-specific.

```
WARC-Type: warcinfo
WARC-Date: 2013-01-18T19:28:02Z
WARC-Record-ID: <urn:UUID:cb486d10-619c-11e2-911b-005056887b67>
Content-Type: application/warc-fields
Content-Length: 85
description: http://id.kb.dk/authorities/agents/kbDkDomsBmIngest.html
revision: 2041

WARC/1.0
WARC-Type: metadata
WARC-Target-URI: urn:UUID:c9db2170-619c-11e2-911b-005056887b67
WARC-Refers-To: <urn:UUID:41d153d0-0099-11e2-9397-005056887b67>
WARC-Date: 2013-01-18T19:27:59Z
WARC-Block-Digest: sha1:62cc454ef47c7d54b77f871ab1ffd3f580307414 -
WARC-Record-ID: <urn:UUID:c9db2170-619c-11e2-911b-005056887b67>
Content-Type: text/xml
Content-Length: 13926
<?xml version="1.0" encoding="UTF-8"?>
<mets:mets xmlns:mets="http://www.loc.gov/METS/" ...>
 <mets:metsHdr CREATEDATE="2013-01-18T19:28:01.025+01:00">
  <mets:agent ID="kbDk" ROLE="CREATOR" TYPE="ORGANIZATION">
   <mets:name>kbDk</mets:name>
   <mets:note>kbDkInternal</mets:note>
  </mets:agent>
  ...
 </mets:metsHdr>
 <mets:dmdSec CREATED="2013-01-18T19:28:01.035+01:00" ID="Mods1">
  <mets:mdWrap MDTYPE="MODS">

      ...
  </mets:mdWrap>
 </mets:dmdSec>
 <mets:amdSec>
  <mets:techMD CREATED="2013-01-18T19:28:01.426+01:00" ID="PrOb1">
   <mets:mdWrap MDTYPE="PREMIS:OBJECT">
    <mets:xmlData>
     <!-- The object which the metadata relates to.
          It is the same as the WARC-Refers-To field, but it is only in the WARC specification
          for optimization of access functions.
          It is the same as for the WARC record of WARC-Type: resource specified below.
     -->
     <premis:object ... /premis/v2/... xsi:type="premis:file">
      <premis:objectIdentifier>
       <premis:objectIdentifierType>UUID</premis:objectIdentifierType>
       <premis:objectIdentifierValue>41d153d0-0099-11e2-9397-005056887b67
       </premis:objectIdentifierValue>
      </premis:objectIdentifier>

       <premis:compositionLevel>0</premis:compositionLevel>
       <premis:fixity>
       <!-- Is the same as for the WARC record of  WARC-Type: resource specified below.
       -->
```

Fig. 14.7 WARC file containing a METS file bundling PREMIS with other metadata schemas (*Note* This example is in PREMIS 2. In PREMIS 3.0. linkingIntellectualEntityIdentifier is replaced by relatedObjectIdentifier. Additionally, a preservationLevelType semantic unit is added. In this example, « bitSafetyHigh » would be split into 2 semantic units: preservationLevelType (value : « Bit preservation ») and preservationLevelValue (value : « High »). In the mets:digiProv section, further mets:digiprovMDs are needed since they are of different METS MDTYPEs)

```
            <premis:messageDigestAlgorithm>MD5</premis:messageDigestAlgorithm>
            <premis:messageDigest>3f349a40b0c47bb070ea6bdd2759a731
             </premis:messageDigest>
          </premis:fixity>
          <premis:size>137362594</premis:size>

          <premis:format>
            <premis:formatDesignation>
              <premis:formatName>TIFF Image</premis:formatName>
              <premis:formatVersion>6.0</premis:formatVersion>
            </premis:formatDesignation>
          </premis:format>
          <premis:objectCharacteristicsExtension>
            <mix:mix xsi:schemaLocation="http://www.loc.gov/mix/v20 ... ">
              <mix:BasicDigitalObjectInformation>
                …
            </mix:mix>
            </premis:objectCharacteristicsExtension>
          </premis:objectCharacteristics>
          <premis:linkingIntellectualEntityIdentifier>
          <!-- The linkingIntellectualEntityIdentifier refers to the Intellectual Entity which the
                object is a Representation of – e.g. a landing page for different Representations of
the
                same intellectual content including migrations over time and dissemination
          versions.-->
            <premis:linkingIntellectualEntityIdentifierType>UUID
            </premis:linkingIntellectualEntityIdentifierType>
            <premis:linkingIntellectualEntityIdentifierValue>
            41d153d1-0099-11e2-9397-005056887b67
            </premis:linkingIntellectualEntityIdentifierValue>
          </premis:linkingIntellectualEntityIdentifier>
        </premis:object>
      </mets:xmlData>
    </mets:mdWrap>
  </mets:techMD>
  <mets:rightsMD CREATED="2013-01-18T19:28:01.455+01:00" ID="MoRi1">
    …
  </mets:rightsMD>
  <mets:digiprovMD CREATED="2013-01-18T19:28:01.456+01:00" ID="Premis1">
    <mets:mdWrap MDTYPE="PREMIS">
      <mets:xmlData>
        <premis:preservationLevel xmlns:xlink="http://www.w3.org/1999/xlink"  … >
          <premis:preservationLevelValue>bitSafetyHigh</premis:preservationLevelValue>
          <premis:preservationLevelDateAssigned>2013-01-18T19:28:01.458+01:00
          </premis:preservationLevelDateAssigned>
        </premis:preservationLevel>
        <premis:preservationLevel xmlns:xlink="http://www.w3.org/1999/xlink" ...>
          <premis:preservationLevelValue>logicalStrategyMigration
          </premis:preservationLevelValue>
          <premis:preservationLevelDateAssigned>2013-01-18T19:28:01.459+01:00
          </premis:preservationLevelDateAssigned>
        </premis:preservationLevel>
```

Fig. 14.7 (continued)

```
            …
          </mets:xmlData>
        </mets:mdWrap>
      </mets:digiprovMD>
      <mets:digiprovMD CREATED="2013-01-18T19:28:01.460+01:00" ID="PrEv1">
        <mets:mdWrap MDTYPE="PREMIS:EVENT">
          <mets:xmlData>
            <premis:event xmlns:xlink="http://www.w3.org.1999/xlink" … >
            <premis:eventIdentifier>
              <premis:eventIdentifierType>UUID</premis:eventIdentifierType>
              <premis:eventIdentifierValue>
              e814a0cc-0230-43aa-b9a5-38fb38298557
              </premis:eventIdentifierValue>
            </premis:eventIdentifier>
            <premis:eventType>ingestion</premis:eventType>
            <premis:eventDateTime>2013-01-18T19:28:01.462+01:00</premis:eventDateTime>
            <premis:linkingAgentIdentifier>
              <premis:linkingAgentIdentifierType>kbDkInternal
              </premis:linkingAgentIdentifierType>
              <premis:linkingAgentIdentifierValue>kbDkDomsBmIngest (2041)
              </premis:linkingAgentIdentifierValue>
            </premis:linkingAgentIdentifier>
            <premis:linkingObjectIdentifier>
              <premis:linkingObjectIdentifierType>UUID</premis:linkingObjectIdentifierType>
              <premis:linkingObjectIdentifierValue>
               41d153d0-0099-11e2-9397-005056887b67
              </premis:linkingObjectIdentifierValue>
            </premis:linkingObjectIdentifier>
            …
          </premis:event>
        </mets:xmlData>
      </mets:mdWrap>
    </mets:digiprovMD>
  </mets:amdSec>
  <mets:fileSec>…</mets:fileSec>
  <mets:structMap>…</mets:structMap>
</mets:mets>

WARC/1.0
WARC-Type: resource
WARC-Target-URI: urn:UUID:41d153d0-0099-11e2-9397-005056887b67
WARC-Date: 2013-01-18T19:27:59Z
WARC-Block-Digest: md5:3f349a40b0c47bb070ea6bdd2759a731
WARC-Record-ID: <urn:UUID:41d153d0-0099-11e2-9397-005056887b67>
Content-Type: image/tiff
Content-Length: 137362594
...<tiff bitstream included here> ...
```

Fig. 14.7 (continued)

Metadata schemas already exist that provide domain-specific fields to describe content; it would have been unnecessary to duplicate this information in PREMIS. Instead one reuses general descriptive metadata and combines it with the more specific PREMIS preservation metadata. As such, they are often used in conjunction with PREMIS, sometimes in the same *metadata container* file. Below, we

provide a nonexhaustive list of the descriptive metadata schemas used most in conjunction with PREMIS and their domain of application:

- domain-agnostic formats: Dublin Core [44]
- library-specific formats: MARCXML [45], MODS [5]
- archive-specific formats: EAD [6]
- museum-specific formats: LIDO [46], VRACore [47].

Additionally, preservation metadata that supports the preservation of the content of the digital object, rather than its form, such as significant properties or rights information, can, in some cases, apply to the *Intellectual Entity*.

Since preservation tools and activities need to access both descriptive and PREMIS metadata in an efficient and coordinated way it is important to specify how the two forms of metadata should be related.

14.4.2 PREMIS and Descriptive Metadata

Three alternatives can be applied, listed below:

1. To capture this information one can store the descriptive metadata along with the AIP: here one has to rely on a metadata container format to wrap both the PREMIS and descriptive metadata. Here, the preservation and curation are combined in a single system.
2. Handle the descriptive metadata in separate access system (Integrated Library System, Electronic Records Management System…): here, a link to the *Intellectual Entity* description can be used as long as it can be permanently and uniquely identified;
3. Combine both approaches, with the preservation repository ensuring preservation of the descriptive metadata curated in the access system, and a link to external, up-to-date descriptive metadata. In such a case, procedures have to be defined when updated metadata needs to be preserved.

Robust bridges between two systems are crucial for approaches 2 and 3 above. In PREMIS version 2 the linkingIntellectualEntityIdentifier is used to link a digital object *Representation* to its description in an *Intellectual Entity*, thereby relating core preservation and descriptive metadata. (Note that linkingIntellectualEntityIdentifier is replaced by *relatedObjectIdentifier* in PREMIS 3.) It is important to assign robust identifiers for the digital objects and their descriptive records that can be used for cross-referencing. The implementing organization must commit to the persistence of those identifiers.[3] In a nutshell, they need to be unique within a clearly defined

[3]On persistent identifiers, see the guidelines published at http://ands.org.au/guides/persistent-identifiers-expert.html and the following report: http://nbn-resolving.de/urn:nbn:de:gbv:7-isbn-90-6984-508-3-8.

context, at a particular point in time and through time (the latter meaning in most cases, non-reattribution of the identifier over the long term). Identifiers need to remain persistent, which is best achieved using opaque identifiers that one is not tempted to change,[4] and using identifiers that are implementation independent and not broken by a technological change.[5] It is perfectly possible to delete an object that has been given a persistent identifier, provided the deletion event is traced with appropriate metadata, and the identifier of the deleted object is not reassigned to a new one. Last but not least, the resulting persistent identification commitment should be documented by the organization. Persistent identification commitment can be based on in-house definitions or on persistent identifiers standards, such as ARK, UUID, Handle or DOI [48].

14.5 Conclusion: Making PREMIS Fit with Other Schemas

Making PREMIS fit with other schemas forces one to have a clear idea about two things:

- Understand what is in and what is outside that scope of PREMIS.
- Consider your systems requirements where a choice has to be made about your metadata implementation.[6]

14.5.1 What PREMIS Is and What It Is Not

- PREMIS is *core preservation metadata*, therefore, it is specific to the digital objects, and only considers their preservation over the long term.
- PREMIS is *technically neutral*, therefore it needs to be complemented by format-specific metadata to express information relating to image, text, video, sound, office documents, or container files.
- PREMIS is *agnostic* about how you bundle the digital object with its metadata. This is achieved by packaging formats.

[4]This means a number like 123456789, or an alphanumeric identifier like b6f5g56dcm9.

[5]This is particularly the case of URIs, e.g., a persistent URI should look like http://www.example.com/id/123456 and should not look like http://www.example.com/index.php?type=premis:object&id=123456, which will break when the retrieval mechanism is not implemented in PHP anymore.

[6]See the three first chapters of this book for an overview of implementation questions to consider; Chap. 13 for serialization options; Chaps. 15 through 17 for systems options.

As a result, you need to articulate how PREMIS descriptions interoperate with the other metadata formats. This is achieved by two means:

- *Metadata container formats* that wrap PREMIS metadata with other metadata that might be stored with the digital object, such as format-specific technical metadata or descriptive metadata.
- *Persistent identifiers* that provide stable and unambiguous identification of the digital objects and metadata that might be curated outside of the Information Package or outside of the preservation repository.

14.5.2 Coping with Overlap

Other metadata formats overlap with PREMIS. This is especially the case of specific technical metadata formats and of metadata container formats. How should these overlaps be resolved?

Schemas that overlap for a particular metadata element may differ in how well they fit systems requirements.

- one might be more granular than the other, such as a set of XML elements versus one single element,
- one might be more extensible,
- one may be repeatable and the other not, such as an XML element versus a single XML attribute,
- and so on.

If both schemas fit your requirements, you have the following basic options:

- Express the information at *a single place*, unless the field is mandatory in the other schema—In that case, it recommended to duplicate the information rather than use a local version of the standard schema to make the mandatory fields optional.
- Express the information in *both schemas*. In that case, you have to make sure that, whenever you update the metadata, it is updated at both places to guarantee understandability and consistency. Or there needs to be well-documented procedures for which place carries the most trustworthy set of values.

