Op standaarden gebaseerde interfaces voor het verzamelen en verwerven van objecten uit digitale bewaarplaatsen

Standards-Based Interfaces for Harvesting and Obtaining Assets from Digital Repositories

Jeroen Bekaert

Promotoren: prof. dr. ir.-architect E. De Kooning, dr. H. Van de Sompel
Proefschrift ingediend tot het behalen van de graad van
Doctor in de Ingenieurswetenschappen

Vakgroep Architectuur en Stedenbouw
Voorzitter: prof. dr. B. Verschaffel
Faculteit Ingenieurswetenschappen
Academiejaar 2005 - 2006

UNIVERSITEIT
GENT

dedicated to P.B. (in memoriam)

English Summary

Various communities are deploying and maintaining digital repositories, as a serious and long-term commitment to store, safeguard, and share rich collections of digital assets. Each community has a different perspective on the design and management of digital repositories. For example, the learning community is developing repositories that focus on use and re-use of learning objects; the digital library community focuses on storage and access of scholarly articles; the cultural heritage community works on curation of digital images, and so on. This focus on particular materials and application domains has led to a variety of parallel technical approaches used in the design and implementation of repository systems, and, as a result to a serious lack of cross-community and cross-repository interoperability. The net result of this lack of interoperability is the inability to use and re-use assets stored in heterogeneous repositories in cross-repository services and service workflows. As a result, the promise and potential of a truly interconnected digital knowledge environment remains unfulfilled.

This general problem presents itself at a smaller, yet significant, scale in the context of the ongoing deployment of institutional repositories. These repositories are increasingly used by universities and research institutions as tools to take on the stewardship of the materials created by their constituency, and to make them accessible to the general public. Several institutional repository systems have been developed and adopted successfully, including

DSpace (Massachusetts Institute of Technology & Hewlett-Packard Labs), Fedora (Cornell University & University of Virginia), and EPrints (University of Southampton). All of these institutional repository systems are conceived as autonomous software components and their content is accessed using diverse proprietary interfaces. This lack of common repository interfaces hinders the realization of the vision of a new scholarly communication system that would have these institutional repositories – and other scholarly repositories – as its core components, as realizing that vision would require the development of federations of such scholarly repositories, the provision of cross-repository service overlays, and the expression and invocation of automated workflow applications.

The study that is object of this doctoral dissertation is framed in this problem space, and focuses on the definition and design of two essential interfaces to digital repositories: An interface to enable harvesting of batches of digital assets (harvest interface) and an interface to enable obtaining disseminations of a single digital asset (obtain interface).

The characteristics and requirements for such harvest and obtain interfaces have been explored over the course of two elaborate experiments. A first experiment focused on the multi-faceted use of the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) and NISO's OpenURL Framework for Context-Sensitive Services as information access protocols in a repository architecture designed and implemented for ingesting, storing, and accessing a vast collection of digital assets at the Research Library of the Los Alamos National Laboratory. A second experiment aimed at designing and implementing a robust solution for the recurrent and accurate transfer of digital assets from the repository of the American Physical Society to the aDORe environment of the Los Alamos National Laboratory; again the OAI-PMH played a significant role in the deployed solution.

The insights obtained from these experiments are generalized and they result in the definition of generic harvest and obtain interfaces that could be supported across heterogeneous repositories. The proposed protocol-based interfaces operate in a manner that is neutral to the way in which a digital repository stores, manages and identifies its digital assets, and they build upon existing technologies that have recently received considerable buy-in from the digital library and scholarly information communities. These technologies are the OAI-PMH, NISO's OpenURL Framework and several XML-based techniques for the representation and packaging of compound digital assets.

By proposing these repository interfaces, the author hopes to contribute to the ongoing global discourse aimed at reaching new levels of interoperability across systems that store valuable content, and hence towards the realization of a richer scholarly information environment.

Nederlandstalige Synthese

Digitale bewaarplaatsen voor het ontsluiten, opslaan en langdurig bewaren van datacollecties hebben de laatste tien jaar een sterke toename gekend in uiteenlopende disciplines en vakdomeinen. Elke discipline heeft haar eigen visie op het ontwerp en beheer van digitale bewaarplaatsen. Zo zijn er bewaarplaatsen die zich richten op het gebruik en hergebruik van elektronisch leer- en studiemateriaal, digitale bibliotheken voor het stockeren en toegankelijk maken van onderzoeksartikels, bewaarplaatsen voor het ontsluiten van gedigitaliseerd cultureel erfgoed, enzovoort. Deze evolutie kan ongetwijfeld positief worden genoemd, zij het niet dat deze verscheidenheid aan visies en materialen tot een veelvoud aan parallelle technologische oplossingen voor het ontwerpen en ontwikkelen van digitale bewaarsystemen heeft geleid. Het gevolg hiervan is een gebrek aan interoperabiliteit van de verscheidene bewaarsystemen en disciplines. Het gemis aan eenvormigheid en coherentie leidt tot een onvermogen om objecten, die in heterogene digitale bewaarplaatsen gestockeerd worden, te gebruiken en hergebruiken in systeemoverschrijdende *online*-diensten en werkschema-toepassingen. De mogelijkheden van een sterk geconnecteerde digitale kennisomgeving worden bijgevolg niet gerealiseerd, laat staan optimaal benut.

Deze problematiek komt tot uiting op een kleinere, maar belangrijke schaal in de huidige, sterke ontwikkeling van zogenaamd institutionele digitale bewaarplaatsen. Deze laatste worden door universiteiten en andere onderzoeksinstellingen meer en meer als hulpmiddel gebruikt om eigen onderzoeksresultaten en -artikels intern te bewaren en via het Internet toegankelijk te maken voor externe geïnteresseerden. Voorbeelden van kant-en-klare softwarepakketten voor de uitbouw van dergelijke institutionele bewaarplaatsen zijn DSpace (Massachusetts Institute of Technology & Hewlett-Packard Labs), Fedora (Cornell University & University of Virginia) en EPrints (University of Southampton). Elk van deze systemen wordt autonoom ontwikkeld en ontsloten aan de hand van onderling sterk verschillende technische specificaties. Door het gebrek aan uniforme interfaces wordt de realisatie van nieuwe wetenschappelijke communicatienetwerken, gebaseerd op institutionele of andere bewaarplaatsen, aanzienlijk gehinderd. Dergelijke netwerken zijn immers sterk afhankelijk van de ontwikkeling van federaties van bewaarplaatsen en van de uitbouw van *online*-diensten die de verschillende bewaarplaatsen overkoepelen.

Het onderzoek dat in het kader van deze doctorale studie werd uitgevoerd, moet binnen deze interoperabiliteitsproblematiek worden gesitueerd. De studie richt zich op de definitie en het ontwerp van een aantal generieke interfaces voor heterogene digitale bewaarsystemen. Het proefschrift onderzoekt in het bijzonder interfaces voor het verzamelen van bundels van complexe objecten (*harvest interfaces*), en interfaces voor het verwerven van disseminaties van individuele complexe objecten (*obtain interfaces*).

