05 Administrative and preservation metadata

Introduction

During the course of its lifecycle, a single digital archival object requires an extensive amount of associated metadata, so that it can be managed and preserved effectively by the repository, and understood and accessed by the researcher. There are three broad categories of metadata:

- Descriptive metadata: information about the intellectual content of a digital object, which is used to aid identification and discovery of the object by the researcher.
- Structural metadata: information about the relationships between digital objects, which can be very complex in a large hybrid personal archive. Structural metadata also supports the display and navigation of digital objects by users.
- Administrative metadata: information needed by the repository for the long-term management of a digital object, including information about an object's creation, technical information such as file formats, provenance information and information about intellectual property rights (see p. 252).

This chapter of the Workbook is primarly concerned with the metadata that must be recorded for administrative and preservation purposes, though it touches on descriptive metadata where this is relevant, and a single piece of metadata may, of course, fulfil several functions. The following metadata areas are introduced and their application to the context of hybrid or digital personal archives is explored:

- Persistent identifiers (see below).
- Preservation metadata for personal digital archives (see p. 73).
- Using METS for preservation and dissemination of personal digital archives (see p. 117).
- Rights metadata for personal digital archives (see p. 141).
- Metadata for authenticity: hash functions and digital signatures (see p. 152).

Persistent identifiers

Persistent identifiers, often referred to as PIDs, provide a means of connecting and distinguishing between an identifier for an object (which should be permanent) and an object's location (which may change). Researchers use a form of persistent identifier (usually a reference code or shelf-mark) when citing archives or manuscripts in a publication, or when requesting access to them. The manuscript's identifier must be permanent and independent of the manuscript's location so that the source of the researcher's statement can always be accessed, even if the storage location of the manuscript changes. For the identifier scheme to work, a resolver, which knows the location of the manuscript referred to by the identifier, is required:

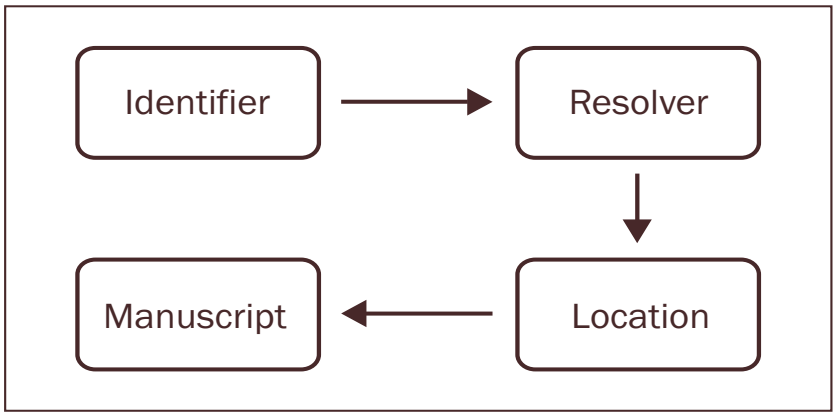

Figure 5: Resolver

Example scenario: When a researcher uses an identifier to request a manuscript in a special collections reading room, the location of the manuscript may be resolved by a staff member who (after consulting a location guide) will retrieve the manuscript from its location and present it to the reader. By maintaining a system to resolve locations from identifiers, special collections staff are able to satisfy reader requests for manuscripts even when their location changes.

Digital objects also require persistent identifiers that connect and distinguish between identity and location. It is possible that locations will change more frequently in the case of digital manuscripts owing to the need for regular refreshment of storage media to guard against media failure. It is also likely that an intellectual entity acquired in digital form must be associated with multiple representations of itself over time, as technological obsolescence requires the repository to migrate away from the formats of the original representation to those accessible using contemporary computing environments.

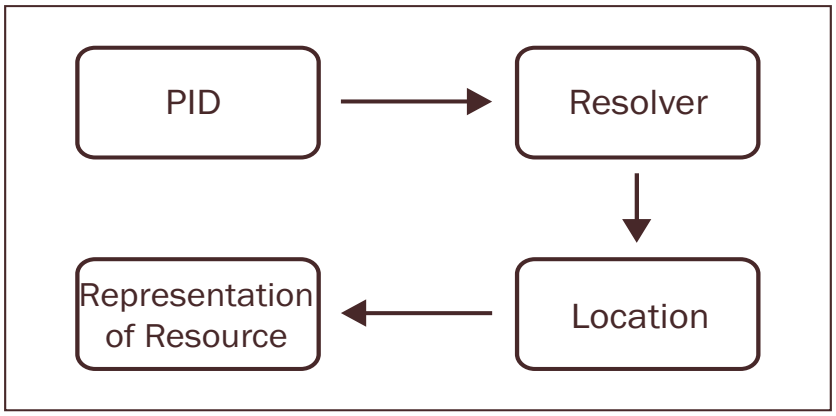

Figure 6: Distinguishing between identity and location

A repository could, in theory, use the same string construction as employed for identifiers of traditional manuscripts in its identifier scheme, though these structures are not usually suited to digital environments. The identifier systems of the Bodleian and John Rylands libraries illustrate this point:

The Bodleian's shelfmarks

The Bodleian uses shelfmarks (which are independent of location) and folio numbers to compile a reference code that identifies its archival materials.

MS. Berlin 102 fol. 230 is the identifier for a letter from C.F. Hardie to Sir Isiah Berlin [1932].

MS. Berlin = Papers of Sir Isiah Berlin

102 = the 102nd shelfmark assigned in the Papers of Sir Isiah Berlin (in this case a box from the series of general correspondence, 1927-97)

fol. 230 = the 230th folio in the box

The John Rylands' reference code

The Rylands assigns a three letter mnemonic to an archive and lower levels of archival description are identified using hierarchical slashes.

RMD/1/2/5 is the identifier for a letter from Bruce Glasier to Ramsay MacDonald, 17 Feb. 1907

RMD = Papers of Ramsay MacDonald

/1 = the first series of material in the Papers (in this case, entitled 'Correspondence and related papers')

/2 = the second subseries of the above series (in this case, representing 'letters from 1907 and the ILP')

/5 = the fifth item in the above subseries

These traditional identifiers are not easily implemented in a digital context. The Bodleian's identifiers contain spaces, inconsistent case and punctuation, and those conforming to the Rylands' system are easily mis-keyed. Neither system accommodates the fact that if digital, a letter could be deposited in one format and subsequently migrated to another, leading to two representations of the same intellectual entity; neither could cope with the need to identify constituent files in complex objects, such as websites. Paradigm has therefore concluded that it is preferable to allocate each intellectual entity (digital or otherwise) a traditional identifier at the time of cataloguing so that identifiers within the catalogue are uniform and comprehensible to researchers, but that more granular identifiers designed to persistently identify representations of the intellectual entity will also be required. Employing a system of persistent identification that is more suited to the digital world for identifying original and successor representations of the digital manuscripts in the digital archive repository will facilitate administration and preservation because it enables the repository to assign identifiers on ingest, or on the creation of new representations resulting from preservation actions; the assignment of identifiers cannot wait until the archive is subject to archival description (which is likely to be a considerable period after accession).

The topic of persistent identifiers for digital material has been subject to much debate and multiple schemes that fulfil the same, or similar, objectives have been created. Currently there is little agreement as to which scheme offers the best solution, and each has its own proponents with vested interests in its proliferation. The problem statement offered at the March 2006 meeting of the NISO Identifiers Roundtable sums up some of the difficulties surrounding the topic:

1. *There is no shared view of the nature of an identifier, its properties, and the requirements for its creation and use.*
2. *There is considerable duplicative effort across disciplines and sectors; although each discipline considers its efforts unique because its underlying data is unique, at an information science level they are often pursuing the same ends by similar means.*
3. *Identifiers can only be fully considered in conjunction with their supporting services, including systems for creating identifiers, binding them to information or objects, and resolving an identifier to obtain the associated object or information (metadata) about it.*
4. *Although much of this work is being conducted outside of the traditional library community, it is inescapable that much of it will eventually impinge upon libraries, due*

> *to their traditional role in gathering, archiving and disseminating information across all domains of human activity. The experience of NISO and its member bodies could helpfully inform a broad interdisciplinary discussion of identifiers and their requirements.*
>
> *NISO Roundtable*[1]

What follows includes an exploration of the issues surrounding persistent identifiers, an articulation of some of the envisaged uses of persistent identifiers in the context of preserving digital archives, and an overview of some of the persistent identifier schemes available.

General important characteristics in PIDs

In order to succeed in identifying one digital document as distinct from another, in a world full of easily movable and reproducible digital matter, repositories must employ naming conventions which make names independent of addresses. These must ensure that a name is only used for one 'thing' in a given namespace so that any ambiguity about the identity of individual manuscripts is impossible.

Furthermore, the name must persist in such a way that it unambiguously identifies the manuscript indefinitely, so that when a manuscript is ordered by its name in 500 years time the researcher can be sure of obtaining it. Although schemes have been devised to resolve some of the issues around persistent identifiers, the reality is that much of the complication is social rather than technical. The key is organisational commitment to a method and effective administration of the selected scheme: identifiers can only be persistent if they are managed.

Some of the aspects important to persistent identification have been categorised as:

Removal of potential ambiguities

To provide unambiguous names, it must be known that the name is not already used in a given namespace.

Example:
In this first image, there are three local namespaces which are those of three separate institutions; there is also a global namespace. Local namespaces 1 and 2 each have an object:1 in their namespace; this causes no problems until both organisations decide to put their object:1 into the global namespace – this causes a conflict because two items cannot have the same identifier in one namespace.

1 National Information Standards Organization, 'NISO Digital Identifiers Roundtable', *National Information Standards Organization website.* URL: <http://www.niso.org/news/events_workshops/ID-06-wkshp.html>

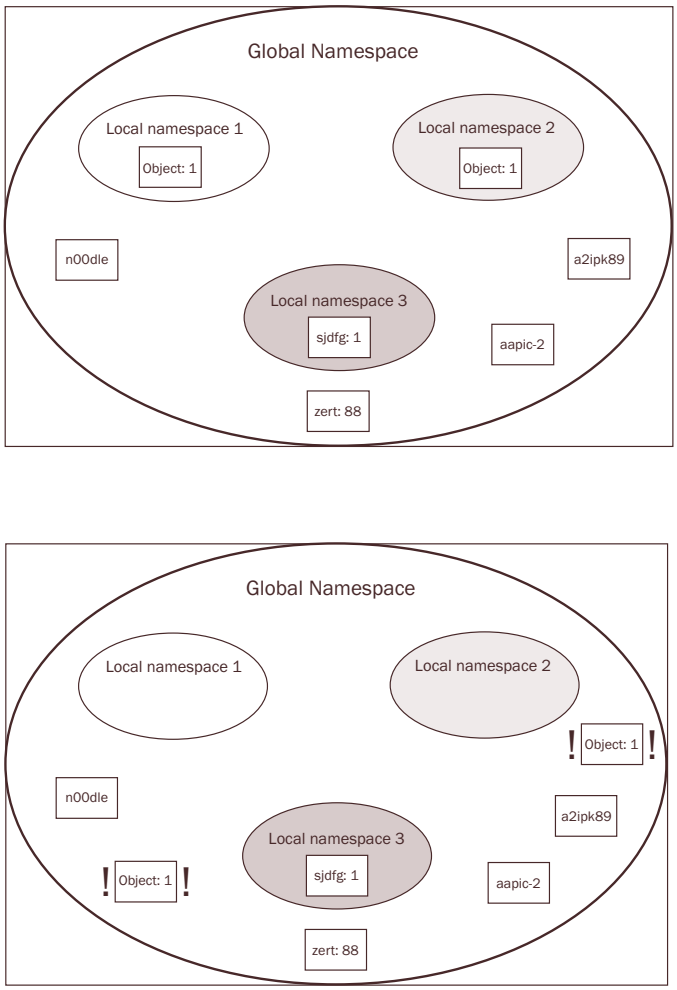

Figure 7: Namespaces

To ensure that conflicts do not happen in a namespace, there must be rules for the apportioning and allocation of names in each namespace. If the identifier is unique within the local namespace, and there is some means of indicating within the identifier which namespace the object belongs to, then uniqueness in the global namespace is ensured.

People friendly

Although computers can easily create unique character strings to represent names (e.g. liduf000alq7t), people prefer units which have some kind of meaning, or which are readable, writable or memorable. However, there are problems associated with using natural language in identifiers; meanings and nuances can change – both over time, and between different cultures and languages. Whilst in an archival context, PIDs should be simple enough for reading room staff to convey over the phone, or for readers to key into a search facility, ideally they should not convey any obvious meaning, and might best be comprised of a simple combination of digits and non-vowel alphabetical characters.

Persistent

Persistence is maintained so long as names continue to be apportioned and allocated according to the rules, and are therefore not used ambiguously, and so long as the current location of an object is known to the resolver of the identifier. The infrastructures responsible for these activities must therefore be evolved and sustained indefinitely. Some of the factors which might affect the longevity of PID systems are:

- **Sustainability of the system and/or its administering organisation(s):** an identifier issued by an organisation which has questionable sustainability has limited credibility as a persistent identifier. This organisation could be the organisation responsible for the repository, or a third party offering a PID service.
- **Popularity of the system:** if a PID service, and the technologies and rules underpinning it, are widely adopted and understood, then a community with an interest in the long-term sustainability of the PID scheme will be formed.
- **Quality of system documentation:** the PID system must be well documented if it is to be understood and implemented over time.
- **Standards compliance:** compliance with web standards, such as URI, can bring interoperability and transparency.
- **Low cost or free:** a repository responsible for preserving digital archives and manuscripts will use a great many PIDs, it is therefore highly desirable that any PID system is economic to administer.
- **Independent of, but interoperable with, other systems:** PIDs must outlast all systems. When repository systems and storage technologies are upgraded, PIDs must remain consistent.
- **Ability to incorporate existing identification schemes:** if there is a long-established persistent identifier scheme already in use within an institution or particular sector, it might be useful to incorporate these identifiers into the namespace of whatever PID system is adopted for the electronic environment (an example might be ISBN identifiers for books).

Retrievable

Once a name is allocated, there is a social expectation that the name should always refer to the item and that the item, or at least information about the item, should be retrievable on production of its name to the correct service. This means that names must be distinct from addresses, so that when the name of the object is given to a service, the service can resolve the current location of the object in order to present it to the user.

Persistent identifiers for a repository of digital archives and manuscripts

The persistent identification of digital objects is vital to the delivery of services associated with the preservation and access of digital archives. To implement an identifier scheme which meets the needs of this context repositories must understand what they expect of such a scheme as comprehensively as possible; only then can the suitability of the various identifier schemes on offer be assessed. Information about some of the most common schemes, such as ARK, DOI, Handle, PURL, URI, URL and URN is provided below; here an anlysis of the persistent identifier requirements for preserving and providing access to personal digital archives is presented.

What do we need to identify?

This understanding of what a repository might need to identify is partly based on the use of identifiers in the *PREMIS Data Dictionary* for preservation metadata (see p. 80).

<table>
<tr><td>The archive The archive, as a collection, must have an identity so that it can be associated with objects belonging to it using their identifiers.

archive:1
archive:2</td><td>An accession In order that digital manuscripts can be associated with an accession, an accession id is needed.

archive:1
    accession:1
    accession:2</td></tr>
<tr><td>A digital manuscript (the intellectual entity) This is the conceptual item that might be described in a catalogue, and will be given a traditional identifier, such as a reference code or shelfmark, when it is catalogued. An example is:

archive:1
    accession:1
        object:1 – Website of Politician X, 20 Jul. 2006 (also known as MS. Eng. 23)</td><td>A representation A representation is a particular instance of an intellectual entity, so MS. Eng. 23 above could have two representations as follows:

archive:1
    accession:1
        object:1 [MS. Eng. 23]
            representation:1 – Website of Politician X, 20 Jul. 2006 (html and jpeg files)
            representation:2 – Website of Politician X, 20 Jul. 2006 (PDF capture of website)</td></tr>
<tr><td colspan="2">A file A representation is composed of one or more files, for example:

archive:1
    accession:1
        object:1 [MS. Eng. 23]
            representation:1 – Website of Politician X, 20 Jul. 2006
            file:2 – html file
            file:3 – jpeg file
            representation:2 – Website of Politician X, 20 Jul. 2006
            file:5 – pdf file</td></tr>
</table>

Different representations will be created for:

- The ‘original’ digital object which was deposited with the archive.
- Successive migrations of the data object deposited with the archive.

A repository’s preservation strategy may require that digital objects be migrated to other formats in response to format obsolescence, or on ingest if the repository has limited its support to a small number of formats. The migrated objects will need their own PID; this PID can be used to relate the migrated object to the object that it was derived from and also to the traditional identifier in the archival catalogue which describes the digital manuscript.

Metadata (and versions of metadata) describing the object

Metadata describing objects may also need identifiers; this enables the repository to associate one piece of metadata to one or more objects. Metadata which may need an identifier includes:

Event metadata: e.g. event:1	Agent metadata: e.g. agent:1
Rights metadata: e.g. rights:1	Descriptive metadata: e.g. catalogue:1

Why uniquely identify?

- To enable users to retrieve objects without knowing their location.
- To enable repositories to change the location of objects internally.
- To enable repositories to share objects with other services where appropriate.
- To enable researchers to cite objects consistently over time.
- To enable repositories to associate entities unambiguously (one to many, many to one or one to one):
    - Objects with objects.
    - Metadata with objects.

Adding persistent identifiers: when to identify?

Labels to identify an object can be added at various times during an object's life, and an object could have several identifiers, simultaneously or consecutively, for different purposes, just as individuals are allocated identifiers by the various systems in which they participate.

Key stages at which an identifier might be added include:

Original object creation
There may come a time when donors provide materials that already possess persistent identifiers. Whether or not these identifiers are retained may depend on the scheme selected by the donor; there might be an ongoing cost associated with doing so. It is possible to associate more than one identifier with an object, so one option may be to use the donor's identifiers in addition to identifiers allocated by the repository. Quite often in paper archives indexing references employed during the records' active life are retained alongside new cataloguing arrangements. If this is done, it should always be clear to the user that the earlier identifier has been superseded by the current, repository-assigned one.

Object creation via migration
If adopting a migration preservation strategy (see Chapter 08 *Digital preservation strategies* p. 235), then the repository may also need to create one or more representations of the original object in more accessible formats. These objects will need their own identifiers and their relationship with the intellectual entity they represent, and thereby with other representations (especially the representation from which they derive), must be expressed.

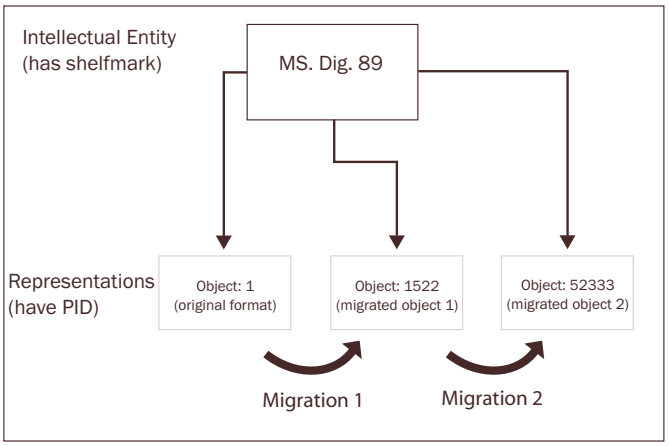

Figure 8: Migration

Object ingest
While archives arrive without persistent identifiers, it seems that ingest (where the repository takes intellectual control of the archive and processes it for preservation) is the most appropriate time to allocate persistent identifiers to the objects to be preserved and for the metadata created about them. If the objects are to be kept in a local dark archive, the identifiers need only be unique within that environment. In a traditional archival context, identifiers (such as reference codes or shelfmarks) are not usually applied at item level until an archive is arranged or described, although a unique accession number is usually assigned to collections or instalments of a collection when they are taken in. In the case of digital archives, some form of item-level identifier must be allocated sooner than this so that preservation metadata can be associated with objects and the original order of an accession can be recorded.

Object dissemination
When Dissemination Information Packages (DIPs) for reader access are prepared, it may be that objects are moved from the restricted dark preservation archive into a more open repository. If publishing objects to an online repository, or allowing others to harvest objects (or their metadata)

for use in their own repositories, the repository must ensure that the identifiers it uses are globally unique. It is this global uniqueness that many of the PID schemes for digital objects aim to provide. It may be useful to employ identifiers which conform to one of these PID schemes even when objects are held in a local dark archive, in order to ease the transition from the closed to the open environment.

Events and agents
Events which need to be recorded must have identifiers as part of their metadata, as must agents associated with events. This allows the repository to associate events with relevant agents, and events with objects to which they pertain.

Adding traditional identifiers, such as reference codes or shelfmarks

It is useful to retain traditional identifiers (such as shelfmarks or archival reference codes) as well as persistent identifiers. PIDs will allow the repository to manage the identity of objects and metadata in a digital environment from the moment an archive is ingested, while shelfmarks can be allocated to digital and analogue items when the archive is being catalogued. The continued use of shelfmarks for digital records in addition to persistent identifiers allows:

- The integration of analogue and digital materials in hybrid collections in a single catalogue.
- The use of familiar references for staff and researchers.
- Some meaning, logic or structure to be incorporated into an identifier.
- The retention of context and provenance.
- The ability to retain one shelfmark for the object which can have several representations.

Shelfmarks or archival reference codes should themselves be persistent and unique, in accordance with ISAD(G)2 (see p. 59).

Implications for researchers

The most important issues for researchers are not likely to be with the shelfmarks and PIDs themselves per se, but with understanding the relationship between different representations of the same manuscript and knowing how to cite them. The repository will need to provide guidance in citation, and those responsible for the design of access mechanisms should build an understanding of an object's digital provenance into the navigation system.

Discovery

The use of PIDs could allow links to a digital object repository directly from an online EAD catalogue (see Chapter 06 *Arranging and cataloguing digital and hybrid archives*), without fear that such a link will break when an object is inevitably moved. The PIDs used in the EAD catalogue should be associated with the Intellectual Entity, rather than the original version of the object. When clicking on the link the user could be presented with a metadata 'title page' for the digital manuscript, so that the original and new representations of the object in question (with their own PIDs), and any metadata about them, can be referenced from a single place. It is also possible that staff and researchers would benefit from a look up table which resolves shelfmarks into PIDs. Having initially encountered the Intellectual Entity via an EAD catalogue, the researcher may also wish in future to return to it directly, e.g. by entering its PID straight into their web browser, rather than going via the catalogue entry.

Citation

Institutions will need to provide citation advice to researchers in the form of worked examples. For digital objects best practice will be to cite the PID using the repository's preferred form along with the item's shelfmark. The shelfmarks will ensure consistency for collections with both a digital and

an analogue component while appending the PID which will allow others to link directly to the exact version of the digital source described by the researcher.

Administering identification

- The repository needs a process to assign identifiers as part of the AIP and DIP creation.
- There must be a procedure in place for updating the locations held for identifiers in the resolver to ensure that identifiers continue to be resolved to the correct location.
- Take-down procedures may be required for instances where a migrated version of an object is no longer the preferred representation. The object and its PID may be retained in the digital archive, but removed from the published repository.
- The repository may need procedures in place to renew subscriptions to relevant services, depending on the methodology selected.
- Responsibility for various aspects of allocating and administering PIDs should be assigned and internal policies produced.
- Comprehensive naming guidelines should be produced to ensure consistency, outlining naming policy, syntax, permitted characters, etc.

The impact of the Web on the relationship between names and addresses

Identifying and locating resources on the Internet and World Wide Web

The Web environment problematises the issue of associating specific digital resources with persistent identifiers and locations because of the ease with which web-based material can be moved and altered.

Before considering the problem of naming and addressing web-based resources, however, it is useful to note the distinction between the Internet and the World Wide Web – terms which are often used interchangeably. The Internet refers to the structure of interconnected computer networks that communicate using Internet Protocol (IP) and Transfer Control Protocol (TCP). The vast collection of interlinked hypertext documents which makes up the Web runs over this network, and access to web documents is provided by Hypertext Transfer Protocol (HTTP) .

The earliest address mechanism used for the Internet was the IP address; IP addresses are commonly written as numbers between 0 and 255, separated by dots. This system is not user friendly, and to address this issue the Domain Name System (DNS) was introduced. This system enables human-readable names to stand in (as indirect names) for IP addresses; multiple IP addresses can then be assigned to a single domain name or vice versa. Domain names take a hierarchical structure and work from right to left. The top-level domain might be a country or community, as in .uk or .fr which appears at the end of a domain name; the next authority might indicate an institution or a sector, like .co. for companies, .ac. for the academic community, etc. Further subdomains are created by the domain owner.

URIs, URLs and URNs

Anyone who uses the Web will be accustomed to locating, requesting and citing digital resources by their Uniform Resource Locator (or URL); however, they may be less familiar with the terms URI and URN.

A Uniform Resource Identifier (URI) is essentially a string of characters used to name or identify a resource, and it can act as a name, a locator or both. A resource is anything that can be identified

by a URI, and does not necessarily have to be in digital format or available via the Internet (e.g. a human being or an abstract concept could be assigned a URI). The URI specification does not require that a URI persists in identifying the same resource over time, although this is a major aim of most URI-compliant schemes. URI identification is carried out by means of an extensible set of registered naming schemes, maintained by the Internet Assigned Numbers Authority (IANA).[1]

URI syntax
A URI has a standardised syntax. The permitted characters come from a limited set comprised of the letters of the basic Latin alphabet, digits and a small number of special characters. It is organised hierarchically, from left to right (rather than right to left as with the DNS), and takes the following form:

[Scheme]:[//][Authority]/[Path]?[Query]#[Fragment]

Hypothetical example:
http://personaldigitalarchive.ac.uk/archive/accession1?ABC#123

Scheme This component is required and is used to identify the scheme being used in the URI (e.g. the HTTP protocol). This is usually given in lower case and is separated from the rest of the URI by a colon. It is followed by the scheme-specific part of the URI, which is largely governed by the specifications of the relevant scheme, although the URI imposes some constraints to ensure consistency.

Authority Many URI schemes include an element for a naming authority, which governs the rest of the namespace in the URI. It is optional and is preceded by a double slash. It can include three subcomponents: userinfo (e.g. user name and scheme-specific information about how to access the resource); host (IP address or registered name); and port (port number).

Path A required component which begins with a slash and contains hierarchical data that (along with the query component) identifies a resource within the scope of the relevant URI scheme.

Query An optional component which contains non-hierarchical data that serves to identify the resource within the scope of the relevant URI scheme.

Fragment Another optional component, which allows indirect identification of a secondary resource by reference to a primary resource and additional identifying information. The secondary resource could be a subset of the primary resource, such as an image that is a constituent file in a web page, or it could be a view of the resource, perhaps the result of a query to a database.

URIs are the principal identifiers used in the Web environment. Whilst a URI can serve the purpose of both naming and locating, these two functions have essentially been separated into two URI subsets: Uniform Resource Locators (URLs, which are in general usage to describe web-based resources) and Uniform Resource Names (URNs).

URLs are used by the HTTP protocol for addressing documents and are intended only for locating resources. In addition to the address protocol used (http://) a URL also contains a network path that includes the domain name or IP address, and further optional paths and parameters. URLs are widely used as identifiers but they are inherently unstable. Users of the Web will be used to the frustration of broken links and error messages resulting from the removal of documents and constantly shifting locations. The recognition of this problem resulted in the search for a persistent identifier scheme, and URNs were introduced as globally unique, persistent identifiers that are independent of location; they have been formally defined and are discussed in more detail below (p. 59). Other persistent identifier schemes have also been established.

1 Internet Assigned Numbers Authority, 'Uniform Resource Identifer (URI) Schemes', *Internet Assigned Numbers Authority website*. URL: <http://www.iana.org/assignments/uri-schemes.html>

Persistent Identifier schemes

Introduction

Persistent identifiers have existed for a long time in the library world in the form of schemes like the International Standard Book Number (ISBN) and Library of Congress Control Number (LCCN). Their importance is also recognised in the archive world: ISAD(G)2 sets out the requirements for a globally unique and persistent archival reference code. At collection level, this consists of: a country code, in accordance with ISO 3166; a repository code in accordance with the national repository code standard (in the UK, this is the ARCHON reference); and a specific local reference code.

Example from the John Rylands University Library: GB-133-RMD.
This is the collection-level code for the Papers of Ramsay MacDonald. The local code is based on a 3-letter mnemonic, and a register of mnemonics is maintained centrally so the same code is not used twice.

Since moving into the digital age the issue of PIDs has become increasingly important, and recent debates over PIDs largely focus on the issue of actionable or resolvable identifiers, i.e. unique names which not only persist over time but will also take the user to the identified archival digital object or to information about that object.

A number of PID schemes are in use in the digital environment; these have been developed by different communities, and some are more suited to certain contexts than others. An institution or sector may find that some elements from a number of different schemes are suitable for their use rather than any one scheme in its entirety; conversely, in some cases none of the existing schemes may be appropriate, in which case an institution may consider implementing an independent solution. If taking the latter approach, the importance of future interoperability should be taken into account; a local scheme should be capable of integration into any wider schemes in operation (either within the same community or globally), which means that adhering to open standards is important. Most of the PID schemes currently in place are still under development, and it may be that some of them do not survive in the long term; however, this situation does give organisations the opportunity to feed into the development of identifier schemes and to work with similar institutions to develop shared solutions.

Probably the most important component of any PID scheme in ensuring persistence is commitment from the organisation, and ensuring that the scheme is administered comprehensively – ultimately a PID can only be associated permanently with any one resource if there is a commitment to maintaining that link; it cannot happen automatically.

An introduction to some of the principal PID schemes follows. This includes: their historical background, how they work, their syntax, the extent of their adoption and who maintains them. The advantages and disadvantages of each scheme are considered from the perspective of an institution which collects the digital and hybrid personal archives of significant individuals such as politicians. No single scheme is picked out as the best, because each institution is likely to have slightly different practices and requirements. Given that such a large proportion of personal archives are likely to be closed for long periods because of data protection and copyright restrictions (see Chapter 09 *Legal issues*), the best approach may be to implement a basic local identifier system which complies with the URI and URN standards and await the widespread adoption of a universal URN resolver.

Uniform Resource Name (URN)

Background

The URN concept originated in 1994 with RFC 1737, 'Functional Requirements for Uniform Resource Names', and a syntax was developed in 1997 (in RFC 2141). The functional requirements include:

- Global scope and uniqueness (one unique name for a specific resource, which has the same meaning everywhere).

- Persistence of the name, regardless of the longevity of the resource it identifies.
- Scalability (the scheme should be able to deal with the quantity of identifiers which will be produced).
- Legacy support (it should accommodate other existing identifier schemes).
- Extensibility.
- Independence (i.e. of the name-issuing authority that is responsible for the scheme).
- Resolution (i.e. if the URN identifies a networked digital resource, it should also enable the user to locate and access the resource, or information about the resource).

URNs are therefore intended as persistent, location-independent resource identifiers, with the capacity for incorporating many different identifier schemes. They remain globally unique and persistent even when the resource becomes unavailable or ceases to exist.

URN syntax

A URN always begins with the 'urn' declaration followed by a colon, and takes the following form:

URN:[Namespace Identifier (NID)]:[Namespace Specific String (NSS)]

Example (urn representing the ISBN of a Hansard Society publication called *A Strategic Guide for Online MPs*):
urn:isbn:0900432160

NID The Namespace Identifier (NID) identifies which namespace is being used, and it comes from the URN Registry maintained by the Internet Assigned Numbers Authority (IANA).[1] This registry lists existing naming schemes, by no means all of which were created specifically for the digital environment, e.g. ISBN, International Standard Serial Number (ISSN) and National Bibliographic Numbers (NBN, a namespace assigned to national libraries for integrating different identification schemes into the same identifier namespace). The NID can consist of letters, numbers and hyphens.

NSS The Namespace Specific String (NSS) follows the NID and is preceded by a colon. The NSS can consist of any characters which may have to be encoded, using the same encoding method as URLs, and the form it takes is dependent on the namespace it comes from (e.g. a string of numbers in the case of ISBN).

Resolving URNs

URNs were developed to be independent of any one resolution service. A number of different approaches have been proposed, although as yet there is no universal resolution service for URNs. This poses an obstacle to their widespread adoption, and there is also some disagreement within the Web and Internet community over whether URNs are necessary at all.

Maintenance and adoption

In order to use URNs as persistent identifiers, an organisation can either work within an existing URN initiative which has been assigned a NID, or (where a new, globally unique approach to identifiers has been developed) obtain a new NID, through a standardised application procedure. The URN Registry gives an indication of URN uptake to date; it includes some major international identifier schemes for resources, people and organisations.

1 Internet Assigned Numbers Authority, 'URN Namespaces', *Internet Assigned Numbers Authority website*. URL: <http://www.iana.org/assignments/urn-namespaces>

Advantages and disadvantages of URNs

Advantages

- URNs are flexible and easy to construct: the NSS can take any form, meaning that other namespaces can easily be mapped into URNs, yet global uniqueness is ensured as long as the NSS is unique within the NID.
- The URN is an open standard and is technology independent.
- Whilst no universal resolver of URNs has yet been developed, they can be used with the DNS and HTTP: a URN can be coded into a URL, and a proxy server used to route URN requests to a host server, enabling users to resolve URNs using a standard web browser.

Disadvantages

- The lack of a universal resolver has hindered the takeup of URNs.
- The ongoing lack of consensus about the value of URNs means that there may be a question mark over their long-term future.
- Existing NIDs may not be suitable for dealing with personal digital archives, so it may be necessary to establish a new NID before using the URN system; this involves developing (to a detailed level) a new identifier scheme and submitting it before joining the queue of NIDs awaiting approval.

Persistent Uniform Resource Locator (PURL)

Background

The PURL system was developed by the Online Computer Library Center (OCLC) in the USA. Its origins lie in library cataloguing applications: PURLs were first implemented in 1996 in the Internet Cataloguing Project – which aimed to advance practice and standards for cataloguing internet resources, and addressed the issue of including URLs in cataloguing records. PURLs were developed as an interim measure until URN technology is fully developed and web browsers are able to recognise URN syntax; they are designed to be automatically translatable to URN architecture when this is established and satisfy as many of the URN requirements as possible using current technology. PURLs provide both a means of identifying a 'general internet resource' and of locating and resolving that resource.

See OCLC's PURL site at <http://purl.org/> for more information.

How do PURLs work?

A PURL is essentially a URL; the difference is that it does not take the user directly to the location specified by the URL, but to an intermediate PURL Resolution Service. The resolution service associates the PURL with the relevant URL and returns that URL to the user, who can then access the server direct and retrieve the resource.

The creation of PURLs is straightforward and involves a simple web-based application procedure. Once created PURLs exist permanently, although they can be disabled. In order to ensure persistence, any changes in the associated URL need to be made by the creator or owner of the PURL, but the PURL always remains the same. If the link from a PURL to its associated URL is broken because the URL is moved, the PURL and its full history will still be available as long as the PURL Service itself is maintained.

PURL syntax

A PURL identifier takes the following form:
[Protocol]/[Resolver Address]/[Name]

Hypothetical example:
http://purl.abcd.org/ABC/DEF/200

Protocol PURL uses the HTTP protocol.

Resolver Address PURL uses Domain Name Services (DNS) to obtain the IP address assigned to the resolver, e.g. the OCLC resolver address is purl.oclc.org and the National Library of Australia's is purl.nla.gov.au.

Name The Name is assigned by the organisation or individual creating the PURL. Either upper or lower case can be used, although some characters are not permitted. A full list of the characters allowed in the Name component is given in the FAQ page of the PURL site.[1]

Names are organised as a hierarchy of domains (like directory paths), with a top-level domain name separated by a slash from further sub-domains (each separated by a further slash). In the above example ABC reflects the top-level domain, DEF a subdomain of ABC, and 200 the specific document being identified.

In theory, the same name could be given to two different documents; it is the resolver which makes the identifier unique, and it is not possible to create two identical names under the same resolver. The resolver's database contains details of all assigned names, which can be checked before creating a name. This system ensures that the name is unique within its namespace.

Resolving PURLs

Resolution of PURLs is carried out using standardised HTTP redirect by means of the OCLC (or another) PURL resolver. A PURL only resolves to a single URL.

Partial redirections can also be set up, using a domain as the prefix for a localised hierarchy of URLs. If the resolver finds no direct match for a particular PURL, it tries to match it right to left based on the hierarchy of domains represented in the syntax; it then resolves as much as it can find and appends the remaining unresolved portion to the end of the resolved URL.

> *Hypothetical example:*
> A partial redirect is set up for the URL <http://personaldigitalarchive.ac.uk/>. The PURL associated with this is <http://purl.organisation.org./ABC>.
>
> If this URL includes a lower level document which has the URL <http://personaldigitalarchive.ac.uk/a/very/long/document>, and a match for the full name is not found, the resolver will try to find a match by first of all taking away /document, then /long etc. It will eventually find the match for <http://personaldigitalarchive.ac.uk/>, and will automatically append /a/very/long/document, resulting in the PURL: <http://purl.organisation.org/ABC/a/very/long/document>.

This means that an organisation might create a partial redirect as the permanent prefix for an entire website and its components. Users would use the partial redirect as the prefix for all the documents forming part of that site; if the site is moved, only the single partial redirect location needs to be changed, rather than the full URL for each document.

Access and use

In order to become a registered user of a PURL resolver, an individual has to create a user ID and password on that resolver by following the given instructions. A registered user can create a PURL by using an online creation form, as long as the top-level domain of the name exists; if it does not, a request to create a domain name must be made to the resolver's administrator.

It is intended that some degree of access control will ultimately be possible, although this has not yet been implemented. Currently all PURLs are universally resolvable, which means that they can be searched and resolved by any unregistered user. In future, it should be possible to create privately resolvable PURLs, domains and partial redirects which can only be resolved by designated registered users of the particular PURL resolvers where they reside; different levels of access (e.g. read, write, maintenance) could be set for different types of user.

1 Online Computer Library Center, 'PURL Frequently Asked Questions', *PURL website*. URL: <http://purl.oclc.org/docs/purl_faq.html>

Maintenance and adoption

OCLC maintains the PURL server software, and it is made freely available to anyone who wishes to establish their own sub-domain and maintain their own PURLs. To date over 600,000 PURLs have been created.

OCLC has made the source code for PURL available, which means that institutions can install their own PURL resolvers. There is no current list of all the institutions which have set up their own PURL servers, although those which have include the National Library of Australia, the Danish Bibliographical Centre and the U.S. Government Printing Office.

Advantages and disadvantages of the PURL scheme

Advantages

- It is cheap and easy to create and resolve PURLs; making use of existing services means that no new protocols or modifications to client software are necessary, and the software is freely available.
- The system is standards based and compatible with both URI and URN schemes.
- PURLs grew from a library cataloguing context and they could provide an effective means of linking from an EAD catalogue entry to the associated DIP.
- The scheme is now well-established and widely used.
- It is scalable: by using the existing distributed technology of DNS/HTTP, many different PURL servers can be established locally, thus avoiding the overloading of servers and enabling greater local control over PURL creation.

Disadvantages

- PURLs were designed primarily as identifiers for open, web-based resources (essentially 'published' material), but digital archives have different requirements. Repositories for personal digtial archives must identify closed or restricted access material and various metadata. They would therefore need to implement PURLs locally in a manner that prevents access by unauthorised parties.
- They are incapable of dealing with the complexities of any single personal digital archive, which may require different levels of access (e.g. some items may be closed, others subject to access restrictions and others open).
- In a personal digital archive each individual object must be unambiguously identifiable, so a facility like partial resolution is inappropriate.

Handle System

Background

The Handle System was developed in America by the Corporation for National Research Initiatives (CNRI) as part of the Computer Science Technical Reports (CSTR) project; this project, which was funded by the Defense Advanced Research Projects Agency (DARPA), ran from 1992 to 1996. It involved developing and providing network access to a corpus of digitised material from the collections of computer science technical reports held by five major universities; part of the project's work involved developing an architecture for an open distributed digital library (as described in a paper by Robert Kahn and Robert Wilensky in 1995[1]). One of the key concepts which emerged from the project was the idea of handles to provide unique, location-independent persistent identifiers for digital objects. The Handle System was first implemented in autumn 1994.