14.5.3 Document Your Choices

Using PREMIS with other metadata schemas requires implementation choices that need to be documented for both the short and long term.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 05 Jan 2016
2. Library of Congress, METS Editorial Board (2015) Metadata Encoding and Transmission Standard (METS): official web site. http://www.loc.gov/standards/mets. Accessed 05 Jan 2016
3. Library of Congress (2015) NISO metadata for images in XML schema (MIX). http://www.loc.gov/standards/mix. Accessed 03 Jan 2016
4. Library of Congress (2009) Technical metadata for text (textMD). http://www.loc.gov/standards/textMD. Accessed 03 Jan 2016
5. Library of Congress, MODS Editorial Board (2015) Metadata object description schema (MODS): official web site. http://www.loc.gov/standards/mods/. Accessed 09 Jan 2016
6. Library of Congress, Society of American Archivists (2015) Encoded archival description (EAD): official web site. http://www.loc.gov/ead/. Accessed 09 Jan 2016
7. W3C (2014) Resource description framework (RDF). http://www.w3.org/RDF/. Accessed 09 Jan 2016
8. W3C (2015) Linked data. http://www.w3.org/standards/semanticweb/data. Accessed 09 Jan 2016
9. Library of Congress (2013) Preservation Metadata: Implementation Strategies (PREMIS) ontology. http://www.loc.gov/premis/rdf/v1#. Accessed 09 Jan 2016
10. Library of Congress (2016) Linked data service, authorities and vocabularies: preservation schemes. http://id.loc.gov/preservationdescriptions/. Accessed 05 Jan 2016
11. American National Standards Institute, National Information Standards Organization (2006) ANSI/NISO Z39.87-2006 (R2011): data dictionary—technical metadata for still images. http://www.niso.org/apps/group_public/download.php/14698/z39_87_2006_r2011.pdf. Accessed 05 Jan 2016
12. Public Broadcasting Metadata Dictionary Project (2015). PBCore web site. http://www.pbcore.org. Accessed 03 Jan 2016
13. Chou C, Goethals A (2009) Document metadata: document technical metadata for digital preservation. http://fclaweb.fcla.edu/uploads/Lydia%20Motyka/FDA_documentation/documentMD.pdf. Accessed 03 Jan 2016
14. Bibliothèque nationale de France (2011) containerMD. http://bibnum.bnf.fr/containerMD-v1. Accessed 03 Jan 2016
15. Adobe Systems (2016) Adobe PDF reference archives. https://www.adobe.com/devnet/pdf/pdf_reference_archive.html. Accessed 09 Jan 2016
16. PKWARE Inc (2014) ZIP file format specification. http://www.pkware.com/documents/casestudies/appnote.txt. Accessed 03 Jan 2016
17. Free Software Foundation. GNU TAR. http://www.gnu.org/software/tar/. Accessed 03 Jan 2016
18. Internet Archive (1996) Arc file format. http://archive.org/web/researcher/ArcFileFormat.php. Accessed 09 Jan 2016
19. International Organization for Standardization (2009) ISO 28500:2009: Information and documentation—WARC file format. http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=44717. Accessed 03 Jan 2016
20. International Digital Publishing Forum (2015) EPUB. http://idpf.org/epub. Accessed 09 Jan 2016
21. Adobe Systems (2016) Extensible metadata platform (XMP). http://www.adobe.com/products/xmp.html. Accessed 09 Jan 2016
22. Consultative Committee for Space Data Systems, International Organization for Standardization (2012) Reference model for an Open Archival Information System (OAIS): Issue 2. http://public.ccsds.org/publications/archive/650x0m2.pdf. Accessed 06 Jan 2016

23. Jhonas Project (2013) JHoNas final report: Foster WARC usage in scalable web archiving workflows using Jhove2 and NetarchiveSuite. http://netpreserve.org/sites/default/files/resources/jhonas-final-report.pdf. Accessed 09 Jan 2016
24. Consultative Committee for Space Data Systems (2008) XML formatted data unit (XFDU) structure and construction rules. http://public.ccsds.org/publications/archive/661x0b1.pdf. Accessed 03 Jan 2016
25. Kunze J, Littmann J, Madden L, Summers E, Boyko A, Vargas B (2015) The BagIt file packaging format (V0.97). http://tools.ietf.org/html/draft-kunze-bagit-11. Accessed 03 Jan 2016
26. International Organization for Standardization (2005) ISO/IEC 21000-2:2005: Information technology—multimedia framework (MPEG-21)—part 2: digital item declaration. http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=41112. Accessed 03 Jan 2016
27. Dappert A, Enders M (2010) Digital preservation metadata standards. Inf Stand Q 22(2):5–13
28. International Organization for Standardization (2005) MPEG-21 schema files. DIDL XML Schema. http://standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-21_schema_files/did/didl.xsd. Accessed 03 Jan 2016
29. Cundiff MW (2004) An introduction to the Metadata Encoding and Transmission Standard. Libr High Tec 22(1):52–64
30. Library of Congress (2015) METS editorial board. http://www.loc.gov/standards/mets/mets-board.html. Accessed 10 Jan 2016
31. Gartner R, Lavoie B (2013) Preservation metadata. DPC Technol Watch Rep. doi:10.7207/twr13-03
32. Caplan P, Kehoe W, Pawletko J (2010) Towards interoperable preservation repositories: TIPR. Int J Digital Curation 1(5). http://ijdc.net/index.php/ijdc/article/viewFile/145/207. Accessed 03 Jan 2016
33. Florida Center for Library Automation (2011) TIPR web site. http://wiki.fcla.edu:8000/TIPR. Accessed 03 Jan 2016
34. Zierau E (2012) Package formats for preserved digital material. Paper presented at the 9th international conference on preservation of digital objects, Toronto. http://hdl.handle.net/109.1.5/1a40ed85-ad9b-459d-8b35-b763142127ed. Accessed 03 Jan 2016
35. Kim Y, Ross S (2011) Digital forensics formats: seeking a digital preservation storage format for web archiving. Paper presented at the 7th international digital curation conference, Bristol. http://eprints.gla.ac.uk/62565. Accessed 03 Jan 2016
36. Forensics Wiki (2014) Advanced forensics format (AFF). http://www.forensicswiki.org/wiki/AFF. Accessed 03 Jan 2016
37. Guenther R (2008) Battle of the buzzwords: flexibility vs. interoperability when implementing PREMIS in METS. D Lib Mag 14. doi:10.1045/july2008-guenther
38. Brandt O, Enders M, Guenther R, Habing T, Lazzarino F, Redding C, Riley J, Wolfe R (2008) Guidelines for using PREMIS with METS for exchange. http://www.loc.gov/standards/premis/guidelines-premismets.pdf. Accessed 03 Jan 2016
39. Vermaaten S (2010). A checklist and a case for documenting PREMIS-METS decisions in a METS profile. D Lib Magaz 16(9/10). doi:10.1045/september2010-vermaaten
40. Pearce J, Pearson D, Williams M, Yeadon S (2008) The Australian METS profile: a journey about metadata. D Lib Magaz. doi:10.1045/march2008-pearce
41. Zierau E (2013) PREMIS implementation at the royal library of Denmark. http://www.loc.gov/standards/premis/pif-presentations-2013/08PREMIS-Zierau-KB_Case.pdf. Accessed 03 Jan 2016
42. PREMIS Editorial Committee (2012) PREMIS data dictionary for preservation metadata version 2.2. http://www.loc.gov/standards/premis/v2/premis-2-2.pdf. Accessed 05 Jan 2016
43. Zierau E, PREMIS Maintenance Activity (2013) Full WARC example (METS & PREMIS). https://www.loc.gov/standards/premis/pif/2013/ZierauWarcExample.txt. Accessed 13 Feb 2016

44. Dublin Core Metadata Initiative (2015) Web site. http://dublincore.org/. Accessed 10 Jan 2016
45. Library of Congress (2015) MARC21 XML schema: official web site (MARCXML) http://www.loc.gov/standards/marcxml/. Accessed 10 Jan 2016
46. International Council for Museums (ICOM): CIDOC (2010) What is LIDO. http://network.icom.museum/cidoc/working-groups/lido/what-is-lido/. Accessed 10 Jan 2016
47. Library of Congress (2014) VRA CORE: A data standard for the description of images and works of art and culture. https://www.loc.gov/standards/vracore/. Accessed 10 Jan 2016
48. Hilse HW, Kothe J (2006) Implementing persistent identifiers. Consortium of European Research Libraries and European Commission on Preservation and Access. http://nbn-resolving.de/urn:nbn:de:gbv:7-isbn-90-6984-508-3-8. Accessed 10 Jan 2016

Chapter 15
Tools for Working with PREMIS

Carol Chou, Andrea Goethals and Julie Seifert

15.1 Introduction

As of 2016, nearly every major commercial and open-source software system for long-term archiving and preservation of digital content supports PREMIS [1] metadata. Furthermore, there is a healthy ecosystem of smaller tools and services that directly work with PREMIS metadata or integrate with it through the powerful extension schema facility.

This chapter describes a subset of the tools that make up the PREMIS ecosystem. We focus on readily available and actively maintained tools. The ecosystem also includes many custom tools developed for use within a single institution, but we do not describe them here. The ecosystem will change over time as new tools are introduced and others fall by the wayside. Nonetheless, it is useful to provide a snapshot of the toolbox for preservationists at the time of the writing in early 2016.

These tools can be placed into one of three categories:

C. Chou (✉)
Florida Virtual Campus, 5830 NW 39th Ave, Gainesville, FL 32606, USA
e-mail: cchou@ufl.edu

A. Goethals · J. Seifert
Harvard Library, 90 Mt. Auburn, Cambridge, MA 02138, USA
e-mail: andrea_goethals@harvard.edu

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_15

1. Tools that generate or process PREMIS metadata directly
2. Tools that generate metadata that can be embedded in PREMIS as extension schemas
3. Processing or repository software that have a great deal of functionality and generate PREMIS metadata either internally or as an export format

In addition, there are also many tools, software libraries and services that can output non-format-specific metadata that could provide values for PREMIS semantic units using other tools or stylesheets. For example, there are many tools that can generate identifiers (e.g., EZID [2] or NOID [3]), message digests (e.g., md5sum [4], MD5Summer [5] or Hash Generator [6]), file size (e.g., ls or du Unix commands), and format information (e.g., DROID [7] or file [8]). In some cases, this software is used within the tools described in this chapter to determine these metadata elements.

15.2 Tools that Generate or Process PREMIS Directly

This section describes open-source stand-alone tools that facilitate PREMIS implementation either by directly outputting data in PREMIS XML or by processing PREMIS XML data directly. They are often integrated into repository software to support PREMIS implementations.

15.2.1 DAITSS Description Service

The DAITSS [9] Description Service performs format identification, validation, and characterization on a digital object. It uses DROID [7] for format identification and JHOVE [10] for format validation and characterization. The result is a PREMIS XML document that includes a PREMIS Object holding the identification and characterization results, a PREMIS Event describing the validation result and any applicable anomaly, and a PREMIS agent identifying the software agent that generated the PREMIS output.

In addition to using the PREMIS schema, the Description Service embeds metadata standard schemas such as MIX [11], TextMD [12], AES Audio [13] and documentMD [14] in the objectCharacteristicExtension section of the PREMIS Object. To reduce redundant format information, the Description Service consolidates format registry information from DROID with the information returned from JHOVE; it uses the PRONOM format registry [15] identifier as the primary format identifier.

The DAITSS Description Service was developed as part of the DAITSS repository software for the Florida Digital Archive.

Maintainer: Florida Virtual Campus
Main website: http://description.fcla.edu[1]
License: GNU General Public License version 3
Source code: https://github.com/daitss/describe

15.2.2 PREMIS in METS (PIM) Toolbox

The PREMIS in METS (PIM) Toolbox contains a set of tools to support the implementation of PREMIS in the METS container format. This is a widely used combination in digital repository environments. PIM is able to:

1. validate against the PREMIS-in-METS best practice [16];
2. convert a PREMIS document into PREMIS-in-METS and vice versa;
3. generate a PREMIS document via the DAITSS Description Service.

Documents may be converted or validated by providing a URI for the file, uploading the file, or by directly inputting the content into a web form. In addition, a Schematron [17] schema can be downloaded to validate that the metadata conforms to the PREMIS-in-METS guidelines in the local environment.

PIM was originally created by the Florida Center for Library Automation for the Library of Congress in 2009.

Maintainer: Florida Virtual Campus
Main website: http://pim.fcla.edu
License: GNU General Public License version 3
Source code: https://github.com/fcla/pim

15.2.3 PREMIS Event Service

The PREMIS Event Service [18, 19] enables any software system to send PREMIS-formatted events to a shared service using a simple API to be stored and subsequently retrieved or analyzed. It uses the PREMIS data model to provide for events and agents due to its semantic clarity and widespread use in digital library systems. The service is typically installed within an institution to support event logging for digital repository workflows. For example, when a digital object is processed during an archiving workflow (i.e., ingest, deletion, fixity, and replication) the repository sends a PREMIS Event to the service detailing the result of the processing. The service then parses the PREMIS Event into an internal database for future aggregation and analysis. For more information, see Chap. 17 of this book.

[1]As of October 2016, DAITSS description service code has been upgraded to PREMIS 3.0, but https://description.fcla.edu not yet.

Maintainer: University of North Texas Libraries
Main website: http://premis-event-service.readthedocs.org/en/latest/
License: See license at https://github.com/unt-libraries/django-premis-event-service/blob/master/LICENSE
Source code: https://github.com/unt-libraries/django-premis-event-service

15.2.4 Premiser

Premiser is a set of bash shell scripts and stylesheets to convert output from ffprobe [20] and MediaInfo [21] into PREMIS XML. It also provides an interface for adding agents and events to an existing PREMIS document.

Maintainer: Dave Rice
Main website: https://github.com/bavc/premisers
License: unknown
Source code: https://github.com/bavc/premisers

15.3 Tools that Generate PREMIS Extension Schemas

PREMIS support a wide range of standard (or even custom) technical metadata through the use of extension schemas [1]. This mechanism enables implementations to place any metadata into a container, wrap it, and place it in the appropriate location within a PREMIS metadata document. Extension schemas are widely used. Common examples include technical metadata schemas such as MIX [11] for images and TextMD [12] for text files.

The tools described in this section readily produce format-specific technical metadata in at least one XML format that easily extends the technical metadata within PREMIS using the premis:objectCharacteristicsExtension element.