Om deze interfaces te kunnen ontwikkelen, en hun eigenschappen te verkennen, werden twee uitgebreide experimenten uitgevoerd. In een eerste experiment werd onderzoek verricht naar een digitale bewaaromgeving voor het deponeren, stockeren, verwerken en ontsluiten van de omvangrijke datacollecties van de Research Library van het Los Alamos National Laboratory. Zowel het Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) en NISO's OpenURL Framework for Context-Sensitive Services Framework worden hier op een veelzijdige manier ingezet. Een tweede experiment beoogde de ontwikkeling van een raamwerk voor de herhaaldelijke en accurate overdracht van nieuwe en gewijzigde complexe dataobjecten van de digitale bewaarplaats van de American Physical Society naar de digitale bewaaromgeving van het Los Alamos National Laboratory. Ook hier is een belangrijke rol weggelegd voor het OAI-PMH raamwerk.

De inzichten die uit beide experimenten zijn voortgevloeid, worden veralgemeend en monden uit in de definitie van interfaces voor het verzamelen en verwerven van digitale objecten. Deze interfaces zijn inzetbaar in een verscheidenheid aan digitale bewaarsystemen. De voorgestelde interfaces functioneren onafhankelijk van de wijze waarop een digitaal bewaarsysteem data-objecten stockeert, beheert en identificeert. De interfaces worden opgebouwd aan de hand van een aantal recente en eenvoudige technologieën die succesvol geïntegreerd werden in het domein van de digitale bibliotheken en informatiewetenschappen. De technologieën waarvan gebruik wordt gemaakt, zijn het OAI-PMH, NISO's OpenURL Framework en verscheidene XML-gebaseerde technieken voor de representatie en het verpakken van complexe datastructuren.

De auteur hoopt met dit proefschrift een bijdrage te leveren aan het lopende discours omtrent nieuwe niveaus van interoperabiliteit tussen bewaarsystemen die belast zijn met het stockeren van rijke en waardevolle informatie. Tegelijk is het zijn hoop om hiermee de totstandkoming van een krachtige wetenschappelijke informatieomgeving te kunnen bevorderen.

Acknowledgments

A little more than four years ago, after finishing my Master's thesis, prof. Mil De Kooning warmed me up for a grand project: A large amount of architectural drawings, sketches and books lies languishing in dusty basements. Digitizing them and accessing them via the Web should save them from downfall. My area of research was born. I am not only grateful to Mil De Kooning for his never-ending fascination, but also for his attentive ear, his support in getting me a research fellowship with the Research Foundation (Flanders) and his commitment to keeping me free from all sorts of bothers.

Fairly early on, Mil De Kooning's vision was replaced by dire reality. The envisaged research domain of data computing and digital archiving was large, too large. Only after a number of conversations with prof. Rik Van de Walle, of the Department of Electronics and Information Systems at the Ghent University, my initial scepticism was overcome. During the first year of my doctoral research his expertise and comments helped me understand the issues involved in the area of research. He monitored my research with an enriching mixture of generosity and rigidity. Rik Van de Walle also introduced me to the 'world' of international standardization, and actively supported my participation in the Moving Picture Experts Group. Very sincere thanks.

In November 2002, I met dr. Herbert Van de Sompel during a lecture in the Pand-building in Ghent. This crucial evening caused an about-face, not only in my research but also in my personal life. After his lecture we spoke to each other only briefly, but the offer was clear. I could further develop my research at one of the most privileged places in the world, the Los Alamos National Laboratory in New Mexico, better known as Atomic City. The decision didn't take very long, the wheels of the administration turned a little more slowly, but in March 2003, I went to New Mexico, to the Land of Enchantment; initially for nine months, which turned into a period of two years and a half.

During my stay in Atomic City my doctoral research acquired a clear direction. There, I worked in close cooperation with the Prototyping Team, an international team under the direction of Herbert Van de Sompel. I got the opportunity to get to the bottom of the research domains of digital libraries and digital preservation and I could develop my doctoral research into what it is now. In the course of my stay I had ample opportunity to enlarge my scientific visibility. I established close contacts with researchers of, for instance, the Library of Congress, NASA Space Data Center and Cornell University; and have been offered many opportunities to participate in the domain leading meetings and conferences.

I want to thank Herbert Van de Sompel for his immense support, his vision and enthusiasm for my research, for the opportunity he gave me to come to the Los Alamos National Laboratory, for the very interesting, educational but especially pleasant years in Santa Fe.

I thank Rick Luce, Director of the Research Library at the Los Alamos National Laboratory, for the moral and financial support during my stay at the Lab in Los Alamos; colleagues Patrick Hochstenbach (at this moment working at the Ghent University), Johan Bollen, Xiaoming Liu, Luda Balakireva, Henry Jerez (at this moment working at CNRI), Thorsten Schwander, Nathan McFarland and Ryan Chute for the productive but especially pleasant cooperation. Thanks are also due to Carl Lagoze, Sandy Payette, Simeon Warner and Michael Nelson, for reading and commenting upon the work in progress, their intellectual stimulations and constructive criticisms.

During my stay in Santa Fe, the house of Herbert and Katrien became more or less my home away from home. Thanks for the warm reception and the numerous times that I could eat at your table, for the unforgettable mountain hikes, the evenings on the porch with a view of the mountains and the (many) margaritas.

Herbert Van de Sompel guided me on writing this doctoral dissertation and verified language and bibliography. Mil De Kooning proofread the text. Geert Roels assisted with layout and printing of the work. I thank all of them for their contribution.

I am also indebted to my colleagues at Ghent University's Department of Electronics and Information Systems and Ghent University's Department of Architecture and Urban Planning for their hospitality. I especially thank prof. Jan Van Campenhout, Head of the Department of Electronics and Information Systems, for his true moral support. For their assistance and steady logistic support, I thank Rita Breems, Sandra Joliet, Brigitte Schollaert, Mieke Van Damme, Karel Lemaitre and Geert Roels.

Writing a doctoral dissertation is a solitary and mainly long-term occupation. During all those months many people had no choice but to do without me. I want to thank my friends – in particular Johan Wouters and Roeland Dudal – for picking up the thread, even after long periods of communicative silence. I want to thank my parents for their support and never-ending patience; my grandparents for their sincere interest in my research, my foreign experience and my personal happiness. I thank my family for making the effort of trying to piece together what exactly I was doing in the United States of America, every time I returned for an ever shrinking visit to the homeland. After all these months it is – I'm afraid – still not altogether clear to them what exactly I was working on so far away from home.

My last words of gratitude go to Annelies Ryckaert. During this doctoral work our happiness was often put to the test; we were not able to share the extended foreign experiences. Still, she always found the courage to convince me to continue my research. This doctoral dissertation is a tribute to her support, patience, and fortitude.