The system is a general-purpose naming service which provides a mechanism both for assigning persistent identifiers to digital objects and resolving these identifiers to provide users with access to the information necessary to locate, access or otherwise use the digital object identified by the Handle, or (where appropriate) to the resource itself. Information about a digital object's current location is stored in the Handle records, meaning that when this location changes, only the Handle record (rather than the Handle address) needs to be changed.

1 Robert Kahn and Robert Wilensky, *A Framework for Distributed Digital Object Services* (May 1995). Handle: cnri.dlib/tn95-01. URL: <http://www.cnri.reston.va.us/k-w.html>

The Handle System was designed to work independently of the DNS, although it can also work successfully within it.

See the Handle System website[1] for more information.

How does the Handle System work?

The Handle System is comprised of three different elements:

- A set of protocols, which defines how a distributed computer system can store and resolve Handles into the information needed by users to locate, access and use digital or other resources.
- A namespace.
- A reference software implementation of the protocols.

In order to maintain its independence, the Handle System is not based on DNS root servers; it has its own root server, the Global Handle Registry (GHR). The GHR provides the service used to manage lower-level Naming Authorities (NAs). Each NA is an organisation with administrative responsibility for creating and managing Handles within a specified namespace (which may have any number of sub-namespaces) and each local namespace is managed by a Local Handle Service; all of these namespaces must be registered with the GHR. Local Handle Services are reliant on a local Handle server, which can be downloaded and installed by system administrators in a similar way to installing a web server. The Local Handle Service can then establish its own local infrastructure, e.g. it might scale up by adding more servers at local level; there are no limits on the number of sites or servers which make up a local service.

It is recommended that a Handle server should be installed on a machine with an internet presence because the GHR needs to be able to contact the local server. It may, however, be possible to configure the server so that two different IP addresses are used to distinguish between internal and external access.

Handle syntax

A Handle identifier is divided into two parts: a prefix and a suffix divided by a forward slash, taking the following form:

[Handle Naming Authority]/[Handle Local Name]

> Hypothetical example:
> 19123.11/object200

Handle Naming Authority Each NA is assigned a number by the GHR which is globally unique within the Handle System. These numbers are decimal and are assigned sequentially. Each NA can authorise any number of sub-NAs, and a dot (.) is used to express this hierarchy, which should be read from left to right; e.g. in the fictional example above, 19123 would represent the higher-level NA, which might be a national or university library; 11 might be a particular project, programme or department within that institution, although sub-NAs do not necessarily have to be administratively dependent on their parent NA in any way.

Handle Local Name The Local Name is assigned by each individual NA in accordance with its own policies. The Handle System sets no limitation on the syntax of the Local Name, although it must be expressed using characters from Unicode's UCS-2 character set. It should also be unique within the NA, making it globally unique within the Handle System.

Resolving Handles

The Handle system is not limited to naming; it also enables users to resolve Handles into the information necessary to locate, access and use an identified resource (e.g. metadata about a resource,

1 Corporation for National Research Initiatives, *Handle System website.* URL: <http://www.handle.net/>

a request form to apply for access, or information about the location of the resource), or take them directly to the resource itself. The GHR maintains a record of all NA prefixes. If a user wishes to resolve a particular Handle, they send a request to the GHR, which identifies (by its prefix) the NA which assigned the Handle and returns this information to the user, who can then access the relevant Local Handle Service directly, e.g. if a user wants to find out which local service is responsible for the Handle 34567/890, they send a request to the Global Handle Registry to resolve the NA Handle for the prefix 34567.

Caching servers can be associated with local servers: these allow frequently-accessed Handles to be stored and resolved without contacting the GHR.

Each Handle may have a set of values assigned to it in the form of a standardised metadata record which contains information about how the Handle and resource it identifies are accessed and administered, e.g: a detailed set of read/write/execute permissions applicable both to the Handle administrator and the user; location information; and a description of the resource identified by the Handle. The Handle value may be 'privatised' so that only the administrator has read-access, thus keeping some of the data (e.g. location information) inaccessible to the public.

One unique feature of the Handle System is that Handles can refer to copies of the same document held at different locations, helping to assure access when servers are busy or there is high demand, although this is not a high priority for digital archivists, who are generally dealing with unique objects. However, it is possible to include in a Handle's value set references to other Handles which 'add credentials' to the Handle; this might be used to link various different metadata records to a digital object.

Using HTTP, Handles can be resolved by using the resolver service at <http://hdl.handle.net/> or simply appending a Handle to the URL <http://hdl.handle.net/>. The user may be redirected to a URL associated with the relevant resource, or be able to view a list of the Handle's values.

Maintenance and adoption

A number of projects and institutions are currently making use of the Handle System, including: the Defense Virtual Library – a digital library established in America by the Defense Technical Information Center (DTIC), the DARPA and CNRI; the Digital Object Identifier System (DOI), another persistent identification scheme which uses Handles as its naming component; the digital repository software DSpace, which uses Handles to name and provide access to document containers; and the Library of Congress in its National Digital Library Program.

In order to use Handles as persistent identifiers, an institution has to register and establish a Naming Authority. This involves signing a licence agreement with the CNRI, which maintains the system. Although the software is made freely available, in June 2006 a fee was introduced for those wishing to participate formally in the Handle scheme in order to cover operational costs for running the GHR. Registering for a Handle NA number costs $50 and there is an annual service charge of $50. While the registration fee only applies once no matter how many derived prefixes are registered, the annual charge is applied per prefix (e.g. three prefixes would incur a $50 registration fee and a $150 annual charge).

The CNRI hosts the Global Handle System root server. This is also overseen by the Handle System Advisory Committee, which has members drawn from both public and private sectors. The software for client and server can be found at <http://www.handle.net>.

Advantages and disadvantages of the Handle System

Advantages

- The Handle system was one of the earliest PID schemes to be introduced (contemporary with URNs), and is being used by a number of digital libraries and national institutions. It is maintained by a national organisation in the USA, so is stable and well-established.
- It conforms to the functional requirements of the URI and URN concepts, and is independent from, yet interoperable with, current protocols like HTTP.
- Handle syntax is straightforward and is also capable of incorporating existing local identifier systems.
- The system may be adaptable to the different levels of access required for managing personal digital archives: operations on the Handle database are controlled by a detailed authorisation mechanism for security of data; and Local Handle Servers can be configured to allow either internal or external access, which might enable use of Handles as identifiers in a closed environment.
- The distributed model means that local Handle services and NAs have autonomy to manage their own Handles.
- The system is scaleable and might allow smaller institutions to share a local service under the same NA.

Disadvantages

- Whilst not prohibitive, there is nevertheless an initial fee and an annual charge for those participating in the system, whereas ideally a PID system should be free and the software openly available.
- While there are authorisation mechanisms, the system still has a strong emphasis on identifying resources which are openly available via the Web, rather than held in the more restricted context of a digital archive.
- The system includes some optional metadata elements which are superfluous to the needs of a digital archive: extensive metadata is already produced for each digital object (using METS (see p. 117) and PREMIS (see p. 80) in the case of Paradigm), so the production of a value set for each Handle would therefore be unnecessary.
- The character set for Handles is much broader than is permissible for URIs, so institutional naming policies would have to place restrictions on the characters used in order to comply with URI requirements.

Digital Object Identifier System (DOI)

Background

The DOI makes use of the Handle System for resolving identifiers, but Handles are only one component of the DOI System, which provides a complete framework for managing digital objects, including a structured means of identification, description and resolution, along with policies, procedures, business models and application tools. It is designed to be independent of the DNS and HTTP protocol, although can be used with this system via the DOI proxy server at <http://dx.doi.org>.

The DOI System was developed as part of a project run by the Association of American Publishers and was launched in 1997 at the Frankfurt Book Fair. It grew out of publishers' concern about control of intellectual property in the digital environment. Its focus was initially on content identification (i.e. a unique identifier would be assigned to a work at the point of creation); however, it was recognised that the issue of persistent identification has value beyond the world of electronic publishing, and so the DOI was developed as a cross-industry and cross-sectoral non profit-making organisation, managed by the International DOI Foundation (IDF, founded in 1998). The system is intended to provide a generic framework applicable to any digital (or other) logical entity; and a DOI name may be assigned to any item of intellectual property, or the parties, events or agreements involved in an intellectual property transaction.

See the DOI System website[1] for further information; the site includes links to numerous overview documents, frequently asked questions and the complete DOI Handbook which contains policies, procedures and guidelines for participating organisations.

How does the DOI System work?

The DOI system consists of four principal components:

- A naming syntax.
- A resolution service, based on the Handle System.
- A data model, which includes structured metadata based on a data dictionary (the indecs Data Dictionary, or iDD) and a framework for using this.
- Policies and procedures for implementing DOI names in a social infrastructure.

DOI syntax

DOI syntax is specified by a NISO standard (ANSI/NISO Z39.84) and is similar to that of the Handle. The syntax takes the form of a prefix and a suffix divided by a slash, as follows:

[Directory Code].[Registry code]/[Local Name]

> Hypothetical example:
> 10.7890/object786

Directory Code The International DOI Foundation (IDF) is a Naming Authority under the Handle system; it has been allocated the number 10 as its unique identifier and this forms the Directory Code in the DOI namespace. All DOI names therefore begin with the number 10.

Registry Code The Registry Code (preceded in the syntax by a dot) is a unique number assigned by the IDF to an organisation that has been authorised to register DOI names – known as a Registration Agency (RA). Anyone wishing to assign DOI names must work through a RA, which is usually based on a particular 'community of interest' and any organisation representing this interest can apply to become an RA (e.g. the CrossRef RA provides citation-linking services for the scientific publishing sector). The role of an RA includes providing services and day-to-day support to registrants, e.g. the allocation of prefixes, registering DOIs, quality assurance. If there is no suitable RA for an organisation's needs, the IDF itself can act as the 'default' RA.

Local Name The local name suffix can be any alphanumeric string chosen by the registering organisation, which allows existing identification schemes to be incorporated into the DOI namespace. Any characters included in UTF-8 can be used, and the local identifier can go to a very granular level, e.g. identifying a paragraph within a larger document.

Resolving DOIs

Resolution of DOIs is carried out by means of the Handle System (see p. 63). DOIs do not have to resolve directly to the resource identified by the DOI, although they can do this.

In the past DOIs have generally been used to resolve to a single location (a URL, which might be a publisher's website, or a digital repository's website), thus providing a basic tool for persistence. However, DOI has now developed the capability to resolve to multiple associated data (e.g. a number of digital objects, metadata or repository information), which means that resolution can be much more granular. It is also possible to indicate relationships between digital resources (e.g. the same document in different formats, or earlier and later versions of the same document), by declaring related entities in the metadata for a DOI, or resolving from one entity to another. Whilst this is possible using the Handle System alone, the DOI provides a framework whereby relationships are defined through metadata using a semantically interoperable data dictionary.

1 The International DOI Foundation (IDF), *The DOI System website*. URL: <http://www.doi.org/>

The current location of each resource identified by a DOI is stored in the DOI system server, and any changes to this location must be registered there.

In the HTTP world DOIs can be resolved through the DOI resolver at <http://dx.doi.org> and through the global Handle resolver at <http://hdl.handle.net>. The DOI has also applied for a "doi:" URI scheme to allow a DOI to be expressed as a URI without the need to reference specific HTTP servers.

Metadata

The DOI System has a Data Model to ensure that every identified object is unambiguously described in a standardised way which facilitates semantic interoperability and consistency. It is not mandatory for DOI names to make use of this Data Model, although the scheme envisages that many will.

At the most basic level, the DOI Data Model allows a 'kernel' of basic metadata to be attached to a piece of intellectual property (on which optional extended metadata schemes can be built). This kernel declaration takes the form of an XML schema and contains 8 elements (drawn from the iDD) which include information about: what the object is; whether it has any other identifiers; what it is usually called; the identity of its creator or publisher; whether its location is digital, physical, etc; and what type of resource it is (e.g. audio file, pdf document). This kernel metadata at present only relates to 'creations' and different kernels would have to be defined for different types of resource or entity, e.g. people or events.

To provide more granular metadata which is common to a particular community, the DOI allows the establishment of Application Profiles (APs). These are a means of grouping together DOI names with common properties (e.g. they describe entities of the same format, share the same metadata schema, or the same rules for access and use); they ensure that a particular type of DOI name behaves predictably in an application through association with specified services. An AP comprises at minimum a set of structured metadata elements, as well as some rules about policy and procedure. Any existing metadata standard can be used in an AP, but the DOI requires that for full interoperability across the DOI system this should be mapped to the iDD. XML is recommended by the DOI both for kernel metadata and AP metadata extended from the kernel.

The DOI system may be used in a restricted or non-public environment; a 'Restricted' AP is used for this purpose. This ensures local good practice and also means that the private identifiers can easily be moved into the public realm (e.g. as archive material moves from a dark to a light archive at the expiration of copyright protection) without having to be altered or reassigned.

Maintenance and adoption

The central authority and maintenance agency for the DOI System is the IDF, which provides standards and a technical and social infrastructure for DOI users. The IDF is controlled by an executive board elected by members of the Foundation. Membership of the IDF is open to any organisation with a stake or interest in managing information in the digital environment. Current members include publishers, software companies and organisations which represent the interests of publishers or other IPR holders, e.g. the International Publishers Association, the Joint Information Systems Committee (JISC), the Online Computer Library Center, The Open University, and the national libraries of the UK, Germany and the Netherlands. Organisations pay an annual subscription, which varies according to categories of membership – although general membership is $35,000. The fee system was introduced so that the IDF can establish itself as a self-funding body in order to ensure long-term sustainability.

The IDF delegates and licences authority to use the DOI through Registration Agencies, each of which must be a member of the IDF. Each RA can determine its own local policies and make use of DOIs in appropriate ways for its own environment. While the IDF charges RAs an annual fee, it does not stipulate how that sum should be raised (e.g. by charging lower-level organisations for assigning a DOI).

The DOI System has had widespread takeup. Tens of millions of DOI names have already been assigned by several hundred different registered organisations. Many of these are operating in the commercial scientific publishing environment, but some publicly funded projects also participate in the scheme.

Advantages and disadvantages of the DOI System

Advantages

- The scheme is run by an established and robust organisation which is likely to be sustainable in the long-term.
- It has been adopted by libraries as well as commercial organisations.
- It provides an infrastructure for implementing a comprehensive digital identifier system, whilst leaving each RA with a considerable degree of autonomy to implement their own system, e.g. there might be scope for establishing an RA for those working with personal digital archives.
- The possibility of establishing a 'Restricted' Application Profile means that the scheme could be used in a non-public digital repository environment or dark archive as well as an open environment.
- It is standards based and DOI metadata is created using XML, both of which maximise interoperability.

Disadvantages

- There is a strong emphasis in the DOI member list on the commercial sector (e.g. publishing and software), or very large information institutions like national libraries. The annual subscription would be prohibitive for smaller libraries and archives. The alternative approach of working with a larger institution that has RA status may mean that the specific requirements of personal digital archives held in smaller institutions are overlooked.
- Whilst the DOI system offers a sophisticated data model which allows the creation of standardised metadata about digital resources and the grouping of resource-types into Application Profiles, these functions are probably superfluous to the needs of many curators looking after digital archives: extensive metadata is already produced for each digital object (e.g. using METS (see p. 117) and PREMIS (see p. 80) as metadata standards; both provide granular information and deal adequately with issues like relationships between digital objects), and services associated with the digital objects are likely to be managed by the repository. Given the costs involved in subscribing to the DOI system, an institution should probably only sign up if it wishes to take advantage of the full range of functions DOI offers; for the more basic needs of a digital archive, simple identifier systems are probably a more appropriate and cost-effective option.
- DOI currently recommends using the scheme to identify only resources, parties and events associated with intellectual property transactions, whereas Paradigm has identified the need for a wider range of identifiers – e.g. for preservation actions, or agents (repository staff or software) who have carried out preservation actions.

Archival Resource Key (ARK)

Background

ARK is a scheme for the persistent identification of information objects, which can include finding aids and other metadata as well as digital archival objects; however, it can also be used to assign a persistent name to other resources, e.g. physical objects such as books and intangible objects (examples given include diseases, vocabulary terms and performances). In this case persistent identification encompasses both naming and retrieval. It is the most recent of the schemes considered here and was originally developed by John Kunze and R.P.C. Rogers at the US National Library of Medicine. An internet draft outlining the scheme was issued in February 2001, and the current draft in July 2007; the scheme is currently maintained at California Digital Library (University of California).

ARK was developed as an alternative to schemes like PURLs, URNs and Handles, which address the problem of broken URLs by using a stable, indirect hostname scheme. Instead, the ARK scheme

is founded on the principle that persistence is a matter of service, not syntax – it is reliant on the continued stability and support of the service behind the identifiers.

As well as being a globally unique identifier, each ARK is an actionable URL, which links users to:

- A digital object (e.g. a content object which forms part of a personal digital archive), although the scheme acknowledges that this kind of direct access may not be feasible (e.g. in dark archive).
- Metadata about that digital object.
- A commitment statement by the provider.

For further information about the ARK scheme see <http://www.cdlib.org/inside/diglib/ark/>.

How do ARK identifiers work?

In order to assign ARKs, an institution must either become a Name Assigning Authority (NAA) under the scheme or be authorised to allocate names as a sub-authority of a NAA. Each NAA is associated with one or more Name Mapping Authority Hostports (NMAHs), which provide services (such as hosting, access or forwarding) for the digital objects being identified under the scheme; these essentially act as a temporary address where ARK requests are directed in order to make the ARKs actionable. A NMAH may change over time if service providers change and may serve more than one NAA (see below). A single institution can act as both NMAH and NAA; in fact, the scheme recognises that this will be common.

ARKs work well with current protocols like HTTP and DNS, but they are designed to be protocol independent.

ARK syntax

An ARK identifier takes the following general format:

[Protocol]/[NMAH/]ark:/[NAAN]/[Name]/[Qualifier]

> Hypothetical example:
> http://library.manchester.ac.uk/ark:/98765/archive/object35

Protocol This label does not form part of the ARK identifier, but indicates the protocol which is being followed (e.g. http://).

NMAH This part of the string identifies the relevant NMAH or provider of services; it is expressed as a hostname in the same format as a domain name which appears in a URL, e.g. library.manchester.ac.uk. This is mutable and does not form part of the unique ARK identifier.

ark:/ This prefix indicates where the actual ARK identifier begins. It, and the components which follow it, can be extracted and used in other identifier schemes (e.g. as part of a URN), and are easily recognisable by the ark: prefix. Following the ark:/ label are the components which make up the globally unique identifier for the digital object.

NAAN NAAN stands for Name Assigning Authority Number: each NAA is assigned a 5 or 9 digit decimal number as a unique identifier. This element of the ARK string is mandatory because it unequivocally identifies the organisation which assigned the persistent name of the digital object.

Name The Name is a mandatory element of the identifier and is assigned by the NAA. It should be comprised of ASCII characters, although there are four reserved characters which have special meanings; it should be unique within the NAA (ensuring its uniqueness within the system as a whole). The NAAN and the Name taken together form the immutable persistent identifier for the object.

Qualifier This is an optional component of the ARK, and the use of qualifiers (e.g. identifying sub-components or variants of a digital object) is determined by the relevant NAA or NMA. The ARK scheme specifies that hierarchies should be expressed using a path which separates each level with a slash. For example, in a digital archive this could be used to express hierarchies in a file structure. If the Name 3567 is assigned to a folder, the sequential files within that folder might be expressed in ARKs which look something like:

```
ark:/[NAAN]/3567/file1
ark:/[NAAN]/3567/file2
ark:/[NAAN]/3567/file3
```

Similarly, different variants of the same object can be specified by using qualifiers divided by dots. The NAA or NMAH determines what constitutes a variant. In an archival context it might be different representations of the same intellectual entity, or the same digital object in two different formats as a result of migration. Example:

```
ark:/[NAAN]/3567.t44.v23
ark:/[NAAN]/3567.232
```

Resolving ARKs

If a working NMAH is included in the ARK prefix, the user can be taken to the NMAH directly. If the NMAH no longer works (e.g. responsibility has been passed on to another institution), users can locate the new NMAH by identifying the NAA and using the register maintained by California Digital Library to look up current NMAHs that service ARKs issued by that NAA.

The ARK scheme also proposes an alternative method of locating the NMAH using a simplified version of the Name Authority Pointer (NAPTR) method of discovering URN resolvers, whereby a query is submitted to the DNS system requesting a list of resolvers matching a particular NAAN, and responses come back inside NAPTR records.

The ARK scheme also specifies a simple protocol for using HTTP to deliver ARKs, which is known as the Tiny HTTP URL Mapping Protocol (THUMP). It allows the user to enter ARK requests directly into the location field of their browser interface; once they have determined the internet host name and port number of the relevant NMAH, they can send questions to this via a THUMP request (contained within an HTTP request) and receive answers via a THUMP response (in an HTTP response).

ARKs can resolve to the object or object metadata, basic information about the object (who, what, when, where, etc., in relation to the object) as well as a commitment statement which could encompass statements about object permanence, variance (e.g. the conditions under which the object could change, such as format migration) and change history, etc.

Maintenance and adoption

The ARK scheme is currently maintained at California Digital Library (University of California). The NAAN registry (listing NAAs and their associated NMAHs) is also maintained by CDL[1] and mirrored at the NLM. The list of registered NAAs gives an indication of the ARK user community. In January 2007 twenty institutions were listed. Most of these are American, including the Library of Congress and several leading university and digital libraries. France's Bibliothèque Nationale is also a participant; the only UK organisation represented is the Digital Curation Centre. The scheme therefore has some backing among information institutions and the public sector.

The cost of participating is low; there is no subscription fee involved. Any institution can obtain a NAAN by contacting CDL and can then begin generating ARKs; this can be done using any software which produces identifiers that conform to the ARK specification; CDL uses a piece of open-source software called 'noid' (nice opaque identifiers).

1 Name Assigning Authority, *Name Mapping Authority Lookup Table* (12 January 2006). URL: <http://www.cdlib.org/inside/diglib/ark/natab>

Advantages and disadvantages of the ARK scheme

Advantages

- The scheme is standards based and protocol/technology independent.
- It works well either as a simple identification scheme, or as a system for both identifying and accessing digital objects.
- ARKs can be used to identify different types of entity, e.g. they could be used to identify agents and events as well as digital archival objects and metadata records.
- The system was developed in a library context and is designed to meet the needs of digital archivists.
- ARKs can be used in both a closed environment like a dark archive or an open publicly-accessible environment.
- The ARK system makes explicit the importance of organisational commitment to a persistent identifier scheme and writes a requirement for this into the scheme itself.
- It is maintained by a leading institution in the field of digital preservation and has no commercially motivated background (like DOI).
- The model for participating in the ARK scheme is more flexible than some of the other PID schemes: if one institution acts as both NMAH and NAA, it is able to have complete control over its own identification scheme; the possibility of multiple NAAs being connected to one NMAH might also enable one institution to host the digital archives of smaller, less well-resourced institutions.
- The technical requirements for participation are relatively low: currently a normal web server using the DNS.
- Because the scheme is still under development, institutions which choose to participate now can feed into and shape this development.

Disadvantages

- Because the ARK scheme was so recently established, it is difficult to gauge at this stage how popular and long-lived it might be.
- Some elements of the scheme are probably superfluous to the requirements of digital archives, e.g. hierarchies and variants can be defined using METS (see p. 117) and PREMIS (see p. 80) metadata rather than complex identifiers. In reality, it is probably more straightforward to use a simple single-level sequence of identifiers.
- Most institutions are moving towards encoding metadata in XML, which is intended to be reasonably human-legible and facilitates the sharing of data across different information systems. The use of Electronic Resource Citation (ERC) for recording ARK metadata (as recommended by the scheme) may involve both duplication of metadata and the additional task of converting it into a different format.

Identifiers and the Fedora repository software

The Fedora repository (used by Paradigm in its protoype preservation repository, see Chapter 07 *Digital repositories*) assigns identifiers to all the objects ingested into it; these identifiers are unique within the local Fedora repository and may suffice for a dark archive. If the material is to be made available in another namespace, additional identifiers may be necessary. Further information, including rules for the formation of Fedora identifiers, is available at <http://www.fedora.info/definitions/identifiers/>.

Fedora digital objects

The Fedora digital object model represents each object as a container which includes the following basic components:

- **PID** – a unique identifier for the object and its container.
- **Object Properties** – key object metadata to enable management of the object within the repository.
- **Datastreams** – one or more to hold the content or metadata items.

- **Disseminators** – optional pointers to methods to allow runtime representations to be generated.

Fedora identifier syntax

Fedora identifiers take the following form:

[*PID*] / [*Component Name*]

where *PID* is of the form [Namespace Identifier] : [Object Identifier]

```
Hypothetical example(s):
aNamespace:Object23          - unique identifier for the object (PID)
aNamespace:Object23/DC       - component (Dublin Core metadata datastream)
aNamespace:Object23/file1    - component (content datastream)
```

Namespace Identifier The namespace in which the object identifier is unique. For example:

- aNamespace.

Object Identifier A unique string for the object in the namespace. For example:

- Object23.

Component Name A string which identifies a component of the object, this references either:

- A datastream containing:
  - Metadata - aNamespace:Object23/DC.
  - Or a digital file - aNamespace:Object-23/file1.
- Or a method-call to access a run-time service:
  - aNamespace:Object-23/service-tobecalled.

Preservation metadata

Introduction to preservation metadata

This section of the chapter introduces some of the thinking behind preservation metadata for digital materials and applies this to a repository designing a preservation metadata application profile for personal digital archives. It begins with a general overview of preservation metadata issues, highlights some notable thinking in the evolution of the subject, provides an in-depth examination of the use of the *PREMIS Data Dictionary (v 1.0)* as a means of structuring generic preservation metadata and supplies information about schemes designed to meet the technical needs of distinct categories of digital material, such as still images and audio files.

Archivists must create, manage and use preservation metadata in order to administer and maintain access to authentic digital archives, their context and provenance over the long-term. The mechanisms needed to realise these concepts must change in the digital environment, where archives are both technology dependent, interdependent and easily mutable. Paradigm assumes that a repository with a remit to preserve personal digital archives for historical research means to preserve the integrity of the Intellectual Entities in the digital archive and their inter-relationships, not simply an arbitrary collection of digital files and folders placed at the repository.

Intellectual Entities are the conceptual items that will be described by archivists and accessed by researchers. An example of an Intellectual Entity is ‘the personal website of politician X, 26 January 2007’. This website consists of a precisely arranged series of interrelated files and folders, which together produce a Representation of the website. It would be possible to preserve the files that

compose the website without preserving their relationships, but doing so would make it near impossible to recreate a Representation of the Intellectual Entity (the website) for researcher access. To preserve meaningful access to digital archives, we must therefore do more than preserve files.

In the digital environment, Intellectual Entities may acquire several Representations over time as a result of preservation actions: if a file format migration is adopted as a preservation strategy, then each time a File belonging to a Representation is migrated, a new Representation of the Intellectual Entity to which it belonged is created (see p. 55 and p. 235).

This illustration shows the relationships between an Intellectual Entity, its Representations and the Files that belong to those Representations:

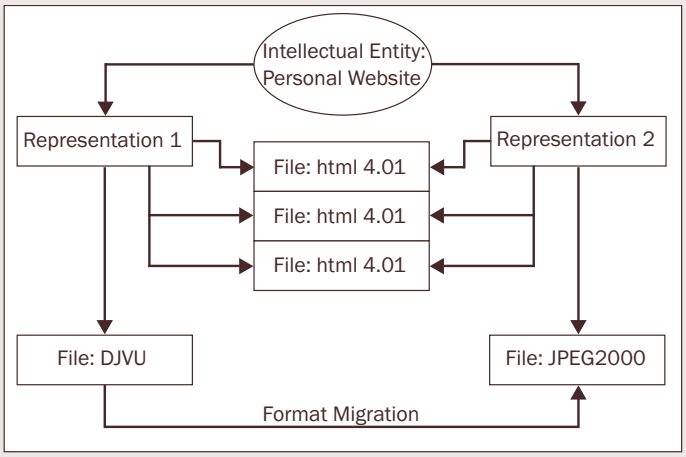

Figure 9: Intellectual Entity, Representations and Files

It is an authentic Representation of the personal website (Intellectual Entity) and its relationship to other Intellectual Entities in the personal archive that the repository wishes to preserve. In order to preserve the personal website, its constituent files must be preserved and metadata about their structure must be created and stored at the Representation level. If one or more of the group of files that constitutes the original Representation must be format-shifted in order to preserve access to the Intellectual Entity, then a new Representation of the same Intellectual Entity is created. In this example, the repository does not support the DJVU image format and has elected to migrate this File to JPEG2000 format, thus creating a new Representation of the personal website. In this way, the ability to reconstruct the Intellectual Entity (and the relationships between the original and subsequent Representations of the Intellectual Entity) is maintained as technology changes. The relationship between the personal website and the other Intellectual Entities in the personal archive is not illustrated here, but must be captured in a METS structure map (see p. 133) detailing the intellectual arrangement of the archive and held with metadata applicable to the collection level.

This model of Intellectual Entities, Representations and Files derives from the *PREMIS Data Dictionary for Preservation Metadata 1.0*, which will be described in more detail below (see p. 82).

Designing a preservation metadata application profile

Repositories will be reliant on metadata to manage born-digital archives in a way that ensures their longevity and proves their integrity; this requires the development of a detailed preservation metadata application profile and guidance in its implementation. When designing such a profile it is useful to articulate the activities that the metadata must support.

Activities the preservation metadata application profile must support

A profile for personal digital archives should enable the repository to undertake, and ultimately automate, manifold activities required to preserve authentic versions of the Intellectual Entities which compose a personal digital archive. It should allow the repository to:

- Identify, locate, understand and render Intellectual Entities over time.
- Record an audit trail of actions performed on Representations of Intellectual Entities for authenticity purposes, including details of which agents undertook events and when.
- Understand the relationships between different Representations of the same Intellectual Entity (e.g. where migration has created a new Representation of an Intellectual Entity, its relationship with the original Representation must be recorded).
- Record the effects of preservation actions on Representations of Intellectual Entities.
- Monitor the robustness of individual Files which compose Representations of the Intellectual Entities.
- Identify Files, and therefore Representations of Intellectual Entities, 'at risk' from obsolescence.
- Perform batch preservation actions on files conforming to a type, which result in new Representations of the Intellectual Entity.
- Profile collections in the repository in order to prioritise the development of preservation strategies appropriate to the file-types present in it.
- Make explicit the tacit knowledge about computing environments on which Representations of Intellectual Entities depend before it becomes extinct.
- Demonstrate that preservation actions undertaken by the repository have been carried out lawfully.
- Ensure there is sufficient information from which to construct the metadata that will be required to support access and use of the Intellectual Entities by future researchers.
- Ensure that the Representations of Intellectual Entities within an archive (which are also records of transactions) retain their evidentiary value.
- Understand the structural relationships between different Intellectual Entities which form part of the same archive.
- Ensure that Intellectual Entities are self-documenting over time.
- Reassemble complex Representations of Intellectual Entities, such as websites, from their constituent digital Files.

Some of the metadata required for these purposes is also useful for other activities, such as informing local preservation repository developments and creating descriptive metadata, and may be re-purposed to fulfill these other needs. The application profile should enable the repository to record information necessary to undertake the activities outlined above on any kind of digital file, and may require sub-profiles to cater for the idiosyncrasies of certain categories of file.

Considerations affecting the design of a preservation metadata application profile

A number of factors must be considered when designing a preservation metadata application profile. These include:

The technical nature of the files to be preserved

Different types of file present different preservation challenges. Examples of 'types' found in a personal archive include audio, diary, document, email, moving image, spreadsheet and still image; all have important technical characteristics which are specific to their type, such as colour depth in a digital still image. Format-specific sub-profiles may also be needed; for example, GIF or JPEG might form sub-profiles of the image profile. Additionally, a repository responsible for personal digital archives may wish to construct a preservation metadata application profile for personal computing software that it has licence to preserve; this would allow it to curate software needed for the extraction of older digital materials not supported by contemporary environments.

Modeling granularity

The many-layered approach (Intellectual Entities > Representations > Files) requires repositories to decide at which level units of metadata are most appropriately recorded and to implement a mechanism for linking between these layers. The repository must also decide at what levels it will link to metadata about associated events, agents and rights. Linking between metadata in this way produces an extensible framework that allows the repository to record metadata at the highest level applicable, and to extend the metadata about an Intellectual Entity over time by creating and linking to: new Representations, events/agents and rights metadata as need arises. Taking all the layers of metadata together supplies the preservation information needed to produce the Archival Information Package that will support the preservation of Intellectual Entities as they evolve over time.

1. **Preservation metadata relating to Intellectual Entities:** this is a conceptual level and applies to notions such as collection, accession and series found in hierarchical archival descriptions, as well as to lower level Intellectual Entities.

Intellectual Entity level	Title	Example metadata
Collection level	'Personal archive of Politician X'	Collection level metadata needed for administration; metadata captured for re-use as descriptive metadata.
Accession level	'Second accession made to archive of Politician X, 16 May 2008'	Accession level metadata needed for administration; metadata captured for re-use as descriptive metadata, such as the original order of the accession.
Series level	'Email archive of Politician X, 1999-2008'	Information about the original email environment; relationship metadata for component email folders.
Subseries level	'Email folder relating to prisons, 2007-2008'	Relationship metadata for component emails.
Item level	'Speech on rising prison populations, delivered 30 January 2007'	Basic descriptive metadata (e.g. creation date and author); metadata describing relationships with Representations of the Intellectual Entity (e.g. the item could exist as a MS Word 11 file and as an OpenDocumentText 1.0 file).

Item level	'Picture of Mrs Williams at Liverpool docks, 1 February 2007'	Basic descriptive metadata (e.g. creation date); metadata describing relationships with Representations of the Intellectual Entity (e.g. the item could exist as an X3F file and as a JPEG 2000 file).

Metadata needed for Intellectual Entities includes descriptive information about content and context, such as the original intellectual arrangement of the digital material in an accession, that must be captured at ingest in order that it may be used for administrative and discovery purposes at the appropriate time. The metadata about Intellectual Entities must also include references to their Representations (the sets of Files and their structural metadata, which rendered together produce a Representation of the Intellectual Entity).

2. **Preservation metadata relating to Representations of Intellectual Entities:** all Representations should link to the Intellectual Entity which they represent and should include structural information which details how to construct the Representation of an Intellectual Entity from its constituent files, as shown in the diagram below:

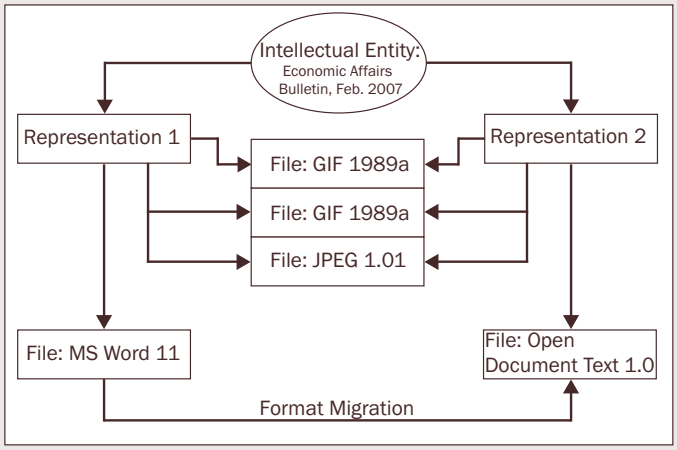

Figure 10: Preservation metadata relating to Representations of Intellectual Entities:

In order to be meaningful, both Representation 1 and Representation 2 must be related to the Intellectual Entity they represent. The Representations must have structural metadata about their component Files in order to produce a Representation of the Intellectual Entity.

The original Representation of an Intellectual Entity should record information detailing the significant properties of the Representation, so that the repository can judge the success of any preservation actions on the Intellectual Entity. Representations should also link to related Representations, e.g. where a migration takes place, a link between the source and derived Representation should be present. The repository must decide whether it will use the Representation level for simple objects consisting of a single digital File.

3. **Preservation metadata at File level:** most technical metadata is associated with the digital Files that compose the Representation of an Intellectual Entity. Such metadata includes information about the file format used and fixity information, such as a checksum or digital signature.

The object characteristics to be preserved

The characteristics, or 'significant properties', to be preserved could vary depending on the class or content of the Intellectual Entity, as well as who created it. Preserving some significant properties may be prohibitively expensive and the decision to preserve them may rest on the potential research value of the archive.

Authenticity requirements
Repositories must consider the level of detail required from audit trails. One helpful approach is to consider what questions archivists or researchers might want to ask in relation to Intellectual Entities and actions taken in respect of them in order to prove that the Intellectual Entities are authentic.

Embed or reference
External registries are being developed for some kinds of preservation metadata, such as file format information. Metadata entries in such registries can be referenced from the repository's metadata rather than held locally if desired. Repositories must balance the need to make metadata generation more efficient with the risks associated with reliance on a third party. The accuracy, coverage and sustainability of external sources of information, along with the cost of creating and maintaining it locally, should be assessed in deciding how much metadata repositories will hold locally and how much will be referenced. One argument in favour of recording metadata locally is that locally determined data structures and content may permit the repository to better query its contents for the purposes of collection profiling and batch preservation actions. An argument against local recording is the cost of creation and future maintenance.

Information required by future researchers
In capturing preservation metadata, repositories ought to consider the kinds of information that future users of the archive will need. Users are likely to be interested in some of the preservation metadata collected as historical information, such as the environment used to create the archive and any passwords that were used to protect certain Intellectual Entities.

Limitations of current tools
The scale of digital archives means that metadata creation must be automated as far as possible. The design of an application profile for preservation metadata must consider how the metadata will be generated and may therefore be limited by the functionality of existing tools.

Interoperability
Selecting common metadata standards will allow repositories to leverage expertise and tools from the community of practice working with those standards. This will reduce costs and risk to the repository, although good records of any profile developed from a standard should be maintained, and copies of documentation relating to the standard held locally or referenced from a sustainable external resource.

The metadata used by the repository should be independent of particular repository software requirements. It should be as easy as possible to move from one repository platform to another.

Preservation supported by the repository
The depth and breadth of metadata required by a repository may depend on the preservation strategies (see Chapter 08 *Digital preservation strategies*) envisaged. Repositories which offer to preserve the original bitstream, but transfer the burden of rendering to the user could operate using a simpler metadata profile than a repository which offers to preserve access to materials for its users. A repository offering preservation of objects conforming to 20 formats may require more metadata, and a more sophisticated metadata model, than a repository which migrates all objects to XML on ingest.

Cost and performance
Metadata can be one of the most costly aspects of digital preservation, and thought must be given to the efficiency of creating metadata conformant to the application profile and the ease of training staff in its use.

Extensibility
Can the application profile be easily evolved and extended to accommodate new circumstances?

Timing

Preservation metadata should be assembled and given structure at ingest to a repository, and may be added to over time in response to preservation actions, availability of new tools, or the transfer of rights, etc. Information about the environment and the intellectual property rights relating to materials is best-obtained from the creators of the material, while other metadata will be compiled via repository processes, using validation tools, fixity checkers, virus checkers, forensic software and metadata extraction tools. Structural information may be evident in the material as transferred, but may need to be extracted and formally recorded.

Preservation metadata and OAIS

Much of the work on developing preservation metadata for digital repositories is informed by the *Reference Model for an Open Archival Information System* (OAIS) (see p. 3); it is therefore worth examining what information OAIS deems necessary for the long-term preservation of digital materials. OAIS specifies that all Content Data Objects which a repository intends to preserve should have relevant Representation Information, Preservation Description Information, Packaging and Descriptive Information.