15.3.1 Tika

Tika is a software toolkit maintained by the Apache Software Foundation that can detect document types and output text or metadata embedded in files. The default output format is XHTML, but it can also output to XMP. It supports a variety of text, document, image, scripting, and container formats, as well as a few audio and video formats. It can be invoked using the API, as a command-line utility, via a GUI or in server mode.

Command-line examples:

java -jar tika-app-1.5.jar test.doc (to output in XHTML format)
java -jar tika-app-1.5.jar - -xmp test.doc (to output in XMP format)

Maintainer: Apache Software Foundation
Main website: http://tika.apache.org/
License: Apache Software License, Version 2
Source code: https://tika.apache.org/source-repository.html

15.3.2 ExifTool

ExifTool is a Perl library and command-line application that provides functionality for extracting, modifying, and manipulating image, audio, and video metadata. It supports over a hundred different file formats. The metadata can be output in a variety of formats such as RDF/XML, XMP, ICC/ICM, MIE, VRD and EXIF, or even to custom formats. EXIF is the Exchangeable Image File Format and provides a method for embedding metadata in image, audio and video files.

Command-line examples:

exiftool -xmp -b a.jpg > out.xmp (to extract complete XMP data and write it to a file)
exiftool -X db.xslx (to extract metadata in RDF/XML format)

Maintainer: Phil Harvey
Main website: http://www.sno.phy.queensu.ca/~phil/exiftool/
License: Perl Licensing
Source code: http://www.sno.phy.queensu.ca/~phil/exiftool/

15.3.3 File Information Tool Set (FITS)

FITS [22] is a tool that can extract and output format-specific technical metadata. It provides a wrapper for third-party tools (e.g., Apache Tika [23], DROID [7], JHOVE [10], file [8], MediaInfo [21]). It provides a uniform way to invoke a wide range of different tools and can also output extracted metadata in a consistent FITS XML format as well as the native output produced by each individual tool. It can also output the metadata in several community standard XML schemas (e.g. AES Audio Object [13], documentMD [14], MIX [11], TextMD [12]) depending on the format of the file object being analyzed. FITS can be invoked using the Java API or as a command-line utility and can be run against a file or directory of files.

Command-line examples:

fits.bat -i image.jpg -xc (to output MIX image metadata and the FITS XML output)
fits.bat -i text.txt -x -o metadata.xml (to output TextMD text metadata to a file)

Maintainer: Harvard Library
Main website: http://fitstool.org
License: GNU Library General Public License, version 3.0
Source code: https://github.com/harvard-lts/fits

15.3.4 JHOVE

JHOVE is a tool written in Java that can identify and validate formats and extract metadata for approximately a dozen formats and variations of these formats. It can be invoked using the API, as a command-line utility or using a GUI. It can output in a JHOVE-specific XML format or for images in MIX within the JHOVE-specific XML. It can be configured to output TextMD for text formats.

Command-line examples:

jhove -c conf/jhove.conf -m PDF-hul -h xml etd.pdf (to output XML for a PDF)
jhove -c conf/jhove.conf -m TIFF-hul -h xml test.tif (to output MIX in XML for an image)

Maintainer: Open Preservation Foundation
Main website: jhove.openpreservation.org and http://openpreservation.org/technology/products/jhove/
License: GNU Library General Public License, version 3.0
Source code: https://github.com/openpreserve/jhove

15.3.5 MediaInfo

MediaInfo is a tool that can identify formats and extract file- and track-level embedded metadata, for many audio and video formats. It is able to output the metadata in XML format. The tool can be invoked using either a GUI or as a command-line utility.

Command-line examples:

mediainfo testfile.wmv --Output = XML (to output metadata in XML format)
mediainfo testfile.wmv --Output = XML --Language = RAW -f (to output full metadata, non-translated in XML format)

Maintainer: Jerome Martinez, MediaArea.net
Main website: http://mediaarea.net/en/MediaInfo
License: MediaInfo(Lib) License
Source code: http://mediaarea.net/en-us/MediaInfo/Download/Source

15.3.6 PBCore Instantiationizer

PBCore Instantiationizer transforms MediaInfo's [19] XML format into the PBCore format [24] developed by the US public broadcasting community to represent audio and video metadata. The transformation is defined using an XML stylesheet, so it can be performed using a command-line tools such as xsltproc. On a Mac, the PBCore Instantiationizer 1.2 Toolset can be easily launched to perform the task.

Command-line examples (adapted from instructions on the AVPreserve website [25]):

xsltproc mediainfo2pbcoreinstantiation.xsl mediainfo.xml > pbcore.xml (to output PBCore metadata from XML output previously output from MediaInfo)
mediainfo --Language = RAW -f --Output = XML file.mov > mediainfo.xml && xsltproc mediainfo2pbcoreinstantiation.xsl mediainfo.xml > pbcore.xml (to output PBcore using MediaInfo in one step)

Maintainer: AVPreserve
Main website: http://www.avpreserve.com/pbcore-instantiationizer/
License: unknown
Source code: MediaInfo to PBCore conversion stylesheet
https://github.com/avpreserve

15.4 Processing or Repository Software that Supports PREMIS

This section describes various feature-filled software packages that either store metadata internally in the PREMIS format or are able to export metadata in the PREMIS format. Most of these examples are digital preservation repository platforms, with BitCurator and Islandora as exceptions.

Data for this section was gathered in 2014 by directly contacting software maintainers. It was reconfirmed and updated in October 2015.

15.4.1 Archivematica

Archivematica is a free and open-source digital preservation system that is designed to maintain standards-based, long-term access to collections of digital objects. It produces PREMIS version 2.2 [26] in METS XML [27], which is stored in the AIP and exported in the DIP. It supports all of the object, agent, event and rights PREMIS entities. Within these entities, it supports all of the mandatory elements and many of the optional elements.

Archivematica uses Python scripts and commands included with its workflow engine to generate PREMIS elements. Prior to AIP storage, users can choose to run the validator component of the PREMIS in METS (PIM) toolbox [28].

Archivematica can be used in conjunction with AtoM (Access to Memory) [29], an open-source archival description tool that implements PREMIS rights and restrictions as actionable statements that affect how digital objects are accessed. Archivematica integrates with DSpace [30], Islandora [31] and ArchivesSpace [32] among other systems that use PREMIS. For more information, see Chap. 16 of this book

Maintainer: Artefactual Systems, Inc.
Main website: https://www.archivematica.org
License: GNU Affero General Public License version 3
Source code: https://github.com/Artefactual/archivematica

15.4.2 BitCurator

BitCurator is an open-source suite of digital forensic and file analysis tools intended to support archival or digital preservation use cases. Its functionality comprises pre-imaging data triage, forensic disk imaging, file system analysis and reporting, identification of private and individually identifying information, and export of technical and other metadata. It has some ability to output metadata in the PREMIS version 2.0 format [33].

The PREMIS output is not validated by BitCurator and may not validate in some situations. It supports the object and event PREMIS entities (not agent or rights). It generates an object instance for each forensically packaged image, and events corresponding to operations performed using BitCurator such as disk image capture. For more information, see Chap. 8 of this book.

Maintainer: BitCurator Consortium
Main website: http://www.bitcurator.net/
License: GNU General Public License version 3
Source code: https://www.github.com/kamwoods/bitcurator

15.4.3 DAITSS

DAITSS is open-source repository software developed for the Florida Digital Archive. It uses PREMIS 2.2 [26] with METS [27]. It creates PREMIS Object, event, and agent instances but does not use PREMIS rights. DAITSS automatically harvests the metadata in the submitted SIPs and uses the DAITSS Description Service to extract the metadata in the digital objects to create the PREMIS Object. The identified format for the digital object is then checked against a format action plan service, DAITSS Action Plan Service, to determine if there is any format migration or normalization needed to be performed for the format and if so, the digital object is converted by the DAITSS Transformation Service. All incoming SIPs[2] are automatically validated and virus checked and the produced AIPs[1] are also validated against the PREMIS schema.

DAITSS also supports SIPs conforming to the TIPR (Towards Interoperable Preservation Repositories) specification. PREMIS metadata in the TIPR packages would be extracted when they are submitted and archived into DAITSS. DAITSS has recently been upgraded to support PREMIS 3.0.

Maintainer: Florida Virtual Campus (originally developed by the Florida Center for Library Automation)

Main websites:

DAITSS: http://daitss.fcla.edu/

Florida Digital Archive: http://fclaweb.fcla.edu/FDA

TIPR: http://wiki.fcla.edu/TIPR

DAITSS Action Plan Service: http://actionplan.fcla.edu

License: GNU General Public License version 3

Source code: https://github.com/daitss/

15.4.4 DSpace

DSpace is open-source repository software that can import and export PREMIS version 1.0 [34] in METS XML. Beginning with version 1.7, DSpace can import and export AIPs in PREMIS-in-METS format, originally created to integrate with DuraCloud. Internally, DSpace does not store metadata in the PREMIS format but when DSpace users export an AIP from DSpace, some of the internal metadata is translated into PREMIS technical metadata about the files in METS wrappers. Of the PREMIS entities, only the object is supported (not agent, event or rights).

DSpace uses a custom PREMISCrosswalk module to read and generate PREMIS metadata. Although it is not packaged to be used independently from DSpace, the code is open and included in the DSpace codebase.

[2]For more information about SIPs and AIPs, see Introductory parts.

By default, DSpace validates the entire AIP on export; this includes the PREMIS elements. It can also be configured to validate the AIP on import.

Maintainer: DuraSpace provides stewardship. A volunteer "Committers Team" perform most development and maintenance.
Main website: http://www.dspace.org
License: DSpace Source Code License
Source code: https://github.com/DSpace/DSpace/
Additional PREMIS-related information:

DSpace AIP format
https://wiki.duraspace.org/display/DSDOC4x/DSpace+AIP+Format
Generating AIPs
https://wiki.duraspace.org/display/DSDOC4x/AIP+Backup+and+Restore
PREMIS Crosswalk
https://github.com/DSpace/DSpace/blob/master/dspace-api/src/main/java/org/dspace/content/crosswalk/PREMISCrosswalk.java

15.4.5 Fedora

Fedora [35] is an open-source repository system for the management and dissemination of digital content. Fedora supports PREMIS for a subset of the elements of the event entity: *eventDateTime*, *eventRelatedAgent*, *eventRelatedObject*, *eventType*, and *eventOutcome*. It also supports the object elements *messageDigest*, *size* and *fixity*. It uses PREMIS version 1.0 [34]. Additionally, Fedora has created two tools for creating PREMIS Events—the Fedora Audit Service and fcrepo-audit.

Maintenance organization or individual(s): DuraSpace
Main URL: http://fedorarepository.org
License: Apache License, Version 2.0
Source code: https://github.com/fcrepo4/fcrepo4

15.4.6 Islandora

Islandora is an open-source digital asset management system for institutions to collaborate, manage, and discover their digital assets. It is built on top of Fedora Commons and Drupal. Islandora supports all PREMIS entities including object, agent, event, and rights.

Internally, Islandora uses an XSLT stylesheet to convert Fedora's [35] XML (FOXML) into PREMIS, which can then be converted into HTML for human viewing using another XSLT stylesheet. Those two stylesheets, as well as the code,

are available under a GPL license. Islandora currently only records fixity events in PREMIS. An enhancement to add more PREMIS Events is currently under consideration. For more information, see Chap. 16 of this book.

Maintainer: Islandora Foundation
Main website: http://islandora.ca/
License: GNU General Public License version 3
Source code: https://github.com/Islandora/islandora_premis

15.4.7 Preservica

Preservica is a commercial digital repository system that can be hosted both on-site and in the cloud. It uses DROID [7] to identify the file formats to be ingested into the repository and checks with the PRONOM technical registry [15] to determine preservation actions needed to be performed on the digital object based on their formats, for example, migration or emulation.

It also uses several proprietary software tools and JHOVE [10] software to validate and characterize digital objects. Preservica uses PREMIS with METS and supports all PREMIS entities including object, agent, event and rights. Out-of-the-box, it currently supports PREMIS schema 2.2 [26]. Preservica is planning to support PREMIS 3.0.

Maintainer: Preservica Ltd.
Main URL: http://www.preservica.com
License: Commercial license
Source code: NA

15.4.8 RODA

RODA is open-source digital repository software that uses PREMIS version 2.2 [26] extensively. It records PREMIS metadata on all actions performed by the repository. It can import all of the PREMIS entities and semantic units. It natively generates all of the PREMIS entities except rights.

To generate PREMIS metadata, the RODA v1.2 uses JHOVE [10] and DROID [7]. Future versions may integrate FITS [22] and thus include additional tools such as FIDO [36] and Apache POI [37]. RODA provides integration points to enable further processing of the output from any of the integrated tools. Any PREMIS metadata that is submitted as a SIP is validated using the PREMIS schema.

Several PREMIS-related enhancements under development including a Report API implemented as an OAI-PMH [38] provider that allows external programs to harvest a repository's PREMIS Events. This enables the preservation

watch systems, such as Scout, to monitor events and provide alerts when they do not meet defined quality levels. These development were supported by the SCAPE [39] project.

Maintainer: KEEP SOLUTIONS http://www.keep-solutions.com
Main website: http://www.roda-community.org
License: GNU Lesser General Public License version 3
Source code: https://github.com/keeps/roda
Additional PREMIS-related information

Report API (PDF): https://github.com/openplanets/scape-apis/blob/master/ReportAPI_V1.0.pdf

15.4.9 Rosetta

Rosetta [40] is a commercial digital preservation system created by Ex Libris. Rosetta implements the PREMIS reference model using METS [27], and it uses PREMIS 2.1 [41]. Additionally, the AIP data model in Rosetta is based on the PREMIS reference model's four levels of objects: *Intellectual Entity*, *Representation*, *File,* and *Bitstreams*. However, there are some differences in how PREMIS and Rosetta each define these entities. In Rosetta, the agent is only an attribute of an external provenance event, since in the other areas, Rosetta is the agent associated with events in the life of the objects and the rights attached to the *Intellectual Entity*. Additionally, according to PREMIS, an *Intellectual Entity* is a set of content that is a single intellectual unit, but is not an object itself. In the Rosetta data model, an *Intellectual Entity* is considered an object and has unique metadata.