Jeroen Bekaert, Ghent, February 2006

Acronyms

AIP (OAIS)	(OAIS) Archival Information Package
API	Application Programming Interface
APS	American Physical Society
CCSDS	Consultative Committee for Space Data Systems
CNRI	Corporation for National Research Initiatives
DIA (MPEG-21)	(MPEG-21) Digital item Adaptation
DID (MPEG-21)	(MPEG-21) Digital Item Declaration
DIDL (MPEG-21)	(MPEG-21) Digital Item Declaration Language
DII (MPEG-21)	(MPEG-21) Digital Item Identification
DIP (MPEG-21)	(MPEG-21) Digital Item Processing
DIP (OAIS)	(OAIS) Dissemination Information Package
DNS	Domain Name System
DOI	Digital Object Identifier
FRBR	Functional Requirements for Bibliographic Records
HTTP	Hypertext Transfer Protocol
IMS-CP	IMS Content Packaging
IPMP (MPEG-21)	(MPEG-21) Intellectual Property Management and Protection
ISBN	International Standard Book Number
ISO	International Organization for Standardization
JISC	Joint Information Systems Committee
KEV	Key/Encoded-Value
LANL	Los Alamos National Laboratory
LCCN	Library of Congress Control Number
METS	Metadata Encoding and Transmission Standard
MODS	Metadata Object Description Schema
MPEG	Moving Picture Experts Group
NASA	National Aeronautics and Space Administration
NDIIPP	National Digital Information Infrastructure and Preservation Program
NEDLIB	Networked European Deposit Library
NISO	National Information Standards Organization
OAI	Open Archives Initiative
OAI-PMH	Open Archives Initiative Protocol for Metadata Harvesting
OAIS	Open Archival Information System
OCLC	Online Computer Library Center
OWL	Web Ontology Language
PREMIS	Preservation Metadata Implementation Strategies
RDD (MPEG-21)	(MPEG-21) Rights Data Dictionary
RDF	Resource Description Framework
REL (MPEG-21)	(MPEG-21) Rights Expression Language
RSS	Really Simple Syndication
SIP (OAIS)	(OAIS) Submission Information Package

SRU	Retrieve URL Service
SRW	Retrieve Web Service
URI	Uniform Resource Identifier
URN	Uniform Resource Name
UUID	Universally Unique Identifier
W3C	The World Wide Web Consortium
XFDU	XML Formatted Data Units
XInclude	XML Inclusion 1.0
XML	Extensible Markup Language

Introduction

Introduction

1. Digital repositories

In laymen's terms, a digital repository is a deposit in which digital materials are stored and can be retrieved for use later. A repository supports methods and workflows to ingest, identify, store, preserve and retrieve digital materials. Digital repositories can be very big or small, storing a large amount of materials or just a single asset. As such, in some contexts a tiny mobile device that contains a few assets can be considered a digital repository, but generally, digital repositories are computer systems that store digital materials using an established data storage mechanism and present it to the world through a public interface.

Currently, as reported by Arms (2000), by far the most common form of digital repository is a web server. A web server is a repository system tasked with the storage of files, each of which is identified by a Uniform Resource Identifier (URI) from the '`http`' namespace (see Section 2.2), and can be accessed using the IETF Hypertext Transfer Protocol (HTTP). Web servers are widely used, and owe much of their success to their simplicity; but some of their simplifying assumptions cause problems for their exploitation in the information domain. For example, web servers support only one data model, a hierarchical file system where digital materials are organized into separate files. Also, access of web servers is tightly controlled by the HTTP protocol which requires the use of URIs from the '`http`' namespace for the identification of stored files. Little planning took place to define mechanisms for long-term persistence and exchangeability; confer the familiar '`404 File Not Found`' error messages, indicating that the web file had been removed, had its name changed, or is temporarily unavailable. As such, although web servers are widely used, other, more advanced types of digital repository systems are needed in the information domain. Such digital repository systems should accommodate the richness of data models that are emerging. But, more importantly, such repositories should focus on guarding and accessing digital materials in the long term.

An increasing range of communities within the information domain are trying to build such – more sophisticated – digital repositories. However, as outlined in Digital Repositories Overview (Heery & Anderson, 2005) of the Joint Information Systems Committee (JISC), each community has a somewhat different focus on the matter: The learning community has a particular interest in repositories that support the IMS Interoperability Specification (IMS Global Learning Consortium, 2003a); the cultural heritage community focuses on the curation of digital images, the digital library community explores the storage and access of research data, the preservation community is working on the deployment of digital preservation models, space agencies are developing mass storage repository systems for satellite and GIS data, and so on. Despite these differences in focus, the intersection across the various communities offers possibilities for various forms of sharing of both content and technologies. As a matter of fact, interesting innovative studies are now emerging, exploring interaction between repository systems from different communities. For example, McLean and Lynch (2004) explored potential interactions between digital library repository systems and e-learning repository systems, with emphasis on the work that needs to be done involving standards, architectural modeling and interfaces – as opposed to cultural or organizational issues – in order to permit these two worlds to co-exist more productively. Recent work by Blinco and others provides a summary of current trends in the development of e-learning repositories, and tries to explore the boundaries with other information communities (Blinco Mason, McLean & Wilson, 2004). And McLean has put forward an 'ecology' of repository services in which he instigates a common understanding within and across domains of service interfaces and required levels of interoperability (McLean, 2004).

Another recent stirring in the area of digital repositories is the development of so-called institutional repositories. As advocated by Van de Sompel (1999) and analyzed by Lynch (2003), the adoption of institutional repositories is rapidly emerging as a new strategy that allows universities and other research institutions to take stewardship of the digital material created by the institution and its community members. Ultimately almost every research institution will be offering some tailored repository services to its community. In response to this trend, a handful off-the-shelf institutional repository systems are being developed by some leading universities, pointing the way forward for other research institutions. Such institutional repository systems include DSpace, a repository system jointly developed by the Massachusetts Institute of Technology Libraries and Hewlett-Packard (Smith et al., 2003; Tansley, Bass & Smith, 2003; Tansley, Smith & Walker, 2005); Fedora, an open-source software jointly developed by Cornell University and the University of Virginia (Payette & Lagoze, 1998; Staples, Wayland & Payette, 2003; Lagoze, Payette, Shin & Wilper, in press) and EPrints, an institutional repository software developed at the University of Southampton, designed more specifically for the deposit of research papers, as opposed to arbitrary digital materials ("EPrints," 2005; Gutteridge, 2002).

All of these repository systems aim at hosting digital materials and at providing tools for storing, managing, identifying and accessing those materials. However, in spite of their similar goals, each of these systems comes with its own perspective on how to accomplish them. Indeed, different repository systems may serve different user communities, have different policies regarding what descriptive metadata is required for deposit, what storage mechanisms to use, which identification mechanisms to employ, what strategies need to be taken for long-term preservation, and so forth. Also, digital repository systems define different public interfaces for the access and interchange of digital materials. This lack of common repository interfaces is a significant limitation in the development of rich service infrastructures and new scholarly communication workflows. The exploration and design of such repository interfaces are the subject of this dissertation.

2. Digital materials

Materials stored in a digital repository can be divided into *data* and *metadata*. Data is a common term encompassing all information resources that are encoded in digital form. Metadata – literally, data about other data – is structured textual information that describes, or makes it easier to retrieve, use, or manage data. Metadata is typically divided into several categories, such as descriptive metadata (e.g., bibliographic records), preservation metadata (used to support and document the digital preservation process of data), and rights metadata (used to manage access restrictions to data) (Arms, 2000).