Representation Information (RI) provides structural and semantic information that permits the interpretation of Content Data Objects (i.e. the archival material accessioned) so that they may be rendered accessible: translating bits into information that is meaningful to the repository's Designated Community (usually a variety of researchers in the case of archival institutions).

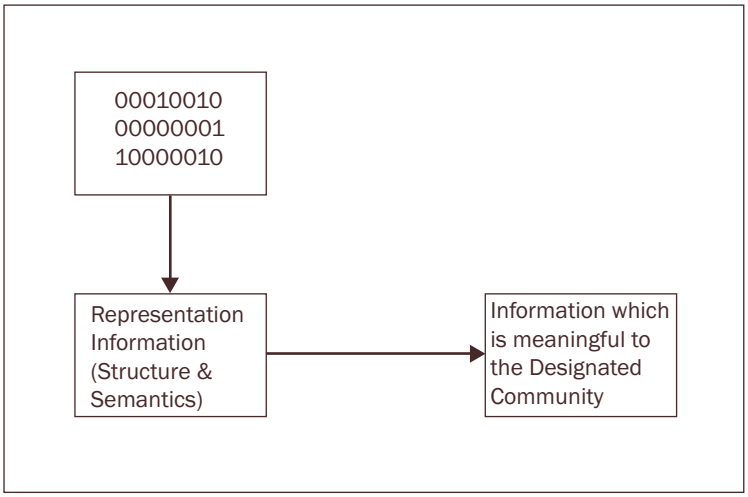

Figure 11: Translating bits into meaningful information

Classes of RI relevant to many individual objects can be identified; RI for common digital object types could therefore be held in one or more central registries, or in a local registry. RI itself may also be in digital form and require its own RI in order to be fully understood; the repository must therefore determine the level of RI to be held locally and/or referred to in external registries.

Example:
A repository accessions a spreadsheet of names and addresses in Lotus 4 spreadsheet format. The spreadsheet is a series of meaningless bits without the RI required to render it visible to the human eye. An understanding of this RI is built into the application software that was originally used to read and write the spreadsheet (Lotus 1-2-3, version 4) and now needs to be made accessible to the archiving repository. The repository may therefore wish to record or refer to information about the Lotus 4 spreadsheet format. If the format documentation is published in Portable Document Format 1.4, then the repository may also require RI to interpret and render files conforming to the PDF 1.4 format.

Preservation Description Information provides information about the individual content data to be preserved. OAIS adopts the classification identified in the 1996 report by Donald Waters and John Garrett entitled *Preserving Digital Information*, which splits this information into four groups:

- **Reference information** – allows for consistent, enduring and unique identification.
- **Provenance information** – details the origin and custody of an information object, including events in its life once submitted to the repository.
- **Context information** – documents the relationship of an information object with its environment, including why it was created and relationships with other information objects.
- **Fixity information** – documents authenticity using mechanisms which evidence that no undocumented alterations have been applied.

Packaging Information binds content information and its metadata together in an Information Package so that the relationship between the two can be sustained over time. METS (see p. 117) is one example of a metadata standard designed to package several bits of metadata and the content information to which the metadata refers.

Descriptive Information does not form part of the Information Package; rather it functions to describe the Package, enhancing access to the content information via finding aids and search and discovery tools. At the preservation stage (represented in OAIS by the Archival Information Package), descriptive information about personal digital archives is likely to be minimal, consisting of automatically extracted information at the inventory level and collection or accession-level contextual information. The depth and quality of descriptive information can be improved when the archive is formally arranged and described (see Chapter 06 *Arranging and cataloguing digital and hybrid archives*) for researcher access, in the form of Dissemination Information Packages (see p. 189) and an EAD catalogue. However, some degree of descriptive metadata is essential for long-term preservation, not least because a long period is likely to elapse before archived digital objects are subject to detailed cataloging.

Introducing PREMIS

OAIS is a reference model not an implementation guide and it is recognised that implementing preservation metadata along the lines that the model suggests relies on a more detailed expression of requirements. Early initiatives to define OAIS' preservation metadata concepts more fully include the work of the CURL Exemplars in Digital Archives (CEDARS) project; National Library of Australia (NLA); National Library of New Zealand (NLNZ) and Networked European Deposit Library (NEDLIB). Each of these initiatives produced an implementable breakdown of semantic units. All demonstrated a desire to embrace a common preservation metadata standard applicable to digital files of all kinds, allowing the same schema to be used regardless of file type or preservation strategy. The NLA and NLNZ schemes also provided an additional layer of metadata geared towards the needs of specific file types. All four undertakings produced valuable results, but the primary aim of the three institutions was to develop schemes that were capable of satisfying local, rather than global, requirements.

In 2003, the Online Computer Library Center (OCLC) and Research Libraries Group (RLG) established a working group (OCLC/RLG Preservation Metadata Framework Working Group)[1] which would build on the work of these early initiatives. Composed of international experts involved in developing or implementing preservation repositories around the globe, the group was charged with investigating preservation metadata-related issues and defining 'an implementable set of "core" preservation metadata elements, with broad applicability to digital preservation repositories'. This aim of 'broad applicability' challenged the group to produce a standard which could be used by a variety of institutions. The working group formed two subgroups: an Implementation Strategies Subgroup, which published a report on the state of the art in this area based on 70 survey responses, and the Core Elements Subgroup, which produced the *PREMIS Data Dictionary 1.0*.

1 Online Computer Library Center, 'Preservation Metadata Framework Working Group', *Online Computer Library Center website.* URL: <http://www.oclc.org/research/projects/pmwg/wg1.htm>

The *PREMIS Data Dictionary 1.0*

The first version of the *PREMIS Data Dictionary*, published in February 2005, was designed to articulate 'things that most preservation repositories are likely to need to know in order to support digital preservation'. While PREMIS defines what it is necessary to know, it does not specify how this information is to be recorded. This means that the core elements of information, or semantic units, suggested by the *PREMIS Data Dictionary* could be held in various ways, such as:

- Policy documents.
- Procedural or workflow documents.
- XML:
    - One or many XML schemas which happen to record the appropriate information.
    - In the PREMIS XML schemas (see below).
    - With or without XML packaging (e.g. METS).
- Database tables.

The Data Dictionary also addresses some implementation issues pertinent to the semantic units it defines, such as repeatability, obligation and controlled vocabulary requirements, and supplies useful examples based on the repositories maintained by working group members.

The PREMIS XML schemas

The PREMIS working group produced five XML schemas to encode the semantic units of the Data Dictionary. As noted above, use of the schemas is not compulsory as PREMIS semantic units may be held in any number of ways, so long as a repository knows the information and is able to supply it on request. The schemas can be used independently or concurrently depending on the profile of the *PREMIS Data Dictionary* being used. The five schemas are:

- PREMIS – a schema to contain the XML produced from the entity schemas.
- Object – a schema for the object entity.
- Agent – a schema for the agent entity.
- Event – a schema for the event entity.
- Rights – a schema for the rights entity.

Developing and maintaining PREMIS

The Library of Congress maintains the PREMIS web pages,[1] wiki and discussion list (the PREMIS Implementers Group list, or PIG for short![2]). PREMIS also benefits from an international editorial committee, which is responsible for coordinating and approving changes to the Data Dictionary and XML Schemas. The Committee is currently undertaking a first review of the *PREMIS Data Dictionary* in light of feedback arising from its first year of usage. A registry of PREMIS implementers is available.[3]

1 Preservation Metadata Implementation Strategies (PREMIS), 'PREMIS Implementors' Group (PIG)', *Preservation Metadata Maintenance Activity website.* URL: <http://www.loc.gov/standards/premis/pig.html>
2 Preservation Metadata Implementation Strategies (PREMIS), 'PREMIS Implementors Group Forum', *Listserv website*. URL: <http://listserv.loc.gov/listarch/pig.html>
3 Preservation Metadata Implementation Strategies (PREMIS), 'PREMIS Implementation Registry', *Preservation Metadata Maintenance Activity website.* URL: <http://www.loc.gov/standards/premis/premis-registry.php>

The PREMIS data model

Although PREMIS builds on the framework established by the OAIS model, the terminology of its Data Dictionary differs somewhat to that of OAIS; the authors attribute this to the transition from conceptual framework (OAIS) to an implementation (PREMIS). The PREMIS data model is composed of five 'entities' relevant to digital preservation. These entities are:

- Intellectual Entities – the conceptual entity, introduced earlier, that can be represented by one or more digital files. Intellectual Entities may also contain other Intellectual Entities (e.g. web pages could be considered sub Intellectual Entities of a website) and may have multiple digital Representations (e.g. a single web page could have one Representation composed of HTML and JPEG files, and another as a PDF file). PREMIS does not define semantic units for Intellectual Entities; these were deemed out of scope because existing descriptive metadata standards tend to be domain-specific and several already exist.
- Object - a discrete unit of information in digital form.
- Event – an action involving at least one Object or Agent known to the repository.
- Agent – person, software or organisation associated with a preservation Event in the life of an Object.
- Rights – assertion of rights pertaining to an Object or Agent in the repository.

The entities defined by PREMIS have relationships with one another. To facilitate these relationships, each entity has an identifier which can be used to point to it from another entity. The arrows in the diagram below show the direction of these relationships.

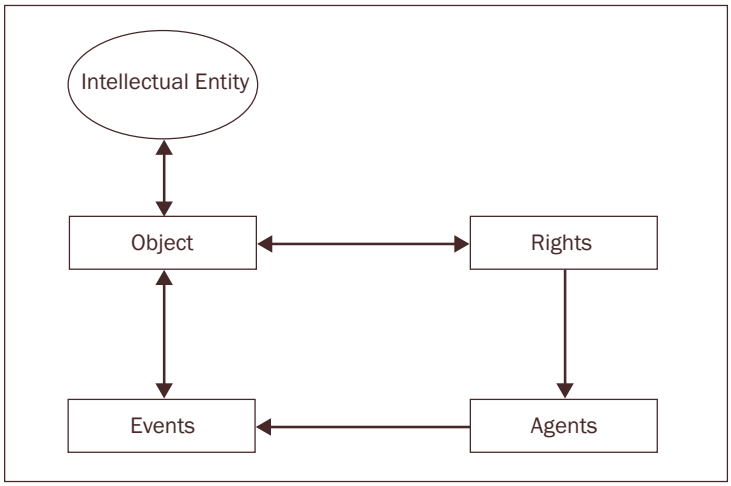

Figure 12: Direction of PREMIS entity relationships

A note on the Object entity

The Object entity is more complex than the other entities and deserves a little more explanation. PREMIS defines three subtypes of Object:

File

'a named and ordered sequence of bytes that is known by an operating system. A file can be zero or more bytes and has one file format, access permissions, and file system statistics such as size and last modification date'.

Examples:
Portable Document Format 1.4 file
WordPerfect for Windows 5.1 file
WordPerfect for Macintosh file
Graphics Interchange Format 1989a file

Bitstream

'contiguous or non-contiguous data within a file that has meaningful common properties for preservation purposes. A bitstream cannot be transformed into a standalone file without the addition of file structure (headers, etc.) and/or reformatting the bitstream to comply with some particular file format'.

Bitstreams which are true files embedded within larger files, are known as filestreams. These have sufficient structural information to stand alone as files when removed from the context in which they were found.

Examples:
An image embedded in a Tagged Image File Format file (bitstream)
A Portable Document Format 1.4 file embedded in a zip file (filestream)

Representation

'the set of files (see p. 82), including structural metadata, needed for a complete and reasonable rendition of a particular Intellectual Entity'.

A Representation object could be a simple object, consisting of a single File to represent an Intellectual Entity, or a complex object, consisting of multiple Files and the structural metadata required to reassemble them into an Intellectual Entity. The diagram below shows two Representations of the same Intellectual Entity, one complex and one simple:

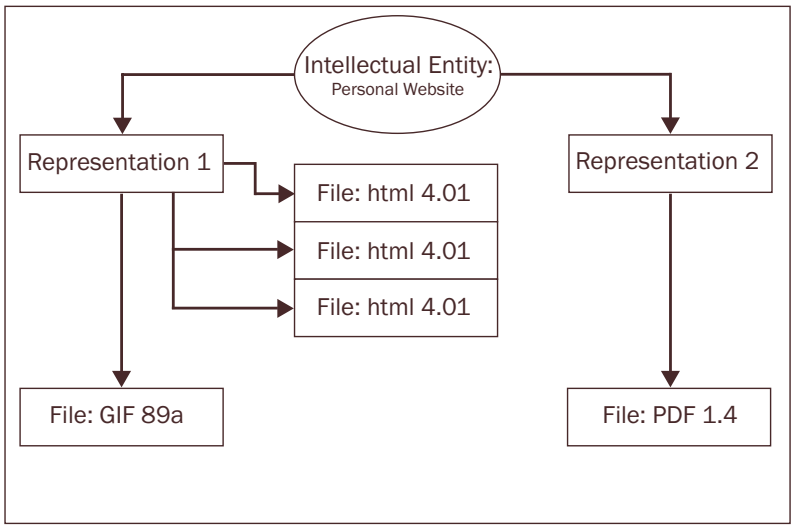

Figure 13: Two Representaions of the same Intellectual Entity

The division of Representation and File in this way enables repositories to record metadata that is relevant to the Representation as a whole and to record metadata about each of its constituent Files. Many of the detailed semantic units in the PREMIS dictionary are not applicable to Representation objects because they are a wrapper object to unite the assembly of the Files which compose an Intellectual Entity, therefore this detail is held in the File objects related to the Representation of the Intellectual Entity.

Further examples to illustrate the relationships between Intellectual Entities, Representations and Files:

Intellectual Entity	Sub-Intellectual Entity	Representation (one or more Files)
Digital correspondence of politician	Messages with or without attachments	Microsoft Outlook for XP personal store file (.pst)
A message about a meeting with an agenda		An email in XML format and attachment in Open Document Format 1.0
Draft article for a magazine		A Microsoft Word 2000 file Tagged Image File Format file
Personal website	Pages from personal website	1 css file 20 html files 1 gif1989a file
Personal website		1 PDF 1.4 file
Page from personal website		1 css file 1 html file

All elements described in PREMIS are as applicable to objects created by the repository itself, through migration events or other preservation actions, as they are to the original objects deposited with the repository as a personal digital archive.

PREMIS does not cover all preservation metadata requirements; along with specifying a local PREMIS application profile, decisions regarding what to use to describe Intellectual Entities, agents, files formats, rights, media, hardware, a repository's business and PREMIS record creation may also be needed.

Strengths and weaknesses of the *PREMIS Data Dictionary*

Strengths

- Understandable, on the whole.
- Allows references to external sources of information, e.g. format registries and policies.
- XML schemas available.
- Supported and maintained by an international editorial committee based at the Library of Congress.
- Several digital repositories are investigating PREMIS.
- Requires establishment of preservation policy to enable preservation level to be specified.
- Grew out of the practical experience of numerous institutions; it is therefore based on consensus and oriented towards implementation.
- Flexibility: no requirements as to how preservation information is stored.
- Generality: applicable to all types of digital material, from personal digital archives to eBooks.
- It can be incorporated into a METS file and therefore combined with other metadata to create an Information Package.
- Ability to link between entities, establishing one-to-one, one-to-many and many-to-one relationships.
- Applicable regardless of object type.
- Repeatable semantic units allow more or less granularity, as required.
- Very few of the PREMIS semantic units are mandatory.

Weaknesses, difficulties and requirements

- Many find the Representation concept difficult to grasp.
- Requires agreement on business rules.
- Requires local agreement on data content standards. This means work on selecting and, in many cases, developing controlled vocabularies based on repository policies which may not yet exist in many repositories.
- Need for local controlled vocabularies may reduce ease of interoperability, unless standards for these emerge.
- Insufficient implementation examples (though more are emerging).
- Difficult to automate creation of metadata structured as in PREMIS at present (though PREMIS does not require this).
- Need to consider fit with overall metadata profile and policy framework.
- Must be supplemented by metadata which can record detailed technical attributes of specific object-types, or media and hardware.
- The values to populate some semantic units defined by PREMIS are currently difficult or impossible to obtain, though this is not a weakness of PREMIS itself.

Assessing the use of PREMIS semantic units and PREMIS XML schemas in the preservation of personal archives

The next portion of this chapter explores the semantic units defined by the *PREMIS Data Dictionary* in a personal archives context and outlines the beginnings of an application profile for personal digital archives. Paradigm decided to use the Object, Agent, Event and Rights XML schemas so that PREMIS metadata could be added to METS files alongside other metadata needed for digital objects in the preservation repository. Paradigm opted not to use the PREMIS container schema, as METS provides adequate metadata packaging. More information about using METS as an Archival Information Package can be found further on in this chapter (see p. 117).

Some PREMIS semantic units are containers that simply group together lower-level semantic components containing data. In order to make the hierarchical structure of semantic units clearer, a hierarchical number is assigned to each unit. For each semantic unit, information about: its usage (whether optional, mandatory, etc.); whether or not it is repeatable; and whether the unit requires a controlled vocabulary is available in the *PREMIS Data Dictionary*.

The <object> entity

1.1 <objectIdentifier> Used to uniquely identify an object in its storage environment; a persistent identifier (see p. 48) according to another model may also be ascribed. As with all identifiers in PREMIS, the type and value of the identifier must be recorded.

In this example the object has an identifier created by the Fedora digital repository software, which will uniquely identify the object in the context of the repository. It also has a Handle (see p. 63), which will uniquely identify the object outside of the repository.

```
<premis:object>
    <premis:objectIdentifier>
        <premis:objectIdentifierType>Handle
        </premis:objectIdentifierType>
    <premis:objectIdentifierValue>http://hdl.handle.net/1842/
    </premis:objectIdentifierValue>
    </ premis:objectIdentifier>
    <premis:objectIdentifier>
        <premis:objectIdentifierType>Local
        </premis:objectIdentifierType>
        <premis:objectIdentifierValue>file:231
        </premis:objectIdentifierValue>
    </ premis:objectIdentifier>
    <!–other metadata–>
</premis:object>
```

1.2 <preservationLevel> Used to record the level of preservation applicable to each object, though it is not clear whether the repository should record its preservation intention or its preservation capability. As the preservation level offered could change over time, recording a date for this statement could be useful; PREMIS does not allow for this at present. To record <preservationLevel> repositories must define levels of preservation and produce a controlled vocabulary. If repositories make preservation level decisions according to classes of material, then fuller information should be available via the repository's policy documents. Where a detailed preservation commitment specific to particular attributes of an item needs to be recorded, this would be recorded elsewhere, perhaps using <premis:significantProperties>.

In this example the repository is undertaking bit-level preservation for the object:

```
</premis:object>
    <!--other metadata-->
    <premis:preservationLevel>bit-level</premis:preservationLevel>
    <!--other metadata-->
</premis:object>
```

1.3 <objectCategory> Recorded as 'Representation', 'File' and 'Bitstream' – the types of object entity described by the *PREMIS Data Dictionary*. Most of the objects in a repository for personal digital archives are likely to be Representations of Intellectual Entities and Files. A fuller explanation of the differences between the object categories is given above (see p. 82).

In this example the object category is a file:

```
<premis:object type="file">
    <!--other metadata-->
    <premis:objectCategory>File</premis:objectCategory>
    <!--other metadata-->
</premis:object>
```

1.4 <objectCharacteristics> Used to group together a series of semantic units applicable to digital Files.

1.4.1 <compositionLevel> Used to record any 'unbundling' required to access a File. Typical 'unbundling' in a personal archives context might include decompression, decryption and the extraction of emails and attachments from an email mailbox format. It is anticipated that most accessions of material will not require unbundling and the value for this element will typically be set at '0'.

1.4.2 <fixity> Used to record information about the message digest value(s) calculated for a File. Fixity information allows the repository to verify that Files have not been altered in an unauthorised or undocumented manner. The element can record a message digest value and the algorithm and agent responsible its creation. Fuller documentation for the creating agent may also be associated with the object by means of a <premis:event> (see p. 93) (the creation of the original digest) and <premis:agent> (see p. 95) (the software and person responsible for the digest). PREMIS recommends that a minimum of two message digest algorithms are used to generate and record hash values for each File. Further information about fixity is available later in this chapter (see p. 152).

1.4.3 <size> The size of files and bitstreams may be recorded in bytes. This information is useful for the management and delivery of objects.

1.4.4 <format> A container unit for collecting format-related information for Files, which may be held locally and/or referenced from external sources. Paradigm recommends that basic format information is held locally using <formatDesignation> and that references to more detailed information in external registries, using <formatRegistry>, also be included.

1.4.4.1 <formatDesignation> <formatDesignation> and its child elements (<formatName> and <formatVersion>) are used to store format information locally, thus guaranteeing continuing access to information about the formats held in the repository without reliance on a third party. It can also be used as a basis for preservation

management searching and reporting (e.g. retrieving Microsoft Word documents of any version).

1.4.4.2 <formatRegistry> The repository must consider which registries it will record references to. Registries developed by the digital preservation community may be more sustainable, but registries designed for other purposes may contain more detailed information. This element is repeatable enabling the repository to record multiple registry entries for any one object. A potential problem is that some registries (e.g. FileInfo.net[1] and Wotsit.org[2]) lack record identifiers, which could be used to point to relevant entries (as required by PREMIS) or do not hold format information for different versions of file formats.

Ideally, registry users will improve the registries they use by contributing new entries for formats which they learn about. Paradigm used the PRONOM[3] registry developed at The National Archives (TNA) in the United Kingdom. The DROID tool supplied by TNA uses a signature file of format information to identify the formats of digital files; DROID is able to download updates of this signature file, enabling the number of formats which it can identify to be extended. In the case of an offline repository like that employed by the Paradigm project, a procedure must be developed for updating the signature files. As other file format registries emerge from the digital preservation community, e.g. the Global Digital Format Registry (GDFR),[4] repositories may wish to record references to these.

1.4.5 <significantProperties> Used to record important, often subjective, characteristics which cannot be recorded elsewhere. Where an item is deemed to have significant properties, the degree to which these are preserved is a measure of success. PREMIS requires no controlled vocabulary to populate this element, but it may be possible to develop structured descriptions of significant properties based on an object's class or content which, in conjunction with repository policies, may either assist automated population of this element or assist automation of actions to be taken on objects in response to the values recorded in it. In an archival context, unstructured information may be preferable, but it is costly to create and use. Deciding which elements of an object constitute significant properties (e.g. content, behaviour, structure, appearance, etc.) might depend on detailed knowledge of the original creator and creating context; and also on how researchers might want to make use of the material.

1.4.6 <inhibitors> Used to record mechanisms which inhibit access to an object, such as passwords. The repository holds metadata which records the inhibitor type, the inhibitor target (e.g. read and write access) and the key to bypass the inhibitor. Repositories must develop a policy for treating items protected by inhibitors. Possible approaches include:

- Retain objects protected by inhibitors, and record information required to bypass the inhibitor.
- Create an inhibitor-free derivation of the object, discard the original object and merely record that the object was subject to an inhibitor of some kind.
- Retain objects protected by inhibitors, record information needed to bypass inhibitors and create an inhibitor-free derivation of the object.

The third option is most suitable for archival repositories because it allows the retention of the original submission while removing the inhibitor from the object as soon as possible (hopefully before obsolescence complicates matters further). The use of a password (and even the password itself!) may be of interest to future researchers. The repository ought to record the removal of inhibitors as a <premis:event> (see p. 93) linking to this event and the newly created inhibitor-free object using the linking mechanisms of <premis:relationship> (see p. 91). The <premis:event> will further link to one or more <premis:agent> (see p. 95) entities responsible for the event. An object with an inhibitor will be assigned a <premis:compositionLevel> (see p. 86) value of 1 (i.e. one level of unbundling required to access the object) and its derivative object will have a <compositionLevel> value of 0. If an object is protected by Digital

1 FileInfo.net, *FileInfo.net website*. URL: <http://www.fileinfo.net>
2 Wotsit.org, *Wotsit.org website.* URL: <http://www.wotsit.org/>
3 The National Archives, 'PRONOM', *The National Archives website.* URL: <http://www.nationalarchives.gov.uk/pronom/>
4 Harvard University Library, *Global Digital Format Registry website.* URL: <http://hul.harvard.edu/gdfr/>

Rights Mechanisms, the repository may require the rightsholders permission to circumvent these; see the Workbook sections on Rights Metadata (see p. 141) and Copyright and Archives (see p. 253).

This example describes a document in Open Document Text Format which has digests calculated using SHA-1 and SHA-256 algorithms, and has been password protected by the creator:

```
<premis:object>
<--!other metadata-->
    <premis:objectCharacteristics>
        <premis:compositionLevel>1</premis:compositionLevel>
        <premis:fixity>
            <premis:messageDigestAlgorithm>SHA-1</premis:messageDigestAlgorithm>
            <premis:messageDigest>9bb7f5d5f5f48afcb516c644981ef4055188abbe</premis:
            messageDigest>
            <premis:messageDigestOriginator>Paradigm preservation repository</premis:
            messageDigestOriginator>
        </premis:fixity>
        <premis:fixity>
            <premis:messageDigestAlgorithm>SHA-256</premis:messageDigestAlgorithm>
            <premis:messageDigest>9f9c98943dfce6aa84a961fd93bd35a1a8992b4c2d4b5e460b2f3
            66ff0fd15d3</premis:messageDigest>
            <premis:messageDigestOriginator>Paradigm preservation repository</premis:
            messageDigestOriginator>
        </premis:fixity>
        <premis:size>26624</premis:size>
        <premis:format>
            <premis:formatDesignation>
                <premis:formatName>Open Document Text</premis:formatName>
                <premis:formatVersion>1.0</premis:formatVersion>
            </premis:formatDesignation>
            <premis:formatRegistry>
                <premis:formatRegistryName>PRONOM</premis:formatRegistryName>
                <premis:formatRegistryKey >fmt/136</premis:formatRegistryKey>
            </premis:formatRegistry>
        </premis:format>
        <premis:inhibitors>
                <premis:inhibitorType>password protection</premis:inhibitorType>
                <premis:inhibitorTarget>All content</premis:inhibitorTarget>
                <premis:inhibitorKey>fr3ddie28</premis:inhibitorKey>
        </premis:inhibitors>
    </premis:objectCharacteristics>
<!--other metadata-->
</premis:object>
```

1.5 <creatingApplication> Used to record the <creatingApplicationName>, <creatingApplicationVersion> and <dateCreatedByApplication>; all three may be of interest to researchers and will assist the preservation process. The element is repeatable to allow the repository to record multiple applications for one object; this information may prove useful as importing and exporting between applications can introduce errors into files. The values for the <creatingApplicationName> and <creatingApplicationVersion> elements may have to be populated manually from a knowledge of the creator's working context, but the <dateCreatedByApplication> (which records the File's last modification date) may be extracted automatically, though it is possible that machine dates may be wrong and that there may be issues with timezones, etc.

A document last modified using OpenOffice.org 2.02 at 16:28:07 on 8th September 2006. The document was co-written by someone using Microsoft Word 97:

```
<premis:object>
    <--!other metadata-->
    <premis:creatingApplication>
        <premis:creatingApplicationName>OpenOffice.org</premis:creatingApplicationName>
    <premis:creatingApplicationVersion>2.02</premis:creatingApplicationVersion>
    <premis:dateCreatedByApplication>2006-09-08T16:28:07</premis:dateCreatedByApplication>
    </premis:creatingApplication>
    <premis:creatingApplication>
        <premis:creatingApplicationName>Microsoft Word</premis:creatingApplicationName>
    <premis:creatingApplicationVersion>97</premis:creatingApplicationVersion>
    </premis:creatingApplication>
    <--!other metadata-->
</premis:object>
```

1.6 <originalName> Used to record the original name of a file, which provides important information about the creator and their record keeping practices. Original file titles should be available to researchers, and in the absence of detailed descriptive metadata at the preservation stage, this information must be stored somewhere. The original filename may also assist the repository in satisfying access requests from creators and in reconstructing the links between Files in complex Representations. The repository must decide whether to store some or all of the filepath as the <originalName>, or merely the name of the File.

```
<premis:object>
    <--!other metadata-->
    <premis:originalName>/Digital Exemplars/Dissemination/Workbook/Preservation Metadata/preservation-metadata-5.doc
    </premis:originalName>
    <--!other metadata-->
</premis:object>
```

1.7 <storage> Used to record the location(s) and medium(s) where objects are stored in order to plan and execute media refreshment and media migration; this element is repeatable to allow multiple locations for copies of the same object to be recorded. In some cases the value may not be the specific medium but the storage system that knows the medium (such as the Fedora repository software). If it is a matter of recording the same storage management system(s) for each object, then it may be more sensible to store this information in policy and procedural documents about the repository, rather than alongside each object. These documents should also detail the age and location of all copies of the repository's objects and plan for replacing media in a cyclical fashion, or when error rates suggest replacement is needed.

An object stored on a Dell Powervault 745N Network Attached Storage device, which is backed up to tape:

```
<premis:object>
    <--!other metadata-->
    <premis:storage>
        <premis:contentLocation>
            <premis:contentLocationType>FullPath</premis:contentLocationType>
            <premis:contentLocationValue>/directory/directory/item.odt</premis:
            contentLocationValue>
        </premis:contentLocation>
        <premis:storageMedium>Dell Powervault 745N</premis:storageMedium>
    </premis:storage>
    <premis:storage>
        <premis:contentLocation>
            <premis:contentLocationType>Physical</premis:contentLocationType>
            <premis:contentLocationValue>Building X Safe N</premis:contentLocationValue>
        </premis:contentLocation>
        <premis:storageMedium>DLT (offsite backup)</premis:storageMedium>
    </premis:storage>
    <--!other metadata-->
</premis:object>
```

1.8 <environment> Used to document hardware and software combinations which support use of the object. The level of detail desirable must be established by the repository, but PREMIS allows the recording of: details about the environment metadata, such as the extent to which the described environment supports the File; non-hardware/software-related dependencies; and hardware and software metadata. Recording environment metadata locally is most useful for objects in obscure formats. Data about relevant software environments for more popular and archival formats is likely to be held in central registries and can simply be referred to using the record ids supplied by those registries and recorded in 1.4.4.2 (<premis:formatRegistry>, see p. 87). The PRONOM registry, for example, has the facility to link formats with software applications which can render them, but this is still relatively immature at present and does not always record the entire software stack that might be needed to access an object, including the software that the application itself might require (e.g. run time libraries, a specific operating system). Registries for hardware environments, which could extend to peripheral devices and cables, are not yet available to the preservation community. The DCC's representation information registry (under development)[1] could hold such information in the future.

Although the <environment> element is intended to record metadata about environments which are known to work with a given object, rather than the environment of an object's creator, it seems sensible to use <premis:environment> to record information about the creators' environment since it should (in theory) satisfy the 'known to work' criteria and records information which may be of interest to future researchers; this may cause some overlap with <creatingApplication>, but <environment> can hold a great deal more information than it is possible to record in <creatingApplication>. To avoid metadata duplication (a single environment could be applicable to multiple objects), repositories may wish to implement a local 'environment registry' (see p. 104) using the PREMIS environment semantic units. Such metadata could also be added to a rudimentary EAD collection level description at ingetst. See Chapter 06 *Arranging and cataloguing digital and hybrid archives* for further details (p. 184).

The environment metadata specified by PREMIS cannot necessarily be extracted from the files submitted to the repository, but might be extracted from the operating system of the environment concerned, via digital forensics software, or by talking to the creator. This relies on the information having been obtained by the archivist at the survey, accession or initial processing stages.

An XML file designed to be viewed using an associated XSLT stylesheet (also in the repository). Not all the dependencies are incorporated here – the representation information network could become endless. This is why external registries are so important for information relating to classes of object.

```
<premis:object>
    <--!other metadata-->
    <premis:environment>
        <premis:environmentCharacteristic>known to work</premis:environmentCharacteristic>
        <premis:environmentPurpose>read</premis:environmentPurpose>
        <premis:dependency>
            <premis:dependencyName>XSL stylesheet</premis:dependencyName>
            <premis:dependencyIdentifier>
                <premis:dependencyIdentifierType>URI</premis:dependencyIdentifierType>
                <premis:dependencyIdentifierValue>http://shuttle.paradigm.ac.uk:8085/fedora/get/
                demo:333/XSL</premis:dependencyIdentifierValue>
            </premis:dependencyIdentifier>
        </premis:dependency>
        <premis:software>
            <premis:swName>Microsoft Windows</premis:swName>
            <premis:swVersion>XP Professional Version 2 Service Pack 2</premis:swVersion>
            <premis:swType>Operating System</premis:swType>
        </premis:software>
        <premis:software>
            <premis:swName>Mozilla Firefox</premis:swName>
            <premis:swVersion>1.5.0.6</premis:swVersion>
            <premis:swType>Renderer</premis:swType>
```

1 Digital Curation Centre, *Representation Information Registry Repository website.* URL: <http://registry.dcc.ac.uk/omar/>

```
        </premis:software>
        <premis:hardware>
            <premis:hwName>Intel Pentium M Processor 1.86GHz</premis:hwName>
            <premis:hwType>Processor</premis:hwType>
        </premis:hardware>
        </premis:environment>
        <--!other metadata-->
</premis:object>
```

1.9 <signatureInformation> Used to record the information needed to verify a digital signature (see p. 152) associated with an object. Some objects may be accessioned with signatures applied by their creators, or repositories could digitally sign objects on ingest. The repository must record the <signatureInformationEncoding> using a controlled vocabulary (e.g. base 64); the name of the signer may be recorded in <signer>; <signatureMethod> records the algorithm (e.g. RSA-SHA1) used to generate the signature and its value should be taken from a controlled vocabulary; <signatureValue> records the actual value of the signature generated; <signatureValidationRules> must be used to record or point to information about the processes required to validate the siganture; <signatureProperties> may optionally be used to record information about the generation of the signature, such as a timestamp. A <keyInformation> container is used to store metadata about the public key which can verify the signature, or decrypt the File (if the algorithm used supports encryption of the signed data). Here the <keyType>, <keyValue> and <keyVerificationInformation>, such as a certificate or certificate chain, is held.

In this example the digital File has been signed by Susan Thomas using the DSA-SHA1 algorithm; no key information is held:

```
<premis:object>
    <--!other metadata-->
    <premis:signatureInformation>
        <premis:signatureInformationEncoding>BASE 64</premis:signatureInformationEncoding>
        <premis:signer>Susan Thomas</premis:signer>
        <premis:signatureMethod>DSA-SHA1</premis:signatureMethod>
        <premis:signatureValue>qUADDMHZkyebvRdLs+6Dv7RvgMLRlUaDB4Q9yn9XoJA79a2882ffTg==
        </premis:signatureValue>
        <premis:signatureValidationRules>Add reference to repository documentation detailing signature
        validation rules</premis:signatureValidationRules>
        <premis:signatureProperties>2006-11-01 10:15:16</premis:signatureProperties>
    </premis:signatureInformation>
    <--!other metadata-->
</premis:object>
```

1.10 relationship Used to describe multi-object relationships of provenance or structure, which may involve an event. Paradigm recommends that the element is used only to record provenance relationships as structural metadata is adequately recorded in METS and EAD. Paradigm used a METS structural map to record the original order of files as accessioned and to record structural metadata for Representations of Intellectual Entities; the final intellectual arrangement of the archive (with its analogue material) will be recorded in an EAD catalogue when the archive is prepared for researcher access.

A vocabulary of relationship types (from the perspective of the object entity being described) must be defined and the repository can then use <relationship> to associate an object with another object and relevant events using the identifiers of those objects and events. A sequence may also be stored for related events.

In this example file:2 was decrypted via event:46 creating file:12 (a decrypted version of file:2). Since there are no other objects related in this way, and no other related events, a dummy value of 0 is recorded in <relatedObjectSequence> and <relatedEventSequence> is omitted.

```
<premis:object>
    <premis:objectIdentifier>
        <premis:objectIdentifierType>Local
        </premis:objectIdentifierType>
        <premis:objectIdentifierValue>file:2
        </premis:objectIdentifierType>
    </ premis:objectIdentifier>
    <--!other metadata-->
    <premis:relationship>
        <premis:relationshipType>Unbundling</premis:relationshipType>
        <premis:relationshipSubType>Decryption</premis:relationshipSubType>
        <premis:relatedObjectIdentification>
            <premis:relatedObjectIdentifierType>Local</premis:relatedObjectIdentifierType>
            <premis:relatedObjectIdentifierValue>file:12</premis:relatedObjectIdentifierValue>
            <premis:relatedObjectSequence>0</premis:relatedObjectSequence>
        </premis:relatedObjectIdentification>
        <premis:relatedEventIdentification>
            <premis:relatedEventIdentifierType>Local</premis:relatedEventIdentifierType>
            <premis:relatedEventIdentifierValue>event:46</premis:relatedEventIdentifierValue>
        </premis:relatedEventIdentification>
        </premis:relationship>
    <--!other metadata-->
</premis:object>
```

1.11 linkingEventIdentifier Used to link objects to events that are not associated with another object, such as format identification, format validation, media refreshment and virus checking, etc. It associates the identifier for an event with the metadata about a File.

In this example file:2 is being linked to event:86

```
<premis:object>
    <premis:objectIdentifier>
        <premis:objectIdentifierType>Local
        </premis:objectIdentifierType>
        <premis:objectIdentifierValue>file:2
        </premis:objectIdentifierType>
    </ premis:objectIdentifier>
    <--!other metadata-->
    <premis:linkingEventIdentifier>
        <premis:linkingEventIdentifierType>Local</premis:linkingEventIdentifierType>
    <premis:linkingEventIdentifierValue>event:86</premis:linkingEventIdentifierValue>
    </premis:linkingEventIdentifier>
    <--!other metadata-->
</premis:object>
```

1.12 linkingIntellectualEntityIdentifier (container) Used to link to descriptive metadata that describes the Intellectual Entity, or to an identifier of an object that is at a higher conceptual level than the object which is currently being described, e.g. to a collection or parent object. Rules about what should be linked to are needed.

In this example the object being described is a personal website (representation:32), which is part of a personal digital archive (collection:34) and part of an accession of that archive with the identifier accession:21:

```
<premis:object>
    <premis:objectIdentifier>
        <premis:objectIdentifierType>Local
        </premis:objectIdentifierType>
        <premis:objectIdentifierValue>representation:32
        </premis:objectIdentifierType>
```

```
    </ premis:objectIdentifier>
    <--!other metadata-->
    <premis:linkingIntellectualEntityIdentifier>
        <premis:linkingIntellectualEntityIdentifierType>Local</premis:linkingIntellectualEntityIdentifierType>
        <premis:linkingIntellectualEntityIdentifierValue>collection:34</premis:linkingIntellectualEntityIdentifierValue>
        </premis:linkingIntellectualEntityIdentifier>
    <premis:linkingIntellectualEntityIdentifier>
        <premis:linkingIntellectualEntityIdentifierType>Local</premis:linkingIntellectualEntityIdentifierType>
        <premis:linkingIntellectualEntityIdentifierValue>accession:21</premis:linkingIntellectualEntityIdentifierValue>
    </premis:linkingIntellectualEntityIdentifier>
    <--!other metadata-->
</premis:object>
```

1.13 linkingPermissionStatementIdentifier Used to link an object to a statement about rights the repository has to undertake preservation-related actions in respect of it using the <premis:rights> entity (see p. 95).

```
In this example the licence by which the repository may undertake preservation actions on a jpeg file (file:8) is expressed in the rights metadata of rights:28.
<premis:object>
    <premis:objectIdentifier>
        <premis:objectIdentifierType>Local
        </premis:objectIdentifierType>
        <premis:objectIdentifierValue>file:8
        </premis:objectIdentifierType>
    </ premis:objectIdentifier>
    <--!other metadata-->
    <premis:linkingPermissionStatementIdentifier>
        <premis:linkingPermissionStatementIdentifierType>Local
        </premis: linkingPermissionStatementIdentifierType>
        <premis:linkingPermissionStatementIdentifierValue>rights:28
        </premis:linkingPermissionStatementIdentifierValue>
    </premis:linkingPermissionStatementIdentifier>
</premis:object>
```

The <event> entity

The entity is provided to enable the repository to record events in the life of an object which illustrate its digital provenance. It is not mandatory to record events in PREMIS, but repositories caring for personal digital archives are likely to want event metadata for audit trails. Events that a repository of digital archives might want to record include:

- Details of the accession – who and when.
- Details of the ingest transaction – who and when.
- Details of preservation actions taken.
- Details of events which change an object.
- Validation against format.
- Fixity checks – initial digest calculations and records of re-calculations which fail to correspond with the initial digest.
- Restoration of object from last-known good version (where bit loss is discovered by a fixity check and the File can be restored from a good version in backup).
- Pre-accession events which have changed an object, where this information is available; though PREMIS does not cover events which take place outside of the repository.