Most of the tools Rosetta uses are based on internal code. Additionally, customers can include PREMIS metadata directly in the deposited SIP and can use Rosetta's API to generate specific events.

The company, along with the Rosetta Advisory Group, is currently in the process of determining the scope and timeline for adding PREMIS 3 enhancements to Rosetta.

Maintenance organization or individual(s): Ex Libris
Main URL: http://www.exlibrisgroup.com/category/RosettaOverview
License: Commercial license
Source code: Not available.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata, version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 29 Dec 2015
2. California Digital Library (2015) EZID. http://ezid.cdlib.org. Accessed 28 Oct 2015
3. University of California Curation Center (2015) NOID (nice opaque identifier). https://wiki.ucop.edu/display/Curation/NOID. Accessed 28 Oct 2015
4. md5sum (nd) https://en.wikipedia.org/wiki/Md5sum. Accessed 14 Feb 2016
5. Pascoe L (2015) MD5Summer [Software]. http://www.md5summer.org/. Accessed 28 Oct 2015
6. SecurityXploded (2015) Hash generator [software]. http://securityxploded.com/hashgenerator.php. Accessed 28 Oct 2015
7. The National Archives (2015) Droid (Digital Record Object Identification) [software]. http://www.nationalarchives.gov.uk/information-management/manage-information/policy-process/digital-continuity/file-profiling-tool-droid/. Accessed 28 Oct 2015
8. Zoulas C, Darwin I (2003) File (version 4.00) [software]
9. Florida Virtual Campus (nd) DAITSS. https://github.com/daitss/describe. Accessed 14 Feb 2016
10. Open Preservation Foundation (2015) JHOVE [software]. http://openpreservation.org/technology/products/jhove/. Accessed 28 Oct 2015
11. Library of Congress (2015) MIX (metadata for images in XML standard). http://www.loc.gov/standards/mix/. Accessed 28 Oct 2015
12. Library of Congress (2015) TextMD (technical metadata for text). http://www.loc.gov/standards/textMD/. Accessed 28 Oct 2015
13. Audio Engineering Society (2011) AES standard for audio metadata—audio object structures for preservation and restoration
14. Florida Virtual Campus and Harvard Library (2012) Document metadata: document technical metadata for digital preservation. http://library.harvard.edu/preservation/digital-preservation_current-projects.html. Accessed 28 Oct 2015
15. The National Archives (nd) The technical registry PRONOM. http://www.nationalarchives.gov.uk/PRONOM/. Accessed 21 Feb 2016
16. Guenther R, Wolfe R et al (2008) Guidelines for using PREMIS with METS for exchange. http://www.loc.gov/standards/premis/guidelines-premismets.pdf. Accessed 28 Oct 2015
17. Wikimedia Foundation (2015) Schematron. https://en.wikipedia.org/wiki/Schematron. Accessed 14 Feb 2016
18. Schultz M, Phillips M, Eisenhauer S, Krabbenhoeft N (2014) Building institutional capacity in digital preservation. University of North Texas Digital Library. http://digital.library.unt.edu/ark:/67531/metadc181668/. Accessed 28 Mar 2014
19. Phillips M, Schultz M, Nordstrom K (2014) PREMIS event service, University of North Texas Digital Service. http://digital.library.unt.edu/ark:/67531/metadc40413/m1/1/. Accessed 28 Mar 2014
20. FFmpeg developers (2015) ffprobe [software]. https://ffmpeg.org/ffprobe.html. Accessed 28 Oct 2015
21. MediaArea.net SARL (2015) MediaInfo [Software]. https://mediaarea.net/en/MediaInfo. Accessed 28 Oct 2015
22. Harvard Library (2015) FITS (file information toolset) [software]. http://www.fitstool.org. Accessed 28 Oct 2015
23. Apache Software Foundation (2016) Apache Tika—a content analysis tool [software]. http://tika.apache.org. Accessed 21 Feb 2016
24. Pbcore Advisory Sub-Committee (nd) Public broadcasting metadata dictionary project (PBCore). http://pbcore.org. Accessed 21 Feb 2016
25. AVPreserve (2015) https://www.avpreserve.com/. Accessed 28 Oct 2015

26. PREMIS Editorial Committee (2012) PREMIS Data dictionary for preservation metadata version 2.2. http://www.loc.gov/standards/premis/v2/premis-2-2.pdf. Accessed 05 Jan 2016
27. Library of Congress, METS Editorial Board (2015) Metadata Encoding and Transmission Standard (METS): official web site. http://www.loc.gov/standards/mets. Accessed 05 Jan 2016
28. Florida Virtual Campus (nd) PREMIS IN METS toolbox [software]. http://pim.fcla.edu. Accessed 17 Feb 2016
29. Artefactual (2016) AtoM [software]. https://www.Artefactual.com/services/atom-2/. Accessed 17 Feb 2016
30. Duraspace (2016) DSpace. http://www.dspace.org
31. Islandora Foundation (2016) Islandora [software]. http://islandora.ca/. Accessed 21 Feb 2016
32. ArchiveSpace Governance Board (nd) ArchiveSpace. http://archivesspace.org. Accessed 21 Feb 2016
33. PREMIS Editorial Committee (2008) PREMIS data dictionary for preservation metadata version 2.0. http://www.loc.gov/standards/premis/v2/premis-2-0.pdf. Accessed 05 Jan 2016
34. PREMIS Working Group (2005) Data dictionary for preservation metadata version 1.0. http://www.loc.gov/standards/premis/v1/premis-dd_1.0_2005_May.pdf. Accessed 05 Jan 2016
35. DuraSpace (2015) Fedora [software]. http://www.fedora-commons.org/. Accessed 28 Oct 2015
36. Farquhar A, de Rooij M (2013) FIDO (format identification for digital objects) [software]. http://openpreservation.org/technology/products/fido/. Accessed 28 Oct 2015
37. Apache Software Foundation (2015) Apache POI [software]. https://poi.apache.org/. Accessed 28 Oct 2015
38. Open Archives Initiative (nd) Protocol for metadata harvesting (OAI-PMH). https://www.openarchives.org/pmh. Accessed 21 Feb 2016
39. SCAPE (Scalable Preservation Environments) (2014) http://www.scape-project.eu/. Accessed 28 Oct 2015
40. French A (2012) Rosetta | PREMIS: an overview. Paper presented at iPRES, Toronto, Canada, 2 October 2012. http://www.loc.gov/standards/premis/pif-presentations-2012/RosettaPREMIS.pdf. Accessed 28 Mar 2014
41. PREMIS Editorial Committee (2011) PREMIS data dictionary for preservation metadata version 2.1. http://www.loc.gov/standards/premis/v2/premis-2-1.pdf. Accessed 05 Jan 2016

Chapter 16
PREMIS in Open-Source Software: Islandora and Archivematica

Mark Jordan and Evelyn McLellan

16.1 Introduction

Open-source software is software whose source code is made freely available for use, modification, and redistribution. Although there are many different models for developing and sustaining open-source tools, the tools are often developed in a collaborative and open environment, with development, technical, and user documentation made available online and community adoption supported by public discussion lists and user groups. The last few years have seen a considerable increase in the number of open-source software tools that have been or are being developed for use by archives and libraries. These tools provide a broad range of functionalities required for digital preservation, management of digital objects within a repository, cataloging or archival description, and provision of online access. Many of these tools are now implementing all or part of the PREMIS Data Dictionary to record detailed technical, preservation, provenance, and rights information about digital holdings. This chapter describes implementations by two software tools, Islandora [1] and Archivematica [2], providing practical examples of how PREMIS can be used to support the ability of archives and libraries to preserve digital holdings and make them accessible over time.

M. Jordan (✉)
W.A.C. Bennett Library, Simon Fraser University, 8888 University Drive, Burnaby, BC V5A 1S6, Canada
e-mail: mjordan@sfu.ca

E. McLellan
Artefactual Systems Inc., Suite 201–301 6th Street, New Westminster, BC, Canada
e-mail: evelyn@artefactual.com

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_16

16.2 PREMIS in Islandora

16.2.1 Overview of Islandora

Islandora integrates Fedora Commons [3] (for low-level repository services), Drupal [4] (for user-facing website and workflow), and SolR [5] (for text indexing and searching) into a general-purpose repository platform that has been deployed at over one hundred sites around the world, providing a wide variety of repository applications and services from simple image collections to platforms for writing critical editions [6]. Islandora supports a wide variety of content through the use of "solution packs," which are Drupal modules and accompanying tools that handle ingestion, derivative creation, and display of content. Solution packs also provide default metadata creation and maintenance forms tailored to the content type. Supported content types include images, video, audio, PDFs, paged content such as books and newspapers, Web archives, and spreadsheets; adding new content types is relatively easy.

Islandora currently (January 2016) uses version 3.x of the Fedora Commons repository platform, but work is underway to rewrite Islandora so that it can use Fedora 4.x. This redevelopment will have a direct impact on Islandora's PREMIS implementation, as detailed in the "Future Directions" section below.

16.2.2 Islandora's Digital Preservation Functionality

Islandora performs a variety of preservation functions that can be expressed within PREMIS, and it can also integrate with other systems that provide additional preservation functionality. These preservation functions fall into three categories: file format normalization, preservation-specific added value tasks, and integration with dedicated digital preservation platforms.

First, several of the content type specific solution packs mentioned above normalize files that users add to Islandora either through a web form or via a batch loading tool. For example, if a user uploads a PDF file, Islandora will create a PDF/A derivative of the file and store it in addition to the original (Islandora always retains the originally uploaded file, regardless of what format it is). Most solution packs normalize files into web-friendly formats for presentation to end users, whereas only a few normalize into preservation-friendly formats. Work is underway to explore how Islandora might leverage normalization policies defined in the Archivematica Format Policy Registry, a shared database of policies, commands and tools for format identification, normalization, metadata extraction, and other preservation processes [7].

Second, Islandora, like Drupal, is highly modular. Most of its preservation-specific functionality is encapsulated in one or more task-specific modules:

- Integration with the File Information Toolset (FITS)[1]: when a user uploads a file, or a file is ingested via a batch loader, FITS is run against the file and its output is stored along with the original file. FITS validates a file's format and extracts various types of information used to characterize the file.
- Enabling of checksum generation on files, and periodic auditing of checksum integrity: Fedora Commons can generate checksums on files for the system to perform fixity check, which indicates whether a file has been tampered with; the Islandora Checksum and accompanying Checksum Checker modules allow Islandora administrators to enable checksum generation and to configure how often regular file integrity audits (i.e., fixity checks) are performed. If an audit detects a file integrity problem, the discrepancy is logged and the administrator is notified.
- Generation of Bags (content packaged in the Library of Congress BagIt format [8]): individual Islandora objects, and entire Islandora collections, can be exported as Bags. The module that performs this functionality provides a wide range of options for creating these Bags. BagIt is widely used throughout the digital preservation community as a trustworthy and flexible container for sets of related files.
- Integration with storage platforms: A number of Islandora modules exist that will store data in cloud storage platforms such as CloudSync, and DuraCloud. Work is underway to integrate Islandora with LOCKSS networks as well [9].

Third, Islandora can integrate with other applications as part of preservation workflows. The best example of this is the Archidora plugin, a module which automatically moves Islandora content into the Archivematica digital preservation platform [10] (Fig. 16.1).

An institution may choose to enable and configure functionality from these three categories to operationalize their digital preservation policies and strategies. Some, but not all, of the outcomes of this functionality is captured by the Islandora PREMIS module.

16.2.3 The Islandora PREMIS Module

The Islandora PREMIS module produces XML and HTML serializations of a subset of the PREMIS Data Dictionary version 2.2 for objects in an Islandora repository. It documents all fixity checks performed on content files, includes agent entries for the institution and for the Fedora Commons software and maps contents of each object's rights elements from Dublin Core metadata [11] to PREMIS *rightsExtension* elements. It generates the PREMIS XML on demand, when a user of the repository requests to see the data. The module retrieves data from a variety

[1]For further information on FITS, see Chap. 15.

of sources (some examples are provided in the table below) and converts the data into valid PREMIS XML. The user can then view the XML or download it. The user can choose to view an HTML version of the data as well. Only users with sufficient privileges have access to the PREMIS data; end users of the repository cannot view or download it. It is also possible to generate the PREMIS XML via automated processes; for example, it is possible to include it in the Bags Islandora can generate.

Islandora's ability to achieve a conformant implementation of the PREMIS Data Dictionary is to a large extent determined by the way that Islandora objects are modeled, which in turn is informed heavily by the Fedora Commons (version 3) Digital Object Model [12]. The Fedora Commons data model defines "objects" and "datastreams." Objects are the highest level residents within a Fedora Commons repository and have a persistent identifier, properties, and datastreams. An object's persistent identifier (called a "PID") is unique within the repository. Properties include the creation and last modification date of the object and its label (which acts in many contexts as a simple title). Datastreams contain digital content, and are in many cases equivalent to ordinary files; they also have identifiers, which, when combined with the PID of their parent object, are unique in a repository, and they have properties. Viewed from the perspective of the PREMIS Data Model, Fedora Commons objects often (but not always) encapsulate PREMIS *Intellectual Entities* or *Representations*, and Fedora Commons datastreams are equivalent to PREMIS *Files*.

Fedora objects reserve a small number of datastreams for internal use. One of these is known as the Audit Log, which is an XML file that records specific types of events performed on all datastreams belonging to an object. The Islandora PREMIS module takes most of the data it converts into into PREMIS XML (and then into HTML for human consumption) from the Audit Log. A partial sample of the HTML output, from York University's Islandora repository [13] showing a single fixity check event and agent and rights entities, is shown in Fig. 16.2.