The distinction between data and metadata often depends upon the context. For example, catalog records or Abstracts and Indexes are usually considered to be metadata, because they describe other data. But in an online catalog or a digital repository of abstracts they are considered the actual data of that catalog or digital repository, respectively.

In this dissertation, the terms *descriptive metadata* and *secondary data* are introduced. The former is used when referring to descriptive catalog records or Abstracts and Indexes, no matter the context in which those reside. The latter is used to refer to all kinds of auxiliary structured information (about other data) that is not considered expressive descriptive metadata. This distinction is needed to reflect the reality that records from Abstracting and Indexing databases play an important role in the digital library and information domain. These descriptive metadata records – in many cases very expensive to create – are often hosted as 'stand-alone' files, in that they do not come with their full-content counterparts. As a result, they can hardly be considered secondary data because there is no primary datastream to which to attach them. In contrast with descriptive metadata, secondary data per se, must not be treated as datastreams in their own right, yet are created for the sole purpose of supporting the storage and retrieval of such datastreams.

The term *content* is used when the emphasis is on the intellectual information or knowledge being represented by data, rather than the data representation itself. Typically, the *content* is the information of primary interest to a consumer.

2.1. Digital materials and their representation

In order for data to be organized efficiently, the granularity of the data must be set to match the level of abstraction at which a consumer or downstream application wants to refer to that data. While an isolated sequence of bits (also known as a stream of data or *datastream*) is typically considered the basic entity of information in the digital world, a consumer often refers to digital material at a different level of abstraction than the individual datastream. A consumer refers to such things as articles, e-books, web pages, lectures, presentations, and so forth.

When consulting – for example – an article in digital form, the article may be instantiated in many datastreams that, in some way, are related to each other. These datastreams may have different formats, different rights and permissions, different preservation requirements, minor differences of content, and so forth; but for some purposes, consumers are considering these datastreams as part of the same entity. The rationale for doing so is twofold:

- The same content may be stored in various digital formats. For example, an article may be formatted as a high resolution master PDF data file, a lower resolution PDF viewing file, a text format with SGML markup, and so on. Similarly, a picture may be stored in TIFF, JPEG and JPEG2000 formats.
- Datastreams may be part of a broader sequential or hierarchical structure. For example, a digitized book may consist of chapters, pages, sections, frontmatter, and table of contents; a web page may include several pages of text containing markup, embedded images, and links to other web pages; materials may belong to collections – these may be groupings

based on repository-specific criteria or collections in the traditional, custodial sense – and so forth.

To this end, within the information domain, data is typically organized in the form of so-called digital assets. A *digital asset* is an abstract information construct that conceptually aggregates multiple streams of relevant data, descriptive metadata and secondary data so that multi-part information may be managed by a single entity. A digital asset typically consists of one ore more datastreams (or bit sequences), secondary data pertaining to the digital asset and/or its datastreams, and a globally unique identifier of the digital asset. A digital asset can be a simple structure consisting of one datastream, or can be a more complex structure containing multiple datastreams combined with a variety of secondary data. In such frameworks as the Functional Requirements for Bibliographic Records (FRBR) (International Federation of Library Associations [IFLA], 1998) and the <in*d*ecs> Metadata Framework (Rust & Bide, 2000), a digital asset is aligned with Work, Expression and Manifestation (ProductType).

One category of digital assets that have a more complex aggregated data structure is e-books. An e-book is typically identified by means of an International Standard Book Number (ISBN) (Hakala & Walravens, 2001) and comprised of multiple datastreams. The e-book may consist of a 1200 dpi master PDF data file, a lower resolution PDF viewing data file, and several auxiliary data files providing extra background information on the subject described in the book. In addition to the ISBN identifier, secondary data pertaining to the e-book may capture identifiers of related books, specify whether the e-book is part of a collection, provide preservation information, and so on.

The seminal Kahn/Wilensky Framework (Kahn & Wilensky, 1995) calls such an aggregated content structure a digital object. Kahn & Wilensky define a digital object as *'an abstract data type that has two components, data and key-metadata. The data is a set of bit-sequences. The key-metadata includes a handle, that is, an identifier globally unique to the digital object; it may also include other metadata, to be specified'*. Another core characteristic of a digital asset is that it can be composed of several other digital assets. The Kahn/Wilensky Framework refers to such constructs as composite digital objects. A digital object that is not composite is said to be elemental.

In this dissertation, the terms digital object and *digital asset* will be used interchangeably. Individual bit sequences pertaining to a digital asset will be referred to as constituent *datastreams*. Auxiliary structured information about a digital asset and/or its constituent datastreams is referred to as *secondary data*. A digital asset must carry a unique persistent *identifier* (More information on identifiers is provided in the Section below).

A digital asset is structured on the basis of a *data model*. A data model consists of several abstract entities, each of which can be used to conceptually represent the multi-part data structure of a digital asset, that is expose the correlations between the digital asset and its constituents. The resulting abstract data structure can then be serialized as a tangible document (also called a package) using a *packaging format*. This package conveys the actual data structure of the digital asset, contains (pointers to) its datastreams and secondary data, and includes the identifier(s) of the digital asset. In the FRBR model and the <in*d*ecs> Metadata Framework, this package aligns with an Instance and Item, respectively.

Since the publication of the Kahn/Wilensky framework, several attempts have emerged from different communities aimed at creating packaging formats (LC, 2001). Of specific relevance in the context of this dissertation that focuses on interoperability are packaging techniques based on W3C's Extensible Markup Language (XML) (Bray, Paoli, Sperberg-McQueen, Maler & Yergeau, 2004). Such techniques include the Metadata Encoding and Transmission Standard (METS) (Library of Congress [LC], 2005a), an initiative of the Digital Library Federation; the IMS Content Packaging (IMS-CP) XML Binding (IMS Global Learning Consortium, 2003b), a representational technique predominantly used in the educational domain; the XML Formatted Data Units (XFDU) (Consultative Committee for Space Data Systems Panel 2 [CCSDS], 2004b), an international standard under development by the Consultative Committee for Space Data Systems (CCSDS) and aimed at aligning a representational approach with the Open Archival Information System (OAIS) Reference Model (International Organization for Standardization [ISO], 2003a), and the MPEG-21 Digital Item Declaration (MPEG-21 DID) (ISO, 2005a), an international standard developed by the Moving Picture Experts Group (MPEG). The latter will be discussed in detail in Section 2 of Chapter 1.

2.2. Digital materials and their identification

Each digital asset has a globally unique identifier. Such unique identifiers are needed for communication within and between applications. Applications use identifiers to reliably reference digital assets, to share those references with other applications and to instantiate relationships between digital assets (for instance, between a descriptive metadata record and the digital asset it describes). Identifiers are typically defined in the context of a specific domain or subspace, called the namespace, and represent points in that namespace. An identifier must be unambiguous in its declared namespace, and must point to only one 'thing' (e.g., a digital asset). Note that the opposite does not hold: A digital asset may have multiple unique identifiers.