- De-accessioning, disposal/deletion, which might take place both during initial appraisal and much later at the point of detailed cataloguing.
- The opening of the archive to researchers and any other changes to access conditions.
- Events relating to the object's metadata: creation, transformation, modification and deletion.

It is also useful to think about types of events:

- Events which change the object, thus creating a new Representation of the Intellectual Entity, e.g. migration of a File.
- Events which do not change the object, e.g. fixity check.
- Events which change the metadata of the object.
- Deletion of objects.

Decisions about whether or not to record an event, or whether to link an agent record to an event record, should take account of:

- The degree of change incurred to the object and its metadata by the event.
- The purpose of the audit trail that recording events, and their agents, creates.

Example: as a matter of routine, the repository undertakes regular fixity checks of its objects. It does not record a FixityCheck event unless the fixity check shows that an object has changed.

PREMIS suggests that event metadata should be stored separately from the digital object itself and its identifiers and linking mechanisms provide the means of linking event metadata to metadata about other entities in the model. Paradigm recommends that each <premis:event> is recorded in its own METS file (in the Digital Provenance <digiprovMD> section), and linked to relevant object(s) and agent(s) concerned using the PREMIS linking semantic units. The repository will need rules to define whether an event applies to a Representation or a File.

The semantic units provided for recording events are very simple:

2.1 <eventIdentifier> Used to record the type and value of the event's identifier.

2.2 <eventType> Used to record the type of event; taken from a controlled vocabulary.

2.3 <eventDateTime> Used to record the date and time of the event and encoded in ISO 8601 format.

2.4 <eventDetail> Used to record unstructured information about events.

2.5 <eventOutcomeInformation> Container element used to record information about the outcome of an event using <eventOutcome> with a controlled vocabulary and <eventOutcomeDetail> for unstructured information.

2.6 <linkingAgentIdentifier> Used to record the identifier value and type of an agent associated with the event and may be repeated enabling multiple agents to be associated with an event. Also includes the <linkingAgentRole> subelement, which allows the role of the agent to be recorded using a controlled vocabulary (e.g. authoriser or virusChecker). An agent could perform more than one role concurrently, therefore <linkingAgentRole> is repeatable.

2.7 <linkingObjectIdentifier> Used to record the identifier value and type of an object associated with the event; it can be repeated to associate numerous objects with a single event.

The <agent> entity

The purpose of the agent entity is to supply information about the agents (persons, organisations, or software) associated with preservation events or with the management of rights related to one or more objects in the repository. The provision of a unique identifier serves to distinguish between agents and can be used to link an agent with one or more events. Groups of agent types, which can be allocated rights to authorise or undertake certain events, may be created.

Recording agent metadata linked to event metadata is useful in the following circumstances:

- Where a change is made to a object, we might want to record the human and/or software agent responsible.
- Where an item is removed from a repository, we might want to record the details of the responsible agent.

If the agent metadata is to be used for troubleshooting, it may be necessary to record information about agents in some detail, for example the environment details for virus checking software may impact on the success of the virus check, and if the check fails to turn up information about a subsequently discovered virus, it may be helpful to have details of the virus checker and its environment available.

At present, the agent entity is very simple consisting of very few semantic units. Karen Coyle has suggested changes to the entity, for better recording of rights-related information (see p. 142).

3.1 <agentIdentifier> Used to record the type and value of the agent identifier.

3.2 <agentName> Used to record a reader-friendly descriptor of the agent.

3.3 <agentType> Used to record a value from a controlled vocabulary to denote the type of agent; PREMIS suggests person, organization and software. It might be useful to have more granularity than this, e.g. software - virus checker.

The <rights> entity

The PREMIS working group established only a bare minimum of rights metadata in version 1.0 of the Data Dictionary: the rights entity only allows repositories to record permissions granted by rightsholders in respect of preservation actions on the objects in which they hold rights. It is not a vehicle for rights metadata relating to rights for access and use by end-users, and cannot record information about the copyright status of an object.

The Library of Congress commissioned Karen Coyle to prepare a report on PREMIS' capacity for rights metadata in preparation for the first review of the Data Dictionary. Coyle's report, published in December 2006, recommends a number of changes to this element, for better recording of rights-related information. In particular, Coyle recommends that the addition of semantic units which would allow repositories to act in accordance with statutory rights be added to the standard. This would be of great benefit to repositories of archives and manuscripts which can usually seek explicit permissions only from the donor of the archive, meaning that material in which third-party copyright resides must be managed according to statutory, rather than contractual, provisions. For more about intellectual property rights relating to digital archives in preservation and access, see Chapter 09 *Legal issues* (see p. 252); also see further in this chapter for more on rights metadata (see p. 141).

4.1 <permissionStatement> A container for metadata elements used to record details of permission(s) granted by the rightsholder to the repository to undertake acts in respect of one or more objects to which the rightsholder hold the rights.

4.1.1 <permissionStatementIdentifier> Used to record the identifier type and value for the permission statement.

4.1.2 <linkingObject> Repeatable element used to link to object(s) covered by the <permissionStatement>. Ideally this should link to Intellectual Entities, which are more stable

than their Representations and Files.

4.1.3 <grantingAgent> Used to identify the granting agent; if details of the agent are held in a PREMIS agent entity, the <agentIdentifier> should be used.

4.1.4 <grantingAgreement> Contains semantic units to describe the granting agreement. A <grantingAgreementIdentification> may be recorded, using either a repository-style identifier (e.g. if an agreement is held in the repository as a digital image of the donation agreement, or an email from the donor, then the agreement will have a formal id type and value) or a less formal means of identification (e.g. Agreement with Donor Y, 28 January 2007 - paper file held by Bodleian Secretariat). The actual text of the agreement, or a paraphrase or other description of it may also be held in <grantingAgreementInformation>.

4.1.5 <permissionGranted> A container for details of the permission granted.

4.1.5.1 <act> Used to record the action that the repository is permitted to undertake. The element is repeatable and must be populated from a controlled vocabulary, using terms such as 'replicate', 'migrate' and 'modify'. PREMIS suggests that the vocabulary be the same as that used for eventType values (see p. 94) in the event entity. It might be difficult and inflexible for an archival repository to enumerate and predict all acts required and seek permission to undertake them; a less granular approach may be more suitable.

4.1.5.2 <restriction> Used to record any constraint relating to the <act>, such as frequency, quantity, need to inform rightsholder, etc.

4.1.5.3 <termOfGrant> Container element used to record the term of the grant using an ISO 8601 encoded <startDate> and <endDate>. The start date will probably be the date of the agreement governing the placement of the archive at the repository and the end date the date that copyright expires, if this can be ascertained.

4.1.5.4 <permissionNote> Allows the repository to make a note of further explanations. It could be used to note that permission has been sought from the principal creator/depositor of the archive to which the object(s) belong, but that other rightsholders are unknown, or have not granted permissions for preservation acts.

Connecting entities

It is useful to be aware of the linking mechanisms in PREMIS. Connections between the various PREMIS entities can be made using identifiers in the following ways:

Objects

Linking objects to Intellectual Entities	Via object entity's <linkingIntellectualEntityIdentifier>.
Linking objects to events	Objects and events can be linked both ways. • Via object entity's <relationship><relatedEventIdentification>; used for connecting objects that are related by an event, such as a migration that created a derivation of an object. • Via object entity's <linkingEventIdentifier>; used to record simple events which do not create new objects, e.g. virus check. • Via event's optional <linkingObjectIdentifier>.
Linking objects to agents	Indirectly via event entity's <linkingAgentIdentifier>.
Linking objects to rights	Via object entity's <linkingPermissionStatementIdentifier>.

Events

Linking events to Intellectual Entities	Indirectly, via object entity's <linkingIntellectualEntityIdentifier>.

Linking events to objects	Events and objects can be linked both ways. • Via object entity's optional <relationship><relatedEventIdentification>; used for connecting objects that are related by an event, such as a migration that created a derivation of an object. • Via object entity's <linkingEventIdentifier>; used for recording simple events, which do not create new objects. • Via event's optional <linkingObjectIdentifier>.
Linking events to agents	Via event entity's <linkingAgentIdentifier>.
Linking events to rights	Indirectly via object entity's <linkingPermissionStatementIdentifier>.

Agents

Linking agents to Intellectual Entities	Indirectly. The object entity's <linkingIntellectualEntityIdentifier> links to Intellectual Entities, and objects can be linked to events, and events can in turn link to agents.
Linking agents to objects	Indirectly. Objects can be linked to events, and events can in turn link to agents.
Linking agents to events	Via event entity's <linkingAgentIdentifier>.
Linking agents to rights	Indirectly. The object entity's <linkingPermissionStatementIdentifier> links to the rights entity, and objects can be linked to events, and events can in turn link to agents.

Rights

Linking rights to Intellectual Entities	Indirectly via the object entity's <linkingIntellectualEntityIdentifier>.
Linking rights to objects:	Via the rights entity's <linkingObject> and/or via the object entity's <linkingPermissionStatementIdentifier>.
Linking rights to events	Indirectly, via the object entity's relationship(s) with the event entity.
Linking rights to agents	Indirectly via the object entity's relationship with the event entity which can link to agent using <linkingAgentIdentifier>.

Paradigm's proposed use of PREMIS for personal digital archives

The tables below suggest what preservation metadata should be recorded at different levels of the object entity; the Bitstream level is not included as Paradigm has opted not to record metadata at this level. Obligation is expressed as follows:

M = mandatory
R = recommended
MA = mandatory if applicable
O = optional
O (M if...) = optional, unless a specified container element is used when usage becomes mandatory

Some of these elements require controlled vocabularies; vocabularies for many elements do not yet exist, some need development locally, others need encoding rules. Rather than attempt to create vocabularies wholesale, repositories may do better to create a list of preferred terms that can be selected from and added to as the repository's needs grow through the accumulation of different materials.

Object entity - Representations of Intellectual Entities:

PREMIS elements to be recorded in PREMIS object XML and wrapped with other metadata in METS documents describing Representations of Intellectual Entities. The METS documents for Representations will also contain a structural map detailing how constituent Files must be assembled to recreate the Representation; some degree of human intervention may be required for complex objects, in bringing together Files into Representations of Intellectual Entities.

	Semantic unit	Semantic unit component	Value	Source	Paradigm Obligation	PREMIS Obligation
1.1.1	objectIdentifier	objectIdentifierType	Local		M	M
1.1.2		objectIdentifierValue	<Fedora PID>	Obtained from namespace used.	M	M
1.2	preservationLevel		Full	Supplied by the digital curator. If automated, this could be set to default to certain values depending on object characteristics such as format. A repository for personal archives may well default to 'full preservation'.	M	M
1.3	objectCategory		Representation	Manual/semi-automatic, depending on type.	M	M
1.4.5	objectCharacteristics	significantProperties	<details of significant properties>	Manual unstructured input at present; work on defining significant properties in a way that might allow some automation of this metadata is being sponsored by the JISC.	O	O
1.5.1	creatingApplication	creatingApplication-Name	<creating application name(s)>	Derived from file or from contextual information obtained from survey or creator. Use a controlled vocabulary.	R	O
1.5.2		creatingApplicationVersion	<creating application version number(s)>	Derived from file or from contextual information obtained from survey or creator. Use a controlled vocabulary for consistency.	R	O
1.5.3		dateCreatedByApplication	<last modified date>	The date of last modification is extracted by the NLNZ tool as <File><FileDateTime><Date><Time>; Jhove extracts this as <lastModified>. Must be encoded using ISO 8601.	M	O
1.10.1	relationship	relationshipType	derivative	Metadata recording relationships of digital provenance created between objects as part of preservation actions should be created as part of those actions.	MA	O (M if <relationship> used)
1.10.2		relationshipSubType	is derived from		MA	O (M if <relationship> used)
1.10.3.1	relationship - relatedObjectIdentification	relatedObjectIdentifierType	Local		MA	O (M if <relationship> used)

1.10.3.2		relatedObjectIdentifier-Value	<Fedora PID for source object>		MA	O (M if <relation-ship> used)
1.10.3.3		relatedObjectIdentifi-erSequence	<number in sequence>	0 is used where no sequence is present.	MA	O (M if <relation-ship> used)
1.10.4.1	relationship - relatedE-ventIdentification	relatedEventIdentifier-Type	Local		MA	O (M if <relation-ship> used)
1.10.4.2		relatedEventIdentifier-Value	<Fedora PID for event which relates objects>		MA	O (M if <relation-ship> used)
1.10.4.3		relatedEventSequence	<number in sequence>		MA	O
1.11.1	linkingEventIdentifier	linkingEventIdentifier-Type	Local		MA	O (M if <linkin-gEventIdentifier> used)
1.11.2		linkingEventIdentifier-Value	<Fedora PID for event>	Must be an the value of an existing event identifier.	MA	O (M if <linkin-gEventIdentifier> used)
1.12.1	linkingIntellectualEnti-tyIdentifier	linkingIntellectualEnti-tyIdentifierType	Local		M	O (M if <linkingIn-tellectualEntityI-dentifier> used)
1.12.2		linkingIntellectualEnti-tyIdentifierValue	<Fedora PID for Intel-lectual Entity METS file>	Record the identifier of the Intellectual Entity to which the Representation relates.	M	O (M if <linkingIn-tellectualEntityI-dentifier> used)
1.13.1	linkingPermissionState-mentIdentifier	linkingPermissionState-mentIdentifierType	Local		MA	O (M if <linking-PermissionState-mentIdentifier> used)
1.13.2		linkingPermissionState-mentIdentifierValue	<Fedora PID for permis-sion statement>	Must be the value of an existing permissionState-mentIdentifierValue.	MA	O

Object entity - Files

PREMIS elements to be recorded in PREMIS object XML as part of METS dpcuments documenting Files. The METS documents may also include metadata specific to the type of object (see below for more information on this, p. 110).

	Semantic unit	Semantic unit component	Value	Source	Paradigm Obligation	PREMIS Obligation
1.1.1	objectIdentifier	objectIdentifierType	Local		M	M
1.1.2		objectIdentifierValue	<Fedora PID>		M	M
1.2	preservationLevel		Full		M	M
1.3	objectCategory		File		M	M
1.4.1	objectCharacteristics	compositionLevel	<numeric value for levels of unbundling required>	Ingest tools should recognise files with multiple composition levels and flag these for the curator's attention. Files with 0 composition levels should be assigned a value of 0 automatically.	M	M
1.4.2.1	objectCharacteristics – fixity	messageDigestAlgorithm	<algorithm name>	There are many tools available. Jhove (can do CRC32, MD5 and SHA-1 checksums); since v. 2.2, Fedora can calculate checksums on ingest.	M	O (M if <fixity> used)
1.4.2.2		messageDigest	<digest value>		M	O (M if <fixity> used)
1.4.2.3		messageDigestOriginator	<details of agent>		M	O
1.4.3	objectCharacteristics	size	<size in bytes>	NLNZ tool extracts this as <File><Size>; Jhove extracts it as <size>.	M	O

1.4.4.1.1	objectCharacteristics – format - formatDesignation	formatName	<format name>	The DROID tool developed by The National Archives in the United Kingdom outputs format name, if it identifies the format. DROID currently exports its results in Comma Separated Value (CSV) and XML formats. Jhove extracts this as <format> and <version> for formats it recognises. Other registries are also useful, though the retrieval of the information can be a more manual process. A controlled vocabulary will be required.	M	O* (M if <formatDesignation> used)
1.4.4.1.2		formatVersion	<format version>	As above.	R	O*
1.4.4.2.1	objectCharacteristics – format - formatRegistry	formatRegistryName	<registry name>	Use DROID to identify entries in PRONOM; http://filext.com/; use other registries as they mature. Use a controlled vocabulary.	R	O* (M if <formatRegistry> used)
1.4.4.2.2		formatRegistryKey	<id of registry record>	DROID for PRONOM which uses PRONOM unique IDs (PUIDs), e.g. x-fmt/42.	R	O* (M if <formatRegistry> used)
1.4.5	objectCharacteristics	significantProperties	<details of significant properties>	Record in free text at present, but keep watching brief on research into significant properties.	O	O
1.4.6.1	objectCharacteristics - inhibitors	inhibitorType	<type of inhibitor>	Acquired from creator, or via password cracking utilities and classified according to the repository's controlled vocabulary.	MA	O (M if <inhibitors> used)
1.4.6.2		inhibitorTarget	<target of inhibitor>	Establish controlled vocabulary.	MA	O
1.4.6.3		inhibitorKey	<key required to by pass inhibitor>		MA	O
1.5.1	creatingApplication	creatingApplicationName	<creating application name(s)>	Use controlled vocabulary.	R	O
1.5.2		creatingApplicationVersion	<creating application version(s)>	Use a controlled vocabulary so that values which might be recorded differently are consistent, e.g. Microsoft Word 97 is also known as Microsoft Word 8.	R	O

1.5.3		dateCreatedByApplication	<last modified date>	The date of last modification is extracted by the NLNZ tool as <File><FileDateTime><Date><Time>; Jhove extracts this as <lastModified>. Must be encoded using ISO 8601. n.b. Archivists need to record dates to describe provenance, including the date the document was first created, to enable researchers to identify the time period in which the document was refined. This cannot be recorded in PREMIS so should be recorded in a MODS record for the object using <originInfo><dateCreated> and <originInfo><dateModified>.	M	O
1.6	originalName		<original name, including filepath, if known>	NLNZ tool extracts this as <File><Filename><Name><Extension>.	M	O
1.7.2	storage	storageMedium	<storage medium(s)>	Use controlled vocabulary.	M	M
1.8.4.1	Environment – dependency	dependencyName	<name of dependency>	Use controlled vocabulary.	MA	O
1.8.4.2.1	Environment – dependency - dependencyIdentifier	dependencyIdentifierType	Local		MA	O (M if <dependency> used)
1.8.4.2.2		dependencyIdentifierValue	<Fedora PID for object on which object is dependent>		MA	O (M if <dependency> used)
1.10.1	relationship	relationshipType	derivative		MA	O (M if <relationship> used)
1.10.2		relationshipSubType	is derived from		MA	O (M if <relationship> used)
1.10.3.1	relationship – relatedObjectIdentification	relatedObjectIdentifierType	Local		MA	O (M if <relationship> used)
1.10.3.2		relatedObjectIdentifierValue	<Fedora PID for source object>		MA	O (M if <relationship> used)

1.10.3.3		relatedObjectIdentifierSequence	<number in sequence>		MA	O (M if <relationship> used)
1.10.4.1	relationship – relatedEventIdentification	relatedEventIdentifierType	Local		MA	O (M if <relationship> used)
1.10.4.2		relatedEventIdentifierValue	<Fedora PID for event which relates objects>		MA	O (M if <relationship> used)
1.10.4.3		relatedEventSequence	<number in sequence>		MA	O
1.11.1	linkingEventIdentifier	linkingEventIdentifierType	Local		MA	O (M if <linkingEventIdentifier> used)
1.11.2		linkingEventIdentifierValue	<Fedora PID for event>		MA	O (M if <linkingEventIdentifier> used)
1.13.1	linkingPermissionStatementIdentifier	linkingPermissionStatementIdentifierType	Local		MA	O (M if <linkingPermissionStatementIdentifier> used)
1.13.2		linkingPermissionStatementIdentifierValue	<Fedora PID for permission statement>		MA	O (M if <linkingPermissionStatementIdentifier> used)

*either formatDesignation or formatRegistry must be recorded

Repository-level documentation

PREMIS elements that must be recorded in repository-level documentation, such as policy, procedure and architectural documents.

	Semantic unit	Semantic unit component	Value	Paradigm Obligation	PREMIS Obligation
1.2		preservationLevel	The repository should document the level of support, including preservation strategies, for as many of the formats in its care as possible. This might include intention to develop capability, or detail of existing capability.	M	M
1.7.1.1	Storage - contentLocation	contentLocationType	Online and offline.	M	O (M if <contentLocation> is used
1.7.1.2	Storage - contentLocation	contentLocationValue	Record the storage locations used by the repository for all copies of its files. The repository should maintain an awareness of the age and robustness of media in order to perform timely media refreshing.	M	O (M if <contentLocation> is used

Metadata for a local environment registry

Rather than re-recording the same environment metadata for many objects, it would be preferable to point to an entry in an environment register of some kind. The environment semantic units of PREMIS's object entity could be used as the basis for a local environment register: PREMIS object XML could be wrapped in METS files, potentially with references to actual copies of software or hardware maintained by the repository for data extraction purposes. The environments documented could be those used by creators and could be pointed to by one or more Representations using the relationship semantic unit. The information to populate this metadata should be obtained through examination of hardware, software and files used by the creator at survey stage and/or after deposit. It can be extracted from the operating system, or by using forensic software. The metadata fields for the registry are mainly optional, because it may not be possible to acquire all of the information for a creator's environment.

Semantic unit	Semantic unit component	Value	Paradigm Obligation
objectIdentifier	objectIdentifierType	Local	M
objectIdentifierValue		<Fedora PID>	M
preservationLevel		Full	M
objectCategory		Representation	M
environment	environmentNote	Creator's environment	M

environment - software	swName	<software name>	O
environment - software	swVersion	<software version>	O
environment - software	swType	<software type>	O
environment - software	swOtherInformation	<other software info., e.g. licence code, perhaps id/location of local copy of software>	O
environment - software	swDependency	<software dependencies>	O
environment - hardware	hwName	<hardware name>	O
environment – hardware	hwType	<hardware type>	O
environment – hardware	hwOtherinformation	<other hardware info, perhaps reference location is held by repository>	O

Event entity

	Semantic unit	Semantic unit component	Value	Source	Paradigm Obligation	PREMIS Obligation
2.1.1	eventIdentifier	eventIdentifierType	Local		M	M
2.1.2		eventIdentifierValue	<Fedora PID>		M	M
2.2	eventType		<event type>	Controlled vocabulary for event types to be determined.	M	M
2.3	eventDateTime		<date and time of event>	Encode using ISO 8601.	M	M
2.4	eventDetail		<event detail>	Record additional information about the event, if needed.	O	O
2.5.1	eventOutcomeInformation	eventOutcome		Controlled vocabulary of event outcomes.	O	O
2.5.2		eventOutcomeDetail		Record a detailed description of the event outcome, if needed.	O	O
2.6.1	linkingAgentIdentifier	linkingAgentIdentifierType	Local	Record references to as many agents as applicable.	M	O (M if <linkingAgentIdentifier> used

2.6.2		linkingAgentIdentifier-Value	<Fedora PID>		M	O (M if <linkingAgentIdentifier> used
2.6.3	LinkingAgentRole			Use a controlled vocabulary.	O	O
2.7.1	linkingObjectIdentifier	linkingObjectIdentifier-Type	Local	Record references to as many objects as applicable.	M	O (M if <linkingObjectIdentifier> used)
2.7.2		linkingObjectIdentifier-Value	<Fedora PID>		M	O (M if <linkingObjectIdentifier> used)

Agent entity

	Semantic unit	Semantic unit component	Value	Source	Paradigm Obligation	PREMIS Obligation
3.1.1	agentIdentifier	agentIdentifierType	Local		M	M
3.1.2		agentIdentifierValue	<Fedora PID>		M	M
3.2	agentName		<agent name>	Record name of agent; use a controlled vocabulary and rules for creating new names.	M	O
3.3	agentType		<agent type>	Use a controlled vocabulary: person, organisation or software.	M	O

Rights entity

A Rights metadata record should be created for permissions granted to the repository by the donor of the archive. Records can be created for other agents if permissions have been granted from these.

	Semantic unit	Semantic unit component	Value	Source	Paradigm Obligation	PREMIS Obligation
4.1.1.1	permissionStatement - permissionStatementIdentifier	permissionStatementIdentifierType	Local		M	M
4.1.1.2		permissionStatementIdentifierValue	<Fedora PID>		M	M

4.1.2	permissionStatement	linkingObject	<Fedora PID>	Repeat to include as many object references as necessary, though recommended that this be linked to the collection level Intellectual Entity where it details permissions granted by the archive donor.	M	M
4.1.3	permissionStatement	grantingAgent		Record relevant Fedora PID of <agentIdentifierValue>.	M	O
4.1.4.1	permissionStatement - grantingAgreement	grantingAgreementIdentification			MA	O
4.1.4.2		grantingAgreementInformation		Record an object PID, if a copy of the agreement is held in the repository; if the agreement is held in a paper filing system this should be noted; alternatively, note if no permission has been granted.	M	O
4.1.5.1	permissionStatement - permissionGranted	act		Use controlled vocabulary. Recommend high-level rather than low-level acts, such as 'preserve', 'make accessible to researchers'.	M	M
4.1.5.2		restriction		Record a restriction if applicable; e.g. 'make accessible to researchers' may be restricted for a period of years after accession.	MA	O
4.1.5.3.1	permissionStatement – permissionGranted - termOfGrant	startDate		Record the start date of the permission, normally that of the deposit agreement. Encode using ISO 8601.	M	M
4.1.5.3.2		endDate		Record the end date of the permission agreement. Usually end of copyright, unless some other arrangement is made with the creator. Encode using ISO 8601.	M	M
4.1.5.4	permissionStatement - permissionGranted	permissionNote		Use to record any notes relating to permissions.	O	O

Example of PREMIS in action in the context of a personal digital archive

A repository accessions the archive of Politician X; during the negotiation surrounding the transfer of the archive to the repository, Politician X grants the repository permission to undertake preservation actions on the material in the archive to which she holds the rights, and this is documented in the terms of agreement covering the placement of the archive with the repository. The accession of the archive includes a draft speech by Politician X (an Intellectual Entity) in Locoscript format (an Object). The repository's policy determines that all Locoscript files should be migrated to Rich Text Format on ingest, so that they may be accessed on contemporary computing platforms. An archivist authorises the migration, which is performed by some migration software. The end result is a new Representation of the same Intellectual Entity (the speech). Using the PREMIS data model, the following entities are identified and linked together using their identifiers:

archive:1 - Archive of Politician X

This Intellectual Entity will be described by a METS file containing the collection level information that will be needed for the long-term preservation of the archive:

- PREMIS rights metadata which records permissions given by Politician X to the archive to undertake preservation actions on materials to which he holds intellectual property rights.
- PREMIS object metadata to document the environment(s) used by Politician X in the creation and use of her archive.
- Basic collection level EAD metadata to record some descriptive and administrative metadata about the accessions, including reference to related analogue materials in the archive.
- A METS structMap, detailing the original order of the archive as submitted to the repository in the following manner Accession(s) > Physical container(s), e.g. hard disk /CD-R > Folder(s) > File(s).

intellectualEntity:1 - Speech by a politician

MS. Politician 23, the speech of the politician, is the Intellectual Entity that will be described in an archival description. As a result of the migration Event, the speech has two Representations: the original Locoscript file (representation:1) and the Rich Text Format file (representation:2).

Its METS file contains:

- Basic descriptive metadata, such as title and creation and modification dates, extracted from the original file as a MODS record.
- METS structMap which points to the METS files of Representations of the Intellectual Entity.
- METS structMap which points to the METS file of collection, accession or folder objects to which the Intellectual Entity belongs.

representation:1 – the original

This Representation links to intellectualEntity:1, to file:1 and the representation:2.

Its METS file contains:

- PREMIS object metadata for the Representation level.
- METS fileSec detailing the Representation's component Files.
- METS structMap detailing the structure of the Representation's component Files.

file:1 – the Locoscript file

file:1 has a relationship with representation:1. It also has a relationship with its derivative Rich Text Format file and event:1, the migration event.

Its METS file contains:

- PREMIS object metadata for the File level.
- METS fileSec providing the location of the File described.
- METS structMap.

representation:2 – the first migration

The Representation links to intellectualEntity:1, to file:2 and to representation:1.

Its METS file contains:

- PREMIS object metadata for the Representation level.
- METS fileSec detailing the Representation's component Files.
- METS structMap detailing the structure of the Representation's component Files.

file:2 – the Rich Text Format file

file:2 has a relationship with representation:2. It also has a relationship with file:1, from which it is derived and event:1, which created it.

Its METS file contains:

- PREMIS object metadata for the File level.
- METS fileSec providing the location of the File described.
- METS structMap.

event:1 – migration

event:1 has relationships with file:1 (the Locoscript file), file:2 (the Rich Text Format file), agent:1 (Ms Archivist, the agent authorising the event) and agent:2 (the migration software).

Its METS file contains:

- PREMIS event metadata.
- METS structMap.

agent:1 – Ms Archivist

agent:1, the archivist authorising the migration, will be referred to by event:1 (the migration event).

Its METS file contains:

- PREMIS agent metadata.
- METS structMap.

agent:2 – migration software

agent:2, the software performing the migration, will be referred to by event:1 (the migration event).

Its METS file contains:

- PREMIS agent metadata.
- METS structMap.

rights:1

rights:1 records that the politician who created the speech (MS. Politician 23) has granted the repository licence to undertake preservation actions in respect of all the material in the politician's archive to which the politician holds the rights. Rights:1 therefore has a relationship with representation:1 and representation:2 in this scenario.

Its METS file contains:

- PREMIS rights metadata.
- METS structMap.

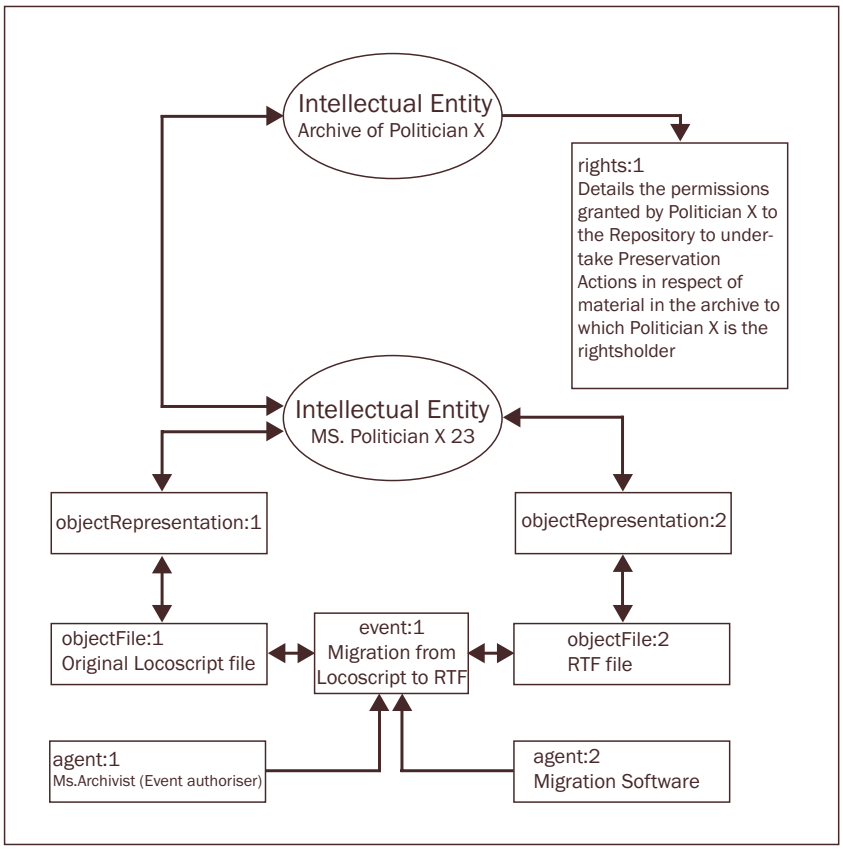

Figure 14: PREMIS Example

Metadata specific to object types

PREMIS provides a generic preservation model applicable to all object types; it does not support the detailed technical metadata that is particular to object types. To give an indication of the nature of the metadata needed for this purpose, a number of metadata schemes being used by digital repositories for technical metadata are listed below. This list is by no means complete, as more schemes for audiovisual material are starting to emerge owing to the growth of web services in this area. It should be noted that some of the more generic elements in these schemes overlap with PREMIS elements, and the repository must decide whether to record them redundantly, or in a single place. More work is needed in understanding the specific technical metadata requirements and how they may be produced before an application profile can be determined.

Metadata for still images

MIX for Images (Z39.87)[1]

Drafted by the National Information Standards Organization, the *Data Dictionary – Technical Metadata for Digital Still Images*[2] is extensive. Containing more than 100 elements, it relates mainly to images produced via digitisation projects, where metadata capture can be embedded in the process, but much of the standard is applicable to digital images entering repositories via digital archives collections. MIX is an XML implementation of the Data Dictionary that is maintained by the Library of Congress. The Jhove tool is capable of outputting some of the MIX metadata set from the GIF, JPEG, JPEG 2000 and TIFF image formats, and tools have been written to extract the MIX XML from the Jhove XML.

The areas covered by the NISO Data Dictionary are as follows:

- Basic Digital Object Information.
- Basic Image Information.
- Image Capture Metadata.
- Image Assessment Metadata.
- Change History.

Harvard Digital Repository Service Image[3]

Metadata set to record details about image attributes and image production (20 elements) developed for Harvard's digital repository, which records metadata about image attributes and the image production process.

National Library of Australia Image[4]

Limited metadata set (10 elements) for still images developed at the National Library of Australia.

Digital Images Archiving Study[5]

The *Digital Images Archiving Study* by Sheila Anderson et al. (March 2006) is a useful overview of issues relating to the preservation of raster and vector images.

Exchangeable image file format (Exif)[6]

Exif is a metadata specification for digital images and was created by the Japan Electronic Industry Development Association (JEIDA). It includes descriptive metadata, such as date and time, and technical metadata about camera settings at image capture.

Metadata for moving images

National Library of Australia Video[7]

Limited metadata set (7 elements) for moving images developed at the National Library of Australia.

---

1 MIX, *NISO Metadata for Images in XML Schema: Technical Metadata for Digital Still Images Standard website.* URL: <http://www.loc.gov/standards/mix/>

2 National Information Standards Organisation, *An American National Standard* (December 2006). URL: <http://www.niso.org/standards/resources/Z39-87-2006.pdf?CFID=2106743&CFTOKEN=20293343>

3 Harvard University Library: Digital Repository Service (DRS), *DRS Documentation Administrative Metadata for Digital Still Images* (March 2004). URL: <http://preserve.harvard.edu/resources/imagemetadata.pdf>

4 National Library of Australia, 'Preservation Metadata for Digital Collections', *National Library of Australia website.* URL: <http://www.nla.gov.au/preserve/pmeta.html>

5 Arts and Humanities Data Service, *Digital Images Archiving Study*. URL: <http://www.jisc.ac.uk/uploaded_documents/FinaldraftImagesArchivingStudy.pdf>

6 Standard of Japan Electronics and Information Technology Industries Association, *Exchangeable image file format for digital still cameras: Exif Version 2.2* (April 2002). URL: <http://www.digicamsoft.com/exif22/exif22/html/exif22_1.htm>

7 National Library of Australia, 'Preservation Metadata for Digital Collections', *National Library of Australia website.* URL: <http://www.nla.gov.au/preserve/pmeta.html>

Library of Congress VideoMD Data Dictionary[1]
Developed as part of the Digital Audio-Visual Prototyping Projects at the Library of Congress, this scheme comprises 16 elements.

MPEG-7[2]
A multimedia content descriptions standard for moving pictures and audio developed by the Moving Pictures Expert Group (MPEG).

Metadata for audio files

Harvard Digital repository service Audio[3]
Metadata set to describe the attributes of an audio file. Comprises 26 elements.

Library of Congress Audio (Source) Data Dictionary[4]
Developed as part of the Digital Audio-Visual Prototyping Projects at the Library of Congress, this scheme comprises 18 elements.

National Library of Australia Audio[5]
Metadata set comprises 7 elements.

MPEG-7[6]
A multimedia content descriptions standard for moving pictures and audio developed by the Moving Pictures Expert Group (MPEG).

ID3[7]
ID3 is a container for metadata stored in MP3 format. Many of the semantic units relate to descriptive metadata, such as title and performer, but there are also units relating to useful administrative metadata, such as file type, size and length.

Metadata for text files

Schema for Technical Metadata for Text by Jerome McDonough[8]
Metadata for text files, with an XML schema, developed by Jerome McDonough at New York University. Responsibility for the maintenance of this schema is unclear. Comprises 19 elements.

National Library of Australia Text[9]
Simple text metadata scheme comprising 5 elements.

---

1 Audio-Visual Prototyping Project, 'VideoMD Data Dictionary', *Audio-Visual Prototyping Project website.* URL: <http://www.loc.gov/rr/mopic/avprot/DD_VMD.html>

2 International Organisation for Standardisation, *MPEG-7 Overview* (October 2004). URL: <http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm>

3 Harvard University Library: Library Digital Initiative, *Administrative Metadata for Digital Audio Files* (February 2004). URL: <http://preserve.harvard.edu/resources/audiometadata.pdf>

4 Audio-Visual Prototyping Project, 'Audio (Source) Data Dictionary', *Audio-Visual Prototyping Project website.* URL: <http://www.loc.gov/rr/mopic/avprot/DD_ASMD.html>

5 National Library of Australia, 'Preservation Metadata for Digital Collections', *National Library of Australia website.* URL: <http://www.nla.gov.au/preserve/pmeta.html>

6 International Organisation for Standardisation, *MPEG-7 Overview* (October 2004). URL: <http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm>

7 ID3. URL: <http://www.id3.org/>

8 New York University Libraries Digital Library Team, 'Schema for Technical Metadata for Text', *New York University Libraries Digital Library Team website.* URL: <http://dlib.nyu.edu/METS/textmd.htm>

9 National Library of Australia, 'Preservation Metadata for Digital Collections', *National Library of Australia website.* URL: <http://www.nla.gov.au/preserve/pmeta.html>

Metadata for databases

National Library of Australia Database[1]
Six elements for database-specific metadata.

Metadata for executables

National Library of Australia Executables[2]
Metadata to record the code type and version of an executable file.

Deciding what metadata is useful for particular content types

Few archivists will have had cause to consider the technical properties of digital content types and may find it difficult to understand the importance and potential usage of the metadata elements chosen by the initiatives listed above. Very few explain the reasoning behind the elements chosen, and the archivist may therefore need to look to other sources to gain an understanding of these elements. The *Sustainability of Digital Formats* web page, hosted by the Library of Congress,[3] is a useful place to learn more about the specifics of formats commonly found in digital collections; the site includes assessments of qualities and function important to content types that should inform metadata application profiles. Implementing many of these metadata is presently difficult as tools which support the extraction of the metadata are not widely available. JISC has initiated a set of studies on the significant properties of software, vector images and moving images, which may provide useful information for developing technical metadata profiles.[4]

Using Metadata Encoding and Transmission Service (METS) for preservation metadata

Like many other digital preservation initiatives, Paradigm used METS as a manner of uniting various kinds of metadata with digital objects. METS and XML-encoded PREMIS could be combined in several ways. Some of the possible options are outlined below:

Option 1 - compound
It is possible to hold all the metadata about an archive in a single METS file, perhaps something along the following lines:

- A MODS record for each item in the dmdSec, based on automatically derived descriptive metadata, allowing the archive to establish some basic intellectual control over the archive.
- A PREMIS object record for each item in amdSec/techMD.
- A type-specific metadata record for each item (e.g. MIX, textMD) in amdSec/techMD.
- A PREMIS rights record in amdSec/rightsMD.
- PREMIS event records for each event in amdSec/digiprovMD.
- PREMIS agent records relating to event record(s) in amdSec/digiprovMD.
- A fileSec inventory, with fileGrp rules for:
    - Intellectual Entities.