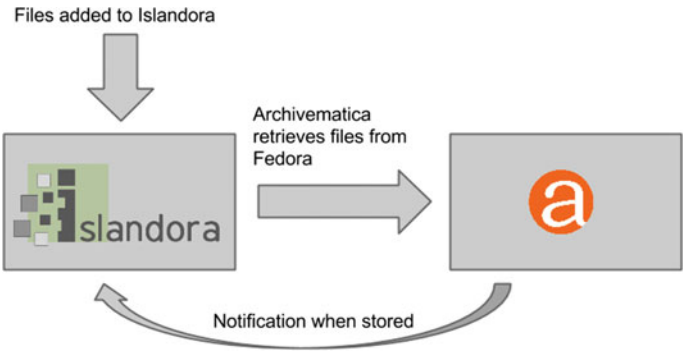
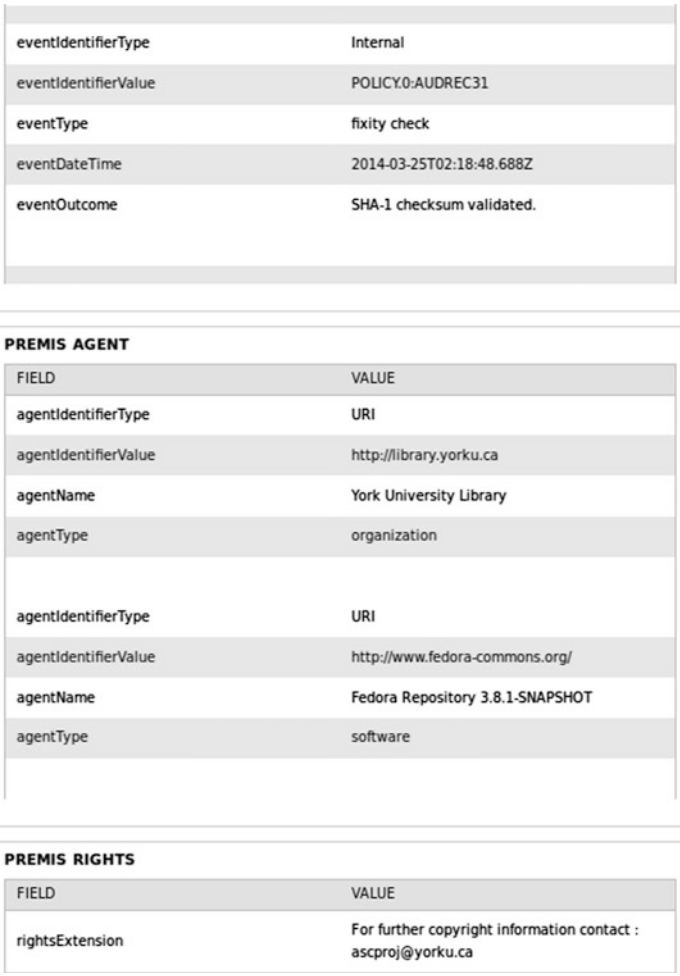


Fig. 16.1 High-level view of Islandora/Archivematica integration. Archivematica ingests content from Islandora and generates AIPs for archival storage



eventIdentifierType	Internal
eventIdentifierValue	POLICY.0:AUDREC31
eventType	fixity check
eventDateTime	2014-03-25T02:18:48.688Z
eventOutcome	SHA-1 checksum validated.

PREMIS AGENT

FIELD	VALUE
agentIdentifierType	URI
agentIdentifierValue	http://library.yorku.ca
agentName	York University Library
agentType	organization
agentIdentifierType	URI
agentIdentifierValue	http://www.fedora-commons.org/
agentName	Fedora Repository 3.8.1-SNAPSHOT
agentType	software

PREMIS RIGHTS

FIELD	VALUE
rightsExtension	For further copyright information contact : ascproj@yorku.ca

Fig. 16.2 Sample HTML output of Islandora PREMIS

In its current version, Islandora PREMIS only takes data documenting fixity checks on datastreams from the Audit Log. However, it also takes rights information from descriptive metadata stored in the datastream containing Dublin Core metadata describing the parent object. The module obtains information that it maps to PREMIS Agent entities from sources that are not datastreams (for example, the Agent information describing the institution is configurable by the Islandora administrator). Table 16.1 documents an illustrative subset of the sources of information that the module maps to PREMIS semantic units.

This selective list of mappings illustrates some important aspects of the data available to the Islandora PREMIS module

Table 16.1 Selected mappings from PREMIS semantic units to sources of data internal to Islandora

PREMIS Semantic Unit	Reference number	Source in Islandora
Object entity		
objectIdentifierValue	1.1.2	Fedora Commons object PID
messageDigestAlgorithm	1.5.2.1	Datastream property foxml:contentDigest/@TYPE
messageDigest	1.5.2.2	Datastream property foxml:contentDigest/@DIGEST
size		Datastream property foxml:datastreamVersion/@SIZE
objectCharacteristicsExtension	1.5.7	Entire FITS XML datastream, if present
Event entity		
eventType	2.2	Templated value "fixity check"
eventDateTime	2.3	Fedora Commons Audit Log entry for fixity check event
eventOutcome	2.5.1	Fedora Commons Audit Log entry for fixity check event
Agent entity		
agentName	3.2	Describing the institution, configured by Islandora administrator; describing the Fedora Commons software, retrieved from software properties
Rights entity		
rightsExtension	4.2	Dublin Core datastream's "rights" element, if present

- Some data are not always present, such as a rights statement. If the Islandora object's Dublin Core metadata does not have anything in its rights element, the PREMIS XML will not contain a *rightsExtension* element.
- Some values are templated, such as the PREMIS *eventType* element. That is, the value is determined by the Islandora PREMIS module in response to a particular type of event.

16.2.4 Future Directions for PREMIS in Islandora

The Islandora community intends to continue developing the PREMIS module in response to existing and new features and capabilities in the Islandora repository platform. A subset of the Islandora Preservation Interest Group [14] is currently investigating how to expand the amount of data converted into PREMIS XML. For example, the PREMIS Event types that could be added based on current capabilities

of Fedora Commons and Islandora include ingestion, normalization, and deletion; event types that could be added based on planned functionality include virus checking. Note that these terms are taken from the Library of Congress Event Types controlled vocabulary [15]; event types that are not members of the vocabulary such as Bag generation could be added as well. In addition, values in the PREMIS *format* semantic unit group (1.5.4) could be populated using PRONOM format IDs taken from datastreams' FITS technical metadata, if present [16].

A significant external factor that will impact the Islandora PREMIS module is the move to supporting Fedora 4, in particular its new Audit Service [17]. The Audit Service will structure and store data documenting events in more flexible ways than Fedora 3's Audit Log does, and will offer standards-based interfaces so that external applications can record events as well. The Islandora community is currently developing a version of its software that will work with Fedora 4, and members of the community are active in planning and developing the Fedora 4 Audit Service. Important design goals of PREMIS support in the new version of Islandora include a migration path that retains metadata used in the current PREMIS 2.2 implementation; the recording of additional metadata in PREMIS; and support for version 3 of the PREMIS Data Dictionary.

16.3 PREMIS in Archivematica

16.3.1 Overview of Archivematica

Archivematica is an OAIS-based free and open-source suite of software tools designed to ingest digital objects and prepare them for long-term preservation [18]. The user interacts with a web-based dashboard to initiate ingest and make decisions about preservation actions. The ingested objects are subjected to a number of preservation "micro-services," or discrete preservation processes, which result in the preparation and storage of Archival Information Packages (AIPs). These packages consist of the original objects, any preservation copies of the objects generated during processing, and technical, preservation and (optionally) descriptive and rights metadata associated with the objects. Because the AIPs are designed to be as self-documenting and system-independent as possible, they rely heavily on recognized standards both for their overall structure and for the metadata they contain. An AIP therefore consists of a simple set of directories, packaged in the Library of Congress BagIt format, which can be opened in any standard file browser. All of the metadata about the AIP contents are contained in a single XML file consisting of PREMIS within METS [19].

16.3.2 Archivematica's PREMIS Implementation

Archivematica's PREMIS implementation is designed to answer in detail questions about the nature, provenance, and preservation of a digital object that is ingested into a repository. What is the object's format? Does it conform to published specifications for that format? When, how, and by whom was it created? What are its technical characteristics? When was it ingested into the digital repository, and what actions were taken on it in the repository and by whom? Was a preservation master of the object created, and if so, how can that master be identified and located? Are there intellectual property or other types of rights associated with the object? Are there restrictions on how the object is preserved or made available? To answer these questions requires an implementation of the PREMIS Data Dictionary that is both highly detailed and well structured.

At the time of this writing, Archivematica implements version 2.2 of the Data Dictionary. The preservation micro-services that Archivematica performs are very much centered on individual files—most commonly text files, office documents, images, and multimedia files. Micro-services such as fixity checks, virus scans, format identification and validation, metadata extraction and format normalization are run on files and the tool outputs recorded at the *File* level. For this reason, Archivematica implements the PREMIS Object entity for the *File* category only, even though PREMIS 2.2 permits users to capture information about *Representations* and *Bitstream* objects in addition to *Files*. Each digital object that is ingested into Archivematica has a corresponding PREMIS Object entity in the METS file. Each Object entity is associated with numerous Events, and each Event has related Agents. Optionally, each Object can also be associated with multiple Rights statements and permissions (see Chap. 12 for details about the Rights Entity implementation in Archivematica).

16.3.3 Describing an Object

Archivematica generates or extracts large volumes of technical information about each ingested digital object, the most basic and essential of which is identification, fixity, file size, composition level (whether or not the object is encrypted or compressed) and format identification. All of this information is captured in the relevant PREMIS semantic units. The information about file format is recorded as a link to the appropriate PRONOM format registry key, with the format name taken from the name in the PRONOM entry. Archivematica also records information about the original filename, if that name had prohibited characters removed during ingest. If a preservation version of the object is made, the links between the original object and the preservation version are recorded in the PREMIS relationship semantic unit. Typical PREMIS Object metadata for an ingested file might therefore look like the following:

objectIdentifierType: UUID [note: stands for unique universal identifier]
objectIdentifierValue: 10a5f06a-6edf-474f-bc27-0b3cb8c8a033
compositionLevel: 0
messageDigestAlgorithm: sha256
messageDigest: 34833a7405974850123afa56de2dfa67f318eb056...
size: 54272
formatName: JPEG File Interchange Format
formatVersion: 1.01
formatRegistryKey: fmt/43
formatRegistryName: PRONOM
objectCharacteristicsExtension: [metadata extraction tool outputs]
originalName: Landing zone
relationshipType: derivation
relationshipSubType: is source of [note: this object is the source for a derivative file that was generated and is itself described as a related object]
relatedObjectIdentifierType: UUID
relatedObjectIdentifierValue: cd0626d4-d962-4392-af60-039bd2d017f1

Archivematica makes extensive use of the PREMIS *objectCharacteristics Extension* semantic unit to capture detailed outputs from format identification, validation, and metadata extraction tools that are run during ingest. These tool outputs are used to supplement the summary information that is captured in the elements described above, and contain a wealth of information on an object's technical characteristics. The screenshot in Fig. 16.3 shows a snapshot of some of these outputs for a JPEG file:

The format identification and extracted metadata in the Archivematica PREMIS file are designed to deliver, in a standardized way, information needed to assist repositories to make informed decisions about actions required to ensure that the ingested objects can be read and rendered authentically over time. The fixity information ensures that there will be a means to objectively determine whether or not a digital object has been altered in some way, either intentionally or accidentally. The information in the Object entity section in the Archivematica METS file is therefore essential for long-term preservation of ingested digital objects. However, it is not the whole story.

16.3.4 What Has Been Done to the Object?

The PREMIS Event entity is designed to provide a structured way of recording what actions a repository has taken to preserve a digital object it has ingested. In determining the types of events to capture, the Archivematica team has made extensive use of the controlled vocabulary examples provided in the PREMIS Data Dictionary and other terms included in the Library of Congress *eventType*

Fig. 16.3 Excerpt of tool output to be used in *objectCharacteristicsExtension* field

vocabulary [15]. Using commonly accepted terms for standard preservation actions encourages data consistency across repositories and supports an emerging community consensus on what constitutes core preservation functionality. When there is no suitable term in the controlled vocabulary list, Archivematica's developers submit suggestions to the PREMIS Editorial Committee for possible inclusion in future versions of the vocabulary.

The PREMIS Events in a typical Archivematica AIP METS file include ingestion, fixity check, message digest calculation, virus check, format identification, validation, and normalization. For objects that have prohibited characters removed from their filenames, a name cleanup event is added; an unpacking event is included for objects that have been extracted from a zipped directory or other type of package. When a file is normalized, a new digital object is generated, and a creation event is captured for that new object.

The purpose of creating a PREMIS Event is to provide an audit trail of the actions taken against the digital object, to provide information on when, how, and by whom the action was taken, and to indicate the outcome or output of the action. A typical Event might therefore look like the following:

eventType: fixity
eventDateTime: 2014-07-17T21:20:07
eventDetail: program="python"; module="hashlib.sha256()"
eventOutcome: Pass
eventOutcomeDetailNote: 99ea9022d2cbb615d0d2b47eeeb44355d12234cb680[…] verified
linkingAgentIdentifier: [links to Agents]

Note that each Event has three associated Agents—the software system and version (for example, Archivematica 1.4); the logged-in user who initiates or approves digital preservation actions; and the name of the organization responsible for ingest. The name of the organization is configured when Archivematica is installed, and all Agents are generated automatically and linked to their associated Events during ingest and processing.

16.3.5 Packaging It All Up

The METS file in Archivematica includes all of the PREMIS metadata described above, packaged in accordance with the guidelines issued by the PREMIS Editorial Committee in 2008 [20]. The METS file acts as a container for the PREMIS elements and for non-PREMIS descriptive metadata such as Dublin Core, and also uses METS-specific elements and attributes to provide information about the overall structure of the AIP. The METS fileSec and structMap sections can be read independently of the PREMIS elements to provide a manifest of all the digital objects in the AIP and to describe the relationships between them. For example, the fileSec can show whether the AIP includes not just ingested digital objects but also related OCR text files, license files, and submission documentation such as donor agreements or transfer forms. The structMap can be used to indicate physical and logical relationships between objects; for example, if the original Submission Information Package consisted of a set of nested directories, the structMap can map out the hierarchy and can be used to assign archival levels of description to the hierarchy when access copies of the objects are uploaded to an access system. The METS therefore serves as a vehicle for data exchange and AIP documentation, while the PREMIS elements provide extensive and highly detailed technical and provenance information required to plan and implement ongoing preservation actions on the ingested objects.