In the Internet, data are globally identified by means of Uniform Resource Identifiers (URIs). The syntax of a URI is specified by means of a URI Scheme (Berners-Lee, Fielding & Masinter, 1998). The URI Scheme defines the namespace of the URI, and thus may further restrict the syntax and semantics of identifiers using that scheme. The official registry for a URI Scheme is maintained by IANA [http://www.iana.org/assignments/uri-schemes]. For example, the identifier '`http://foo.org/myexample/`' uses the namespace '`http`'. The latter is defined by means of URI Scheme RFC 2616 (Fielding et al., 1999). And, the identifier '`urn:isbn:0-395-36341-1`' uses the namespace '`urn`'. The Uniform Resource Name (URN) namespace is defined by means of URI Scheme RFC 2141 (Moats, 1997).

In order to take full advantage of the richness and complexity of the aforementioned digital assets, rich services will have to be built that allow for the assets to be consumed in many different ways. The process of requesting a service about an identified object is called identifier resolution. For this, an identifier is submitted to a networked system; and in response, one ore more pieces of information related to the identified data are returned. Such information may include the location of the identified data, or the location of services pertaining to that data. In order for a namespace to support resolution, two infrastructural components must be in place: First, a system providing resolution functionality (also known as a resolver), and second, a protocol to communicate with such a resolver. Different identifier namespaces require different resolution needs. A brief exploration of important identifier namespaces and their corresponding resolution mechanisms is provided below.

Data identified by a URI from the '`http`' namespace can be obtained simply by using the URI as their network location on the Web. Identifiers from the '`http`' namespace have a built-in resolution mechanism. They are resolved using the HTTP protocol (Fielding et al., 1999) and by relying on a Domain Name System (DNS) to map the domain name to an IP address. The IP address can be used to communicate with the Internet provider hosting the identified data. This mechanism is widely adopted by web servers, and because of its simplicity, is extremely powerful. However, it poses a long-term problem for the information domain (Lynch, 1997). The problem is the need for persistence. If data resources are identified by a location derived from a domain name, the resolution of the data resource will fail if the data resource is moved to a different network location or if the domain name ceases to exist. Another disadvantage of merging identification and location is the short-term life expectancy of the identifiers, as resolution protocols may evolve, change or get obsolete over time. The resolution provided by the HTTP protocol is flat (i.e., one-on-one resolution), and does not support linking the identified digital asset to rich services (i.e., multiple resolutions).

Because of this, the URN namespace has been introduced. In contrast with identifiers from the '`http`' namespace, identifiers from the '`urn`' namespace are intended to be persistent, and location-neutral; that is, they identify digital assets, in a manner that is neutral to the location of these assets. For these very reasons, the URN namespace has been warmly embraced by the digital library domain and publishers of electronic material. For example, the namespace '`urn:isbn`' – the International Standard Book Number – played a central role in facilitating business communications between booksellers and publishers, '`urn:issn`' is the namespace of periodical and serial publication identifiers, and '`urn:nbn`' is the namespace of national bibliographic numbers.

The URI Scheme for the URN namespace does not impose a resolution mechanism, but does not impede resolution either. It is believed that the provision of a resolution mechanism for identifiers from URN namespaces is an issue to be decided on – and devised – by local user communities. As different communities may have different resolution needs, each such community may suggest and implement resolution functionality on an individual basis. For example, RFC 2168 (Daniel & Mealling, 1997) and RFC 2169 (Daniel, 1997) specify how to use the HTTP protocol and DNS system for requesting URN resolution services. But, since URNs have no built-in resolution mechanism, identifiers from the '`urn`' namespace may also be resolved using other resolution protocols and service, including Z39.50 (National Information Standards Organization [NISO], 2002), Handle (Sun, Lannom and Boesch, 2003), and RWHOIS (Williamson, Kosters, Blacka, Singh & Zeilstra, 1994). The decoupling of identifiers and resolution offers the flexibility to develop multiple rich resolution mechanisms, each of which can be tailored to the respective market needs. For example, Die Deutsche Bibliothek developed an HTTP-based resolver for identifiers from the '`urn:nbn:de`' namespace. A Web interface to this resolver can be found at [http://nbn-resolving.de/]. Also, researchers from the Online Computer Library Center (OCLC) developed a web service, called xISBN, that takes as input an identifier from the '`urn:isbn`' namespace, and returns a list of ISBNs that are associated with the same work as the submitted ISBN (Hickley, Toves and Young, 2005). Such ISBNs may identify paperback or hardcover versions of a book, online or print versions, new editions, and so on. The mapping between the various ISBN identifiers is based on the FRBR model (IFLA, 1998). Identifiers and responses are interchanged using the HTTP protocol.

A similar approach has been taken by the recently approved info URI Scheme (Van de Sompel, Hammond, Neylon & Weibel, 2005), defining the '`info`' namespace. The info URI Scheme has been created to expedite the referencing by URIs of data resources that have identifiers in public namespaces but have no representation within the URI allocation. For various reasons many of the public identifiers used by the library and publishing domain are not valid URIs. The info URI Scheme provides a bridging mechanism to allow such public identifiers to become part of the URI allocation. Examples include, Library of Congress Control Numbers (LCCN), Digital Object Identifiers (DOI), National Library of Medicine PubMed identifiers, National Aeronautics and Space Administration (NASA) Astrophysics Data System Bibcodes, and identifiers minted by the repository of the Research Library of the Los Alamos National Laboratory.

For example, LCCNs are identifiers assigned by the Library of Congress to descriptive metadata records. As LCCN syntax existed prior to the Internet, LCCN identifiers are not valid URIs. As a consequence, LCCN identifiers do not exist naturally in the networked world because the Internet only recognizes URIs as a means to identify data resources. Based on the info URI Scheme, the data resource with LCCN '`n78-890351`', in a Web environment can be referred to as '`info:lccn/n78890351`'.

The effort to create the info URI Scheme emerged from the process to standardize the OpenURL Framework for Context-Sensitive Services (See Section 4 of Chapter 1) which requires the ability to describe data resources by means of globally unique identifiers. Both the info URI Scheme and the OpenURL Framework are guided by the National Information Standards Organization (NISO), a non-profit association accredited by the American National Standards Institute. Public namespaces declared under the info URI Scheme are regulated by a registration authority. The registry is available at [http://info-uri.info/]. Following the approach taken by the URN namespace, the info namespace exists merely for identification purposes, and adds no associated dereferencing or rich resolution mechanisms to a representation of the resource identified by the URI. Although, registered namespaces that reside under the info namespace, such as '`info:doi`', may provide rich resolution services on an individual basis.

Another namespace that resides under the info URI umbrella, is '`hdl`', the namespace for Handles (Sun, Lannom & Boesch, 2003). The handle namespace is developed and maintained by the Corporation for National Research Initiatives (CNRI) and steered by the publishing industry. Within the handle namespace, every handle consists of two parts: A naming authority, or prefix, and a unique local name under the naming authority, otherwise known as its suffix. For example, '`10.1045/june2005-bekaert`' is a handle for an article published in D-

Lib Magazine. It is defined under the Handle Naming Authority '`10.1045`', and its local name is '`june2005-bekaert`'. The uniqueness of a naming authority and a local name under that authority ensures that any handle is globally unique. Handles are resolved using the Handle System (Sun, Lannom & Boesch, 2003). Communication with the Handle System is carried out using the Handle System Protocol (Sun, Reilly, Lannom and Petrone, 2003). To enable resolving Handles in a Web-based environment, CNRI maintains a proxy server [http://hdl.handle.net/] that understands both the Handle System Protocol and the HTTP Protocol. A web browser may direct a handle to this proxy for resolution of the handle.