1 National Library of Australia, 'Preservation Metadata for Digital Collections', *National Library of Australia website.* URL: <http://www.nla.gov.au/preserve/pmeta.html>
2 National Library of Australia, 'Preservation Metadata for Digital Collections', *National Library of Australia website.* URL: <http://www.nla.gov.au/preserve/pmeta.html>
3 The Library of Congress, *Sustainability of Digital Formats: Planning for Library of Congress Collections website.* URL: <http://www.digitalpreservation.gov/formats/>
4 JISC, 'Significant Properties ITT'. *Jisc website.* URL: <http://www.jisc.ac.uk/fundingopportunities/funding_calls/2007/03/significant_properties_itt.aspx>

    - Representations of Intellectual Entities.
    - Files.
- A structMap detailing the original order of the Intellectual Entities in the archive.
- Several structMap sections detailing the structure of each Representation of an Intellectual Entity.

Advantages

- Requires only one METS file.

Disadvantages

- The resulting METS file would be enormous and difficult to work with.
- The resulting METS file would be complex, especially as new Representations, Files, events and agents are created through migrations.
- Requires careful planning and control over the assignment of identifiers and linking mechanisms.

Option 2 – atomistic

Split up the archive into numerous, smaller, METS files and use the linking mechanisms in METS and PREMIS to connect these.

Option 2a

Create separate METS files for:

- Intellectual Entities.
- Representations of Intellectual Entities.
- Files.
- Agents.
- Events.
- Rights.

Option 2b

Create separate METS files for:

- Intellectual Entities, which include metadata for:
    - Representations of the intellectual entity.
    - Files which make-up the Representation(s).
- Agents.
- Events.
- Rights .

Option 2c

Create separate METS files for:

- Intellectual Entities, which include metadata for:
    - Representations of Intellectual Entities.
    - Files.
    - Agents.
    - Events.
    - Rights.

Option 2d
Create separate METS files for:

- Intellectual Entities.
- Representations of Intellectual Entities, which include metadata for:
    - Files.
- Agents.
- Events.
- Rights.

Advantages

- Results in smaller METS file sizes.
- Results in simpler METS files from day one and through subsequent migration pathways.
- Avoids unnecessary duplication of metadata: where the same metadata (e.g. rights) relates to numerous objects, it is possible to link from all of those objects to a single METS file containing the common information.

Disadvantages

- A number of METS files may be needed to assemble a complete AIP for transfer to another repository.
- Requires careful planning and control over the assignment of identifiers and linking mechanisms.

Ids

Much in METS and PREMIS is dependent on ids. The repository must develop schemes for automatic assignation of identifiers to metadata, files, rights, agents and events; such schemes must take account of various ID-related issues, including scalability. See earlier in this chapter for more on Persistent Identifiers (see p. 48).

Choosing tools for metadata generation

Generating all the preservation metadata needed for preserving personal digital archives cannot be a fully automatic process at present, though the DROID, Jhove and National Library of New Zealand Metadata Extract (NLNZ) tools do produce some of the values repositories might like to record. A key problem is that no tool currently generates metadata marked up in the PREMIS XML schemas, and though PREMIS does not require that the PREMIS XML schemas be used, it seems sensible to use a single mark-up standard, which will interoperate with future tools and repositories, rather than add multifarious kinds of tool-specific metadata to METS files. Working with fewer schemas will also facilitate the training of staff in using the repository.

Tools developed by digital preservation specialists

DROID[1]

Developed by The National Archives in the United Kingdom, DROID (Digital Record Object Identification) uses a signature file to identify formats it knows, and returns metadata to the user which could be used to populate the <formatDesignation> and <formatRegistry> semantic units. This information can be exported from the tool in CSV and DROID's own XML formats.

1 DROID. URL: <http://droid.sourceforge.net/wiki/index.php/Introduction>

Jhove[1]

Developed by JSTOR and Harvard, Jhove (JSTOR/Harvard Object Validation Environment) supports mainly open formats; these include profiles of versions of the following: AIFF; ASCII; Bytestream; GIF; JPEG; JPEG2000; PDF; TIFF; UTF8; WAVE and XML.

Jhove uses its modules to parse an object in order to identify it. The file will be run through each module until it is identified, or does not identify with any module. It can be configured to look at magic numbers instead. For files that do not conform to these types, Jhove can produce a small set of metadata according to its default profile.

Jhove produces its metadata according to its own XML schema, but images are described using MIX.

NLNZ tool[2]

The NLNZ tool supports the following proprietary formats:

- Images: BMP, GIF, JPEG, TIFF.
- Office docs: MS Word (versions 2,6), WordPerfect, OpenOffice (v1), MS Works, MS Excel, MS PowerPoint, PDF.
- Audio and video: WAV, MP3
- Markup language: HTML, XML.

It also has a default profile for other formats. The NLNZ uses file extension to determine format (not such a reliable method) and provides a mime type.

The NLNZ tool produces its metadata according to its own schema.

Other tools

Software developers and the public are becoming increasingly interested in creating and exploiting metadata, and there are a number of tools for working with metadata available commercially and in open source software repositories such as Sourceforge. The Cairo project,[3] which is developing a tool for ingesting digital archives and metadata into a repository, is conducting a survey of such tools and assessing their utility in supplying metadata for preserving digital objects. Initial impressions include:

- Tools tend to support a single content type (many are for still images).
- More generic tools produce less detailed metadata.
- Licensing is unclear is some instances.
- Documentation is often poor.
- Little is available for older digital formats.
- Metadata mark-up and values are extremely varied.
- The metadata produced would need crosswalking to appropriate standards.

1 Harvard University Library, 'JHOVE - JSTOR/Harvard Object Validation Environment', *Harvard University Library website.* URL: <http://hul.harvard.edu/jhove/>

2 National Library of New Zealand, *Metadata Extraction Tool*. URL: <http://meta-extractor.sourceforge.net/>

3 Cairo Project, *Cairo Project website.* URL: <http://cairo.paradigm.ac.uk>

Using METS for the preservation and dissemination of digital archives

Introduction

During recent years an increasing quantity and variety of digital material has been created or held by libraries, and consequently many standards have been developed to encode different categories of metadata for specific object types. There is no catch-all standard which accommodates the needs of every digital object type, and the lack of consensus on which standards to use can cause interoperability problems, especially when metadata or objects need to be transferred between repositories. As well as metadata specific to particular object types, all digital objects require different levels and types of metadata at different points in their lifecycle; all of this diverse metadata needs to be associated or packaged with the object it describes.

The Metadata Encoding and Transmission Standard (METS) was developed to deal with these issues. It is an XML Schema designed as an overall framework within which all the metadata associated with a single digital object can be stored or referred to. It enables effective management of digital objects within the repository, acts as a standard for transferring metadata within repositories, facilitates access and navigation by the researcher, and links the digital object and its metadata inextricably together. METS offers significant benefits to archivists, but its usage will not solve interoperability problems; only agreement on common METS profiles (see p. 120) can do this.

METS is intended to act as an Information Package, as defined by the Reference Model for an Open Archival Information System (OAIS). It can deal with all the categories of metadata specified by OAIS as necessary to the preservation of a Content Data Object, i.e.

- Content Information (describing the content data object and its representation information).
- Preservation Description Information (reference, provenance, context and fixity information).
- Packaging Information (METS is designed to package metadata and objects together).
- Descriptive Information (describing and enhancing access to the content information).

A METS file contains separate sections for descriptive, administrative and structural metadata; each section is linked to the others by means of a comprehensive system of unique identifiers. It allows two approaches to the storage of descriptive and administrative metadata: they can either be held internally within the METS file, or held externally and referenced from within the file. While METS does not dictate the content of the metadata, the METS Editorial Board recommends a number of other metadata schemas (known as Extension Schemas) which can be incorporated into the METS file or referred to from it (see p. 119).

A general introduction to METS for archivists is provided here in order to examine how it might be used to hold all of the metadata that must be captured for the kinds of digital objects typically found in personal archives. The suggestions made derive from Paradigm's experiences; they do not constitute a fully drafted METS profile, but provide a basis to build on.

Background to METS

In the late 1990s a digitisation initiative called the Making of America II (MOA2) Testbed Project identified categories for descriptive, structural and administrative metadata types. An XML Document Type Definition (DTD) was developed for the project. In 2001, this DTD was reviewed and revised, under the sponsorship of the Digital Library Federation; the outcome was Version 1.0 of the METS XML Schema, which inherited the broad metadata structure set out by the MOA2 project.

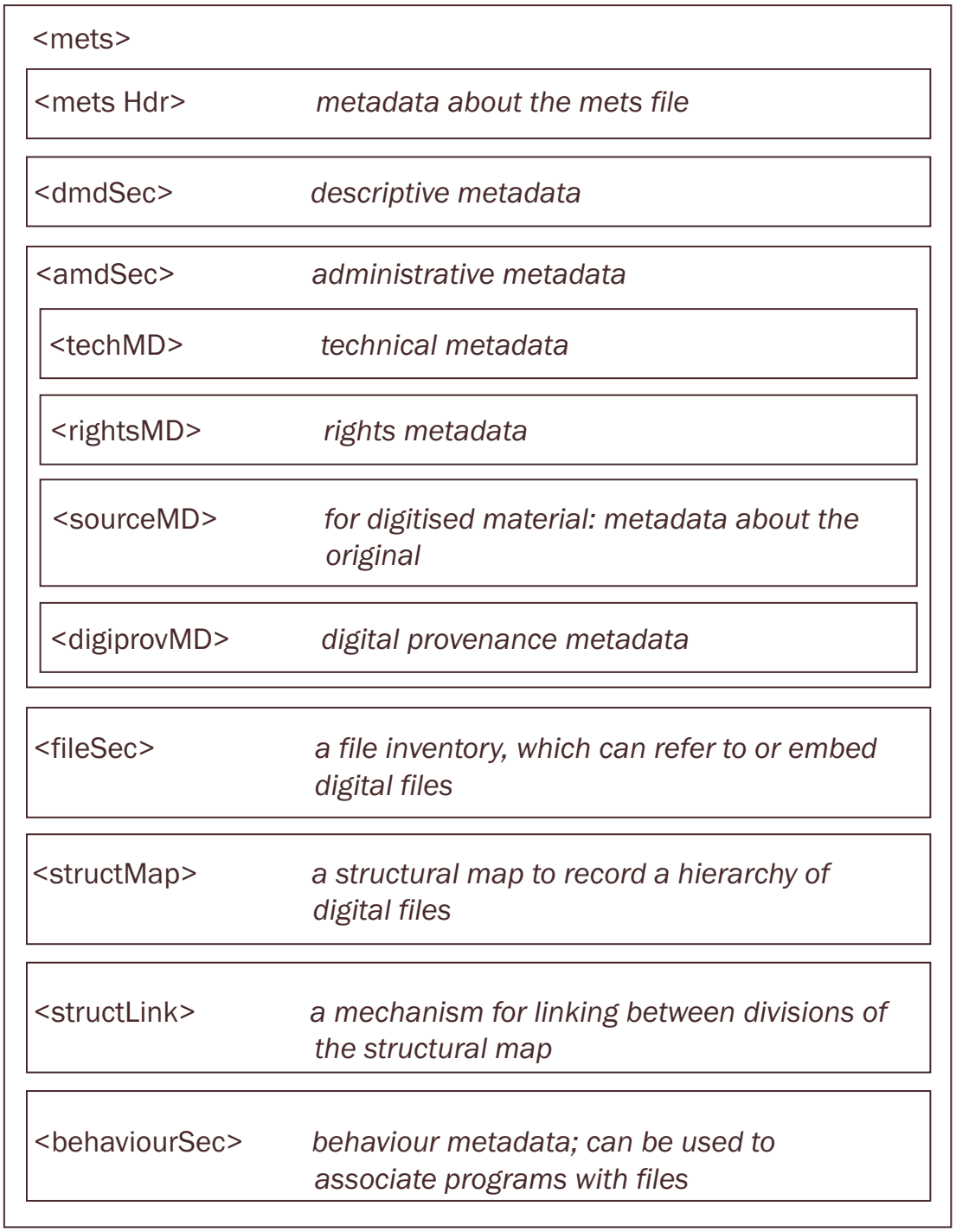

Figure 15: METS Sections

The Library of Congress is the maintenance agency for METS.[1] It is governed by the METS Editorial Board, which also promotes the standard, endorses best practice and supports the METS community.

The standard has been widely adopted by institutions and projects worldwide. The METS Implementation Registry[2] contains information about METS projects both planned, in progress and fully implemented, which have been registered with the METS Board. UK users of METS include the National Library of Wales, Oxford Digital Library and the UK Data Archive.

Using other schemas with METS

Other metadata schemas can be incorporated into a METS file or referred to from it. Some of these are included in the schema as values within the MDTYPE attribute associated with the <mdWrap> and <mdRef> elements. Not all of these have XML schemata that are endorsed by the METS Editorial Board: approved schemes are known as METS Extension Schemas and a list of these is provided on the official METS website;[3] usually the Board endorses a particular XML schema only when it has been officially sanctioned by the organisation supporting its development. External schemas commonly used include:

For descriptive metadata:

- Simple Dublin Core: developed by the Dublin Core Metadata Initiative to produce a core set of metadata terms for all kinds of digital objects, and to promote resource discovery across domains. Commonly used for the purposes of OAI-PMH metadata harvesting.
- Metadata Object Description Schema (MODS): developed in a joint initiative led by the Library of Congress; richer than Dublin Core and designed specifically to work with METS. In order to create self-describing digital objects, Paradigm used MODS in its METS files to record item-level descriptive metadata that can be automatically extracted.
- Encoded Archival Description (EAD): administrative and descriptive metadata developed specifically to encode archives and manuscript collections.
- Machine Readable Cataloguing (MARC) in the form of the MARCXML Schema: widely used by libraries to describe analogue and digital materials.
- Visual Resources Association (VRA): a scheme for describing visual images.
- Text Encoding Initiative Header (TEIHDR): a schema for encoding metadata associated with TEI-encoded texts.

For administrative metadata:

- Schema for Technical Metadata for Text (TextMD): for textual documents.
- NISO Technical Metadata for Digital Still Images (XML encoded as MIX): metadata scheme described by a data dictionary that can be used to describe a number of formats of still images.
- Technical metadata for audiovisual formats as specified by the Library of Congress A/V prototyping project (LC-AV).
- Schema for Rights Declaration (METSRights): designed for use with METS to record basic metadata about the intellectual property rights associated with a digital object or its parts (see p. 145).
- Preservation metadata developed by the OCLC-RLG Preservation Metadata Implementation Strategies Working Group (PREMIS); some PREMIS XML schemas have been developed.

1 Network Development and MARC Standards Office, *Metadata Encoding & Transmission Standard website.* URL: <http://www.loc.gov/standards/mets/>

2 Network Development and MARC Standards Office, 'METS Implementation Registry', *Metadata Encoding & Transmission Standard website.* URL: <http://www.loc.gov/standards/mets/mets-registry.html>

3 Network Development and MARC Standards Office, 'METS Extension Schemas', *Metadata Encoding & Transmission Standard website.* URL: <http://www.loc.gov/standards/mets/mets-extenders.html>

Other types of metadata can also be used with METS, by selecting the OTHER value in the MDTYPE attribute and naming the scheme within the OTHERMDTYPE attribute.

METS profiles

METS profiles describe the application of METS in a specific project or institution. These profiles are expressed in XML and provide detailed guidance for others in creating and processing METS documents, for a particular class, conforming to that profile. For example, the Library of Congress are creating a different METS profile for typical Library items, such as books, photographs and CDs. Institutions creating profiles may register them with the Library of Congress Network Development and MARC Standards Office after they are approved by the METS Editorial Board; such profiles will be published so that others can create METS files that conform to them.

A schema for compiling a METS profile is provided via the METS website; projects and institutions which register a profile should ensure that their METS files reference this profile.

Strengths of METS

- METS is a standard maintained by the Network Development and MARC Standards Office of the Library of Congress, and is non-proprietary. Any system capable of handling XML documents can be used to create, store and deliver a METS file, thereby mitigating problems of software obsolescence.
- It is written in XML, which is robust as an archival medium and is readily interchangeable because it uses standard ASCII code rather than a binary format to encode its data.
- As an XML Schema it has certain advantages over a DTD. Schemas are generally richer and more powerful than DTDs: they provide greater control of the content and usage of elements and attributes; they are extensible to future additions, meaning that METS can accommodate new standards as they are developed in the future; and they support the use of multiple XML namespaces, which allows different kinds of metadata to be encoded in the same document.
- METS has the ability to deal with a wide variety of materials; this is particularly useful in the context of personal archives which typically contain a wide range of object types – including text documents, images, email directories, websites and blogs.
- METS was specifically designed to act as an OAIS Information Package: it can deal with all categories of OAIS metadata; it packages this metadata with the digital object it describes, ensuring that the object is self-documenting over time; and it intellectually links together all the categories of metadata for an object, even if they are stored in separate locations.
- It was developed by the library community and has been widely adopted in preservation repositories, although its use in the context of personal digital archives has been little explored to date.
- It effectively expresses the hierarchical structure of digital objects. This is very useful in the case of personal archives, as it enables the creator's original structure of directories and folders to be maintained.
- Whilst METS has the capability to deal with large and complex digital objects which might be comprised of many files, it is also useful for dealing with individual files (e.g. a single word-processed document). The latter is useful for digital archives, where long-term digital preservation requires that extensive metadata be recorded for each individual digital object.
- The possibility of creating multiple structural maps in a METS document means that whilst it is invaluable for preserving original hierarchical relationships between objects, archivists can also take advantage of its capacity for sorting and reordering records in varied ways for researcher access.
- It is relatively easy for archivists familiar with XML to pick up: it copes well with the kind of hierarchical structures archivists are used to dealing with, and the mark-up will be relatively familiar to archivists who are used to encoding archival finding aids in EAD.
- It is extensible: new versions of metadata may be incorporated alongside older versions of metadata, thus providing an audit trail.

Weaknesses of METS

- The very flexibility of METS can raise interoperability problems. It does not ensure standardisation because it does not operate as a metadata standard, rather as a framework within which metadata can be stored.
- Whilst using METS Profiles can mitigate these problems to an extent and facilitate manual cross-mapping, this still does not allow the automatic transfer of files between systems.
- At present, and certainly in the case of personal digital archives (which are not only extremely varied but also require large amounts of preservation metadata), METS documents largely have to be generated manually; this is unfeasible in the long-term, and solutions will have to be found.
- METS relies on the effective use of unique identifiers. This can be difficult to administer.

Alternatives to METS

METS is not the only framework for packaging disparate metadata; other XML schemas for use in the digital preservation environment have also been proposed. They include:

- The Sharable Content Object Reference Model (SCORM):[1] a packaging model developed for use with learning objects.
- The MPEG-21 Digital Item Declaration Language (DIDL):[2] used in commercial applications and has some proponents in the digital library community. It is an abstract model but also has an XML syntax and acts as a metadata format containing all the necessary elements for automated metadata harvesting.
- Resource Description Framework (RDF):[3] a family of World Wide Web Consortium specifications; designed as a metadata model using XML, it has also subsequently been taken up as a general method of modeling knowledge through a variety of syntax formats.
- IMS Content Packaging Specification (IMS-CP):[4] developed by the IMS Global Learning Consortium for defining interoperability between systems used with learning objects.
- XML Formatted Data Unit (XFDU):[5] currently under development by the Consultative Committee for Space Data Systems (CCSD), who produced the OAIS model.

METS and the Paradigm Project

Paradigm identified METS as the most appropriate means of storing all the metadata required for long-term preservation; each digital object in a personal archive should have an associated METS document which wraps up, or points to, all the metadata needed to preserve that object, thus forming an Information Package. Relationships with other digital objects will be made manifest by means of the METS structural map mechanism, which details the hierarchy for the entire accession, and internally each object can also record metadata about associated child and parent objects.

METS documents will also be needed for:

- Intellectual constructs, such as folders used by creators in arranging their archives, accessions and collections.
- Metadata about agents, events and rights (as defined by the PREMIS standard, see p. 80).

This Workbook chapter will concentrate on the use of METS as an Information Package for digital objects as defined by the OAIS model. OAIS defines three types of Information Package, and the METS document for a digital object will differ slightly according to which role it is fulfilling at any one

1 Advanced Distributed Learning, 'SCORM', *Advanced Distributed Learning website.* URL: <http://www.adlnet.gov/scorm/>
2 MPEG Industry Forum, *MPEG Industry Forum website.* URL: <http://www.mpegif.org/>
3 W3C, 'Resource Description Framework (RDF)', *W3C website.* URL: <http://www.w3.org/RDF/>
4 IMS Global Learning Consortium, 'Content Packaging Specification', *IMS Global Learning Consortium website.* URL: <http://www.imsglobal.org/content/packaging/>
5 Consultative Committee for Space Data Systems (CCSD), *XFDU website.* URL: <http://sindbad.gsfc.nasa.gov/xfdu/>

stage of the lifecycle:

- As Submission Information Packages (SIPs): in some contexts, data creators supply data preservers with structured metadata. In these instances METS can be used to wrap the objects and the metadata together, and the preserving service could impose a standard METS profile for this purpose. This is highly unlikely in the context of a collecting archive, but might happen where the data creators are library staff undertaking a digital project of some kind.
- As Archival Information Packages (AIPs): this is the key stage in the long-term digital preservation of digital objects. Each digital object will have its own dedicated METS file containing comprehensive administrative metadata and will link to related METS files detailing relevant events, agents and rights which are associated with the digital object. This group of linked METS files therefore constitutes the AIP for a single digital object. There will also be collection and accession AIPs which provide comprehensive structural maps pointing to the AIPs of their children digital objects. There is likely to be less descriptive metadata at this stage than at the next.
- As Dissemination Information Packages (DIPs): METS can act as a delivery package for researchers, who will usually identify the object they are interested in via an EAD catalogue. On calling up an item, they will receive both the digital content itself and some relevant metadata from the METS file; there is likely to be a higher proportion of descriptive information at this stage and rather less administrative metadata. See Chapter 06 *Arranging and cataloguing digital and hybrid archives* for further information about DIPs for individual objects (see p. 189). The DIP should be relatively straightforward to build by extracting the relevant metadata from the current version of the AIP for any single digital object.

Paradigm's progress with METS was stalled by the fact that personal archives contain a huge variety of digital objects with complex relationships. This requires the specification of a detailed content model and the means to automate the generation of METS documents which subscribe to that model. Presently, such METS documents can only be crafted by hand, assisted by the use of various (non-uniform) metadata extraction tools and registries. This is clearly unfeasible as a long-term solution. The CAIRO Project[1] has therefore been set up to address this issue; it aims to develop an integrated, automated workflow, which will produce repository-independent metadata packages in the form of METS documents, that will provide the basis for long term lifecycle management.

Structure of a METS file

While the METS schema is very flexible, it is also tightly structured. A METS document is comprised of seven principal sections:

1. **METS Header <metsHdr>** Contains brief descriptive information about the METS document itself, including date of creation and last modification, current status, names of the agents who have played some role in relation to the document, and the nature of that role.
2. **Descriptive Metadata Section <dmdSec>** Contains descriptive metadata, supplying information on the intellectual content of an object which is necessary for users to find an item and assess its value for their research. It may contain the metadata itself, or point to metadata held outside the METS document. Multiple instances of both external and internal descriptive metadata may be included. For external metadata the <mdRef> element allows the provision of a URI for that metadata.
3. **Administrative Metadata Section <amdSec>** Contains technical information about the digital object, rights management information and provenance information. It is divided into four main sections: technical metadata (re. file creation, format and use characteristics); IPR metadata (re. copyright, licensing etc); source metadata (re. the

1 Cairo Project, *Cairo Project website.* URL: <http://cairo.paradigm.ac.uk>

analogue source from which a digital object derives, where relevant); digital provenance metadata (re. source of files, relationships between files, information about any migration or other preservation activities undertaken).

4. **File Section <fileSec>** A list of the file(s) that make up the digital object. Each is given an ID and its physical location is indicated. The files can be grouped within FileGrp elements, to provide for subdividing the files by object version or other criteria, e.g. file type, size, structure.
5. **Structural Map <structMap>** This is the heart of a METS document and is the only mandatory section. It indicates, by means of a hierarchical structure, how the various components of the digital object (if more than one) relate to each other, and facilitates navigation by the end user. It also includes links to the relevant content files and the metadata relating to each content file.
6. **Structural Link Section <structLink>** This section contains only one (repeatable) element, <smLink>, which facilitates hyperlinks between items within the structural map. This is a useful facility when using METS to present websites or other hypermedia.
7. **Behaviour Section <behaviorSec>** Provides information on how particular components of the digital object should be rendered for the user. This may include information on specific software packages to be used, or on particular parameters to be used when rendering a file.

The descriptive and administrative metadata sections, the file section, and the structural map, can all be allocated unique identifiers; by means of the IDREF mechanism, these IDs can be used to link all of these sections together.

This overview will take each section in order, although the METS *Primer and Reference Manual*[1] recommends starting with the file section, followed by the structural map, when creating a METS document.

In the examples below, a single email is used as an example content file. For the purposes of long-term preservation and packaging, each digital object within the Paradigm testbed (like this email) will have its own METS document, which links to other METS documents describing the preservation agents, events and rights associated with the digital object. The purpose of these METS documents is to facilitate long-term preservation and to create a self-describing Information Package (AIP), rather than to facilitate navigation and user access. In order to facilitate access (via the DIP), Paradigm proposes using a higher level METS document which, by means of its structural map, will be used to organise lower-level METS documents to provide different methods of access for the user. Some examples of alternative ways of using the structural map are also included here. None of these examples reflect definitive Paradigm practice; they are merely offered as possible models of working.

Root element The root element <mets> provides the overall container for the information being stored or transmitted, which is held within the seven sections listed above. This element can contain a number of attributes; all of these are optional, although the *METS Primer and Reference Manual* strongly recommends using the OBJID attribute, which is the primary identifier assigned to the METS document as a whole and operates to tag the METS document to external systems.

The root element can also contain the namespaces (xmlns) and schema instance locations (xsi) of the external standards referenced in the METS record. This is shown in the example AIP METS root below, which contains: descriptive metadata encoded in both Dublin Core and the Metadata Object Description Schema (MODS); administrative metadata encoded in TextMD, METSRights and PREMIS; and XML Linking Language (xlink) for linking between the File Section and Structural Map.

1 METS Editorial Board, *<METS> Metadata Encoding and Transmission Standard: Primer and Reference Manual*, (Version 1.6, 30 September 2006). URL: <http://www.loc.gov/standards/mets/METS%20Documentation%20final%20070930%20msw.pdf> [last accessed 25 Oct 2007]

Example:

```
<mets:mets xmlns:mets="http://www.loc.gov/METS/"
  xmlns:mods="http://www.loc.gov/mods/v3"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:mix="http://www.loc.gov/mix/"
  xmlns:metsrights="http://cosimo.stanford.edu/sdr/metsrights/"
  xmlns:premis="http://www.loc.gov/standards/premis"
  xmlns:textMD="http://dlib.nyu.edu/METS/textmd.htm"
  xsi:schemaLocation="http://www.loc.gov/METS/
  http://www.loc.gov/standards/mets/mets.xsd
  http://www.loc.gov/mods/v3
  http://www.loc.gov/standards/mods/v3/mods-3-0.xsd
  http://purl.org/dc/elements/1.1/
  http://dublincore.org/schemas/xmls/simpledc20021212.xsd
  http://www.loc.gov/mix/
  http://www.loc.gov/standards/mix/mix.xsd
  http://cosimo.stanford.edu/sdr/metsrights/
  http://cosimo.stanford.edu/sdr/metsrights.xsd
  http://www.w3.org/1999/xlink
  http://www.loc.gov/standards/mets/xlink.xsd
  http://www.loc.gov/standards/premis
  http://www.loc.gov/standards/premis/PREMIS-v1-0.xsd
  http://dlib.nyu.edu/METS/textmd.htm
  http://dlib.nyu.edu/METS/textmd.xsd" OBJID="representation:1233" LABEL="Folder">
```

Here, the OBJID attribute is used to record the identifier that will be assigned to the object, perhaps by the digital repository software (e.g. Fedora). There is also a label to identify the resource to human users, which includes the subject line of the email.

METS Header <metsHdr> The METS Header contains some basic information describing the METS document itself. This might include details of the document author or any other agents who have played a role in the METS document (e.g. editor); this does *not* relate to agents who are associated with the digital object described by the METS document. The <metsHdr> element can also include dates and times of creation and modification; and any alternative identifiers for the document.

Example:

```
<mets:metsHdr CREATEDATE="2006-11-21T15:10:56">
  <mets:agent ROLE="CREATOR" TYPE="ORGANIZATION">
    <mets:name>Paradigm project</mets:name>
  </mets:agent>
</mets:metsHdr>
```

Descriptive Metadata Section <dmdSec> Archivists have long been accustomed to producing descriptive metadata so that researchers can identify and retrieve archival content, so this <dmdSec> section will probably be most familiar to those working with traditional archives.

The <dmdSec> of METS is repeatable; this allows descriptive metadata to be recorded for each separate item or component in the METS document. In the case of the testbed digital objects held by Paradigm, each of which has its own individual METS file, the advantage has a different emphasis. At each stage of a digital archival object's lifecycle, differing quantities of descriptive metadata will be required. For example, at AIP stage administrative metadata is most important and descriptive metadata might be represented simply in the form of a basic MODS or Dublin Core record. MODS is the option selected by the Bodleian Library, both because it is richer than Dublin Core and because it is used in various digital library contexts at Oxford, and is therefore useful for local interoperability. At DIP stage, a considerably higher proportion of descriptive information is needed to facilitate intellectual access for researchers. Existing MODS metadata could be retained

and possibly enhanced, and a Dublin Core record could be added to items published in an online repository for the purpose of OAI-PMH harvesting. The object would also have an additional layer of descriptive metadata in the form of an entry (although not necessarily to item level) in a detailed EAD catalogue (referred to from the item's METS DIP) for the archive of which it forms a part.

METS does not define the content of descriptive metadata elements; instead it allows descriptive metadata from other schemes to be incorporated in a METS file using one of two methods: it can either be embedded in the METS file itself, using the <mdWrap> element, along with the <xmlData> element (if the metadata is in XML form) or the <binData> (if not in XML); or it can be stored in an external file and referred to from the METS file by means of a URI, using the <mdRef> element.

A unique ID attribute can be assigned to each <dmdSec> element, which facilitates linking from other sections of the METS document. There is also an optional GROUPID attribute which is used to indicate that different metadata sections may be considered as part of a group; this is useful for grouping changed versions of the same metadata if previous versions are maintained in a file for tracking purposes.

The <dmdSec> element can also include an ADMID attribute, which can be used to link it to relevant administrative metadata sections that relate to the digital object described; this is done by citing as attribute values the IDs allocated to each administrative metadata section.

Internal IDs play an important role in any METS document. Each descriptive and administrative metadata section is given an ID which is unique within the METS file; these can be referred to from other elements, as described above. This allows units of information which appear in dispersed locations across a METS document to be linked to all their appropriate contexts.

The Paradigm model uses <mdWrap> to embed a MODS record within the METS document for a digital object at AIP stage; at DIP stage (for items published to online repositories) a Dublin Core record could also be embedded using <mdWrap> and all items belonging to catalogued archives would include the <mdRef> element to link to an external EAD catalogue entry.

Paradigm recommends that the following attributes be associated with the <mdRef> element:

- MIMETYPE, to specify the MIME type of the external EAD document.
- LABEL to identify the linked EAD record for researchers.
- XPTR would be used where a link to a specific point in the EAD document (e.g. the section of the catalogue which describes the digital object represented by the METS document) is required.
- LOCTYPE, which specifies the locator type (e.g. URL).
- MDTYPE, which allows us to specify what form of metadata is being referenced; the METS schema identifies a number of values for this attribute (as well as an additional OTHER option). EAD is supplied as one of the allowable terms.

Example DIP:
This example illustrates what the descriptive metadata sections of a METS document might look like in a DIP for a single email published to an online repository. Much of the information supplied in the Dublin Core and MODS records will be extracted automatically, including: the title (the subject line of the email, in this case 'Latest draft of election press release'); the name of the creator (the sender of the email, in this case a member of politician's staff); and the date and time the email was sent. The EAD reference code for the email is ABC/1/3/9/670/1 (this is also given as the identifier in both the Dublin Core and MODS records). Each descriptive metadata section is also given a unique identifier (for referencing from elsewhere) and an explanatory label for researchers. Sections of administrative metadata relating to the same digital object are referenced using the ADMID attribute. Policy on formats for IDs will be decided at local level, as they are not intended as external identifiers.

The ENCODING attribute value of "w3cdtf" simply refers to the way in which the date is represented (based on a profile of the ISO 8601 standard specifying the pattern YYYY-MM-DD).

```
<mets:dmdSec ID="DMD01M" ADMID="AMDRts01M AMDdprov01M">
    <mets:mdWrap MIMETYPE="text/xml" MDTYPE="MODS" LABEL="MODS Metadata">
        <mets:xmlData>
            <mods:mods version="3.0">
                <mods:identifier>GB-0133-ABC/1/3/9/670/1</mods:identifier>
                <mods:titleInfo>
                  <mods:title>Latest draft of election press release</mods:title>
                </mods:titleInfo>
            <mods:name type="personal">
                <mods:displayForm>Name of sender (member of politician's staff), expressed
                  using NCA Rules</mods:displayForm>
                <mods:role>
                  <mods:roleTerm type="text">creator</mods:roleTerm>
                </mods:role>
            </mods:name>
            <mods:originInfo>
                <mods:dateCreated encoding="w3cdtf">
                2005-04-28T10:15:00+01</mods:dateCreated>
            </mods:originInfo>
            <mods:abstract>Used to record a brief account of content. This would probably include any
            automatically extracted metadata which does not appear in other fields, including: the recipi-
            ent's identity and email address; the date and time the message was received; message priority
            level; information about attachments (this example email has a Microsoft Word document as
            an attachment, which is discussed below, see p. 134); and the identities of any other recipients
            included in the cc field of the email.</mods:abstract>
            </mods:mods>
        </mets:xmlData>
    </mets:mdWrap>
</mets:dmdSec>
<mets:dmdSec ID="DMD02M" ADMID="AMDRts01M AMDdprov01M">
    <mets:mdWrap MIMETYPE="text/xml" MDTYPE="DC" LABEL="Dublin Core Metadata">
        <mets:xmlData>
            <dc:dc>
                <dc:identifier>GB-0133-ABC/1/3/9/670/1</dc:identifier>
                <dc:title>Latest draft of election press release</dc:title>
                <dc:creator>Name of politician's staff member, expressed using NCA Rules</dc:creator>
                <dc:date>2005-04-28T10:15:00+01</dc:date>
                <dc:description>Brief account of content, including automatically extracted metadata, as in
                the <MODS:abstract> field above.
                </dc:description>
            </dc:dc>
        </mets:xmlData>
    </mets:mdWrap>
</mets:dmdSec>
<mets:dmdSec ID="DMD3M">
        <mets:mdRef MIMETYPE="application/xml" LABEL="EAD catalogue description" XPTR="abc23"
        LOCTYPE="URL" xlink:href="http://www.paradigm.ac.uk/ABC.xml" MDTYPE="EAD"/>
</mets:dmdSec>
```

In the <mdRef> element here, the XPTR attribute points to a unique ID which has been added as an attribute value within a specific EAD component level tag (at either folder or item level) in an external EAD catalogue, e.g. <c03 id="abc23">. For more information on linking between EAD and METS documents, see Chapter 06 *Arranging and cataloguing digital and hybrid archives* (p. 174).

Administrative metadata section <amdSec> The <amdSec> acts as a holder for the key information which is central to long-term digital preservation - enabling the repository to manage the material effectively, ensuring that the digital object is authentic and clarifying intellectual property rights in the object. At AIP stage, administrative metadata will be extensive, while at DIP stage less of this information will be included.

The <amdSec> contains four principal subelements, all of which are optional and repeatable. METS does not prescribe the content of any of these administrative metadata sections, although

it recommends a number of extension schemas for each type of metadata. As with descriptive metadata, administrative metadata may be embedded using <mdWrap> or stored externally and referred to using <mdRef>. The four main sub-sections within <amdSec> are as follows:

Technical metadata <techMD> Contains information about the generation of the digital object represented by the METS document, including its creation, format and use characteristics. Where relevant, METS extension schemas specific to particular object types, e.g. MIX for images and TextMD for text, should be used.

Intellectual property rights metadata <rightsMD> Contains information about any copyright and licensing attached to the digital object. The METSRights schema has been specifically developed for recording this kind of information in METS.

Source metadata <sourceMD> Used to record information about the analogue source of a digitised record (perhaps the MARC record for a digitised book). This element is not relevant to born-digital material.

Digital provenance metadata <digiprovMD> Records information about master/derivative relationships between the current digital object and its earlier forms, as well as recording information about format transformations and other preservation actions undertaken by the repository in relation to the object. Some XML schemas have been produced to encode the core digital preservation metadata elements specified by the *PREMIS Data Dictionary* (see p. 80), and these have been adopted as METS Extension Schemas. The model described here includes all PREMIS object metadata within the <digiprovMD> element of METS, but creates separate METS files for objects, agents, events and rights. Best practice combining METS/PREMIS is still nascent and some advocate splitting PREMIS entities across various amdSec sections (e.g. see p. 113).

METS allows the entire <amdSec> to be allocated a single unique ID by which it can be referred to from other parts of the METS document by means of the ADMID linking attribute.

Example:
<mets:amdSec ID="AMD01M">
</mets:amdSec>

Each of the four major elements within <amdSec> can also be allocated a unique ID. Where a single METS document represents a single object (as with the Paradigm model), it may be sufficient to use but one ID at the highest <amdSec> level; however, to avoid ambiguity (for example, when creating a DIP METS file from the data in an AIP file) or to allow for the addition of extra metadata sections in future), it may be advisable to include IDs for <techMD>, <rightsMD> and <digiprovMD> separately.

The XML examples below are based on the email described in the <dmdSec> example above and give a basic indication of the kind of metadata included in <amdSec> subsections for digital object METS documents at the AIP stage. Information on which elements of administrative metadata might be included in a DIP are given in Chapter 06 *Arranging and cataloguing digital and hybrid archives* (see p. 189).

Technical metadata <techMD> For email and other text-based documents, the TextMD schema is used to embed technical metadata in the METS document, where such information does not duplicate the PREMIS record contained in <digiprovMD>. The METS Editorial Board recommends the texMD schema for encoding technical information about textual documents, whether digitised or born-digital.

Example:
In this simple example, the TextMD record is embedded within the METS document in XML form; the <techMD> element has been given a unique identifier, and a human-readable label is provided. The character set employed by the digital object is given (as stipulated in the TextMD Schema) using a controlled vocabulary established by the Internet Assigned Numbers Authority (IANA). The language of the email is given using the ISO 639-2 code to denote English, and the default font of the message is also supplied.

```
<mets:techMD ID="AMDTech01M">
    <mets:mdWrap MIMETYPE="text/xml" MDType="TextMD" LABEL="Technical Metadata for Text">
        <mets:xmlData>
            <textMD:textMD>
                <textMD:encoding>
                  <textMD:encoding_software version="6.2">oXygen</textMD:encoding_software>
                </textMD:encoding>
                <textMD:character_info>
                  <textMD:charset>ANSI_X3.4-1968</textMD:charset>
                  <textMD:byte_order>little</textMD:byte_order>
                  <textMD:byte_size>400</textMD:byte_size>
                  <textMD:character_size> </textMD:character_size>
                  <textMD:linebreak>CR/LF</textMD:linebreak>
                </textMD:character_info>
                <textMD:language>eng</textMD:language>
                <textMD:font_script>Times New Roman</textMD:font_script>
                <textMD:markup_basis>HTML</textMD:markup_basis>
                <textMD:textNote>Notes not covered elsewhere</textMD:textNote>
            </textMD:textMD>
        </mets:xmlData>
    </mets:mdWrap>
```

Intellectual property rights metadata <rightsMD> The Intellectual property rights (IPRs) associated with a digital object have a bearing both on preservation activities carried out within the repository (making multiple copies for preservation purposes, migrating into different formats, etc.) and on access and use by researchers (obtaining copies, downloading, quoting, etc.). Paradigm used the <rightsMD> to record IPR information which relates to access and use; information will be drawn from here to form the rights metadata section of the DIP, although some information held in the AIP, such as rights holder's contact details, may not be published in the DIP. IPR information relating to preservation actions will be recorded in a separate METS document using the PREMIS rights entity within <digiprovMD> and will form part of the comprehensive IPR record required at AIP stage; this will not be made available to users as part of the DIP.