16.4 PREMIS in Open-Source Software and Services

The number and diversity of open-source tools for digital preservation and access has grown dramatically over the last few years, and shared digital preservation services using open-source tools have been emerging and becoming more robust at

the same time. This chapter has focused on PREMIS implementations in Islandora and Archivematica, but a number of other tools and services also implement PREMIS to some degree. Examples of digital preservation tools implementing PREMIS, such as DAITSS [21], a digital preservation system in use at the Florida Digital Archive, and RODA [22], a digital preservation and repository system developed in Portugal, both have highly detailed PREMIS implementations. The DSpace [23] digital repository platform stores Archival Information Packages with PREMIS metadata wrapped in METS; DSpace implements the PREMIS Object entity only (i.e., no Events, Agents, or Rights). BitCurator [24], a suite of tools used for forensic disk preservation, generates PREMIS Object, Event and Agent entities for each data forensic tool that is used to process a disk image. ArchivesSpace [25], a tool used to generate EAD finding aids for archival repositories, includes a PREMIS Rights implementation, as does AtoM [26], a tool used for archival description and provision of online access to digital objects. An example of a shared online service that features a robust PREMIS implementation is HathiTrust [27], a consortium of research libraries that manages preservation and access services for images and text files.[2]

The proliferation of PREMIS implementations across open-source software tools and shared platforms has a number of benefits. Such implementations provide a means for archives and libraries to use PREMIS without having to develop local expertise and in-house templates and tools. This means that PREMIS can be introduced to the broadest possible audience, including institutions with few technical resources and limited budgets. Using PREMIS improves the ability of the software tools to meet functional requirements associated with preserving and providing access to digital content; the detailed Object entity elements generated by Archivematica, for example, support an institution's ability to undertake format normalization and emulation actions on ingested objects. The use of PREMIS also provides a means for content to be exchanged between systems, or for tools to be integrated with one another to enhance and supplement each others' functionalities. It is not difficult to imagine circumstances in which several of the tools mentioned above are used by the same institution to accomplish complex workflows, with metadata being exchanged across or referenced by different tools, or aggregated to provide a more complete body of information about a digital object. The combination of freely available tools and standardized, exchangeable metadata thus holds great potential for enhancing the ability of institutions to preserve and provide access to diverse bodies of digital holdings.

[2]Examples of digital preservation tools implementing PREMIS, such as DAITSS, RODA, DSpace are mentioned in Chap. 15. A Bitcurator introduction is provided in Chap. 8.

References

1. Islandora website: http://islandora.ca/. Accessed 16 Jan 2016
2. Artefactual Systems. Archivematica website: http://archivematica.org/. Accessed 16 Jan 2016
3. Fedora website: http://fedorarepository.org/. Accessed 07 Feb 2016
4. Drupal website: https://www.drupal.org/. Accessed 07 Feb 2016
5. Solr website: http://lucene.apache.org/solr/. Accessed 07 Feb 2016
6. Islandora Foundation Islandora website. http://islandora.ca/. See also http://islandora.ca/islandora-installations. Accessed 05 Jan 2016
7. Artefactual Systems (2015) Format policy registry (FPR). https://www.archivematica.org/en/docs/fpr. Accessed 05 Jan 2016
8. The BagIt File Packaging Format: https://tools.ietf.org/html/draft-kunze-bagit-13. Accessed 28 Jan 2016. Accessed 07 Feb 2016
9. CloudSync: https://wiki.duraspace.org/display/CLOUDSYNC11/Fedora+CloudSync+1.1. DuraCloud: http://www.duracloud.org/. LOCKSS (Lots of Copies Keep Stuff Safe): http://www.lockss.org/. All accessed 28 Jan 2016
10. Artefactual Systems, Discovery Garden Archidora: Archivematica Integration. https://github.com/discoverygarden/archidora. Accessed 05 Jan 2015
11. Dublin Core Metadata Initiative. DCMI Metadata Terms. http://dublincore.org/documents/dcmi-terms/. Accessed 07 Feb 2016
12. Fedora Commons. Fedora Digital Object Model. https://wiki.duraspace.org/display/FEDORA38/Fedora+Digital+Object+Model. Accessed 05 Jan 2015
13. York University York University Digital Library. https://digital.library.yorku.ca. Accessed 05 Jan 2016
14. Islandora Preservation Interest Group. https://github.com/Islandora/Islandora-Preservation-Interest-Group. Accessed 05 Jan 2015
15. Library of Congress LC Linked Data Service: Authorities and Vocabularies: Event Type. http://id.loc.gov/vocabulary/preservation/eventType. Accessed 05 Jan 2016
16. PRONOM: http://www.nationalarchives.gov.uk/PRONOM/Default.aspx. FITS: http://projects.iq.harvard.edu/fits/home. Accessed 28 Jan 2016
17. Duraspace Organization (2015) Fedora design: audit service. https://wiki.duraspace.org/display/FF/Design+-+Audit+Service. Accessed 05 Jan 2016
18. Artefactual Systems (2015) Archivematica website. http://www.archivematica.org. Accessed 05 Jan 2016
19. METS: Metadata Encoding and Transmission Standard. http://www.loc.gov/standards/mets. Accessed 07 Feb 2016
20. Brandt O, Enders M, Guenther R, Habing T, Lazzarino F, Redding C, Riley J, Wolfe R (2008) Guidelines for using PREMIS with METS for exchange. http://www.loc.gov/standards/premis/guidelines-premismets.pdf. Accessed 05 Jan 2016
21. DAITSS stands for Dark Archive In the Sunshine State. https://daitss.fcla.edu/. Accessed 16 Jan 2016
22. RODA stands for Repository of Authentic Digital Objects. http://www.roda-community.org/. Accessed 16 Jan 2016
23. DSpace website: http://dspace.org/. Accessed 16 Jan 2016
24. http://www.bitcurator.net/. Accessed 16 Jan 2016
25. http://archivesspace.org/. Accessed 16 Jan 2016
26. AtoM stands for Access to Memory. https://www.accesstomemory.org/. Accessed 16 Jan 2016
27. https://www.hathitrust.org/. Accessed 16 Jan 2016

Chapter 17
Case Study: Implementing an Open-Source and In-House Developed PREMIS Events and Agents System

Mark Edward Phillips and Daniel Gelaw Alemneh

17.1 Introduction

One approach for implementing digital preservation metadata involves building the service alongside an existing platform. This provides a modular design option supplementing existing infrastructure with additional preservation functionality. By loosely coupling a PREMIS [1] implementation to the primary repository, each system is able to change over time independently of the other as long as they maintain the linkage between the systems, often times by using unique identifiers. Implementing PREMIS services may be approached in this manner and can supplement existing infrastructure where preservation metadata is either not adequately documented, or where local requirements surpass what is present in available repository infrastructure. This article illustrates this approach with a case study on implementing PREMIS for a particular purpose.

17.2 PREMIS Event Service

The University of North Texas Libraries took a modular approach in developing a system to log and store PREMIS Events and Agents within its locally developed repository structure. The team at UNT wanted to capture events that were important in the lifecycle of digital objects and to store those events in order to provide transparent information about the quality and status of the digital objects they were

M.E. Phillips (✉) · D.G. Alemneh
University of North Texas, UNT Libraries, 1155 Union Circle #305190,
Denton, TX 76203-5017, USA
e-mail: Mark.Phillips@unt.edu

D.G. Alemneh
e-mail: Daniel.Alemneh@unt.edu

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_17

tasked with preserving. To this end it investigated a number of software implementations but was not able to find a modular tool that would allow it to integrate event tracking into the repository without a large amount of modification to the underlying architecture of the repository [2]. The UNT Libraries decided to develop the "PREMIS Event Service" [3] which could be used to log, store, and query PREMIS Events. The software implementation was designed using the principle of Representational State Transfer, commonly known as REST [4, 5], to create a loosely coupled implementation that would work alongside the existing digital repository infrastructure. The team additionally made use of the Atom Publishing Protocol (ATOMPub) [6] as a core component of this REST implementation.

A PREMIS Event is submitted to the system by wrapping a PREMIS record serialized in XML within an ATOMPub wrapper to the REST interface as shown in Fig. 17.1. Within the repository, a submitted PREMIS XML Event is deconstructed

```
<?xml version="1.0"?>
<entry xmlns="http://www.w3.org/2005/Atom">
  <title>6ca80534090a471495374f5f1aeb5ed4</title>
  <id>http://coda.library.unt.edu/APP/event/6ca80534090a471495374f5f1aeb5ed4/</id>
  <updated>2014-06-12T11:06:44Z</updated>
  <content type="application/xml">
    <premis:event xmlns:premis="info:lc/xmlns/premis-v2">
      <premis:eventIdentifier>
        <premis:eventIdentifierType>
          http://purl.org/net/untl/vocabularies/identifier-qualifiers/#UUID
        </premis:eventIdentifierType>
        <premis:eventIdentifierValue>
          6ca80534090a471495374f5f1aeb5ed4
        </premis:eventIdentifierValue>
      </premis:eventIdentifier>
      <premis:eventType>
        http://purl.org/net/untl/vocabularies/preservationEvents/#ingestion
      </premis:eventType>
      <premis:eventDateTime>2014-06-12T05:15:47</premis:eventDateTime>
      <premis:eventDetail>
       Ingest of codaf8td performed by CODAProcess.py, from dropbox at
       /data07/coda-008_dropbox/1.ToCODA
      </premis:eventDetail>
      <premis:eventOutcomeInformation>
        <premis:eventOutcome>
          http://purl.org/net/untl/vocabularies/eventOutcomes/#success
        </premis:eventOutcome>
      </premis:eventOutcomeInformation>
      <premis:linkingAgentIdentifier>
        <premis:linkingAgentIdentifierType>
          http://purl.org/net/untl/vocabularies/identifier-qualifiers/#URL
        </premis:linkingAgentIdentifierType>
        <premis:linkingAgentIdentifierValue>
          http://coda.library.unt.edu/agent/codaIngest
        </premis:linkingAgentIdentifierValue>
      </premis:linkingAgentIdentifier>
      <premis:linkingObjectIdentifier>
        <premis:linkingObjectIdentifierType>
          http://purl.org/net/untl/vocabularies/identifier-qualifiers/#ARK
        </premis:linkingObjectIdentifierType>
        <premis:linkingObjectIdentifierValue>
          ark:/67531/codaf8td
        </premis:linkingObjectIdentifierValue>
      </premis:linkingObjectIdentifier>
    </premis:event>
  </content>
</entry>
```

Fig. 17.1 Example ingest event wrapped in an ATOMPub entry wrapper

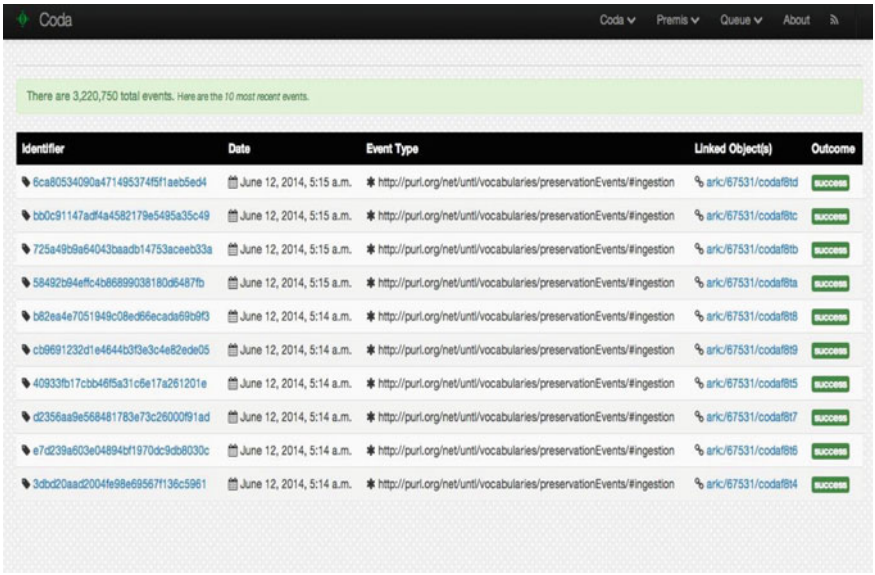


Fig. 17.2 End user interface for PREMIS Event service

into the individual semantic units of the Event model and stored in a standard relational database. This database provides a number of end user interfaces that allow for the usage of the PREMIS Event Service in a number of applications. In addition to the HTML-based end user interface shown in Fig. 17.2, the REST API provided by the ATOMPub interface makes integrating the tool with other systems straightforward.

In addition to modeling the PREMIS Event type, the PREMIS Agent was also modeled in order to record parties responsible for the PREMIS Events in the system and to allow for the creation and maintenance of unique identifiers for them. In the implementation at the UNT Libraries, these PREMIS Agents represent organizations and software agents, which interact with digital objects in the preservation repository.

17.3 PREMIS Event Service Implementation Decisions

Once the PREMIS Event Service was developed, implementers had to decide which events to capture, how to uniquely identify these events, and at what granularity (bitstream, file, object, collection, or other grouping) they should reference a digital object. At the time of implementation the UNT Libraries defined the following event types as those that should be logged and saved as part of a digital object's lifecycle.

These events include

- ingestion
- replication
- fixity check
- virus check
- migration

The UNT team minted identifiers locally to represent these concepts and made assertions of equivalence to the same concepts in the Library of Congress id.loc.gov Service once the Preservation Vocabularies were added to that system [7].