A core attribute of the Handle System is its support for multiple resolutions; that is, a single handle can resolve to multiple locations and services. For example, multiple locations that point to the same copies of a digital asset – for instance, to help balance loads when documents are in high demand – or multiple locations that point to the same asset, represented in a variety of data formats or value-adding services pertaining to the identified asset, and so on. Because of this, methods will need to be devised to make a choice between different options. At its simplest, a consumer may be provided with a list from which to make a manual choice. However, to enable an intelligent and automated selection from the list of locations, the association of a well-defined set of secondary data has to be provided during the resolution process.

It is fair to say that – overall – the Handle System is a reliable infrastructure providing expressive multi-resolution capabilities, and using stable and open protocols. Presently, several millions of handles have been issued, by various naming authorities. The Handle System has been well-integrated in several research activities, and in many applications, such as the DOI System (The International DOI Foundation [IDF], 2005) and DSpace (Smith et al., 2003; Tansley et al., 2003; Tansley et al., 2005).

From this, one may argue that handles are an outstanding candidate for identification and potentially could be adopted by all players in the information domain. The overall use of handles brings along rich resolution mechanisms, but more importantly, would support the interchange of all types of digital assets in a consistent and interoperable manner.

Nonetheless, handles will never oust other identifier namespaces. One of the main reasons is the fact that the use of an identifier scheme cannot be enforced. Different content owners use and have been using different identifier schemes, including legacy schemes. The choice of one scheme above another is influenced by cultural, political, organizational and/or technical motives. For example, the International Standard Audiovisual Number, ISAN, is a URN namespace used for the identification of audiovisual works; ISBN is a legacy scheme used for the identification of books; DOI is a system to identify intellectual property in an interoperable manner, and is mainly supported in the scholarly information domain; identifiers from the HTTP namespace are being used in the web environment, and so on. In order for these identification schemes to co-exist, a resolution framework must be devised that acknowledges the existence of such a 'world' of diversity. And such a resolution mechanism can only be established through the clear separation of identification and resolution processes. The latter has also been argued by Paskin (1999). As advocated by Paskin, disconnecting identification from resolution, and making resolution protocol-neutral, is highly desirable and would offer a structural sound solution that facilitates the evolutionary continuity from current to future protocols, and solves the claim for persistence of identification beyond location.

In this dissertation, a core set of interfaces will be devised to enhance interchange of digital assets across fairly heterogeneous repositories. Because such repository interfaces should be implementable across a broad variety of repository systems, they have to accommodate any type of identifier that is used for the identification of digital assets in a repository. Also, because only a few existing identifiers provide built-in resolution (with varying expressiveness), the interfaces can hardly rely on such built-in resolution protocols and mechanisms for providing services about the identified data in an interoperable manner. Instead, as will be shown, services will be requested, and built-in resolution is by-passed, using such technologies as the NISO OpenURL Framework for Context-Sensitive Services (NISO, 2005) and the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) (Lagoze, Van de Sompel, Nelson & Warner, 2003b).

3. The need for standardized interfaces for digital repositories

To illustrate the diversity of public interfaces in the current established digital repository landscape, a short overview of the interfaces of the aforementioned leading institutional repository systems is provided below. All of them provide access to stored digital assets; yet, different access interactions and diverse sets of technical standards are employed.

- ***DSpace*** (Smith et al., 2003; Tansley et. al, 2003; Tansley et al., 2005): A DSpace digital repository system is accessible via a Web user interface, and its underlying Application Programming Interface (API). The Web user interface enables human-based access by allowing consumers to search, browse and deposit digital assets and to perform workflow tasks on those digital assets. The Web user interface is implemented using both Java Servlets and Java Server Pages: Java Servlets receive incoming HTTP requests and handle the processing and business logic; and forward the request to a particular Java Server Page for display. In addition to this Web interface, DSpace supports the OAI-PMH v2.0 as a data provider. OAI-PMH support was implemented using OCLC's OAICat open-source software (Online Computer Library Center [OCLC], 2005a) to make Dublin Core descriptive metadata records (from DSpace Items) available for harvesting. Also, at the time of writing, efforts are ongoing at defining a Lightweight Network Interface (Stone, 2006) based on the WebDAV protocol (Goland, Whitehead, Faizi, Carter & Jensen, 1999) and tailored to the DSpace Data Model. However, it should be observed that, as WebDAV is an extension to the HTTP/1.1 protocol, and because the HTTP/1.1 protocol does not separate identification from location, the Lightweight Network Interface does not introduce a long-term solution to the resolution problem. This is in contrast with DSpace's long-term vision to serve as a repository system that also addresses long-term preservation concerns.

- ***GNU EPrints*** (Gutteridge, 2002; EPrints.org, 2005): Eprints is an open source digital repository system, developed at the University of Southampton, that allows researchers to 'self-archive' their publications. EPrints 2 (the second version of EPrints) runs as an apache module (using mod_perl), and uses MySQL to store descriptive metadata. The actual files are stored in a UNIX file system. EPrints 2 is accessible using various methods. First, a Web user interface is introduced to browse, search and deposit research articles and to perform review processes on those articles. Next, EPrints 2 also provides several repository interfaces that allow for machine-based access of the hosted materials. Descriptive metadata from an EPrints repository are made harvestable to agents using the OAI-PMH. By default unqualified Dublin Core is supported; but also more expressive descriptive metadata formats can readily be configured. Also, in version 2.3 of the EPrints software, the latest additions to the EPrints repository are made available via Really Simple Syndication (RSS) feeds.

EPrints also allows incremental harvesting of datastreams constituting the articles. To this end, a set of ad-hoc conventions are proposed that build on top of the OAI-PMH (Tourte & Powell, 2004). The datastreams themselves are exposed on a web server, and can be fetched using their HTTP location. However, as reported by Van de Sompel, Nelson, Lagoze and Warner (2004), the proposed techniques introduce methods that interfere with the generality and specificity of the OAI-PMH. The techniques assume the existence of a splash-page and require a harvester to be able to determine that it is harvesting from an OAI-PMH repository that follows the proposed convention.

The initial focus of the GNU EPrints software has been on the process of depositing articles and promoting discovery and access. However, if the EPrint content of these repositories is to continue to be made available into the future, the concept of long-term preservation needs to be bought into the equation. Again, it should be noted that addressing such functionality by heavily relying on web server related techniques may jeopardize the long-term vision.