Example from Object AIP:
Rights information for access and use is wrapped in the METS document using the METSRights schema (see p. 145). In this simple example – based on our single email message – the rights category is given as "Copyrighted", because textual documents like this will usually remain in copyright until 70 years after the death of their creator. Other values METSRights provides for this attribute include "Public Domain" (for material which is out of copyright), "Licensed" (for material which is subject to a licence; this might be a creative commons licence or a specific licence granted to the digital repository), "Contractual" (which might apply in the case of principal archive creators who grant copyright permission as part of their deposit or donation agreement with the repository) as well as "Other" to cover alternative statuses.

While contact details are included in this example, in reality it is likely to be impossible to record (or even to discover) current contact details for copyright holders in individual records like this, unless they have created a substantial proportion of the material held in the archive.

The <Context> element has been used to outline the access and use rights of one category of researcher (academic users), although further user categories (e.g. "General public", "Managed Grp") could be created. Because this email is still in copyright, it will only be made available under strict conditions (outlined in the <ConstraintDescription> element), and the attributes within the <Permissions> element indicate that the specified user group can discover the object (by means of descriptive metadata) and display it on the screen, but they are not permitted to duplicate, modify or delete it. The copyright holder has given permission for academic researchers to make one copy of the digital object to disk. Donor-stipulated access conditions like this are recorded in the AIP for administrative purposes; the DIP would derive its METSRights metadata from the AIP in order to provide researchers with details of access conditions.

```
<mets:rightsMD ID="AMDRts01M">
<mets:mdWrap MIMETYPE="text/xml" MDType="METSRights" LABEL="Rights Declaration
information">
    <mets:xmlData>
        <metsrights:RightsDeclarationMD>
        <metsrights:RightsDeclaration RIGHTSCATEGORY="Copyrighted">
            The content of this email remains in the copyright of the creator or their estate.</metsrights:
            RightsDeclaration>
            <metsrights:RightsHolder>
                <metsrights:RightsHolderName>
                Name of the member of politician's staff who composed and sent the email message.
                </metsrights:RightsHolderName>
                <metsrights:RightsHolderContact>
                <metsrights:RightsHolderContactDesignation>
                Name or title of the contact person/institution holding rights; this will often be the same as
                the rights holder.
                </metsrights:RightsHolderContactDesignation>
                <metsrights:RightsHolderContactAddress>
                Use to record the postal address of rights holder.
                </metsrights:RightsHolderContactAddress>
                </metsrights:RightsHolderContact>
            </metsrights:RightsHolder>
            <metsrights:Context CONTEXTCLASS="Academic User">
                <metsrights:UserName USERTYPE="Group">
                Academic and other bona fide researchers.
                </metsrights:UserName>
                <metsrights:Permissions DISCOVER="true" DISPLAY="true" COPY="true"
                DUPLICATE="false" MODIFY="false" DELETE="false" PRINT="false"></metsrights:
                Permissions>
                <metsrights:Constraints>
                <metsrights:ConstraintDescription>
                This item can only be made available to registered readers who have signed a declaration
                form; it will be made available in the reading room on a non-networked PC. The copyright
                holder has given permission for researchers to make one copy of this item to disk for private
                study purposes. If a researcher wishes to publish or quote from a copyright item in a publica-
                tion, permission must be sought from the copyright holder.
                </metsrights:ConstraintDescription>
                </metsrights:Constraints>
            </metsrights:Context>
            </metsrights:RightsDeclarationMD>
        </mets:xmlData>
</mets:mdWrap>
</mets:rightsMD>
```

Digital provenance metadata <digiprovMD> The <digiprovMD> element records information which allows both repository staff and users to understand what modifications have been performed on a digital object during its lifecycle in order to judge whether and how those processes might have altered or corrupted the 'original' object. Four PREMIS XML Schemas have been produced to represent four of the entities outlined in the *PREMIS Data Dictionary*. These schemas: object, agent, event and rights - may be used in conjunction with METS. Paradigm proposes that METS documents for each of the four entities be created and linked together using the linking elements supplied by PREMIS. In this example all PREMIS metadata will be recorded in the <digiprovMD> of METS. The rights metadata held by PREMIS is that which establishes the right of the repository to undertake preservation actions on the digital objects in its care. Format-specific technical metadata will be recorded in <techMD>.

Example from object AIP:
This example deals with PREMIS' object entity (see p. 82), which is analogous to the digital object (e.g. an email) that is the target of preservation.

The example below gives the unique identifier for the file generated by the Fedora repository software (and also encoded as the OBJID attribute of the METS document); full preservation is being undertaken, and the file is not encrypted in any way, so its composition level is set to 0. Information is provided within the <fixity> element on a checksum carried out on the object, and its size is recorded in bytes. The email was created in Microsoft Outlook 2002 and arrived at the repository in its original format (personal folders, or .pst). It was subsequently extracted and normalised to XML format using Xena software. The PREMIS <relationship> element records the relationship between the normalised email (file:2345) and the original mailbox (file:2301) from which it was derived; the normalisation is recorded as an event in a separate METS document (event:312). There is also a reference to the rights information relating to this email, which is recorded in another METS document (rights:2303).

```
<amdSec>
 <mets:digiprovMD ID="file2345AMDdprov01M">
   <mets:mdWrap MIMETYPE="text/xml" MDTYPE="PREMIS" LABEL="PREMIS object preservation
   metadata">
     <mets:xmlData>
       <premis:object type="file">
         <premis:objectIdentifier>
           <premis:objectIdentifierType>Local</premis:objectIdentifierType>
           <premis:objectIdentifierValue>file:2345</premis:objectIdentifierValue>
         </premis:objectIdentifier>
         <premis:preservationLevel>Full</premis:preservationLevel>
         <premis:objectCategory>File</premis:objectCategory>
         <premis:objectCharacteristics>
           <premis:compositionLevel>0</premis:compositionLevel>
           <premis:fixity>
             <premis:messageDigestAlgorithm>MD5</premis:messageDigestAlgorithm>
             <premis:messageDigest>9744c39e74af829d21f272e7654b1429
             </premis:messageDigest>
           </premis:fixity>
           <premis:size>12288</premis:size>
           <premis:format>
             <premis:formatDesignation>
               <premis:formatName>Xena email format RFC 2004/2
               </premis:formatName>
             </premis:formatDesignation>
           </premis:format>
           <premis:significantProperties>This unstructured element should detail important characteris-
           tics of the digital file which cannot be recorded elsewhere.</premis:significantProperties>
         </premis:objectCharacteristics>
         <premis:creatingApplication>
           <premis:creatingApplicationName>Xena Lite 3.0</premis:creatingApplicationName>
           <premis:dateCreatedByApplication>2006-11-21T10:10:03
           </premis:dateCreatedByApplication>
         </premis:creatingApplication>
         <premis:relationship>
           <premis:relationshipType>derivation</premis:relationshipType>
           <premis:relationshipSubType>XenaNormalisation</premis:relationshipSubType>
           <premis:relatedObjectIdentification>
             <premis:relatedObjectIdentifierType>Local</premis:relatedObjectIdentifierType>
             <premis:relatedObjectIdentifierValue>file:2301</premis:relatedObjectIdentifierValue>
             <premis:relatedObjectSequence>1</premis:relatedObjectSequence>
           </premis:relatedObjectIdentification>
           <premis:relatedEventIdentification>
             <premis:relatedEventIdentifierType>Local</premis:relatedEventIdentifierType>
             <premis:relatedEventIdentifierValue>event:312</premis:relatedEventIdentifierValue>
             <premis:relatedEventSequence>0</premis:relatedEventSequence>
           </premis:relatedEventIdentification>
         </premis:relationship>
         <premis:linkingPermissionStatementIdentifier>
           <premis:linkingPermissionStatementIdentifierType>Local</premis:
           linkingPermissionStatementIdentifierType>
           <premis:linkingPermissionStatementIdentifierValue>rights:2303</premis:
           linkingPermissionStatementIdentifierValue>
```

```
          </premis:linkingPermissionStatementIdentifier>
        </premis:object>
      </mets:xmlData>
    </mets:mdWrap>
  </mets:digiprovMD>
</amdSec>
```

Example from rights AIP

The rights section of the PREMIS record focuses on permission granted by the depositor of the archive to enable the repository to take various actions in relation to the digital material for preservation purposes; this permission is granted for a finite term in a 20-year deposit agreement.

```
<mets:amdSec>
  <mets:rightsMD ID="rights2303AMDrights01M">
    <mets:mdWrap MIMETYPE="text/xml" MDTYPE="PREMIS" LABEL="PREMIS rights preservation
    metadata">
      <mets:xmlData>
        <premis:rights>
          <premis:permissionStatement>
            <premis:permissionStatementIdentifier>
              <premis:permissionStatementIdentifierType>Local</premis:
              permissionStatementIdentifierType>
              <premis:permissionStatementIdentifierValue>rights:2303
            </premis:permissionStatementIdentifierValue>
          </premis:permissionStatementIdentifier>
          <premis:linkingObject>file:2345</premis:linkingObject>
          <premis:grantingAgent>The name of the person(s) holding rights in the email should be record-
          ed here.</premis:grantingAgent>
          <premis:grantingAgreement>
            <premis:grantingAgreementIdentification>Deposit agreement dated 5 July 2005.</premis:
            grantingAgreementIdentification>
            <premis:grantingAgreementInformation><p>Permission granted by the depositor for the re-
            pository to undertake any necessary preservation actions on the digital material which forms
            part of the deposit.</p> </premis:grantingAgreementInformation>
          </premis:grantingAgreement>
          <premis:permissionGranted>
            <premis:act>Any necessary preservation action</premis:act>
            <premis:termOfGrant>
                <premis:startDate>2005-07-05</premis:startDate>
                <premis:endDate>2025-07-05</premis:endDate>
            </premis:termOfGrant>
          </premis:permissionGranted>
        </premis:permissionStatement>
      </premis:rights>
    </mets:xmlData>
  </mets:mdWrap>
</mets:rightsMD>
</mets:amdSec>
```

Example from event AIP

The event section of PREMIS is used to record significant events in the life of the object. This event records the normalisation of the Microsoft Outlook pst file which created the email and attachment digital object, one of these emails was the object described above. There are references to two different agents (both represented by separate METS documents): the archivist (agent:10) who authorised the normalisation event, and the software (agent:111) which executed the event.

```
<mets:digiprovMD ID="event312AMDdprov01M">
  <mets:mdWrap MIMETYPE="text/xml" MDTYPE="PREMIS" LABEL="PREMIS event preservation
  metadata">
    <mets:xmlData>
      <premis:event>
        <premis:eventIdentifier>
          <premis:eventIdentifierType>Local</premis:eventIdentifierType>
          <premis:eventIdentifierValue>event:312</premis:eventIdentifierValue>
```

```
        </premis:eventIdentifier>
        <premis:eventType>XenaNormalisation</premis:eventType>
        <premis:eventDateTime>2006-11-21T10:10:03</premis:eventDateTime>
        <premis:eventOutcomeInformation>
          <premis:eventOutcome>Success</premis:eventOutcome>
        </premis:eventOutcomeInformation>
        <premis:linkingAgentIdentifier>
          <premis:linkingAgentIdentifierType>Local</premis:linkingAgentIdentifierType>
          <premis:linkingAgentIdentifierValue>agent:10</premis:linkingAgentIdentifierValue>
          <premis:linkingAgentRole>Authorizer</premis:linkingAgentRole>
        </premis:linkingAgentIdentifier>
        <premis:linkingAgentIdentifier>
          <premis:linkingAgentIdentifierType>Local</premis:linkingAgentIdentifierType>
          <premis:linkingAgentIdentifierValue>agent:111</premis:linkingAgentIdentifierValue>
          <premis:linkingAgentRole>Executing Programme</premis:linkingAgentRole>
        </premis:linkingAgentIdentifier>
        <premis:linkingObjectIdentifier>
          <premis:linkingObjectIdentifierType>Local</premis:linkingObjectIdentifierType>
          <premis:linkingObjectIdentifierValue>file:2301</premis:linkingObjectIdentifierValue>
        </premis:linkingObjectIdentifier>
        <premis:linkingObjectIdentifier>
          <premis:linkingObjectIdentifierType>Local</premis:linkingObjectIdentifierType>
          <premis:linkingObjectIdentifierValue>file:2345</premis:linkingObjectIdentifierValue>
        </premis:linkingObjectIdentifier>
      </premis:event>
    </mets:xmlData>
  </mets:mdWrap>
</mets:digiprovMD>
```

Example from Agent AIP

```
<mets:digiprovMD ID="AMDdprov01M">
  <mets:mdWrap MIMETYPE="text/xml" MDType = "PREMIS" LABEL="PREMIS agent preservation
  metadata">
    <mets:xmlData>
      <premis:agent>
        <premis:agentIdentifier>
          <premis:agentIdentifierType>Local</premis:agentIdentifierType>
          <premis:agentIdentifierValue>agent:111</premis:agentIdentifierValue>
        </premis:agentIdentifier>
        <premis:agentName>Xena Lite 3.0</premis:agentName>
        <premis:agentType>Software</premis:agentType>
      </premis:agent>
    </mets:xmlData>
  </mets:mdWrap>
</mets:digiprovMD>
```

File section <fileSec> The file section is used to provide an inventory of, and location for, the data files comprising the digital object being described by the METS document. It can contain one or more file group (<fileGrp>) elements which can be used to organise the individual files (each recorded in a <file> element) into sets, e.g. in the case of digitised images, there might be groups for thumbnails, reference copies and archival masters.

In contrast, the file section of a typical METS document for a born-digital object in a personal archive is likely to be very simple. A METS document representing a single email, for example, will only contain one <fileGrp> and one <file>.

Example:
This example is based on a single email at AIP stage, when it is in archival storage and is linked to its own individual METS document.

Much of the information in the <fileSec> is conveyed by means of attributes. Here, the <fileSec> as a whole has been allocated a unique identifier which allows it to be referenced from elsewhere in the METS document. Similarly, the <fileGrp> and <file> elements also have unique IDs. The <fileGrp> element can contain an ADMID attribute which links to the relevant administrative metadata sections by means of their IDs. The <fileGrp> element does not allow a similar link to descriptive metadata; this is done at <file> level by means of the DMDID attribute.

The USE attribute indicates the intended use of the files within a <fileGrp>. Frequently used values include master, reference or thumbnails for image files. METS does not prescribe values for this attribute, so these should be determined at local level. Here it has been used to indicate that this version of the file is an AIP rather than a SIP or DIP.

At <file> level an attribute to specify the MIME-type of the file is also available.

METS offers two methods of dealing with content files within the <file> element. They can either be embedded within the METS document using the <FContent> element, or held externally and pointed to by means of the <FLocat> element. The latter is the more usual approach and here the location of the email is given as a URL. The href attribute supplies the URL for the location of the file (although technically optional, this is essential when using the <mptr> (see below) element in the structural map); the title describes the meaning of the link in a human-readable fashion; the "new" value for the SHOW attribute indicates that the digital object (the email) would be shown in a new window; and "onRequest" indicates that it should only be shown at the request of the user.

```
<mets:fileSec ID="Fsec01">
    <mets:fileGrp ID="FGrp01" ADMID="file2345AMDdprov01M file2345DMD01" USE="Archival
    Information Package">
        <mets:file ID="em01" MIMETYPE="text/xml" DMDID="file2345DMD01">
            <mets:FLocat LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/
            fedora/get/file:2345" xlink:title="Link to file content" xlink:show="new" xlink:
            actuate="onRequest"/>
        </mets:file>
    </mets:fileGrp>
</mets:fileSec>
```

The <fileSec> can also handle much more complex digital objects: there is a component byte stream (<stream>) element which can be used to record the existence of separate data streams within a particular file (e.g. separate audio and video streams in an MPEG4 file); and a transform file (<transformFile>) element, which provides a means of accessing any subsidiary files listed below a <file> element by indicating the steps required to unpack or transform the subsidiary files.

Structural map <structMap> The <structMap> sits at the heart of a METS document; it organises the digital content represented by the <file> elements in the METS <fileSec> into a coherent hierarchical structure. More than one <structMap> can be included in a METS document, so more than one method of organisation is possible – e.g. physical, logical or a mixture of the two. The structural divisions in the map are represented by division ( ) elements, which can be nested to any depth to allow for very complex hierarchies.

As well as organising content, the <structMap> provides a mechanism for linking content at any hierarchical level with the relevant descriptive or administrative metadata sections in the same METS document.

Nested divisions within the <structMap> can be used to reflect the hierarchy of folders, subfolders and files in a digital accession in an accession-level METS document. The <structMap> for digital objects, such as that representing the single email at AIP stage will be a very straightforward one, containing only one division.

Example:
In the example below, the type of the <structMap> is given as a "single digital file"; METS does not prescribe values for the TYPE attribute, and "physical" or "logical" are the usual examples given; preferred attribute values for this could be established in METS profiles (see p. 120).

The <structMap> only contains one element because the METS document represents a single file. Its type has been set to “email” (again a value not prescribed by METS) and a comprehensive label supplied for the user; labels may differ according to the type of <structMap> which is being presented.

The element contains a file pointer, or <fptr>, element. This is used to link the division to the file content it represents (the email), as recorded in the File Section of the same METS document; the link is made by means of the unique ID (em01) which was allocated to the <file> by means of the ID attribute in the <fileSec>.

```
<mets:structMap TYPE =“Single digital file”>
    <mets:div TYPE=“Email” LABEL =“Email from [name of sender], Latest draft of election
    press release, 2005-04-28T10:15:00+01” DMDID=“DMD01M DMD02M DMD03M”
    ADMID=“AMDTech01M AMDRts01M AMDdprov01M”>
        <mets:fptr FILEID=“em01”/>
    </mets:div>
</mets:structMap>
```

The structural map can also deal with much more complex objects. For example, the <area> element within <fptr> can be used to point to just one portion or area of a file representing as a <file> element in the <fileSec>. In addition, the sequence of files (<seq>) and parallel files (<par>) elements aggregate pointers to files, parts of files, or sequences or sections of files, that must be played or displayed either sequentially or simultaneously to manifest a block of digital content.

The <structMap> is not the final section of a METS document, but often only the five sections covered to this point are all that is needed to represent even fairly complex objects. The final two sections - <structLink> and <behaviourSec> (see p. 123) - are not therefore covered in any detail here.

Alternative uses of the structural map

The repeatable <structMap> element allows archivists to exploit the digital environment to the full by creating multiple arrangements – and therefore multiple means of navigating and accessing – the material in a personal digital archive. The hierarchical nature of the <structMap> is ideal for maintaining the original order of an archive (i.e. the creator’s divisions into different directories, folders, subfolders and files), but it also allows the repository to cater for different categories of user by presenting numerous alternative arrangements of the material. METS does not prescribe any specific types of arrangement, so these might be physical (e.g. s representing page sequence in a digitised book), logical (e.g. s representing poems in a poetry collection, which might span pages in the volume), or a mixture of both. This facility also has great potential for personal digital archives.

The <structMap> in one METS document is not limited to organising the content represented by the <fileSec> of the same METS document. It can also organise content represented by linked, external METS documents. This is achieved by using the METS pointer (<mptr>) element instead of the file pointer (<fptr>) within each : the <mptr> points to content represented by an external METS document, by means of an xlink:href attribute containing a URL marking the location of the relevant METS document.

This means that a digital archive can be arranged by using numerous ‘parent’ METS documents, which do not contain a <fileSec> themselves, but by means of their <structMap> organise further METS documents at a lower level in the hierarchy. Ultimately, the lower level parent METS documents organise and point to METS documents for individual digital objects like the email example explored above.

The following examples take the single email and its METS document as a starting point, and work upwards through this hierarchy of METS documents.

Example 1: Email and attachment (AIP)

This example is based on an email and the attachment; the attachment is in the form of a Microsoft Word file containing a draft of the election press release referred to in the subject line of the email. As an attachment is an integral part of an email, the two digital objects must be unambiguously linked together. This can be achieved by means of a parent METS document containing some basic descriptive metadata, and a <structMap> pointing to the two separate METS documents representing the email message and the attachment. There is no need to include a <fileSec> in this parent document because it does not hold any digital content itself; the administrative metadata specific to each component object will be recorded in its own METS document and does not necessarily need to be represented at this higher level, although a basic <amdSec> could be included. This is a reversal of the rules for descriptive archival cataloguing, where as much common information as possible is given at a higher level; each digital object needs to be self-documenting, so the detail is placed at the lowest possible level.

This example is a rough approximation of what such a METS document might look like at the AIP stage. At DIP stage Dublin Core metadata (for metadata harvesting), a reference to an EAD finding aid and some basic administrative metadata might be included.

```
<?xml version="1.0" encoding="UTF-8"?>
<mets:mets xmlns:mets="http://www.loc.gov/METS/" xmlns:mods="http://www.loc.gov/mods/v3"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:mix="http://www.loc.gov/mix/" xmlns:rts="http://cosimo.stanford.edu/sdr/
metsrights/" xmlns:premis="http://www.loc.gov/standards/premis" xsi:schemaLocation="http://
www.loc.gov/METS/ http://www.loc.gov/standards/mets/mets.xsd http://www.loc.gov/mods/v3
http://www.loc.gov/standards/mods/v3/mods-3-0.xsd http://www.loc.gov/mix/ http://www.loc.
gov/standards/mix/mix.xsd http://cosimo.stanford.edu/sdr/metsrights/ http://cosimo.stanford.
edu/sdr/metsrights.xsd http://www.w3.org/1999/xlink http://www.loc.gov/standards/mets/xlink.
xsd http://www.loc.gov/standards/premis http://www.loc.gov/standards/premis/PREMIS-v1-0.xsd"
OBJID="representation:2344" LABEL="Migrated representation: email and attachment, Latest draft
of election press release">
  <metsHdr CREATEDATE="2006-11-21T15:17:09">
    <agent ROLE="CREATOR" TYPE="ORGANIZATION">
      <name>Paradigm project</name>
    </agent>
  </metsHdr>
  <mets:dmdSec ID="representation2344DMD1" ADMID="representation2344AMDdprov01M">
    <mets:mdWrap MDTYPE="MODS">
      <xmlData>
        <mods:mods>
          <mods:identifier>representation:2344</mods:identifier>
          <mods:titleInfo>
            <mods:title>Migrated Email with attachment: "Latest draft of election press release"</mods:
            title>
          </mods:titleInfo>
          <mods:name type="personal">
            <mods:namePart type="given">Bob</mods:namePart>
            <mods:namePart type="family">Snow</mods:namePart>
            <mods:namePart type="termsOfAddress">Dr</mods:namePart>
            <mods:namePart type="date">1956-</mods:namePart>
            <mods:role>
              <mods:roleTerm type="text">creator</mods:roleTerm>
            </mods:role>
          </mods:name>
          <mods:originInfo>
            <mods:dateCreated encoding="w3cdtf">2006-11-21T10:10:03</mods:dateCreated>
          </mods:originInfo>
          <mods:abstract>Migration 1 of email to [recipient's identity] with attached Microsoft Word file;
          file title "General Election Press Release 20050427".</mods:abstract>
          <mods:relatedItem>
            <mods:identifier>collection:200</mods:identifier>
          </mods:relatedItem>
        </mods:mods>
      </xmlData>
    </mets:mdWrap>
  </mets:dmdSec>
  <mets:amdSec>
```

```
    <mets:digiprovMD ID="representation2344AMDdprov01M">
      <mets:mdWrap MDTYPE="PREMIS" LABEL="PREMIS object preservation metadata"
      MIMETYPE="text/xml">
        <mets:xmlData>
          <premis:object type="representation">
            <premis:objectIdentifier>
              <premis:objectIdentifierType>Local</premis:objectIdentifierType>
              <premis:objectIdentifierValue>representation:2344
              </premis:objectIdentifierValue>
            </premis:objectIdentifier>
            <premis:preservationLevel>Full</premis:preservationLevel>
            <premis:objectCategory>Representation</premis:objectCategory>
            <premis:objectCharacteristics>
              <premis:significantProperties>
              <p>Use this unstructured element to record any relevant characteristics of the two objects
              represented by this METS document: an email in xml format, extracted from a Microsoft Out-
              look 2002 .pst file; with an associated attachment in the form of a Microsoft Word for a Win-
              dows 6.0 file.</p>
              </premis:significantProperties>
            </premis:objectCharacteristics>
            <premis:relationship>
              <premis:relationshipType>derivation</premis:relationshipType>
              <premis:relationshipSubType>XenaNormalisation</premis:relationshipSubType>
              <premis:relatedObjectIdentification>
              <premis:relatedObjectIdentifierType>Local</premis:relatedObjectIdentifierType>
              <premis:relatedObjectIdentifierValue>representation:201</premis:
              relatedObjectIdentifierValue>
              <premis:relatedObjectSequence>1</premis:relatedObjectSequence>
              </premis:relatedObjectIdentification>
              <premis:relatedEventIdentification>
              <premis:relatedEventIdentifierType>Local</premis:relatedEventIdentifierType>
              <premis:relatedEventIdentifierValue>event:312</premis:relatedEventIdentifierValue>
              <premis:relatedEventSequence>0</premis:relatedEventSequence>
              </premis:relatedEventIdentification>
            </premis:relationship>
          </premis:object>
        </mets:xmlData>
      </mets:mdWrap>
    </mets:digiprovMD>
  </mets:amdSec>
  <mets:structMap TYPE="item-level parent document">
    <mets:div TYPE="Email and attachment" LABEL="Email message from [name of sender], Latest
    draft of election press release, 2005-04-28T10:15:00+01, with attached file">
    <mets:div TYPE="email message" LABEL="email message">
      <mets:mptr LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
      file:2345"/>
    </mets:div>
    <mets:div TYPE="attached file" LABEL="attached file">
      <mets:mptr LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
      file:290"/>
    </mets:div>
  </mets:div>
 </mets:structMap>
</mets:mets>
```

In the MODS descriptive metadata section the subject line of the email is given in the <title> field and the title of the attachment is supplied in the <abstract>. The date of creation is given as a span date - from the date when the Microsoft Word attachment was created, to the date when the email was sent. Additional <name> elements should be added if the sender of the email is not the person who authored the press release in the attachment.

There are two elements in the <structMap>, which both point to external METS documents using the <mptr> element: one for the email message itself and the other for the attached Microsoft Word document. The TYPE attribute values are hypothetical; these should be established and recorded in a METS Profile (see p. 120).

Example 2: email folder (DIP)

This example moves up the archival hierarchy one step to an email folder within an email directory; the example represents an email folder called 'Press releases', in which the email and attachment from Example 1 might be stored.

This example contains descriptive metadata in the form of a MODS record as well as some basic administrative metadata; this therefore contains all the metadata needed in a DIP, apart from an additional Dublin Core record and a reference to an EAD finding aid. Most importantly, however, it contains three different structural maps providing users with different ways of accessing the folder contents. The first <structMap> represents the original order of the folder's contents (which is a single sequence of messages arranged chronologically); this is the primary order and the one which would be represented in the EAD catalogue. The second <structMap> is arranged alphabetically by named correspondent; and the last is grouped according to email subject line (allowing users to navigate 'threads' in a particular email correspondence).

```
<?xml version="1.0" encoding="UTF-8"?>
 <mets:mets xmlns:mets="http://www.loc.gov/METS/" xmlns:mods="http://www.loc.gov/mods/v3"
 xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
 instance" xmlns="http://www.loc.gov/METS/" xmlns:mix="http://www.loc.gov/mix/" xmlns:
 metsrights="http://cosimo.stanford.edu/sdr/metsrights/" xmlns:premis="http://www.loc.
 gov/standards/premis" xsi:schemaLocation="http://www.loc.gov/METS/ http://www.loc.gov/
 standards/mets/mets.xsd http://www.loc.gov/mods/v3 http://www.loc.gov/standards/mods/
 v3/mods-3-0.xsd http://www.loc.gov/mix/ http://www.loc.gov/standards/mix/mix.xsd http://
 cosimo.stanford.edu/sdr/metsrights/ http://cosimo.stanford.edu/sdr/metsrights.xsd http://www.
 w3.org/1999/xlink http://www.loc.gov/standards/mets/xlink.xsd http://www.loc.gov/standards/
 premis http://www.loc.gov/standards/premis/PREMIS-v1-0.xsd" OBJID="representation:1233"
 LABEL="Folder">
   <metsHdr CREATEDATE="2006-11-21T15:10:56">
    <agent ROLE="CREATOR" TYPE="ORGANIZATION">
      <name>Paradigm project</name>
    </agent>
   </metsHdr>
 <mets:dmdSec ID="representation1233DMD01F" ADMID="representation1233AMDRts01F
 representation1233AMDdprov01F">
   <mets:mdWrap MDTYPE="MODS">
    <xmlData>
      <mods:mods>
        <mods:identifier>GB-0133-ABC/1/3/9</mods:identifier>
        <mods:titleInfo>
          <mods:title>Email folder: "Press Releases"</mods:title>
        </mods:titleInfo>
        <mods:name type="personal">
          <mods:displayForm>Name of creator</mods:displayForm>
          <mods:role>
            <mods:roleTerm type="text">creator</mods:roleTerm>
          </mods:role>
        </mods:name>
        <mods:originInfo>
          <mods:dateCreated encoding="w3cdtf">2001/2006</mods:dateCreated>
        </mods:originInfo>
        <mods:abstract>Email folder of [name of politician] containing 754 messages, with a total of 367
        attachments.</mods:abstract>
      </mods:mods>
    </xmlData>
   </mets:mdWrap>
   <mets:mdRef MIMETYPE="application/xml" LABEL="View the catalogue for the whole archive"
   LOCTYPE="URL" xlink:href="file:/filepath/eadfile.xml" MDTYPE="EAD"/>
 </mets:dmdSec>
 <mets:amdSec ID="representation1233AMD01F">
   <mets:rightsMD ID="representation1233AMDRts01F">
    <mets:mdWrap MIMETYPE="text/xml" MDTYPE="OTHER" OTHERMDTYPE="METSRights"
    LABEL="Rights Declaration information">
      <mets:xmlData>
        <metsrights:RightsDeclarationMD>
          <metsrights:RightsDeclaration RIGHTSCATEGORY="Copyrighted">Each email remains in the
          copyright of its creator or their estate. The principal archive creator is generally the copyright
```

```
        holder in sent mails.</metsrights:RightsDeclaration>
        <metsrights:RightsHolder>
          <metsrights:RightsHolderName>[Name of principal creator]</metsrights:RightsHolder-
          Name>
          <metsrights:RightsHolderContact>
            <metsrights:RightsHolderContactDesignation>[Name or title of contact person or institu-
            tion holding rights]</metsrights:RightsHolderContactDesignation>
            <metsrights:RightsHolderContactAddress>[Postal address of rights holder or note to see
            reading room staff]
            </metsrights:RightsHolderContactAddress>
          </metsrights:RightsHolderContact>
        </metsrights:RightsHolder>
        <metsrights:Context CONTEXTCLASS="ACADEMIC USER">
          <metsrights:UserName USERTYPE="GROUP">Academic and other bona fide researchers
          </metsrights:UserName>
          <metsrights:Permissions DISCOVER="true" DISPLAY="true" COPY="false"
          DUPLICATE="false" MODIFY="false" DELETE="false" PRINT="false"/>
          <metsrights:Constraints>
            <metsrights:ConstraintDescription>This material is accessible to registered readers
            who have signed a declaration form; it will be made available in the reading room only on
            a non-networked PC. It will be delivered in read-only format, and the making of copies or
            printouts is prohibited unless stated otherwise at individual object-level.</metsrights:
            ConstraintDescription>
          </metsrights:Constraints>
        </metsrights:Context>
      </metsrights:RightsDeclarationMD>
    </mets:xmlData>
  </mets:mdWrap>
</mets:rightsMD>
<mets:digiprovMD ID="representation1233AMDdprov01F">
  <mets:mdWrap MIMETYPE="text/xml" MDTYPE="PREMIS" LABEL="PREMIS preservation
  metadata">
    <mets:xmlData>
      <premis:object type="representation">
        <premis:objectIdentifier>
          <premis:objectIdentifierType>Local</premis:objectIdentifierType>
          <premis:objectIdentifierValue>representation:1233</premis:objectIdentifierValue>
        </premis:objectIdentifier>
        <premis:objectCategory>Representation</premis:objectCategory>
        <premis:objectCharacteristics>
          <premis:size>23040000</premis:size>
        </premis:objectCharacteristics>
        <premis:creatingApplication>
          <premis:creatingApplicationName>Microsoft Outlook 2002
          </premis:creatingApplicationName>
        </premis:creatingApplication>
        <premis:originalName>Press Releases</premis:originalName>
      </premis:object>
    </mets:xmlData>
  </mets:mdWrap>
</mets:digiprovMD>
</mets:amdSec>
<mets:structMap ID="representation1233StructMap01" TYPE="original order">
  <mets:div TYPE="chronological" LABEL="Press Releases email folder in creator's original
  order" DMDID="representation1233DMD01F" ADMID="representation1233AMDRts01F
  representation1233AMDdprov01F">
    <mets:div LABEL="2001-01-15T09:23:13 A. Brown">
      <mets:mptr LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
      file:1234"/>
    </mets:div>
    <mets:div LABEL="2001-01-15T10:01:22 B. Jones">
      <mets:mptr LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
      file:1235"/>
    </mets:div>
```

```
    <mets:div LABEL="2001-01-15T12:01:34 D. Andrews">
      <mets:mptr LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
      file:1236"/>
    </mets:div>
    <!-- more divs would go here -->
    </mets:div>
  </mets:structMap>
  <mets:structMap ID="representation1233StructMap02" TYPE="logical">
    <mets:div TYPE="alphabetical by correspondent" LABEL="Press releases email folder in alphabeti-
    cal order by correspondent" DMDID="representation1233DMD01F" ADMID="representation1233
    AMDRts01F representation1233AMDdprov01F">
      <mets:div LABEL="Messages received from D. Baker">
        <mets:div LABEL="2004-03-02T10:14:41">
          <mets:mptr  LOCTYPE="URL"  xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
          file:1226"/>
        </mets:div>
        <mets:div LABEL="2004-03-24T12:23:10">
          <mets:mptr LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
          file:1243"/>
        </mets:div>
        <!-- more divs would go here -->
      </mets:div>
      <mets:div LABEL="Messages received from A. Brown">
        <mets:div LABEL="2001-01-15T09:23:13">
          <mets:mptr LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
          file:1234"/>
        </mets:div>
        <mets:div LABEL="2002-12-04T10:45:01">
          <mets:mptr  LOCTYPE="URL"  xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
          file:1258"/>
        </mets:div>
        <!-- more divs would go here -->
      </mets:div>
    </mets:div>
  </mets:structMap>
  <mets:structMap ID="representation1233structMap03" TYPE="logical">
    <mets:divTYPE="thread"LABEL="Pressreleasesemailfolderarrangedbythread"DMDID="representa
    tion1233DMD01F" ADMID="representation1233AMDRts01F representation1233AMDdprov01F">
      <mets:div LABEL="Election press release">
        <mets:div LABEL="2005-04-23T09:15:20+01 H. James">
          <mets:mptr  LOCTYPE="URL"  xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
          file:1247"/>
        </mets:div>
        <mets:div LABEL="2005-04-23T16:32:14+01 H. James">
          <mets:mptr LOCTYPE="URL" xlink:href="http://shuttle.paradigm.ac.uk:8085/fedora/get/
          file:1250"/>
        </mets:div>
        <!-- more divs would go here -->
      </mets:div>
    </mets:div>
  </mets:structMap>
</mets:mets>
```

Each of the three structural maps has been given a unique ID to distinguish it from the others. The root element in each case refers (by means of DMDID and ADMID attributes) to the descriptive and administrative metadata relating to the email folder, which is stored in this METS document. The lower level elements point to other METS documents which will contain the same kind of metadata specific to the digital object represented in each. While some of the elements point directly to METS documents for single objects which would include the location of the object content, others point to lower-level 'parent' METS documents containing data about emails and associated attachments.

The first <structMap> above reflects the creator's original order, which means a single chronological sequence of divisions, each representing one email, with a label indicating the date/time of sending and the sender's name. Another <structMap> could be provided in order to nest the divisions into chronological periods based on week or month to make the content more manageable.

The second two structural maps have been classified as “logical” types because they do not reflect the creator’s original order; an artificial arrangement has been imposed on the material with the purpose of facilitating access for users. In the second map the elements are nested into groups based on the name of the correspondent, and in the third example nesting is based on named ‘threads’ (drawn from the email subject line), which have been arranged in alphabetical order.

These possibilities for creating different arrangements of folder content also apply at the higher level of the email directory. While the primary <structMap> would reflect the creator’s arrangement of folders within their directory; additional maps might cut across these folder divisions (which are usually subject-based and ordered alphabetically) to arrange material in different ways, e.g. by correspondent.

Example 3: collection level (DIP)
At the highest level, an overall parent METS document can be used to structure a digital archive in its entirety. Whereas the EAD collection-level description will pull together all the elements (both paper and hard copy) of a hybrid archive in their final archival arrangement, the collection-level METS document can organise the digital elements of the archive into a hierarchical structure which reflects the different components (website, email directory, office files) of the digital archive and their structure into directories, folders, sub-folders and files.

This collection-level file will point, via its <structMap>, to a web of lower level METS documents, most of which will be parent files representing folders (e.g. named email folders or subject folders); these will in turn organise their own contents and point to the lowest level METS documents which each represent a single digital object.

The collection-level METS document would contain some basic descriptive metadata and its administrative metadata would consist of brief information on IPRs which affect the digital part of the archive as a whole. The focus of the document would be the structural map, which is all that is reproduced below. Each would contain a <mptr> element (not included here) which would point to the location of the METS document representing that .

```
<mets:structMap ID=”CollectionSM01” TYPE=”original order”>
  <mets:div LABEL=”Office files”>
    <mets:div TYPE=”Folder” LABEL=”Folder of politician’s staff member A”>
      <mets:div TYPE=”Sub-folder” LABEL=”Annual Reports” />
      <mets:div TYPE=”Sub-folder” LABEL=”General Election” />
      <mets:div TYPE=”Sub-folder” LABEL=”Campaign and Strategy 2004” />
      <mets:div TYPE=”Sub-folder” LABEL=”Press Releases” />
      <!--More divs would go here-->
    </mets:div>
    <mets:div TYPE=”Folder” LABEL=”Folder of politician’s staff member B”>
      <mets:div TYPE=”Sub-folder” LABEL=”Constituents” />
      <mets:div TYPE=”Sub-folder” LABEL=”Crime” />
      <mets:div TYPE=”Sub-folder” LABEL=”Local election” />
      <mets:div TYPE=”Sub-folder” LABEL=”My Pictures” />
      <!--More divs would go here-->
    </mets:div>
    <!--More divs would go here-->
  <mets:div LABEL=”Email directory”>
    <mets:div TYPE=”email folder” LABEL=”Arts, culture and sports briefing” />
    <mets:div TYPE=”email folder” LABEL=”Economy” />
    <mets:div TYPE=”email folder” LABEL=”Local election” />
    <mets:div TYPE=”email folder” LABEL=”Sent items” />
    <mets:div TYPE=”email folder” LABEL=”Trade and industry” />
    <!--More divs would go here-->
  </mets:div>
    <mets:div TYPE=”Website snapshots” LABEL=”Series of dated website snapshots”>
      <mets:div TYPE=”Website snapshot” LABEL=”Snapshot taken on 3 January 2005” />
      <mets:div TYPE=”Website snapshot” LABEL=”Snapshot taken on 3 February 2005” />
      <mets:div TYPE=”Website snapshot” LABEL=”Snapshot taken on 3 March 2005” />
      <!--More divs would go here-->
    </mets:div>
  </mets:div>
</mets:structMap>
```

This diagram gives an indication of the kind of METS-based AIPs that might be created for a personal archive. These represent the digital objects which make up the archive, and intellectual constructs like folders, accessions and collection; they will link to relevant agents, events and rights METS documents.