The UNT Libraries utilize the PREMIS Event Service as a core piece of the management of a digital object's lifecycle in the UNT Libraries' preservation repository. At this time, PREMIS Events are created during ingest of items into the repository, whenever fixity checks are completed, and upon replication of a digital object to offsite storage. Both successful and failing system events are logged with the PREMIS Event Service as a way of monitoring the state objects in the repository. An end user is able to view each event record from the system using a Web browser with the user interface like the one shown in Fig. 17.3. In 2013, the UNT Libraries conducted a full migration of the underlying storage architecture used in its digital repository, and as a result, a migration event was created which documented this lifecycle event for all digital objects which were migrated.

By designing the system using a modular design methodology, the development team at the UNT Library is able to submit PREMIS Events from a number of software workflows and scripts as they are executed. The ease of creating and storing PREMIS Events within the tool facilitates the UNT Libraries' events

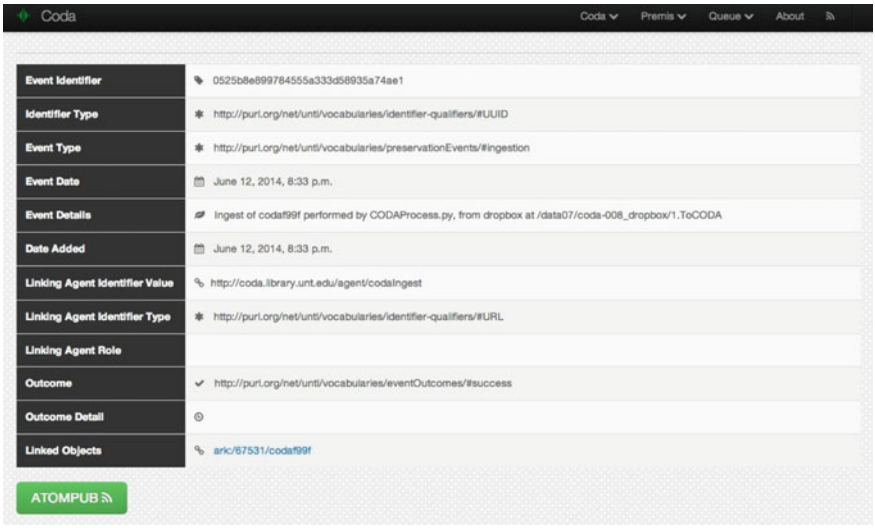


Fig. 17.3 Event record detail view

tracking over the digital object's complete lifecycle. The end user HTML-based user interface allows managers and auditors to understand the flow of digital objects through the various workflows in the repository.

Now in full operation, the PREMIS Event Service plays an integral role in the digital preservation strategy of the UNT Libraries. It provides a modular mechanism to associate PREMIS Events and associated Agents with the digital objects that they describe. In 2014, the UNT Libraries released the PREMIS Event Service as an open-source software project with a BSD license [8].

17.4 Summary and Conclusion

Accurate, accessible, and usable metadata is a vital component in the lifecycle of a digital object. Whether it is descriptive, administrative, technical, or preservation metadata, the need to collect, store, and retrieve this information about digital objects may grow and often times change over its lifecycle. Taking a modular approach to implementing some of the metadata services, such as collecting PREMIS Events, is a viable option, which has been demonstrated by the UNT Libraries in the creation of the PREMIS Event Service. As time goes on, repository infrastructure is guaranteed to change and a modular, loosely coupled infrastructure for the collection, storage, and retrieval of preservation metadata allows for planned change over time in how, when, who, and why this metadata is collected and used. With the goal of ensuring the long-term viability of digital objects in their custody, and with new tools for managing preservation metadata associated with these objects, cultural heritage institutions have a new opportunity to meet this important piece of their mission.

References

1. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation metadata version 3.0. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 05 Jan 2016
2. Phillips M, Schultz M, Nordstrom K (2011) PREMIS event service. In: Sixth annual international conference on open access, 2011, Austin, TX. http://digital.library.unt.edu/ark:/67531/metadc40413/. Accessed 24 April 2016
3. UNT Libraries (2016) PREMIS event service documentation, Release 1.0, January 15, 2016. https://premis-event-service.readthedocs.org/en/latest/. Accessed 13 Mar 2016
4. Fielding RT, Taylor RN (2000) Principled design of the modern Web architecture. In: ICSE '00 proceedings of the 22nd international conference on software engineering, pp 407–416. doi:10.1145/337180.337228
5. Fielding RT, Taylor RN (2002) Principled design of the modern Web architecture. TOIT 2 (2):115–150. doi:10.1145/514183.514185

6. Gregorio J, De Hora B (2007) RFC 5023-the AtoM publishing protocol. Internet Engineering Task Force (IETF) Request for Comments. http://www.loc.gov/standards/premis/FE_Dappert_Enders_MetadataStds_isqv22no2.pdf
7. LC Linked Data Service (2014) Preservation vocabularies: event type. http://id.loc.gov/vocabulary/preservation/eventType.html. Accessed 13 Mar 2016
8. UNT Libraries (2014) Django PREMIS event service. https://github.com/unt-libraries/django-premis-event-service. Accessed 13 Mar 2016

Chapter 18
Conformance with PREMIS

Peter McKinney

18.1 Introduction

In 2010 the PREMIS Editorial Committee released a Conformance Statement [1]. This defined a set of principles "governing a conformant implementation" of the Data Dictionary. A key tenet of the work was to give implementers of the Data Dictionary flexibility to be able to use the Dictionary in ways that allowed them to respond to their own internal preservation processes.

As experience of both digital preservation practices and of implementing PREMIS has grown across the community the Editorial Committee asked a sub-committee to explore again the notion of conformance.[1]

The work began as an effort to extend and refine the existing conformance requirements, but has in addition looked at fundamental questions of preservation functionality and a reassessment of how PREMIS fits into that functionality.

This chapter will discuss the new version of the Conformance Statement [2] and look at the benefits that can be achieved through conformance.

[1]The committee comprised: Angela Di Iorio (Sapienza Università di Roma), Jay Gattuso (National Library of New Zealand), Rebecca Guenther (Library of Congress), Jan Hutař (Archives New Zealand), Amy Kirchoff (ITHAKA), David Lake (NARA), Peter McKinney (National Library of New Zealand), Evelyn McLellan (Artefactual Systems), and Joseph Pawletko (New York University).

P. McKinney (✉)
National Library of New Zealand/Te Puna Mātauranga O Aotearoa, PO Box 1467
Wellington 6140, New Zealand
e-mail: Peter.McKinney@dia.govt.nz

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7_18

18.2 What Does Conformance Mean?

A quick survey of standards bodies globally shows that conformance and compliance are used interchangeably.[2] Conformance, however, is the preferred term used by the PREMIS committee, rather than compliance. We take conformance to be the action of conforming according to "some pattern, model or instruction" [3].

In the heritage sector conforming is a matter of institutional rigor driven primarily by perceived benefit, rather than audited necessity. There are few, if any, fiscal ramifications for erroneously asserting compliance (perhaps withholding of project funding if conformance is a project deliverable could be offered as one example). Which is not to say that there are not negative outcomes: reputational risk is of course a very real danger in the heritage world. The benefits of conformance must therefore be strong enough to convince individuals and institutions to conform. The task then for implementers (potential and definite) is to judge the benefits against any barriers to conformance. Is the pain justified by the benefits gained from conforming?

18.3 Earlier Statement and Discussions About Conformance

The 2010 Conformance Statement used principles to guide implementers. The principles are simple and clear. Conformity is defined at both the Dictionary level and the semantic unit level.

Principles for Semantic units

- If a metadata element shares the name of a PREMIS semantic unit, it must also share its definition. If a metadata element shares the definition of a PREMIS semantic unit but does not share its name, the repository must establish a mapping between the metadata element and its corresponding PREMIS semantic unit.
- Usage requirements specified in the Data Dictionary for a particular semantic unit must be observed. Repeatability, obligation (i.e., whether a semantic unit is mandatory), and applicability (*Bitstream*, *File*, and *Representation*) requirements can be made more stringent, but not more relaxed.

[2]For example, the European Union uses "Conformance Marking," [http://europa.eu/legislation_summaries/other/l21013_en.htm], Standards Australia uses "compliance" [http://www.standards.org.au/OurOrganisation/Pages/Compliance-With-Australian-Standards.aspx], and ISO uses 'conformity' [http://www.iso.org/iso/home/faqs/faqs_conformity_assessment_and_certification.htm]. Accessed 8 January 2016.

Principles for the Data Dictionary

- Include the mandatory semantic units for any Data Model Entity (Objects, Events, Agents, or Rights) supported by the repository.
- Be able to recover all of the information specified in the mandatory PREMIS semantic units from the repository system (regardless of its specific implementation), and associate it with its corresponding Entity.

It has been noted that a number of gaps exist in these principles. The Conformance statement itself notes "it may be the case that certain specific digital preservation contexts…may benefit from adherence to a stricter conformance profile" [4]. It has also been suggested that the benefits achieved through conformance (such as consistency, consensus, sharing and understanding the standard's application over time) are minimized as any conformance statement becomes more flexible [5]. The benefits specified by the PREMIS Editorial Committee that come from conformance are: inter-repository data exchange, repository certification; shared registries; automation/reusable tools; and vendor support.

This is an important issue. If the flexibility is such that loose interpretations can be considered conformant, how then can these benefits be realized? And if the benefits are lessened by a flexible conformance statement, then what is the driver that convinces individuals and institutions to conform to the standard? And if conformance is not undertaken, then the ramification could be that the community no longer has a central core of preservation metadata around which it can understand the digital world, exchange experience and more simply, discuss work with a mutually defined language.

In order for the target benefits to be achieved, the conformance statement must be made stronger. For example, automation and reusable tools require a very strict conformance. If an institution states conformance to PREMIS, it does not mean that they encode their metadata using PREMIS semantic units. If the day-to-day working currency of metadata in a repository is something other than PREMIS, how then can tools and automation services be built that would benefit that repository?

A recent report backs this view. Kool et al. [6] have looked at the viability of using PREMIS metadata to underpin risk assessment of content in digital repositories. They note that there is no requirement to code PREMIS metadata in a repository. Institutions can code metadata in (almost) any fashion they wish, thus hampering any form of risk assessment that uses specific metadata elements. If the elements are not recorded, or recorded in a different form, the risk assessment tool will not be able to use them [7].

18.4 New Conformance Statement

The current conformance working group issued a revision of the 2010 statement for conformance [8]. The principles of conformance remain, but the revision introduces

- Graduated levels of conformance (1, 2, and 3)
- Two sublevels of conformance (Level A and B)
- A distinction between using preservation metadata that can be mapped to the PREMIS Data Dictionary and implementing the Dictionary as an internal metadata schema
- An emphasis on implementing the Object entity at a minimum.

Unlike the original statement, this statement does not explicitly address the concepts of internal and external conformance (i.e., internal use within a repository versus interchange with other repositories); however, the higher the level of conformance a repository achieves, the greater will be its ability to exchange preservation metadata with other repositories.

The primary change is the introduction of clear levels of conformance. These levels are more prescriptive than the previous statement, allowing institutions to define exactly how they conform.

18.4.1 Level 1 Conformance Through Mapping

18.4.1.1 Level 1A: Object Entity Only

A repository uses one or more internal preservation metadata schemas, elements of which can be mapped to PREMIS. Such mapping must satisfy the principles of use at both the semantic unit and Data Dictionary levels. The repository is able to produce documentation demonstrating such mapping, at a minimum, for the Object entity.

18.4.1.2 Level 1B: Object, Event and Agent Entities

A repository uses one or more internal preservation metadata schemas, elements of which can be mapped to PREMIS. Such mapping must satisfy the principles of use at both the semantic unit and Data Dictionary levels. The repository is able to produce documentation demonstrating such mapping, at a minimum, for the Object entity; one or more Agents; and sufficient Event metadata to document actions the repository has taken to preserve the digital objects.

18.4.2 Level 2 Conformance Through Export

18.4.2.1 Level 2A: Object Entity Only

A repository uses one or more internal preservation metadata schemas, elements of which can be exported as PREMIS. Such export must satisfy the principles of use at both the semantic unit and Data Dictionary levels. The repository has established processes and tools in place to perform these exports as a routine operation, and is able to demonstrate such capability, at a minimum, for the Object entity.

18.4.2.2 Level 2B: Object, Event, and Agent Entities

A repository uses one or more internal preservation metadata schemas, elements of which can be exported as PREMIS. Such export must satisfy the principles of use at both the semantic unit and Data Dictionary levels. The repository has established processes and tools in place to perform these exports as a routine operation, and is able to demonstrate such capability for the Object entity; one or more Agents; and sufficient Event metadata to document actions the repository has taken to preserve the digital objects.

18.4.3 Level 3 Conformance Through Internal Implementation

18.4.3.1 Level 3A: Object Entity Only

A repository implements the PREMIS Data Dictionary as an internal metadata schema in a way that satisfies the principles of use at both the semantic unit and Data Dictionary levels and in a form that does not require further mapping or conversion. The repository implements, at a minimum, the Object entity.

18.4.3.2 Level 3B: Object, Event and Agent Entities

A repository implements the PREMIS Data Dictionary as an internal metadata schema in a way that satisfies the principles of use at both the semantic unit and Data Dictionary levels and in a form that does not require further mapping or conversion. The repository implements, at a minimum, the Object entity; one or more Agents; and sufficient Event metadata to document actions the repository has taken to preserve the digital objects.

Level 1 can be considered to be the lowest level of conformance: institutions do not code in PREMIS terms, but can map to it and their use and inclusion of

metadata satisfies the conditions within PREMIS. For example, they note the format identification of a file, but do not call it "formatName," instead using "formatIdentification". As long as the mapping can be made, the meaning is equivalent and it is a mandatory element, this is conformant.

Level 2 is a further step toward supporting interoperability. Over and above mapping metadata elements to PREMIS elements, the institution must be able to export their data in a compliant PREMIS form. While Level 1 and Level 2 may appear on first sight to be extremely similar, if not exactly the same, the work of the conformance group has shown that the step up from mapping to being able to export is actually substantial. It is dependent on up-to-date tools and processes to do such an export. One of the benefits from this level is that it is testable.