- ***Fedora*** (Payette & Lagoze, 1998; Lagoze et al., in press): The Fedora digital repository system defines two APIs for obtaining digital assets and constituent datastreams from a Fedora digital repository: The Fedora-API-A and the Fedora-API-A-LITE. The former is implemented as a Simple Object Access Protocol (Gudgin, Hadley, Mendelsohn, Moreau & Nielsen, 2003), or shortly SOAP, enabled web service and defines a detailed interface for accessing digital assets (and parts thereof) stored in the Fedora repository. The access operations include methods to obtain XML-based packages of digital assets (or Fedora Digital Objects) from the Fedora repository (on a one-by-one basis), and to discover and request individual disseminations of datastreams of a Fedora Digital Object. The Fedora-API-A-Lite defines a lightweight version of the Fedora-API-A and is intended to support a REST-like style of access. Both APIs are closely interwoven with the Fedora data model. In addition, a Fedora digital repository system supports an OAI-PMH provider interface that allows sharing any descriptive metadata format available through the Fedora repository system via dissemination. The repository interface is implemented as a Java Servlet and can be accessed using the OAI-PMH.

Because of this diversity in public repository interfaces, creating cross-repository services and service workflows is very problematic. While the way in which each of those repositories internally manages its stored digital assets is of no importance, a common and – preferably – standardized way to access digital assets from each of those digital repositories is essential to easily allow for the emergence of rich and meaningful cross-repository services, repository federations, and automated workflow applications. Assuming the existence of such common repository interfaces, the following scenarios would become rather straightforward to realize.

- The transfer of digital assets from digital repositories to parties that provide discovery-oriented services over the digital assets. For example, the transfer of collections of scholarly publications from a variety of publishers to a party that provides aggregated search services, the transfer of subsets of various image collections to a service that builds a subject-oriented portal, the transmission of a collection of scholarly publications to a service that extracts bibliographic references from those publications and uses them to build citation indexes, and so forth.
- The transfer of digital assets from information producers to parties that provide digital preservation services. For example, the transfer of content to services charged with content migration and the transfer of collections to facilities that mirror the content to guarantee the existence of safety copies.
- The transfer of digital assets between members of a digital repository federation to gain geographic redundancy in pursuit of backup.
- The transfer of digital assets in a distributed scholarly communication workflow system. As reported by Van de Sompel, Payette, Erickson, Lagoze and Warner (2004), scholarly communication systems should no longer be conceived as autonomous vertical-integrated silos, but as a distributed, loosely coupled system of modular services. In such a new system, each service acts as a hub and provides a specific function pertaining to the scholarly communication process. One such hub could focus on the registration of a scholarly article, another hub is tasked with the certification of the validity of a registered scholarly claim, and yet another on the preservation of the registered article. It goes without saying that a necessary technical step in the development of such a new communication workflow system is the definition of protocol-based interfaces to enable interoperability among scholarly repositories, and between those repositories and the overlaying services.
- The submission of digital assets to a government agency for an official purpose such as the registration of copyright as, for example, was pioneered in the CORDS effort (Nierman, 1997).
- The interchange of digital assets between repositories and services from different application domains. For example, the transfer of digital assets from the digital library domain to repository services inherent to the e-learning system environment.

In recent years, several studies have been carried out – some of which are still under development – that aim at exploring such scenarios. Some of these studies try to tinker a solution that fits their own particular requirements; others can only reconfirm the urgency for the design and development of common protocol-based repository interfaces that support such scenarios as the ones stated above. A brief overview is provided below.

A number of frameworks have been created that focus on the definition of conceptual interfaces and workflow models for the interchange of data from content producers to digital repositories, and from digital repositories to content consumers. Among the most important are the OAIS Reference Model (ISO, 2003a) and the aforementioned Kahn/Wilensky framework for distributed digital object services (Kahn & Wilensky, 1995). Both models provide a basis for further implementation and standardization of interoperability repository interfaces and promote greater awareness of, and support of, interoperability requirements.

- The OAIS Reference Model – an international standard developed by the Consultive Committee for Space Data Systems – identifies, defines and provides structure to the conceptual relationships and interactions between information consumers on the one hand, and preservation-oriented repository systems (also known as OAIS or archival systems) on the other hand. Several pre-defined categories of access requests are distinguished, including so-called Order requests aimed at returning Dissemination Information Packages (DIPs), and Query requests aimed at generating Result Sets. The OAIS Reference Model also defines the conceptual methodology required for mutual interaction between archival systems (CCSDS, 2004a). To this end, a distinction is made between cooperating archival system, federated archival system and archival system that share technological resources. The essential requirement for each of those types of interaction is the existence of a set of mutual Submission Agreements and common technical interfaces that allow OAIS Dissemination Information Packages (DIPs) from one archival system to be ingested as Submission Information Packages (SIPs) in another. The OAIS Reference Model, however, is silent on the technical implementation of such interfaces. Yet, it is interesting to observe that the need for "*technical implementations of the OAIS DIP/SIP interfaces*" has been endorsed by the Joint Information Systems Committee, in the context of its Digital Repositories Programme, as one of many areas where additional standards or specifications are urgently needed (Heery & Anderson, 2005, Annex 2). A brief description of the OAIS Reference Model is provided in Section 1 of Chapter 1.
- The concept of a common interface to a repository of digital assets is also discussed in the Kahn/Wilensky framework for distributed digital object services. There, the conceptual Repository Access Protocol is introduced to allow consumers to request disseminations of a digital object by specifying its unique handle (Sun, Reilly and Lannom, 2003), a service request type, and a set of additional parameters.

Several research projects have explored the actual implementation of interoperable standards-based interfaces supporting the transfer of digital assets from content producers to digital repositories. Such projects include the Networked European Deposit Library (NEDLIB), the BIBLINK project and the Library of Congress's National Digital Information Infrastructure and Preservation Program (NDIIPP).

- The Networked European Deposit Library project (NEDLIB) aimed at defining a workflow for ingesting, storing and accessing digital assets in the context of deposit systems for electronic publications (van der Werf-Davelaar, 1999); and the BIBLINK project focused on establishing authoritative rules for transfer of descriptive metadata between publishers of electronic assets and national bibliographic agencies (Sutton & Clayphan, 1997). Based on a thorough examination of the – at that time – existing practices and enabling technologies, both projects concluded that it was unlikely that all (meta)data formats available on the market could be transferred through a single, standardized protocol-based framework. As a result, the BIBLINK project recognized the existence of a heterogeneous environment by identifying a rather extensive list of (meta)data formats and network protocols that should be supported by a national archive to facilitate transfer of descriptive metadata from publishers to libraries. And, in order to be able to ingest electronic publications into the deposit system, NEDLIB introduced the concept of a pre-processing interface that is tasked with retrieving publications from a publisher, and with repackaging it to the models required by the deposit system. Both projects felt that, under the existing circumstances, aiming for a single standardized transfer framework was unrealistic. The NEDLIB report formulates this as follows:

"*A 'pre-processing' interface is needed because deposit libraries cannot dictate submission formats to publishers. In principle, they have to accept all formats published on the market.*"

In hindsight, it is interesting to observe that several core technologies required for the standardization of a digital asset transfer framework were not yet available. Indeed, the BIBLINK project clearly identifies the need for a standardized packaging technique, but can only conclude that the one proposed in the Warwick framework (Lagoze, 1996; Lagoze, Lynch, Daniel, 1996) lacks maturity. In both projects, an interest in protocols with synchronization capabilities can be detected, but none is able to identify a protocol that meets the requirements. Also, both projects identify the need for ensuring authenticity and integrity of the exchanged information, but no technology can be selected that provides such guarantees across all packaging and transport techniques that are being considered.