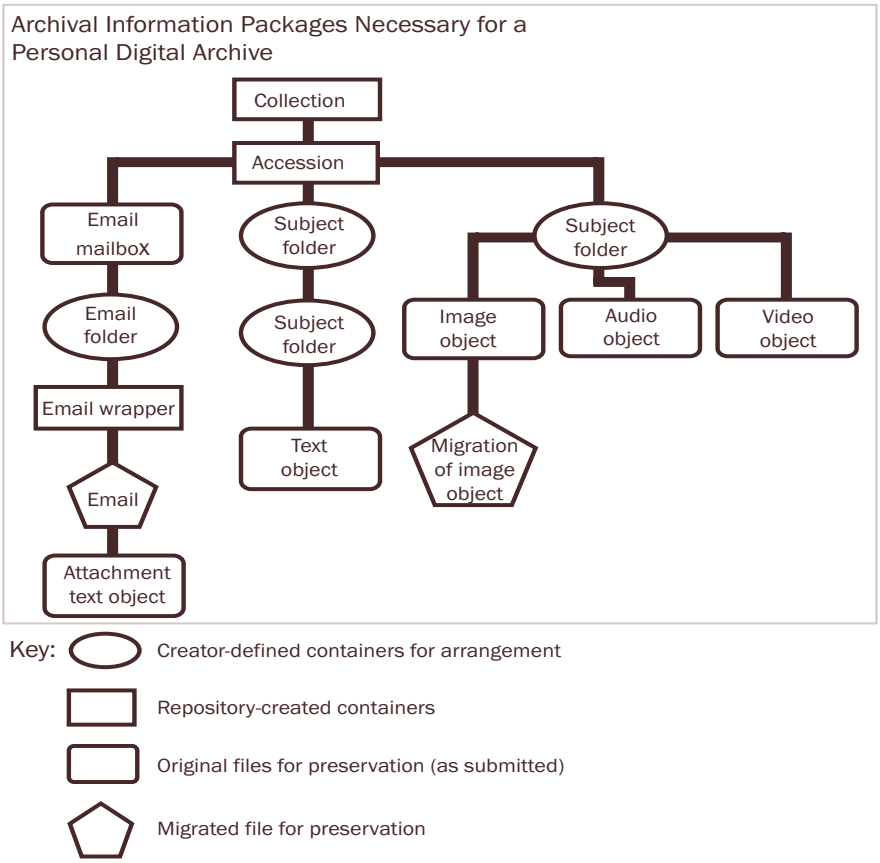

Figure 16: Archival Information Packages Necessary for a Personal Digital Archive

Rights metadata for personal archives

Introduction

Chapter 09 *Legal issues* describes the kinds of Intellectual Property Rights which archivists need to be aware of when collecting and managing digital archives (see p. 252). Copyright is the most important of these because all of the material in a digital archive is likely to be covered by copyright law. Some items may also be subject to additional permissions and constraints on use which have been granted by licence. In some cases these will be licences agreed between the repository and the rights holder, and in others they might be licences, such as Creative Commons licences, which have already been attached to items within an archive by its creator; legally binding contracts or licences can override copyright law.

Copyright restrictions have an important bearing on two of the core functions of a digital archive:

- **Preservation:** preserving digital objects requires the making of multiple copies and sometimes modifying the original object in some way, e.g. saving multiple copies of the bitstream, refreshing the bitstream to new media, migrating a digital object to different formats or transforming it in some other way. Currently, all of these preservation activities can potentially infringe copyright, although this situation may change

soon: The *Gowers Review* (published in December 2006, see p. 263) recommends that UK law should be amended to allow libraries to copy the master copy of any work for 'archival purposes', and to make further copies to mitigate against future wear and tear; and to format shift archival copies to ensure that records do not become obsolete.

- **Access:** in the digital environment, even viewing an archival record on screen or across a network creates a copy of that record which may infringe copyright law; copying, downloading or printing works may also be infringing activities. See Chapter 09 *Legal issues* for more information about copyright in relation to researcher access (see p. 260).

Paradigm therefore recommends obtaining explicit permission (see p. 258) from the principal rights holder in each archive (usually the donor or depositor) to make multiple copies for preservation purposes and to establish acceptable access terms for researchers; this should form part of the negotiation at accession and should be documented in the resulting donation or deposit agreement. Any permissions and constraints granted in a donation or deposit agreement should be fully documented in the rights metadata which accompanies a digital archive (and the component objects which make up that archive) throughout its lifecycle in the repository; the agreement or licence itself is a separate legally binding document (usually in hard copy format) which is described by, but not part of, the rights metadata.

However, obtaining a licence from the principal rights holder does not take into account the large number of other rights holders (often hundreds or even thousands) who are likely to be represented in any typical digital or hybrid personal archive. While archivists may be able to obtain similar permissions from other major rights holders in a collection (e.g. the most significant correspondents of the archive's creator), they will usually have to rely on copyright law and its 'fair dealing' (see p. 259) provisions when managing the rest of the material in a digital archive. The rights metadata for a digital object or collection should therefore also include information about the material's copyright status.

Recording rights information like this about digital objects is of crucial importance, both to:

- **The repository:** it enables the repository to demonstrate that it is acting within the boundaries of the law when carrying out preservation activities and permitting access to collections.
- **The researcher:** it enables the users of collections to understand the rights status of any archival material they consult, how they are permitted to use the material, and to identify and locate the rights holder when they wish to use material in a way which is restricted by legislation (e.g. to seek permission for publication).

Karen Coyle (a metadata expert and former employee at the California Digital Library), has proposed that detailed copyright metadata for each digital object or collection of digital objects be created. Based on registration forms used by the U.S. Copyright Office, she suggests that the following main areas be addressed in this metadata:

- General rights information (overview).
- Copyright status.
- Publication status.
- Dates (year of copyright creation; year of copyright renewal).
- Copyright statement.
- Country of publication or creation.
- Creator (name, date, contact).
- Copyright holder (name and contact).
- Publisher (name and contact, year of publication).
- Administrative data including contact information.

Unlike published books, which carry details of publication date and other copyright information, archival material does not come with such codified information. In order to produce a full copyright record, some research may need to be undertaken in addition to extracting relevant information from the digital object itself. Digital archivists are unlikely to have the resources to gather such information about the thousands of digital objects in their collections: unlike bibliographic data, which can be shared, archives are unique. Metadata extraction tools could lessen the burden if adapted to output some of the required data; even so, the file metadata that might be extracted - such as creator names and dates - is not always accurate and will probably require review.

Standards for digital rights management

Linking IPR information to groups and single digital objects is a core activity of a digital repository. While all digital objects will be subject to copyright law, some materials will be covered by different conditions under contract or licence. Rights metadata must therefore express both of these.

There are various metadata standards which include fields for statements of digital rights. These standards have been developed to address the issue of digital rights management in different contexts and to serve different purposes. Some have grown out of the e-commerce community (e.g. to protect rights in music downloads which are purchased under licence), while others have been developed within the digital library community to describe the rights connected with particular information resources. Most of them are specifically intended for a digital context (e.g. Creative Commons and MPEG-21/5); others are flexible (e.g. METSRights and ODRL do not exclude non-digital resources); and others need to be adapted to the digital environment (e.g. EAD, which was designed to catalogue archives in any format but has a strong emphasis on traditional paper-based material).

Some proprietary rights expression languages exist too, e.g. Adobe Content Manager, which allows the creator of a PDF document to add digital rights mechanisms to restrict how document is used by others; Microsoft Windows rights management solutions are expressed in eXtensible rights Markup Language (XrML), which is owned by ContentGuard. See Paradigm's *Guidance for creators* (*Appendix B: Guidelines for creators of personal archives*, p. 286) for an outline of the problems associated with such tools for digital curators. Rather than simply expressing the terms of contracts or licences, these rights languages act as components of a digital rights management system which includes machine-enforceable implementations of certain rights and constraints on the user.

Below is an overview and brief description of the main open standards for digital rights management; these allow the expression of rights statements associated with a particular digital object or resource, rather than forming the basis of machine-enforceable technical or copy protection measures. Most are forms of descriptive metadata, which are aimed at imparting rights information to the users of a digital resource, although maintaining access-related rights metadata is also crucial for the digital curators who manage the material. PREMIS, however, focuses specifically on the rights information digital archivists need to know in order to carry out preservation activities. The most useful standards for archivists are those which allow the expression of general copyright conditions, rather than being limited to describing detailed contractual conditions such as specific permitted acts; 'fair dealing', for instance, is open to interpretation and is not easily codified into permitted or restricted acts.

Dublin Core

Simple Dublin Core has 15 elements which may be used to describe a resource. One of these is specifically for the description of IPR rights attached to one or more digital objects: <dc:rights>.

For more information on Dublin Core see <http://dublincore.org/>.

<dc:rights> This field can be use to record information about the date of creation/publication, the owner of the rights, as well as information about the access conditions. Alternatively, the field may contain a URL which points to this.

Example:
<dc:rights>Access limited to members**</dc:rights>**
<dc:rights>http://www.bodley.ox.ac.uk/dept/pubs/copyright/**</dc:rights>**

Qualified Dublin Core

Qualified Dublin Core extends the 15 core descriptive elements, providing a more granular metadata structure. See 'Section 3: Other Elements and Element Refinements' of the DCMI Metadata Terms[1] for a full list of these elements.

Elements relevant to rights are:

<dcterms:accessRights> Information about who can access the resource or an indication of its security status.

<dcterms:dateCopyrighted> Date of a statement of copyright.

<dcterms:license> References a legal document giving official permission to do something with the resource, preferably via a URI. However, this might also be a hard-copy deposit or donation agreement.

<dcterms:rightsHolder> A person or organisation owning or managing rights over the resource.

Metadata Object Description Schema (MODS)

MODS is an XML schema which was developed as a bibliographic element set based on MARC fields, but may be used for a variety of purposes, including descriptive metadata for archive material; it is richer than simple Dublin Core. See <http://www.loc.gov/standards/mods/mods-outline.html> for a full outline of elements and attributes used in MODS, and <http://www.loc.gov/standards/mods/v3/mods-userguide.html> for more detailed user guidelines. The MODS elements which relate to rights are:

<accessCondition> Information about restrictions imposed on access to a resource. It can simply contain a free text description of any rights associated with a digital object or collection; alternatively, it can be made more specific using the TYPE attribute, although there is no controlled list of values defined for this attribute. The two examples provided in the MODS user guidelines divide the use of the element into access and use: "restrictionOnAccess" and "useAndReproduction".

<originInfo><copyrightDate> Used to record the date of copyright.

Encoded Archival Description (EAD)

Chapter 06 *Arranging and cataloguing digital and hybrid archives* provides an introduction to the arrangement and cataloguing of digital archives using EAD, which is a widely adopted standard for encoding archival finding aids modelled upon the *International Standard Archival Description (General)*. The official EAD website is at <http://www.loc.gov/ead/> and the full tag library is available at <http://www.loc.gov/ead/tglib/index.html>.

EAD includes two elements relevant to IPR:
<accessrestrict> Conditions Governing Access[2] The name of this element is taken from the equivalent ISAD(G) field. According to the EAD tag library it is intended to provide 'information about conditions that affect the availability of the materials being described. May indicate the need for an appointment or the nature of restrictions imposed by the donor, legal statute, repository, or other agency. May also indicate the lack of restrictions. Do not confuse with Conditions Governing Use <userestrict>, which designates information about limitations on the use of the described materials after access has been granted'.

1 Dublin Core Metadata Initiative, 'Section 3, Other Elements and Element Refinements', *Dublin Core Metadata Initiative website.* URL: <http://dublincore.org/documents/dcmi-terms/#H3>
2 Network Development and MARC Standards Office, 'Encoded Archival Description Tag Library, Version 2002 : <accessrestrict> Conditions Governing Access ', *Encoded Archival Description Version 2002 website.* URL: <http://www.loc.gov/ead/tglib/elements/accessrestrict.html>

This can be used at collection or lower levels to record information on material which is closed or restricted for copyright reasons (e.g. in the case of personal digital archives, much copyright material will only be made available to readers under strictly controlled conditions, see p. 259).

<userestrict> Conditions Governing Use[1] This element equates to ISAD(G) 'Conditions governing reproduction'. It is used to record 'information about conditions that affect use of the described materials after access has been granted. May indicate limitations, regulations, or special procedures imposed by a repository, donor, legal statute, or other agency regarding reproduction, publication, or quotation of the described materials. May also indicate the absence of restrictions, such as when copyright or literary rights have been dedicated to the public. Do not confuse with Conditions Governing Access <accessrestrict>, which designates information about conditions affecting the availability of the described materials. Preferred Citation <prefercite> may be used in conjunction with <userestrict> to encode statements specifying how the described materials should be referenced when reproduced, published, or quoted by patrons'.

METSRights

METSRights is an extension schema to the popular METS packaging metadata standard. The XML Schema and examples for METSRights can be found at <http://www.loc.gov/standards/rights/METSRights.xsd>. There is no separate documentation for this standard, though the schema itself does provide this.

METSRights has the advantage of being specifically intended to express the rights associated with digital objects and has been designed for use with digital library materials. It is divided into three principal sections, although the highest, root, level also has attributes which enables the specification of the kind of rights being described, e.g. copyrighted, licenced, public domain, contractual, or other. The three main sections are:

- **<RightsDeclaration>** a broad declaration of the rights associated with a digital asset or part of a digital asset intended to inform the user community of these rights.
- **<RightsHolder>** details of any person or organisation holding some rights to a given digital asset or part of a digital asset. Contains subelements to specify rightsholder details: the name of the rights holder, the name of the person or organisation acting as a contact for the rightsholder, along with contact addresses, telephone numbers and email addresses.
- **<Context>** describes the specific circumstances associated with who has what permissions and constraints. Given contexts can be related to specific types of individual or institution who may have a stake in a digital object, e.g. academic user, general public, repository manager, managed group, institutional affiliate and others. Specific permissions can be stated, e.g. discover, display, copy, duplicate, modify, delete, print, other; and Boolean values can be used to indicate particular constraints or restrictions placed on users within a given context, e.g. quality, format, payment, re-use, other.

See above (p. 129) for an example of METSRights in use.

Creative Commons

Creative Commons[2] is a form of licensing which enables copyright holders to grant some of their rights to the public while retaining others through a variety of licences. The licences were developed in recognition of the fact that many rights holders do not wish to restrict the use of their materials as rigidly as the default copyright protections and may in fact wish to encourage re-use of their creations. Creative Commons allows creators to generate licences for their materials very simply, by completing an online form. The licences have three parts:

- A machine readable part - this allows search engines to search specifically for materials that can be re-used without tracing and seeking the permission of rights holders.

1 Network Development and MARC Standards Office, 'Encoded Archival Description Tag Library, Version 2002: <userestrict> Conditions Governing Use', *Encoded Archival Description Version 2002 website.* URL: <http://www.loc.gov/ead/tglib/elements/userestrict.html>

2 Creative Commons, *Creative Commons website.* URL: <http://www.creative-commons.org/>

- A human readable part - this part allows creators to understand the licence they are granting and potential re-users to understand the licence that is granted them.
- A part for legal professionals - this part is the legalese that ensures that licences are legally sound.

The Creative Commons Licences include three major characteristics:

- Permissions - rights granted by the licence.
- Prohibitions - things prohibited by the licence.
- Requirements - restrictions imposed by the licence.

Each work covered by a Creative Commons licence has a Creative Commons graphic displayed or embedded in it to alert users to the presence of a licence.

Creative Commons licences were designed to facilitate the re-use of web-based works, though they can be used in other contexts. Whilst most of the material in a personal digital archive is likely to be private in nature (e.g. day-to-day correspondence and working papers), some record creators may have attached Creative Commons licences to their works – usually electronic publications and other web-based material intended for public consumption. Digital curators need to be aware of any such works included in the archives they administer, because these licences take precedence over copyright protection – usually allowing digital material to be used more freely. In addition to setting out the specific rights attached to a digital object, a Creative Commons licence has optional fields in which to record both the creator's and copyright holder's names, enabling archivists to identify rights and rights holders for inclusion in the metadata record.

Open Digital Rights Language

The Open Digital Rights Language (ODRL) Initiative[1] is an international effort aimed at developing and promoting an open standard for the Digital Rights Management expression language.

ODRL is designed to express all the elements of a digital rights licence to which a resource is subject, including identification of parties and all possible permissions covered by a licence. It has no mandatory elements, it allows the expression of very complex statements and is extensible, so it can be used in a variety of contexts. It also provides the semantics to express policies which might be enforced by a machine-actionable DRM system.

The ODRL Model (v. 1)

The ODRL model[2] defines three core entities: assets, rights and parties.

Assets are any uniquely identifiable content. They may also contain components.

Rights include:

- Permissions - usages or activities allowed over the Assets (e.g. print an image Asset) – which may have:
    - Constraints – these limit the use(s) allowed in some way (e.g. a particular category of user can print an image Asset up to a maximum of 1 times).
    - Requirements – these are 'obligations needed to exercise the Permission' (e.g. pay £1.50 to print the image).
    - Conditions – these specify scenarios where renegotiation of Permissions may be needed (e.g. if the user's credit card expires, Permission to print the image ceases).

Parties include end users and rights holders and may be individuals, organisations, or defined roles.

1 ODRL Initiative, *ODRL Initiative website.* URL: <http://odrl.net/>
2 W3C, *Open Digital Rights Language (ODRL) Version 1.1* (September 2002). URL: <http://www.w3.org/TR/odrl/>

Using these three entities ODRL can express Offers and Agreements. Offers are proposals by rights holders for specific rights over their Assets. Agreements are where Parties enter into contracts with specific Offers.

Karen Coyle suggests that it might be possible to develop preservation-related permission statements using rights languages like ODRL, but to date the standards have focused on current e-commerce applications and do not have a specific language for preservation actions. ODRL also focuses on rights granted by licence rather than copyright law which is more important to digital archivists.

MPEG-21 Part 5 (MPEG-21/5)

MPEG-21[1] was developed by the Moving Picture Experts Group and consists of a number of standards relating to digital multimedia resources. It is in seven parts, of which part 5 is the rights expression language and part 6 is a data dictionary containing terms for use in the rights expression language (which is not yet complete). MPEG-21/5 was developed for the commercial sector and derived from ContentGuard's XrML language; it can be machine-actionable (i.e. interact with software or hardware to enforce licensed permissions and restrictions). While it was primarily intended for the licensing of digital audio and video resources, it is also a general-purpose rights language which can be applied to other digital objects; MPEG-21 is organised into separate sections so that particular sections (like MPEG-21/5) can be removed and used in other contexts. MPEG-21/5 is also the only rights expression language to be issued by a formal standards body, as ISO/IEC 21000-5.

It is based on the idea of a digital Resource, one or more Principals with a stake in that Resource (these can be either machine or human, e.g. a particular type of user), a set of Rights which are associated with the resource and Conditions to which these rights are subject. A Right is described as a verb; the Resource is the object of the Right and a Condition describes rules under which Rights can be exercised.

Like ODRL, MPEG/21/5 includes an extensive list of possible rights because it is intended to cover a wide range of situations; it is a generalised standard and can be used as the basis for developing more specific languages designed to deal with particular functions, e.g. the Open eBook Forum have extended MPEG-21/5 to support the management of rights for e-books. To date it has not been explored in depth as a possible archival rights solution and, like ODRL, it has the disadvantage of being unable to record metadata about copyright, the most important information in a digital archive context.

Preservation Metadata Implementation Strategies (PREMIS)

The *PREMIS Data Dictionary* provides a list of elements along with information about how to apply these in order to support the long-term preservation of digital objects. The data dictionary in its entirety is available online,[2] and an introduction to PREMIS in the context of personal digital archives is provided earlier in this chapter.

PREMIS includes a rights entity (see p. 95) which is designed to document rights relating to preservation actions rather than to access and use by researchers. It is relatively specific: it allows the expression of a structured permission statement relating to a digital object or collection, which sets out the rights of the repository to undertake preservation activities on that material, granted in a legally-binding agreement by the principal rights holder. It includes elements to record: details of the permission statement; the digital object(s) and agent(s) concerned in the agreement; information about the granting agreement; details of the permission granted, including permitted acts and restrictions; with a start and end term for the grant. PREMIS suggests employing a controlled vocabulary to record different types of act; however, it may be preferable to make a more general statement of permission which encompasses any kind of preservation action which might become necessary over time as technologies change.

Whilst the initial PREMIS Working Group chose only to establish the minimal metadata needed by a digital repository to document preservation-related rights, the Library of Congress subsequently

1 Organisation Internationale De Normalisation, *MPEG-21 Overview v.5* (October 2002). URL: <http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm>

2 PREMIS Working Group, *Data Dictionary for Preservation Metadata* (May 2005). URL: <http://www.oclc.org/research/projects/pmwg/premis-final.pdf>

commissioned Karen Coyle to provide a study of *Rights in the PREMIS Data Model*[1] to inform the PREMIS Editorial Committee in producing its first revision of the data dictionary. In her report, published in December 2006, Coyle makes clear the distinction between rights that have a statutory basis and those that have a contractual basis. She points out that the PREMIS metadata standard relies almost entirely on a view of preservation rights as explicit permissions, whereas many repositories have to rely on copyright law and policy as the foundation for their preservation actions. This is certainly the case with collecting institutions which take in personal archives, where the only contract is usually with the donor of the archive, and many other third party rights holders are represented in the collections.

Coyle recommends the addition of some new data elements to the PREMIS Rights entity to record preservation actions that have been interpreted to be permitted by law or statute; she also suggests including elements to record information about the copyright status of an object, where known. The additional metadata should therefore express: the law or regulation governing the preservation mandate or right, when preservation is done under those auspices; ownership of the IPRs in the resource at the time of entry into the repository; the public domain status of the work; and when preservation is based on a specific permission, details of the permission and its grantor.

Her recommendations include the following:
An <agentRole> element should be added to the PREMIS agent entity (see p. 95). The agent entity can be linked to the rights entity to identify the granting agent involved in any permissions agreement. The <agentRole> element would help to clarify the relationship of the granting agent to the IPRs in the digital object, e.g. they might be the actual rights holder, or a legal representative of the rights holder.

A new element <copyrightInformation> with subelement <copyrightStatus> should be added to the rights entity to allow digital curators to record whether or not a digital object is in copyright, and when the copyright status of an object is unknown. Coyle also suggests providing a more granular record of copyright status by including either:

- The CopyrightMD XML schema[2] developed by the California Digital Library (currently in a test phase) includes elements for: creator name and dates; rights holder name and contact; a copyright status attribute; dates of creation, publication and copyright registration.
- Copyright elements similar to those produced by the Cedars Project[3] which include fields for: a copyright statement, name of publisher, date of publication, place of publication, rights warning (i.e. a warning that the digital object may be subject to copyright or database right, presumably intended for users), and rights holder or contact.

A new element <permittedByLicense> should follow the copyright element. This would simply contain as subelements the two existing PREMIS elements <grantingAgent> (with the addition of <grantingAgentRole>) and <grantingAgreement>.

This should be followed by a new element <permittedByStatute> to record whether a repository is undertaking preservation actions based on legislation rather than a specific licence or permissions. Instead of an agent, this scenario involves a specific <jurisdiction> (the first subelement) e.g. the UK. The second subelement is <statute> containing two further subelements <statuteIdentification> and <statuteCitation>, to record the law invoked and the specific part of that law on which any preservation activities are being based. The final element <determinationDate> is intended to record the date on which the decision was made to proceed with preservation activities on this basis, e.g. when the copyright status of a work changes as a result of legislative change, or if a work comes into the public domain.

Coyle also notes that currently the final rights entity element (<permissionGranted>) includes the mandatory subelement <act>. The *PREMIS Data Dictionary* suggests using a controlled vocabulary

1 Karen Coyle, *Rights in the PREMIS Data Model: A Report for the Library of Congress* (December 2006). URL: <http://www.loc.gov/standards/premis/Rights-in-the-PREMIS-Data-Model.pdf>
2 California Digital Library, 'copyrightMD Schema', *California Digital Library website.* URL: <http://www.cdlib.org/inside/projects/rights/schema>
3 Cedars Project, *Metadata For Digital Preservation The Cedars Project Outline Specification Draft For Public Consultation*. URL: <http://www.leeds.ac.uk/cedars/documents/Metadata/cedars.html>

for the content of this element (examples provided include replicate, migrate and modify). Coyle recommends that this should be more flexible; laws and agreements can be vague and subject to interpretation, e.g. the 'fair dealing' provisions, or a broad agreement with a depositor granting the repository permission to do whatever is necessary to preserve a digital object in an accessible way. She recommends that <act> should either be able to codify generalised permissions like this, or should be defined as optional rather than mandatory.

Rights metadata and personal digital archives

In the context of personal digital archives, it is important to include a rights clause in all donation and deposit agreements, in which explicit permission is granted by the donor or depositor for the repository to undertake preservation and access activities (see p. 258). This means that at least the material under the copyright of the donor or depositor is covered by a formal permissions agreement, and the conditions of this agreement should be recorded in the rights metadata which accompanies the collection and its component parts. Copyright status metadata should be created for the rest of the material in the archive.

It is important to extract as much IPR information about a digital archive as possible at an early stage in the relationship between donor and repository. Curators should take the opportunity to discuss rights with the donor or depositor and perhaps to contact, or at least identify, other significant rights holders in the material. More detailed and accurate rights information can be obtained at this stage than if the creation of rights metadata is left until later in the record's lifecycle. All of the rights information obtained should be recorded as part of the AIP.

Below are some suggested approaches to recording rights metadata about the kinds of digital objects found in a personal archive.

Rights metadata at AIP stage

Rights information which has a bearing on preservation activities should be recorded as comprehensively as possible in the AIP metadata record. Full rights information regarding access and use should also be recorded as part of the administrative metadata; elements of this can be extracted for researcher use as part of the DIP at a later stage. Rights information need not be included as part of the descriptive metadata of an object's AIP since these are held in a stand-alone preservation environment and are not accessible to researchers.

The METS document for each digital object should include a <rightsMD> element within the Administrative Metadata Section (<amdSec>). This will be used to record IPR information relating to access and use of the digital object by researchers. The METSRights schema can record this information and should indicate whether access and use of the digital object is to be provided under statutory provisions ('copyrighted') or under contractual conditions ('licensed', 'contractual'); or whether the object is in the 'public domain'. Most digital objects will not be in the public domain for many years, and most of the material in a digital archive will not be subject to a licence, so digital archivists are likely to be basing access conditions on copyright law.

METSRights should also be used to record the full name and contact details of the rights holder or their legal representative, where this information can be ascertained. The <RightsHolderComments> element might be used to indicate where rights holder information was unavailable or could not be traced; in many cases digital archivists will not have the time and resources to trace every single rights holder represented in an archive when that information cannot be extracted from the archive itself. Although it includes general elements like <RightsDeclaration> (to contain a broad declaration of rights associated with a digital asset), the METSRights schema does not include specific elements in which to record the kind of details about copyright status recommended by Coyle (see p. 145), e.g. the name and dates of the author as well as the rights holder if there is a distinction, and the date of creation of the object. The copyrightMD schema could be used to record such information. Archivists should also monitor the work which is currently underway to see whether METSRights elements can be represented as an application profile of ODRL; this might

enable users to enhance the METSRights record with more granular copyright information.

The <context> element of METSRights uses attributes to express the types of user to which particular permissions or constraints relate. The OTHER attribute provides a means to define local categories, e.g. Registered Readers; Remote users who have filled in an online copyright declaration form, etc. Specific types of permitted act can be defined, e.g. discover, display and copy. See Chapter 09 *Legal issues* for the access conditions that are likely to apply to personal digital archives (see p. 259) which are still in copyright.

In cases where the creator of a digital object has attached a Creative Commons licence to their work, this should also be indicated in METSRights. In some cases creators might use a Creative Commons licence to dedicate their work to the public domain, in which case there will be no restrictions on its re-use. In other cases a creator will retain some rights over the material and any specific permissions and constraints should be recorded using the <Context> element of METSRights; these can be recorded using Boolean values, and a fuller explanation of these might be recorded in the <ConstraintDescription> element.

In 2005 a specification for an ODRL Creative Commons Profile was produced. This describes how the semantics of Creative Commons licences can be represented using a Profile of the ODRL Rights Expression Language; ODRL is XML-based, so this information might be extracted and included in an AIP METS file. This could be done by adding an additional <rightsMD> section, using the <mdWrap> element, and the OTHER attribute to denote that XML metadata which does not come from one of the METS extension schemas is being used.

The digital object METS document will be linked by an identifier to a related METS document containing the information which forms the PREMIS Rights entity for that object; this records rights associated with preservation activities rather than researcher access. Storing this information separately means that all the digital objects in an archive which are under the copyright of the same person and subject to the same conditions can refer to a single rights record; an example of a complete rights entity appears earlier in this chapter (see p. 131). While the rights entity is not a mandatory element of PREMIS, Paradigm recommends that it is used.

Currently the PREMIS rights entity only allows the recording of a limited number of specified actions (based on a controlled vocabulary) that are covered by explicit conditions. These would be agreed as part of a donation or deposit agreement, and so are only likely to apply to material in the copyright of the archive's principal creator. Currently UK copyright law is ambiguous on the subject of copying digital material for preservation purposes, and it is not clear whether digital archivists can rely on legislation as a mandate for preservation copying (see p. 258). However, the *Gowers Review* (see p. 263) may alter this situation, and in future archivists may be able to claim the right to copy for preservation purposes under copyright law. Should this come to pass, it would be very helpful if Karen Coyle's suggested amendments (see p. 147) to the PREMIS Rights entity are adopted and added to the Rights schema. The inclusion of further metadata elements from the copyrightMD schema or Cedars (as recommended by Coyle) would provide an even more granular record.

In the same way that digital archivists need to carry out a 'technology watch' while objects are in archival storage, it is also important to maintain a 'rights watch' to monitor any changes in legislation, as well as changes in the rights status of the material in the preservation repository's care (e.g. through the expiration of copyright).

Rights metadata in the DIP

The user of an archive will not need access to preservation-related rights information. However, it is important that they are given comprehensive access-related rights information, especially about the copyright status of an item. This should be recorded as part of the DIP for each digital object in an archive.

The DIP should include information drawn from the METSRights record relating to permissions and constraints on access and use; the contact details of the rights holder or their representative

should be omitted from this public record because including these without first obtaining permission would contravene the *Data Protection Act* (see p. 250). A statement should be added in the <ConstraintDescription> field to indicate that users should contact the digital repository if they wish to seek permission from the rights holder for re-use of any copyrighted digital material.

Rights information could also be included in the Dublin Core record for a digital object for OAI-PMH harvesters. An ODRL/DCMI Profile Working Group has been set up, which plans to develop a way of making combined use of the rights-related DCMI metadata terms and the ODRL rights expression language. This will enable richer rights management information to be captured along with DCMI descriptive metadata.

Rights metadata in the EAD catalogue

The EAD catalogue provides the final, comprehensive layer of descriptive metadata, which enables the researcher to determine whether or not the content of an archive is relevant to their research. On identifying a relevant digital object, the researcher will be able to move from the catalogue entry to the DIP via a link. The DIP, rather than the EAD catalogue, should contain the principal rights metadata record for the researcher because:

- By the time a researcher is at the stage of calling up a DIP, they are likely to be interested in using the digital object in some way and need to be made aware of permitted uses and any constraints.
- In most cases, digital archivists are unlikely to have the time or resources to catalogue to item level in EAD, so detailed object-specific rights metadata will not form part of the EAD catalogue.

The primary rights-related information in the EAD catalogue should occur at collection level; in a hybrid archive this will draw together relevant rights information about both the paper-based and digital components of an archive. An outline of suggested content for the <accessrestrict> and <userestrict> elements at collection level is given in Chapter 06 *Arranging and cataloguing digital and hybrid archives* (see p. 185). At lower levels, a brief <accessrestrict> statement can be used to indicate if component items are closed or subject to access restrictions.

Rights metadata and controlling access

Most of the material in a personal digital archive will remain in copyright for many years and will only be made accessible in a managed searchroom environment. Sometimes, however, it may be possible to obtain a licence from a rights holder authorising the repository to make material available to researchers remotely, with certain conditions attached, e.g. access might be limited to authorised remote users or to a restricted number of computers; or there may be limits on the number of each item which can be printed. In this case, digital archivists may choose to employ rights expression languages which could be embedded in software to control usage of material. Coyle, for instance, suggests that the Federated Digital Rights Management (FDRM)[1] architecture could in future be developed into a system for enforcing access rights based on licence statements like those recorded in METSRights.

However fully a repository documents the IPRs associated with the digital objects in its collections, and however effective the repository's rights management policy, Paradigm also recommends publishing an institutional 'take down' policy, whereby material can be removed from public access at the request of a rights holder. This is crucial in the context of personal archives, where large numbers of third party rights holders are represented in collections and explicit permission cannot be sought from each to copy or make available the documents to which they hold the rights. Having a comprehensive rights management policy does at least demonstrate that a repository is acting responsibly, obtaining permission where possible and complying with the relevant legislation where this is not possible.

1 Karen Coyle, 'The "Rights" in Digital Rights Management', *D-Lib Magazine*, 10, 9 (September 2004). URL: <http://www.dlib.org/dlib/september04/coyle/09coyle.html>

Metadata for authenticity: hash functions and digital signatures

Introduction

Authenticity and integrity are important characteristics of archives and it is natural that archivists are concerned about ensuring the ongoing authenticity of digital archives, which are so easily and near undetectably altered. This ongoing authenticity is also important to the creators of material, as well as to the researchers who will rely on it to inform their research. The OAIS model (see p. 3) requires that 'fixity information' be held for digital objects so that their moral and physical integrity can be verified over time. Hash functions and digital signatures are two means of creating and validating such fixity information and both have a number of potential applications in the acquisition, management and dissemination of personal digital archives. Paradigm has therefore explored the nature and uses of hash functions and digital signatures; an overview of these technologies and consideration of how they might be useful to archivists is provided here.

Archival uses for fixity information

Fixity information documents authentication mechanisms used to ensure that the materials stored by a preservation repository have not been altered in an undocumented manner. Creating and verifying fixity information is therefore an integral part of the management process for authentic digital objects ensuring that the repository can be confident of the authenticity and integrity of its digital objects. Many repositories will wish to record some level of fixity information about the digital files and metadata (and versions thereof) that they manage. Not only is fixity information an important part of the OAIS information model, it is also included in a number of metadata schemas for digital objects including PREMIS (see p. 80), METS (see p. 117) and others.

When should fixity information be created and verified?

Fixity information can be created or verified at numerous stages in the preservation workflow. The management policy for the archive, its context and the level of confidence required will dictate when and what type of fixity information is created and how often it is verified. Appropriate points in the lifecycle for generating and verifying fixity information include:

- At point of creation – creators could themselves decide to create fixity information for some of their digital files and perhaps to use a digital signature to encrypt sensitive data.
- At point of accession – archivists or creators could create fixity information when accessioning digital archives which can later be used to verify that digital archives have not been subject to any unauthorised transformation during the process of transferring material to the Library and ingesting it into a digital repository.
- At point of ingest – archivists record fixity information as part of the preservation metadata of an archive, which can be used thereafter to verify the continuing authenticity of the material.
- At point of transformation - changes to metadata and the creation of new versions of digital objects through file format migrations can be a necessary part of the digital preservation process, but fixity information is required to provide an audit trail of such actions so that new versions of objects and their metadata are trustworthy.
- As part of normal maintenance routines – fixity information may be used to ensure the continuing authenticity of material when undertaking regular maintenance activities, such as system back-up.
- At point of dissemination – it is possible that researchers may expect repositories to sign the digital archives they make available, so that their authenticity is assured.

Types of fixity information

The types of fixity information explored by Paradigm included hash functions and digital signatures. These are also the kinds of fixity information recommended by the *PREMIS Data Dictionary for Preservation Metadata*.[1]

Hash functions

Hash functions are a method of computing a unique, fixed-size, string of text from a digital object of any size which can act as a fingerprint for the data. By calculating and storing the hash value of data, and later subjecting the same data to the same hash function and comparing the resultant hash values, it is possible to identify whether the data has been altered in some way. Even small alterations to data produce dramatically different hash values (also known as message digests), as shown in the example below:

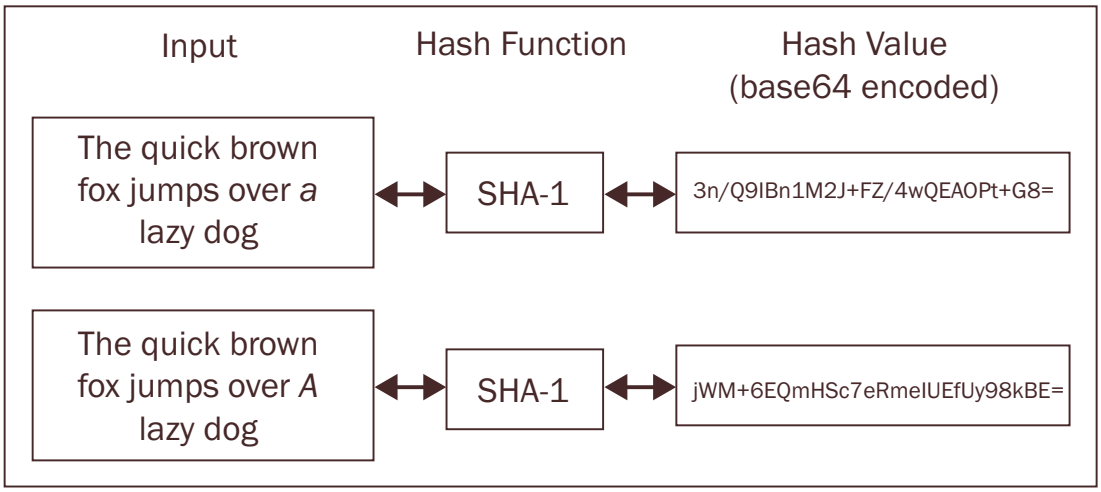

Figure 17: Hash Function

There are a number of different hash functions available. These break down into three groups: checksums, cryptographic hash functions and cyclic redundancy checks.

Checksums Checksums are relatively simple hash functions calculated from the value of the bytes in the data being checksummed. Example algorithms include:

- sum8 (8 bits)
- sum16 (16 bits)
- sum24 (24 bits)
- sum32 (32 bits)

Cryptographic hash functions Cryptographic hash functions are particularly suitable for archival purposes. Cryptographic hash functions provide additional security, though not all algorithms are equally secure. Simpler algorithms (e.g. MD5) have been compromised and are therefore suitable for monitoring stored data for accidental damage, but not for securing data against malicious alterations. The more complex algorithms enable detection of more kinds of errors. Some digital repositories have chosen to record several hash values for each digital object using multiple algorithms. Examples of cryptographic algorithms include:

- HAVAL (125 to 256 bits).
- MD5 (128 bits).
- SHA-1 (160 bits).
- SHA-256 (256 bits).

1 PREMIS Working Group, *Data Dictionary for Preservation Metadata* (May 2005). URL: <http://www.oclc.org/research/projects/pmwg/premis-final.pdf>

- Tiger (192 bits).
- Whirlpool (512 bits).

Cyclic Redundancy Checks (CRC) CRCs are generally used for checking the integrity of stored data or data in transmission. Examples of CRC algorithms include:

- CRC 16 (16 bits).
- CRC 32 (32 bits).

The different hash functions require varying amounts of processing power to calculate their hash values. The hash functions with more complex algorithms, or that produce longer hash values, require more processing power but provide more reliable fixity information.