While we would not describe Level 3 as the "top level," it is the "purest" level in terms of PREMIS. It requires institutions to record metadata using absolute PREMIS terms. There is no mapping as the elements are named exactly the same and the optionality and repeatability are no less stringent.

All levels are split into two sublevels. The first (the "A" category) is for those institutions that only deal with the Object entity of the Data Dictionary. The second (the "B" category) is for those institutions that deal with the Object entity and Agents and Events. The reason for this split is to cover institutions that are not implementing the full Data Dictionary; some institutions may only be at the beginning stage of implementing a digital preservation program and have chosen to implement the Object entity—the entity that controls the items being preserved. These institutions should be able to record a level of conformance. However, it should be recognized that the subcategory A is not the complete set of metadata that a full digital preservation should collect. Institutions should strive to attain a "B" category.

This "B" subcategory expands the number of entities required to be conformant to include Agents and Events. These were felt to be the most crucial of the remaining entities required to be conformant as they note what actions have been undertaken to preserved objects and who or what carried out that action. In essence, they help maintain the integrity of the preserved object. The Rights entity is deliberately not part of either subcategory. Experience has shown that there are many other ways of capturing rights information, whereas for the other entities, PREMIS is the primary expression of the required metadata.

18.5 What Are the Benefits from Each Level?

Each level has different positives (and negatives) for any implementing institution.

Level 1 conformance demonstrates that an institution has a clear understanding of PREMIS. It also allows that institution to take advantage of the community development of PREMIS. It does not, however, require the institution to take full advantage of using the exact language of PREMIS, forcing the adoption and maintenance of mappings between their own metadata dictionary and PREMIS.

Level 2 conformance allows implementers to realize further benefits such as data exchange and the use of shared registries. However, to achieve some benefits, such as automation/reusable tools, more complex system behavior may be required than that which level 2 allows.

Level 3 allows for the possibility for all of the benefits mentioned above to be realized (inter-repository data exchange, repository certification; shared registries; automation/reusable tools; and vendor support). Level 3 allows tools to be built or adapted to output into PREMIS elements, processes can be defined using PREMIS elements, and shared policies can be written that hook directly into actionable metadata. At this level, the real benefits of adopting a standard can be felt. Institutions can take full benefit of community development and can in turn take part in that community. The digital preservation community, while growing, is still clamoring for experience and expertise. Level 3 implementers will be able to help build that experience and can do so through the common language of PREMIS.

Level 3 has a further layer of benefit that means there is no requirement to develop and maintain mappings and tools that can export data into a PREMIS form. There is also no requirement to maintain knowledge of an internal metadata dictionary. These benefits should be key to institutions that are considering using a closed or proprietary system.

18.6 Notes and Examples

18.6.1 PREMIS as a Component of a Preservation System

The conformance statement does not presuppose that a preservation system will contain only one metadata schema. Systems utilize many different schemas across all of their functions. For example, many will use METS to wrap other metadata schemas. Descriptive metadata schemas are abundant (MODS, Dublin Core, EAD, etc.) and technical metadata also comes in many flavors (AES, MIX, EXIF, etc.)

It is also clear that systems do not only deal with one way of storing and using metadata. Systems will often split data across various parts of databases in order to maximize system performance and user experience. This is in addition to any more permanent storing of the preservation metadata, in what may be termed the permanent AIP. Ex Libris's Rosetta system [9] and Preservica [10] for example, utilize databases to store metadata in addition to a more static storage of an AIP on the permanent storage layer. Archivematica [11] also makes use of databases as a precursor to a more static AIP structure.

In many ways, in terms of the conformance levels, it does not matter which part of the metadata is deemed to be conformant. For example, at level 1, it is most likely that the mapping would be against the schema used for the more static AIP. However, the level 2 export could be more sensibly generated by a system from the more active database stored metadata. For level 3 however, the pure

implementation of PREMIS, the system would have to use PREMIS across all metadata stores, where it was relevant. This is because level 3 requires that any user, at any point could analyze that system (either through its user interface or through a static AIP) and instantly recognize PREMIS elements.

18.7 Is Conformance the End?

18.7.1 Translation of PREMIS

Translation is a critical aspect to be wary of in conformance work. It is one thing to be conformant (either at Level 1, 2, or 3) with recording metadata in line with a data dictionary, but it is another to say that the recorded metadata is being generated or used in the "correct way". Translation is a negotiation, rather than an exact science. One word, when translated, can have a profound effect on the concept or message that is being conveyed [12]. Such translations have to be negotiated not only between the author and translator, but between the cultures that support the languages. For example, a direct translation of the word "rat" is required if it is significant to convey a specific animal that plays a role in the transmission of bubonic plague. But in other examples, a direct translation could be less helpful if the receiving culture would not understand that "rat" could be used to convey the notion of a dishonest person.

There should be little room for translation of a standard however. It is vital across preservation that the scope of translation is minimized. Negotiation should be minimal, but there are examples across PREMIS implementations where this is the case. Two examples are instructive.

> PREMIS states that audio data within a WAVE file is *Bitstream* level [13]. NLNZ currently chooses to write this information at the *File* level. Correspondence on the PREMIS listserv has indicated that other implementers write the audio information also to the *File* level. Should this be classed as nonconformance?
>
> Further examples have brought to light other areas of practice that deviate from the PREMIS examples. The Data Dictionary uses two more examples to highlight the difference between *File* and *Bitstream*. A single image TIFF file should have all information about the image written to the *File* level, but in a multi-image TIFF, all the image information should be written to discrete *Bitstream* levels (one for each image within the file). Again, discussion on the listserv showed that practice deviated from the prescribed classification of the object subtypes. Some institutions wrote all image information, irrespective of whether or not it came from a multipage TIFF, to the *Bitstream* level. NLNZ and NLB currently write it all to the *File* level [14].

These two examples (WAV and TIFF) show that a translation has taken place in institutions in terms of defining the type of object they are preserving: is it bitstream or a filestream? Who is using PREMIS in the "right way" in these two examples? The conformance statements do not really help in this area. It expresses that implementers must be semantically correct and implement the correct elements of

the Data Dictionary, however, it does not note that the specified classification of object types must be used.

It should be clear that conformance (at least with the original statement) does not necessarily imply uniformity across implementations. To take this a step further, claiming conformance does nothing to show that the implementer uses PREMIS information in its preservation activities, or if it does, that PREMIS information is utilized in the "correct" fashion.

The question should be posed: "can conformance to PREMIS be equated with best practice in digital preservation activities?"

To arrive at an answer, a more holistic view of digital preservation practice is required and brings the work closer to audit and certification. It is important to ensure that PREMIS is fit for its purpose and aligns with other standards. The next step in the conformance work is therefore to explore the functionality of preservation programs and understand (or perhaps more correctly, reassess) the role that preservation metadata plays in that functionality. Once metadata and practice are fully integrated, then the community can begin to make more elaborate trust statements about their entire program. The question therefore can become: "does conformance to PREMIS support the trustworthiness of a programme?"

In order to approach an answer, samples of AIPs from various repositories that are using the Data Dictionary are required. The Editorial Committee have tried on a few occasions to collect implementers' AIPs, but have achieved minimal success. Why this is the case is unclear; perhaps exposure of metadata elicits a certain institutional nervousness.

The first task with the AIPs collected is to manipulate them in order to be able to compare them. As AIPs encompass more than just preservation metadata and very few organizations implement only a pure version of PREMIS for their preservation metadata, the comparison is hindered by the fact that there are other elements to account for. For example, the National Library of New Zealand uses Ex Libris's Rosetta system. The data model for this system contains over 220 elements, contrasting with the 190+ elements in PREMIS version 3.0, and this is before considering whether all elements of PREMIS are actually reflected in the system's data model. It is therefore a nontrivial exercise to both create a mapping to PREMIS where it does not already exist and then using that mapping to actually transform that information into PREMIS (this is reflected in the difference between Level 1 and Level 2). What matters most in this work is understanding what generates the metadata and when it is used. One approach is to filter the metadata by functional area. However, work to date has shown that classifying the functions of a preservation program was a large task; questions of PREMIS conformance quickly turn into questions of best practice in digital preservation, which is perhaps going beyond the remit of the Editorial Committee.

A quick survey of relevant areas (audit tools, papers, standards, community groups, event lists) reveals that there are high-level functional views of a preservation repository that are agreed upon [ingest, data management, archival storage, administration, access, preservation planning] but no codified agreement on the actual detail of these [what is run during ingest and in what order? How many

copies are stored and on what type of storage?]. Which is not to say that they are not known, but rather that the community has tacit agreement of the details rather than explicit statement on what detailed functions are required for best practice.

Despite these caveats, the Editorial Committee is eager to continue this work. The outcomes should be

- A more complete understanding of what activities institutions undertake when preserving objects
- An understanding of how those institutions generate and use preservation metadata during the activities
- A reassessment of whether PREMIS semantic units fulfill all of the activities
- An exploration of how PREMIS conformance can be used in any audit process.

In terms of the latter, there is a current suite of audit mechanisms available to the community, so the concept is not unfamiliar, if perhaps not fully utilized [15]. The matter here is whether stating conformance with the Data Dictionary could automatically give a positive grading in certain parts of the audit (the object management aspect for example). Members of the group have begun to look at such a question and are mapping PREMIS to parts of the CCSDS audit standard.

18.8 Conclusion

We are, as a community, still in the beginning years of preservation in practice. There are tools in place, the community is growing and there is experience coming from preservation programs that have been running for multiple years. However, it is clear that we are still exploring and understanding the world in which we work. When compared with the physical preservation community, our practice is not yet stabilized. There is no agreed set of ethics, there is no peak body to which we all subscribe, there are no agreed and solid set of functions underpinned with a full set of policies and guidance.

The work of the PREMIS conformance committee is looking to undertake a reassessment. It has generated a new set of conformance levels that tighten the previous statements. These levels take into account scenarios that come from daily preservation routines and allow a clearer statement from institutions as to how they conform with PREMIS.

Finally, proposed work linking the levels of conformance with the functionality of preservation repositories may also bring further benefits to the digital preservation community and help ensure PREMIS remains a practical tool for the preservation of digital objects.

References

1. PREMIS Editorial Committee (2010) Conformance implementation of the PREMIS data dictionary, October 2010. http://www.loc.gov/standards/premis/premis-conformance-oct2010.pdf. Accessed 8 Jan 2016
2. PREMIS Editorial Committee (2015) Conformance implementation of the PREMIS data dictionary, April 2015. http://www.loc.gov/standards/premis/premis-conformance-20150429.pdf. Accessed 8 Jan 2016
3. OED Online (2014) "Conform, v." Oxford University Press, June 2014. Accessed 13 July 2014
4. PREMIS (2010) http://www.loc.gov/standards/premis/premis-conformance-oct2010.pdf, p 6
5. Jailani H, McKinney P (2012) Compliance conundrums: implementing PREMIS at two national libraries, IS&T archiving conference 2012 (ARCHIVING 2012), pp 244–249
6. Kool W, van der Werf T, Lavoie B (2014) Preservation health check: monitoring threats to digital repository content. Dublin, Ohio: OCLC Research. http://www.oclc.org/content/dam/research/publications/library/2014/oclcresearch-preservation-health-check-2014-a4.pdf. Accessed 8 Jan 2016
7. Kool W, van der Werf T, Lavoie B (2014) Preservation health check: monitoring threats to digital repository content. Dublin, Ohio: OCLC Research, p 13. http://www.oclc.org/content/dam/research/publications/library/2014/oclcresearch-preservation-health-check-2014-a4.pdf. Accessed 8 Jan 2016
8. PREMIS Editorial Committee (2015) Conformance implementation of the PREMIS data dictionary, April 2015. http://www.loc.gov/standards/premis/premis-conformance-20150429.pdf. Accessed 8 Jan 2016
9. http://www.exlibrisgroup.com/category/RosettaOverview. Accessed 8 Jan 2016
10. http://preservica.com/. Accessed 8 Jan 2016
11. https://www.archivematica.org/en/. Accessed 8 Jan 2016
12. Eco U (2004) Mouse or rat?: translation as negotiation. Orion, London, p 2004
13. PREMIS Editorial Committee (2015) PREMIS data dictionary for preservation, version 3. p. 13. http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf. Accessed 14 Jan 2016
14. Jailani H, McKinney P (2012) Compliance conundrums: implementing PREMIS at two national libraries, IS&T archiving conference 2012 (ARCHIVING 2012), p 248
15. Consultative Committee for Space Data Systems (CCSDS) (2011) Audit and certification of trustworthy digital repositories, recommended practice, issue 1

Glossary

This glossary defines central terms that are not explained elsewhere in the book.

Archive [OAIS term] An organization that intends to preserve information for access and use by a Designated Community

Archival Information Package (AIP) An Information Package which is preserved within an OAIS. Implementation-wise, an Archival Information Package will typically contain digital objects that a given Archive wants to preserve, associated with metadata informing their preservation

Dissemination Information Package (DIP) An Information Package, derived from one or more AIPs, and sent by Archives to the Consumer in response to a request to the Archive

Emulation The act of replacing an obsolete original computer software or hardware (usually called the guest) by another computer software or hardware that imitates its behavior (called the host). Emulation is a preservation strategy that allows one to render or execute obsolete digital objects by changing their environment leaving the content files untouched

Information Package A container typically composed of Content Information (the objects to preserve and their Representation Information) and Preservation Description Information (their preservation metadata), wrapped together with Packaging Information. The Information Package is the base unit for manipulating digital objects in an Archive. The OAIS Archive is responsible for defining the level(s) of granularity which correspond to an Information Package

Migration A transformation of an object creating a version in a more contemporary format

Submission Information Package (SIP) An Information Package that is delivered by the Producer to the OAIS for use in the construction or update of one or more AIPs and/or the associated Descriptive Information

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7

Index of General Terms

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7

Index for PREMIS Semantic Units

A. Dappert et al. (eds.), *Digital Preservation Metadata for Practitioners*,
DOI 10.1007/978-3-319-43763-7