Digital signatures

Digital signatures are more complex than hash functions. A digital signature is a string of bits that is computed from the data being signed (or its hash value) and the key of the entity (such as a person or organisation) performing the signing. The combination of these inputs to the digital signature permit the recipients of the signed data to verify both the authenticity of the data source as well as the integrity of the data received.

Public key cryptography, the basis for digital signatures, utilises asymmetric encryption which uses a pair of keys – a 'private key' and 'public key'. The private key is securely kept by its owner, and the public key (its partner) can be distributed to those whom the holder of the private key wishes to exchange encrypted or signed data. Public key cryptography therefore enables:

- Data confidentiality – as long as the private and public keys are held securely by their respective owners, public key cryptography enables the confidential exchange of data over potentially insecure networks.
- If a private key is used to encrypt data (some digital signature algorithms encrypt the signature rather than the data being signed) then this data can only be read by the holders of the corresponding public key.
- If a public key is used to encrypt data then this data can only be read by the holder of the corresponding private key.
- Data integrity – the authenticity of the data signed with a private key can be verified by recipients in possession of its corresponding public key. Any alterations, whether deliberate or accidental, can easily be spotted by the validation process because a digital signature will not verify as authentic if either the signed data or the signature is altered.
- Originator authentication - the identity of the originator of data signed with a private key can be verified by recipients in possession of the corresponding public key because the private key is held only by the originator and it cannot be forged so long as the originator keeps it secret.
- Non-repudiation - digital signatures created with private keys support non-repudiation because they are partly computed from the data being signed and because the public key can only decrypt information encoded using the corresponding private key, which is tied to a specific entity.

The use of asymmetric keys provides a mechanism analogous to, and potentially more robust than, the traditional signature or seal more familiar to archivists. It produces, in effect, a digital signature.

How digital signatures work

The private key of the originator is used as input to the algorithm which transforms the data being signed (or its hash value). This transformation can only be reversed, and the data decrypted and accessed, by use of the originator's public key, which is provided to the recipient(s) by the originator.

Creating a digital signature with a private key

Data encryption using asymmetric keys is an expensive operation directly proportional to the size of the data being encrypted; it potentially doubles the size of the data increasing the processing power and bandwidth required to process and transfer the data. A more efficient approach is to first use a secure cryptographic hash function (such as SHA-1) which can take large objects of varying size and produce a unique fixed-size hash value or message digest. The much smaller hash value can then be encrypted with the private key of the originator to produce the digital signature.

Having calculated the message digest this can be encrypted using the private key of the originator to produce the digital signature, as shown in the diagram below:

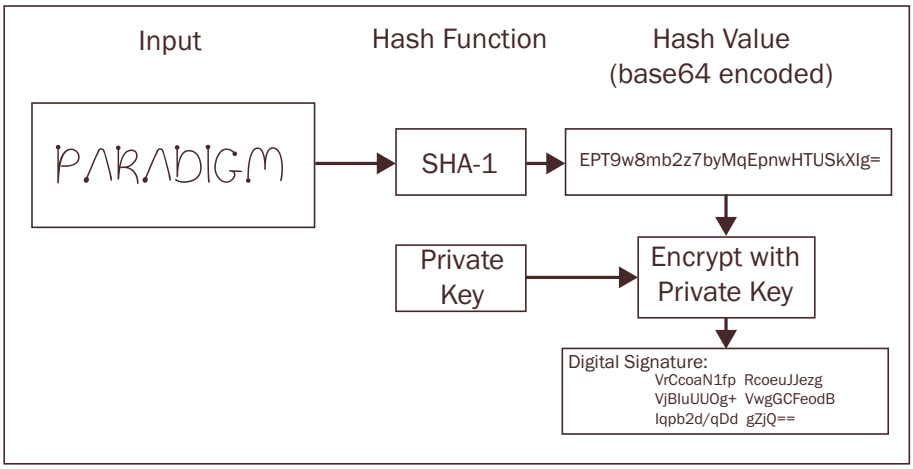

Figure 18: Creating a digital signature

Verifying a digital signature created with a private key

The recipient must de-crypt the digital signature using the public key of the originator and recalculate the hash value of the corresponding digital object. If the calculated hash value does not match the result of the decrypted signature, either the object has been altered since being signed, or the signature was not generated with the corresponding private key of the originator.

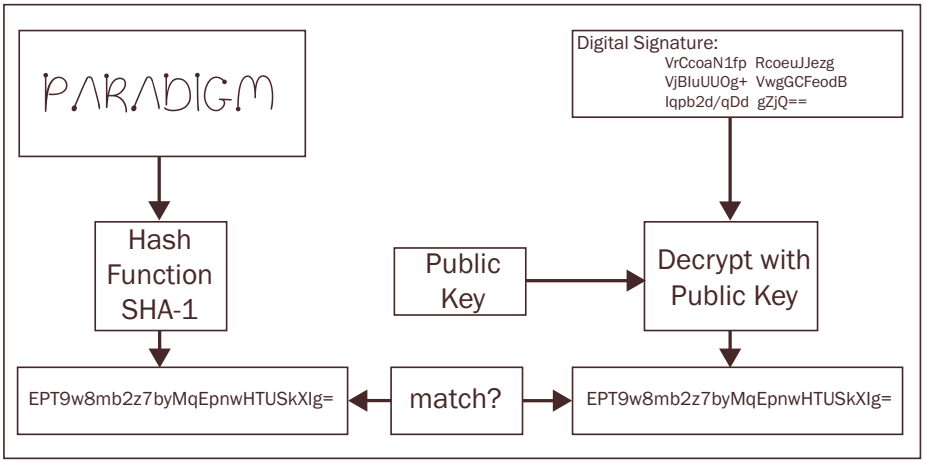

Figure 19: Verifying a digital signature

Digital signature algorithms

Just as there are a number of algorithms for creating hash values, there are also a number of digital signature algorithms. Two of the most commonly used are RSA and DSA.

RSA The RSA digital signature algorithm was developed by Ron Rivest, Adi Shamir and Leonard Adleman at Massachusetts Institute of Technology (MIT) in 1977. RSA can also be used to encrypt and decrypt the data being signed. RSA does not mandate the use of a particular hash function, so the security of the signature and encryption are partly dependent on the choice of hash function used to compute the signature.

DSA The DSA (Digital Signature Algorithm) is defined by the Digital Security Standard (DSS) and was developed by the National Institute of Standards and Technology (NIST) in 1991. The algorithm requires a SHA-1 digest to compute its digital signature. The DSA algorithm does not encrypt the data being signed, it purely produces a signature that allows the recipient to verify the authenticity and provenance of the data. DSA signatures can be created as quickly as RSA signatures, but their verification can take much longer.

The role of trust and certification in validating public keys

The receiver must have confidence that the public key does actually belong to the originator otherwise substitution by a false public key would enable a "man in the middle attack" to compromise the data. One mechanism for asserting the validity of the relationship between the originator and their public key relies on certificates. These are issued by trusted certification authorities who generate and digitally sign certificates binding entities (such as people and organisations) to their public keys. Unfortunately, mechanisms enabling us to trust the signature of the trusted authority on the certificate are also needed; this creates what is known as a 'chain of trust', which can become complex and expensive to manage.

An alternative, or complementary, approach is that of self-certification using a key and certificate management utility. One such utility is KeyTool (a Java based tool) which manages a database (keystore) which can contain cryptographic keys, X509 certificate chains and trusted certificates. Utilities of this type permit users and organisations to administer their own private/public key pairs and associated certificates for use in data integrity and authentication services using digital signatures and to self-authenticate to other users/services.

Digital repositories supplying signed objects to users may wish to establish confidence in their public key through the use of a certificate, but this will only be worthwhile if users understand its purpose.

Storing fixity information as XML metadata

If digital repositories mean to use fixity information then some provision for its storage and association with the digital objects the fixity information refers to will be needed. Here the *W3C Recommendation XML-Signature Syntax and Processing* (for digital signatures), the *PREMIS Data Dictionary* (for hash values and digital signatures) and the Fedora digital repository (for hash values) are examined as potential modes of recording fixity information.

W3C Recommendation XML-Signature Syntax and Processing

The *W3C recommendation XML-Signature Syntax and Processing*[1] provides a detailed marking up of digital signatures in XML, optionally locating signed objects (of any type) within XML documents or referencing externally held signed objects. It also allows users to sign parts of documents. The location of the signed content relative to the XML signature may be one, or a combination of, the following:

- Enveloping: the signature is a parent element of the data object within the same XML document (the signed content is found within an <Object> element of the signature).

1 W3C, *XML-Signature Syntax and Processing* (February 2002). URL: <http://www.w3.org/TR/xmldsig-core/>

- Enveloped: the signature is a child element of the data object within the same XML document (the signed content includes a signature element as a sub-element).
- Detached: the signed content is completely detached from the signature as either elements within the same XML document (as sibling elements or sub-elements of sibling elements) or the objects are external resources not contained within the same XML document.

As part of the process of creating the XML-encoded signature(s) it is necessary to produce a canonical form of the XML. Canonicalisation standardises aspects of the XML document which may not necessarily impair on the meaning of the document (such as line breaks or excessive whitespace) but would give rise to different hash values and thus different digital signatures. XML editors usually support XML canonicalisation of two kinds: Inclusive XML Canonicalization (XMLC14N[1]) and Exclusive XML Canonicalization (EXCC14N[2]).

Creating and verifying XML signatures using the W3C recommendation

XML signature creation
In this example two digital objects are to be signed with a single digital signature:

- The Paradigm logo (http://www.paradigm.ac.uk/images/paradigm.gif).
- The Paradigm home page (http://www.paradigm.ac.uk/index.html).

1. For each object a <Reference> element is created containing:
    - The location (URI) of the object.
    - An ordered list of the transforms (or processing steps) that were applied to the content of the referenced resource before its digest was calculated.
    - The actual algorithm used (such as SHA-1) to calculate the digest value.
    - The digest value (base64 encoded) for the identified object in the <DigestValue> element.
2. These <Reference> elements are collected within the <SignedInfo> element along with:
    - The canonicalisation method (e.g. XMLC14N as used in the example below) to be applied to the <SignedInfo> element.
    - The signature algorithm to be applied to the <SignedInfo> element.
3. The <SignedInfo> element does not include explicit signature or digest properties (such as date or calculation time), if these are required they can be associated via a <SignatureProperties> element attached to an <Object> element.
4. The populated <SignedInfo> element is then canonicalised using the specified <CanonicalizationMethod>.
5. Finally the <SignatureMethod> which is a combination of a digest algorithm and a key dependent algorithm (e.g. DSA-SHA1) is applied to the canonicalised <SignedInfo> element and the digest result is placed in the <SignatureValue> element.

XML signature verification
The verification of an XML signature consists of of two phases:

1. Signature validation
This comprises the verification of the <signatureValue> of the <SignedInfo> element:
    - The digest of the <signedInfo> element is recalculated using the digest algorithm specified in the <SignatureMethod> element.
    - The public key from <KeyInfo>, or from an external source, is used to verify that the <SignatureValue> matches the recalculated <SignedInfo> digest.
2. Reference validation
This comprises the verification of the <DigestValue> of each <Reference> element
    - The <SignedInfo> element is canonicalised using the algorithm specified in <CanonicalizationMethod>.

1 W3C, *Canonical XML: Version 1.0* (March 2001). URL: <http://www.w3.org/TR/2001/REC-xml-c14n-20010315>
2 W3C, *Exclusive XML Canonicalization: Version 1.0* (July 2002). URL: <http://www.w3.org/TR/2002/REC-xml-exc-c14n-20020718/>

- For each referenced object in the canonicalised <SignedInfo> the recipient must:
  - Obtain a copy of the object.
  - Apply any transforms specified to the object.
  - Regenerate the digest for the transformed object using the <DigestMethod> specified in its <Reference> element.

Validation fails if the generated digest value and the <DigestValue> in the <Reference> do not match.

Arguably the popularity of packaging metadata standards such as METS, which allow the referencing and embedding of metadata and digital files in a single XML file, make the aggregation features of the *W3C XML-Signature Syntax and Processing* redundant in a digital library context. The *PREMIS Data Dictionary* also specifies that digital signatures are applicable only to files and bitstreams for preservation purposes and the ability to sign an aggregation of files is therefore not required. Despite this, the W3C Recommendation remains the de-facto standard for encoding digital signatures and its definition of the processing rules around digital signatures and the semantic units needed to record them is useful.

PREMIS metadata for hash values and digital signatures

The *PREMIS Data Dictionary for Preservation Metadata*[1] provides semantic units for the recording of hash values collected under the <fixity> unit and recommends that preservation repositories store hash values calculated using at least two hash algorithms for each file. Using the PREMIS Object XML schema, hash value information for files can be recorded as follows:

```
<premis:object>
  <--!other metadata-->
    <premis:fixity>
      <premis:messageDigestAlgorithm>SHA-1</premis:messageDigestAlgorithm>
      <premis:messageDigest>9bb7f5d5f5f48afcb516c644981ef4055188abbe</premis:
      messageDigest>
      <premis:messageDigestOriginator>Paradigm preservation repository</premis:
      messageDigestOriginator>
    </premis:fixity>
  <--!other metadata-->
</premis:object>
```

PREMIS borrows some elements from the *W3C recommendation XML-Signature Syntax and Processing* in defining semantic units necessary to record metadata about digital signatures. This metadata includes:

- The value of the digital signature itself.
- The name of the hash function algorithm and digital signature algorithm used to produce the digital signature.
- The parameters associated with these algorithms.
- The chain of certificates (if certificates are used to bind the signing entity to its public key) needed to validate the signature.

The resulting metadata might look something like this:

```
<premis:object>
  <--!other metadata-->
    <premis:signatureInformation>
      <premis:signatureInformationEncoding>BASE 64</premis:signatureInformationEncoding>
      <premis:signer>Susan Thomas</premis:signer>
      <premis:signatureMethod>DSA-SHA1</premis:signatureMethod>
      <premis:signatureValue>qUADDMHZkyebvRdLs+6Dv7RvgMLRIUaDB4Q9yn9XoJA79a2882ffTg==
      </premis:signatureValue>
```

1 PREMIS Working Group, *Data Dictionary for Preservation Metadata* (May 2005). URL: <http://www.oclc.org/research/projects/pmwg/premis-final.pdf>

```
      <premis:signatureValidationRules>Add reference to repository documentation detailing signature
      validation rules</premis:signatureValidationRules>
      <premis:signatureProperties>2006-11-01T10:15:16</premis:signatureProperties>
    </premis:signatureInformation>
  <--!other metadata-->
</premis:object>
```

Repositories employing digital signatures must store their own private and public keys securely. PREMIS also recommends that repositories store the definitions of algorithms and relevant standards used in their context so that these methods could be reimplemented if necessary. If the digital preservation community can agree on a small number of standards, these could perhaps be stored by a central registry, such as the OMAR Representation Information Registry[1] being developed by the DCC.

More information about PREMIS for digital archives can be found in earlier in this chapter (see p. 80).

Using Fedora for recording hash values

As of version 2.2, the Fedora digital repository software provides the facility to calculate, store and verify hash values for all files and metadata managed by the repository; it supports the recording of a single value using one of the following algorithms: MD5, SHA-1, SHA-256, SHA-384 and SHA-512.

The resulting metadata is held as part of the FOXML (Fedora's native XML metadata standard, which can be exported to METS) and looks like this:

```
<--!other metadata-->
  <foxml:datastream CONTROL_GROUP="M" ID="thumbnail" STATE="A" VERSIONABLE="true">
    <foxml:datastreamVersion CREATED="2007-02-07T15:32:05.802Z" ID="thumbnail.0"
    LABEL="Thumbnail image of Louis in the Sun" MIMETYPE="image/jpeg" SIZE="0">
      <foxml:contentDigest DIGEST="8316de8d1432a3df74c4f1c4f530187e469a1bff" TYPE="SHA-
      1"/>
      <foxml:contentLocation REF="http://shuttle.paradigm.ac.uk:8080/fedora/get/paradigm:401/
      thumbnail/2007-02-07T15:32:05.802Z" TYPE="INTERNAL_ID"/>
    </foxml:datastreamVersion>
  </foxml:datastream>
<--!other metadata-->
```

More details are available in the release notes for Fedora 2.2.[2]

Tools for creating and verifying digital signatures, keys and certificates

Tools for creating and verifying digital signatures tend to be developed for use from the command-line without assistance of a graphical user interface (GUI); the GUI interfaces that are provided sometimes lack the features of their command-line equivalents. Many are deployed as a back-end or engine for other applications and may be integrated with file manager/browsers (e.g. KDE, Gnome and MS Windows Explorer) or embedded in applications such as OpenOffice, MS Office or email clients. Some tools, for example Jacksum, are hash engines, others like GnuPG are examples of cryptographic engines supporting the Public Key Infrastructure (PKI) infrastructure.

Paradigm conducted a brief survey of tools, focused on cross-platform and open source tools, supporting the generation and validation of hash values and digital signatures. The project was interested both in:

- GUI-based programmes for non-technical users.
- Programmes that can be easily incorporated into automated work-flows by developers.

1 Digital Curation Centre, *Representation Information Registry Repository website.* URL: <http://registry.dcc.ac.uk/omar/>

2 Fedora Project, 'Checksums on Datastreams in Fedora', *Fedora Project website.* URL: <http://www.fedora.info/download/2.2/userdocs/server/features/checksumming.html>

The tools surveyed fell into the following categories: application libraries providing underlying support for the algorithms and data stores required by the command-line tools for use by developers and associated graphical user interface (GUI) front-ends. The following table provides a summary of how these relate:

Interface		Application Library	Algorithms
GUI	**Command-line**		
portecle		Bouncycastle JCE	Public-Key Cipher Hash
	GPG	(libgcrypt)	Public-Key Cipher Hash Compression
Kgpg			
GPGee			
GPA			
	jacksum	Jacksum	Hash
Hasher			
	keytool jarsigner	Sun (JCA/JCE)	Public-Key Cipher Hash W3C XML Signature & Encryption
		Apache XML Security	W3C XML Signature & Encryption
	xmlsec	XML Security Library	W3C XML Signature & Encryption

The following table summarises the features of the tools surveyed:

	Tool Summaries
BouncyCastle[1]	A Java Cryptographic Library that provides a set of independent application programming interfaces (APIs) for use in: • Digital signatures • Message digests (hashes). • Encryption (symmetric/asymmetric keys block/stream ciphers). • Key generation and management. • Certificates and certificate validation.
Portecle[2]	A Java GUI based on the Bouncycastle cryptographic libraries for creating, managing and examining key stores, keys, certificates, certificate requests and certificate revocation lists. Portecle also enables the user to covert between various keystore formats which would be of assistance in managing collections which are protected or signed by different providers.
GnuPG (GNU Privacy Guard)[3]	GnuPG (GNU Privacy Guard) is a complete implementation of the OpenPGP standard defined by RFC2440. GnuPG, also known as GPG (the name of its command-line tool) supports: • Encryption. • Digital signatures. • Key management system. • Access modules for public key directories. • Features for easy integration with other applications. • It has a range frontend applications, including KGPG, GPA, GPGee (Windows). • Version 2 of GnuPG also provides support for S/MIME (Secure / Multipurpose Internet Mail Extensions is a standard for public key encryption and signing of e-mail encapsulated in MIME).

1 Bouncy Castle. URL: <http://www.bouncycastle.org/>
2 Portecle. URL: <http://portecle.sourceforge.net/>
3 GnuPG. URL: <http://www.gnupg.org/>

	• Libgcrypt – a general purpose cryptographic library based on the code from GnuPG project. It provides functions for all cryptographic building blocks: • Symmetric ciphers (AES, DES, Blowfish, CAST5, Twofish, Arcfour). • Hash algorithms (MD4, MD5, RIPE-MD160, SHA-1, TIGER-192). • Message Authentication Codes - MACs /HMACs. • Public key algorithms (RSA, ElGamal, DSA). • Large integer functions, random numbers, etc.
KGPG[1]	A KDE (KDE is a desktop environment for Linux and Unix) GUI for GnuPG that supports key signing, importing and exporting. It can be integrated with other KDE tools such as the Konqueror file browser/manager.
GPA[2]	GPA (GNU Privacy Assistant) is a Windows GUI for the GnuPG application library.
GPGee[3]	GPGee is a Windows GUI for GnuPG adding support via a context menu for: signing, signing and encrypting, encrypting, verifying and decrypting. It works on multiple files at once.
Jacksum[4]	Jacksum is an Open Source, platform independent, Java utility for calculating and verifying checksums, hash values and file timestamps.
Hasher[5]	HasherGUI is a GUI for Jacksum. It currently supports some of the hash functions, such as MD5, SHA-1, SHA-256, SHA-512, MD4, CRC, etc., provided by Jacksum.
Jarsigner[6]	A JAR Signing and Verification Tool which is a command-line java based application and part of the Sun Java Development Kit (JDK).
Java Security Libraries (jca/ jce)[7]	Basic functionality for using cryptographic techniques is provided by the Java Cryptography Architecture (JCA) which focuses on authentication; the Java Cryptography Extension (JCE) provides a framework for implementations of encryption, key generation and key agreement, and Message Authentication Code (MAC) algorithms.
keytool[8]	Part of the Sun Java Development Kit (JDK), keytool is a command-line Java based application which allows users to manage their own public/private key pairs and associated certificates as well as storing the certificates (public keys) of other users and services.
Apache XML Security[9]	Version 1.4 provides a Java library implementing the standard Java Application Programming Interface (JSR105: XML Digital Signatures) for creating and validating XML Signatures as defined by the W3C XML Digital Signature Specification. There is also a cross-platform C++ library implementation (Version 1.3)
XML Security Library[10]	XML Security Library is a C library based on LibXML2. The library supports all the features and algorithms described in the W3C XML Digital Signature and Encryption Specification, it provides an API to sign prepared document templates, add signature(s) dynamically to a document or verify the signature(s) in the document.

Some of these command-line and GUI tools are explored further in how-tos available in the online version of the Workbook.[11]

1 KGPG. URL: <http://developer.kde.org/~kgpg/index.html>
2 GPA. URL: <http://wald.intevation.org/projects/gpa/>
3 GPGee. URL: <http://gpgee.excelcia.org/>
4 Jacksum. URL: <http://sourceforge.net/projects/jacksum/>
5 HashergGUI. URL: <http://sourceforge.net/projects/hashergui>
6 Jarsigner. URL: <http://java.sun.com/javase/6/docs/technotes/tools/windows/jarsigner.html>
7 Sun Microsystems, 'Security', *Java website.* URL: <http://java.sun.com/javase/6/docs/technotes/guides/security/index.html>
8 Sun Microsystems, 'keytool - Key and Certificate Management Tool'', *Java website.* URL: <http://java.sun.com/j2se/1.5.0/docs/tooldocs/windows/keytool.html>
9 The Apache-XML Project, 'XML Security', *The Apache Software Foundation website.* URL: <http://java.sun.com/javase/6/docs/technotes/tools/windows/keytool.html>
10 XML Security Library. URL: <http://www.aleksey.com/xmlsec/index.html>
11 Online Paradigm Workbook. URL: <http://www.paradigm.ac.uk/workbook/metadata/authenticitytools.html>

Useful resources

Persistent Identifiers

Persistent Identifiers

Digital Curation Centre, *Proceedings of the DCC Workshop on Persistent Identifiers, 30 June–1 July 2005* (Glasgow).
URL: <http://www.dcc.ac.uk/events/pi-2005/>

ERPANET, *Persistent Identifiers*, Final Report of the ERPANET Workshop on Persistent Identifiers, 17-18 June 2004 (University College Cork, Ireland).
URL: <http://www.erpanet.org/events/2004/cork/Cork%20Report.pdf>

Hilse, Hans-Werner and Kothe, Jochen, *Implementing Persistent Identifiers* (Consortium of European Research Libraries, November 2006).
URL: <http://webdoc.sub.gwdg.de/edoc/ah/2006/hilse_kothe/urn%3Anbn%3Ade%3Agbv%3A7-isbn-90-6984-508-3-8.pdf>

The National Library of Australia, *Persistent Identification Systems, Report on a consultancy conducted by Diana Dack for the NLA* (May 2001).
URL: <http://www.nla.gov.au/initiatives/persistence/PIcontents.html>

NISO, *Report of the NISO Identifiers Roundtable, 13-14 March 2006* (National Library of Medicine, Bethesda, Maryland, USA).
URL: <http://www.niso.org/news/events_workshops/ID-workshop-Report2006725.pdf>

URI

Berners-Lee, T., *Universal Resource Identifiers in WWW*, RFC 1630 (June 1994).
URL: <http://www.ietf.org/rfc/rfc1630.txt>

Berners-Lee, T., Fielding, R., and Masinter, L., *Uniform Resource Identifier (URI): Generic Syntax*, RFC 3986 (January 2005).
URL: <http://www.ietf.org/rfc/rfc3986.txt>

Daniel, R. and Mealling, M., *Resolution of Uniform Resource Identifiers using the Domain Name System*, RFC 2168 (June 1997).
URL: <http://www.ietf.org/rfc/rfc2168.txt>

URL

Berners-Lee, T., Masinter, L., and McCahill, M., *Uniform Resource Locators (URL)*, RFC 1738 (December 1994).
URL: <http://www.faqs.org/rfcs/rfc1738.html>

URN

Moats, R., *URN Syntax*, RFC 2141 (May 1997).
URL: <http://www.ietf.org/rfc/rfc2141.txt>

Internet Assigned Numbers Authority, 'URN Namespaces', *Internet Assigned Numbers Authority website.*
URL: <http://www.iana.org/assignments/urn-namespaces>

Sollins, K., and Masinter, L., *Functional Requirements for Uniform Resource Names*, RFC 1737 (December 1994).
URL: <http://www.faqs.org/rfcs/rfc1737.html>

URIs, URLs and URNs

Mealling, M., and Denenberg, R., *Report from the Joint W3C/IETF URI Planning Interest Group: Uniform Resource Identifiers (URIs), URLs, and Uniform Resource Names (URNs): Clarifications and Recommendations*, RFC 3305 (August 2002).
URL: <http://www.ietf.org/rfc/rfc3305.txt>

W3C, 'Naming and Addressing: URIs, URLs, ...', *W3C website.*
URL: <http://www.w3.org/Addressing/>

PURL

Online Computer Library Center, *PURL website*.
URL: <http://purl.org/>
Includes links to overview documents and FAQs.

Handle System

Corporation for National Research Initiatives, 'Handle Resolver Service', *The Handle System website*.
URL: <http://hdl.handle.net/>

Corporation for National Research Initiatives, *The Handle System website*.
URL: <http://www.handle.net/>

Kahn, Robert, and Wilensky, Robert, *A Framework for Distributed Digital Object Services* (May 1995). Handle: cnri.dlib/tn95-01.
URL: <http://www.cnri.reston.va.us/k-w.html.>

Sun, S., Lannom, L., and Boesch, B., *Handle System Overview*, RFC 3650 (November 2003).
URL: <http://www.ietf.org/rfc/rfc3650.txt>

Sun, S., Reilly, S., and Lannom, L., *Handle System Namespace and Service Definition*, RFC 3651 (November 2003).
URL: <http://www.ietf.org/rfc/rfc3651.txt>

Sun, S., Reilly, S., Lannom, L. and Petrone, J., *Handle System Protocol (ver 2.1) Specification*, RFC 3652 (November 2003).
URL: <http://www.ietf.org/rfc/rfc3652.txt>

DOI

The International DOI Foundation (IDF), *DOI System website.*
URL: <http://www.doi.org/>
Including overviews, DOI handbook etc.

The International DOI Foundation (IDF), 'Resolve a DOI', *DOI System website.*
URL: <http://dx.doi.org>

Paskin, Norman, 'DOI: a 2003 progress report', *D-Lib Magazine* (June 2003).
URL: <http://www.dlib.org//dlib/june03/paskin/06paskin.html>

Paskin, Norman, *Digital Object Identifier*, presentation from the ERPANET Workshop on Persistent Identifiers, 17-18 June 2004 (University College Cork, Ireland).
URL: <http://www.erpanet.org/events/2004/cork/presentations/040617PaskinDOIpresentation.pdf>

ARK

California Digital Library, 'Archival Resource Key (ARK), *California Digital Library website.*
URL: <http://www.cdlib.org/inside/diglib/ark/>

Kunze, John A., *Towards Electronic Persistence Using ARK Identifiers* (July 1993).
URL: <http://www.cdlib.org/inside/diglib/ark/arkcdl.pdf>

Kunze, J., and Rodgers, R.P.C., *The ARK Persistent Identifier Scheme*, Internet Draft (23 August 2006).
URL: <http://www.ietf.org/internet-drafts/draft-kunze-ark-14.txt>

Fedora

Fedora, 'Fedora identifiers', *Fedora website.*
URL: <http://www.fedora.info/definitions/identifiers/>

Preservation metadata

Anderson, Sheila, et al., *Digital Images Archiving Study*, (March 2006)/
URL: <http://www.jisc.ac.uk/uploaded_documents/FinalDraftImagesArchivingStudy.pdf>

Caplan, Priscilla, and Guenther, Rebecca, 'Practical Preservation: The PREMIS Experience', *Library Trends*, 54, 1 (Summer 2005), pp. 111-124.

Caplan, Priscilla, 'Preservation Metadata', *Digital Curation Manual*, (July 2006, Version 1.0).
URL: <http://www.dcc.ac.uk/resource/curation-manual/chapters/preservation-metadata>

The Cedars project, *CEDARS Guide to Preservation Metadata*, (March 2002).
URL: <http://www.leeds.ac.uk/cedars/guideto/metadata/guidetometadata.pdf>

Harvard University Library, *Administrative Metadata for Digital Audio Files* (2004)
URL: <http://preserve.harvard.edu/resources/audiometadata.pdf>

Harvard University Library, *DRS Documentation Administrative Metadata for Digital Image Files* (2004)
URL: <http://preserve.harvard.edu/resources/imagemetadata.pdf>

Harvard University Library, *Global Digital Format Registry (GDFR) website.*
URL: <http://hul.harvard.edu/gdfr/>

Harvard University Library, 'JHOVE (JSTOR/Harvard Object Validation Environment)', *Harvard University Library website.*
URL: <http://hul.harvard.edu/jhove/>

Lavoie, Brian, and Gartner, Richard, 'Preservation Metadata', *DPC Technology Watch Series Report* 05-01 (September 2005).
URL: <http://www.dpconline.org/docs/reports/dpctw05-01.pdf>

Lee, Bronwyn, Clifton, Gerard, and Langley, Somaya, *PREMIS Requirement Statement Project Report*, (ASPR, National Library of Australia, July 2006).
URL: <http://www.apsr.edu.au/publications/presta.pdf>

The Library of Congress, *Sustainability of Digital Formats: Planning for Library of Congress Collections website.*
URL: <http://www.digitalpreservation.gov/formats/>

Lupovici, Catherine, and Masanès, Julien, *Metadata for the Long Term Preservation of Electronic Pulications* (September 2000).
URL: <http://nedlib.kb.nl/results/NEDLIBmetadata.pdf>

The National Archives, 'PRONOM: Online Registry of Technical Information', *The National Archives website.*
URL: <http://www.nationalarchives.gov.uk/pronom/>

National Library of Australia, 'Preservation Metadata for Digital Collections: Exposure Draft' (1999), *National Library of Australia website*.
URL: <http://www.nla.gov.au/preserve/pmeta.html>

National Library of New Zealand, 'Metadata Standards Framework – Metadata Implementation Schema' (July 2003), *National Library of New Zealand website.*
URL: <http://www.natlib.govt.nz/files/nlnz_data_model.pdf>

National Library of New Zealand, 'Metadata Standards Framework – Preservation Metadata Data Model' (Revised) (July 2003), *National Libary of New Zealand website*.
URL: <http://www.natlib.govt.nz/catalogues/library-documents/downloadpage.2007-02-15.6613783926>

National Library of New Zealand, 'Preservation Metadata Extract Tool', *National Library of New Zealand website.*
URL: <http://www.natlib.govt.nz/about-us/current-initiatives/metadata-extraction-tool>

Network Development and MARC Standards Office, 'ECHO Dep Generic METS Profile for Preservation and Digital repository Interoperability' (2005), *Metadata Encoding & Transmission Standard website.*
URL: <http://www.loc.gov/standards/mets/profiles/00000015.html>

OCLC/RLG Working Group on Preservation Metadata, *A Metadata Framework to Support the Preservation of Digital Objects* (June 2002).
URL: <http://www.oclc.org/research/projects/pmwg/pm_framework.pdf>

OCLC/RLG Working Group on Preservation Metadata, *Preservation Metadata for Digital Objects: A Review of the State of the Art* (January 2001).
URL: <http://www.oclc.org/research/projects/pmwg/presmeta_wp.pdf >

PREMIS Implementers Group (PIG), *PREMIS Implementers Group website.*
URL: <http://www.loc.gov/standards/premis/pig.html>

PREMIS Implementers Group (PIG), 'Swiki', *PREMIS Metadata Maintenance Activity website.*
URL: <http://www.loc.gov/standards/premis/pig.html>

PREMIS Working Group, *Data Dictionary for Preservation Metadata: Final Report of the PREMIS Working Group* (May 2005).
URL: <http://www.oclc.org/research/projects/pmwg/premis-final.pdf>

Preservation Metadata Implementation Strategies (PREMIS), *Preservation Metadata Maintenance Activity website.*
URL: <http://www.loc.gov/standards/premis/>

Waters, Donald and Garrett, John, 'Preserving Digital Information', Report of the Task Force on Archiving of Digital Information, commissioned by The Commission on Preservation and Access and The Research Libraries Group (May 1996).
URL: <http://www.rlg.org/legacy/ftpd/pub/archtf/final-report.pdf>

METS

Cantara, Linda, 'METS: The Metadata Encoding and Transmission Standard.' *Cataloging and Classification Quarterly,* 40 (2005), pp. 237-253.
Caplan, Priscilla, 'Preservation Metadata', *DCC Curation Manual Instalment* (July 2006).
URL: <http://www.dcc.ac.uk/resource/curation-manual/chapters/preservation-metadata/preservation-metadata.pdf>

Day, Michael, 'Metadata', *DCC Curation Manual Instalment* (November 2005).
URL: <http://www.dcc.ac.uk/resource/curation-manual/chapters/metadata/metadata.pdf>

Gartner, Richard, *METS: Metadata Encoding and Transmission Standard* (October 2002).
URL: <http://www.jisc.ac.uk/uploaded_documents/tsw_02-05.pdf>

METS Editorial Board, *<METS> Metadata Encoding and Transmission Standard: Primer and Reference Manual*, (30 September 2007).
URL: <http://www.loc.gov/standards/mets/METS%20Documentation%20final%20070930%20msw.pdf> [last accessed 25 Oct 2007]

Network Development and MARC Standards Office, *Metadata Encoding & Transmission Standard (METS) website*.
URL: <http://www.loc.gov/standards/mets/>
Includes the current version of the Schema, the Profile Schema and related documentation, including a METS Overview and Tutorial.

Oxford Digital Library, 'JISC-funded METS Awareness Training Project website. Includes a series of powerpoint training presentations, and useful METS links', *Oxford Digital Library website*.
URL: <http://www.odl.ox.ac.uk/projects/projects_mets.htm>

Rights metadata

Coyle, Karen, *Rights Expression Languages: A Report for the Library of Congress* (February 2004).
URL: <http://www.loc.gov/standards/relreport.pdf>
Coyle, Karen, 'The "Rights" in Digital Rights Management', *D-Lib Magazine*, 10, 9 (September 2004).
URL: <http://www.dlib.org/dlib/september04/coyle/09coyle.html>

Coyle, Karen, 'Descriptive metadata for copyright status' *First Monday,* peer reviewed journal on the Internet, 10, 10 (October 2005).
URL: <http://www.firstmonday.org/issues/issue10_10/coyle/index.html>

Coyle, Karen, *Rights in the PREMIS Data Model: A Report for the Library of Congress* (December 2006).
URL: <http://www.loc.gov/standards/premis/Rights-in-the-PREMIS-Data-Model.pdf>

Creative Commons, *Creative Commons website.*
URL: <http://creativecommons.org/>

Dublin Core Metadata Initiative, *Dublin Core Metadata Initiative website.*
URL: <http://dublincore.org/>

The Library of Congress, 'METSRights', *The Library of Congress website.*
URL: <http://www.loc.gov/standards/rights/METSRights.xsd>

Martin, Mairéad, et. al., 'Federated Digital Rights Management: a proposed DRM solution for research and education', *D-Lib Magazine*, 8, 7/8 (July/August 2002).
URL: <http://www.dlib.org/dlib/july02/martin/07martin.html>
Network Development and MARC Standards Office, *Encoded Archival Description Version 2002 website.*
URL: <http://www.loc.gov/ead/>

Network Development and MARC Standards Office, *Metadata Object Description Schema (MODS) website.*
URL: <http://www.loc.gov/standards/mods/>

Open Digital Rights Language (ODRL) Initiative, *ODRL Initiative website.*
URL: <http://odrl.net/>

Organisation Internationale De Normalisation, *MPEG-21/5* (October 2002).
URL: <http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm>

Preservation Metadata Implementation Strategies (PREMIS), *Preservation Metadata Maintenance Activity website.*
URL: <http://www.loc.gov/standards/premis/>

Metadata for authenticity

The Apache XML Project, 'Apache XML Security', *The Apache Software Foundation website.*
URL: <http://xml.apache.org/security/>

Bouncy Castle
URL: <http://www.bouncycastle.org/>

Fedora, 'Checksums on Datastreams in Fedora: Fedora Release 2.2', *Fedora website.*
URL: <http://www.fedora.info/download/2.2/userdocs/server/features/checksumming.html>

GPA
URL: <http://wald.intevation.org/projects/gpa/>

GnuPG
URL: <http://www.gnupg.org/>

GPGee
URL: <http://gpgee.excelcia.org/>

Hasher
URL: <http://sourceforge.net/projects/hashergui/>

Jacksum
URL: <http://sourceforge.net/projects/jacksum/>

jarsigner
URL: <http://java.sun.com/javase/6/docs/technotes/tools/windows/jarsigner.html>

KGPG
URL: <http://developer.kde.org/~kgpg/>

PGP, 'An introduction to Cryptography', *PGP website.*
URL: <http://www.pgpi.org/doc/pgpintro/>

Portecle
URL: <http://portecle.sourceforge.net/>

PREMIS Working Group, *Data Dictionary for Preservation Metadata: Final Report of the PREMIS Working Group* (May 2005).
URL: <http://www.oclc.org/research/projects/pmwg/premis-final.pdf>

RFC 2828 - Internet Security Glossary (May 2000).
URL: <http://www.faqs.org/rfcs/rfc2828.html>

Sun Microsystems, 'Security', *Java website*.
URL: <http://java.sun.com/javase/6/docs/technotes/guides/security/index.html>
Sun Microsystems, 'Java Cryptography Architecture', *Java website.*
URL: <http://java.sun.com/javase/6/docs/technotes/guides/security/crypto/CryptoSpec.html>

Sun Microsystems, 'keytool - Key and Certificate Management Tool', *Java website*.
URL: <http://java.sun.com/javase/6/docs/technotes/tools/windows/keytool.html>

W3C, 'Canonical XML version 1.0' (March 2001), *W3C website.*
URL: <http://www.w3.org/TR/2001/REC-xml-c14n-20010315>

W3C, 'Exclusive XML Canonicalization 1.0' (July 2002), *W3C website.*
URL: <http://www.w3.org/TR/2002/REC-xml-exc-c14n-20020718/>

W3C, 'XML-Signature Syntax and Processing' (February 2002), *W3C website.*
URL: <http://www.w3.org/TR/xmldsig-core/>

Wikipedia, 'Base64', *Wikipedia website.*
URL: <http://en.wikipedia.org/wiki/Base64>

Wikipedia, 'Hash function', *Wikipedia website.*
URL: <http://en.wikipedia.org/wiki/Hash_functions>

Wikipedia, 'Public-key cryptography', *Wikipedia website.*
URL: <http://en.wikipedia.org/wiki/Public-key_cryptography>

XML Security Library.
URL: <http://www.aleksey.com/xmlsec/>