- More recently, the Archive Ingest and Handling Test, a project of the Library of Congress's NDIIPP program, was set up to test the feasibility of transferring digital collections from one institution to another. As reported by Shirky (2005), the purpose was to test the stresses involved in the wholesale transfer, ingestion, management, and export of a relatively modest digital collection, whose content and form are both static. The test is designed to assess the process of ingest and export, and to identify areas that require further research or development.

  One of the participants, John Hopkins University, focused on the bulk ingest and export of the sample collection in such institutional repository systems as DSpace and Fedora (DiLauro, Patton, Reynolds & Sayeed, 2005). As shown above, each of these institutional repository systems comes with its own public interfaces. To this end, researchers from John Hopkins University developed a repository-agnostic 'layer' tasked with remodelling the data of the sample collection to a data format supported by both DSpace and Fedora, and passing on the remodeled information to the respective repository APIs. By adding such a 'layer', downstream application may interact with a repository without knowing with which repository system (notably DSpace or Fedora) it is interacting; yet, it also moves the burden of providing interoperability from the repository to the added infrastructure. Indeed, as reported by DiLauro et al. (2005), the 'abstraction layer' accesses the DSpace repository using the DSpace API. The Fedora repository is accessed using the SOAP-based Fedora API, and requires that all datastreams be fetched using the HTTP protocol. Consequently, to enable ingestion of the Fedora repository system, the sample data had to be placed behind a web server.

  While the proposed solution may still be manageable in the context of two repository systems, it proves to be increasingly laborious when more repository systems are added.

Also, currently, several studies investigate network-based workflow systems that are able to capture digital information from heterogeneous digital repositories, via public interfaces, and make it accessible, and process it through service-oriented value chains. Such studies include the Pathways project ("Pathways," n.d.; Bekaert, Liu, Van de Sompel, Lagoze, Payette, & Warner, in press) and JISC's e-Framework for Education and Research (Olivier, Roberts & Blinco, 2005).

- The Pathways project – officially called Lifecycles for Information Integration in Distributed Scholarly Communication –, is a recent project carried out by Cornell University and the Los Alamos National Laboratory that aims at exploring broadly applicable models and protocols to support a loosely-coupled, highly distributed, interoperable scholarly communication system ("Pathways," n.d.; Van de Sompel, Payette, Erickson, Lagoze and Warner, 2004). As stated by Van de Sompel et al., the task of implementing a new scholarly communication system holds many complex technical and organizational challenges. While many new repository systems are emerging, they tend to offer little or no interoperability among them. A necessary technical step in the development of a scholarly communication workflow is the development of protocol-based interfaces that enable interoperability among existing repositories, information stores, and services.
- The need for common ways to interchange digital information has also been raised in the context of JISC's e-Framework for Education and Research (Olivier et al., 2005). The e-Framework is an initiative by the UK's JISC and Australia's Department of Education, Science and Training to produce an evolving and sustainable standards-based, service-oriented technical framework to support the education and research communities. The initiative builds on two earlier initiatives, the JISC e-Learning Framework ("JISC E-Learning Programme," 2005) and the JISC Information Environment ("JISC Information Environment," 2003), as well as other service-oriented initiatives in the areas of scholarly information. Each service defined by the Framework is envisaged as being provided as a networked service within an organization through a public interface that other systems can call on and utilize. Again it is recommended that such interfaces are protocol-based (Web service or REST style) and conform to a set of standardized specifications.
- In the context of the JISC Digital Repositories Programme (Heery & Anderson, 2005), several scoping surveys – formally called Linking UK Repositories Scoping Studies – are being undertaken to identify sustainable technical and organizational models to support user-oriented services across digital repositories ("JISC ITT," 2005). These surveys will inform the future funding and development of user-oriented services by the JISC from mid 2006 onwards. An important part of the questionnaires focuses on interoperable protocol-based interfaces and the practicality for a repository of providing such interfaces.

4. Interacting with digital repositories

A first step in exploring common public interfaces for digital repositories lies in trying to conceptualize the highly complex and varied ways in which end-users (or consumers), on the one hand, and downstream applications (or businesses), on the other, interact with a digital repository environment. Interactions between end-users and digital repository systems are also referred to as business-to-consumer interactions; interactions between applications and repository systems may be called business-to-business interactions. Attempts so far at describing such conceptual interactions – in the scholarly information domain – have not been undertaken. Yet, it is worth mentioning that the aforementioned JISC e-Framework for Education and Research (Olivier et al., 2005) and the Digital Library Federation Abstract Services Taskforce (Dempsey & Lavoie, 2005) are both working on the definition of reference models that describe the set of services used in a particular application area. Most likely, an important facet of their research will be the categorization of types of interaction between digital repository services and consumers and/or businesses. Awaiting the results of these activities, a tentative list of core interactions that are likely to be supported in a digital repository environment is provided below. The list is partially inspired by Powell's overview of abstract services in the context of the JISC Information Environment (Powell, 2005). In order to fulfill the requirements imposed by downstream consumers, each of those services may be used selectively in combination.

□ *Browse*: A browse-interface provides a browsable list of hyperlinks that a consumer or business application can follow to access digital assets. In the digital library domain, browsing is an important way for a consumer to discover information. For example, most electronic articles include a list of citations to other articles. And each citation is presented as a hyperlink to the full-text of the referred article. A consumer can browse from one article to another by clicking on such hyperlinks.

□ *Deposit*: A deposit-interface initiates an ingest process by providing a public interface through which digital assets and/or descriptive information can be deposited.

□ *Delete*: A delete-interface provides a mechanism through which digital assets, stored in a digital repository, can be deleted.

□ ***Harvest*: A harvest interface provides a mechanism through which batches of digital assets can be harvested, that is, gathered on a regular basis.** Of specific importance is the concept of so-called selective harvesting. **Selective harvesting allows downstream businesses applications to harvest only those assets that were created or modified within a specified date range.** For example, an application may be interested in harvesting the newly added digital assets of a repository only.

□ ***Obtain*: An obtain interface enables requesting disseminations of a digital asset and its constituents datastreams on a one-by-one basis. Hereby, a dissemination is a view of a digital asset or one of its datastreams that is generated by a service operation.** Both the service operation and the digital asset (or constituent thereof) are identified using a unique identifier. The result of applying the service operation is a media typed bit stream. Of specific interest in current digital repository systems are service operations for the generation of documents or images in various formats or resolutions.

□ *Search*: A search-interface accepts a structured query request and issues a result set in response. A result set is a set of descriptive records for those digital assets in a repository which match the criteria stated in the structured query. In recent years, several (standardized) search protocols have been specified, including the ANSI/NISO Z39.50 Information Retrieval Protocol (NISO, 2002), the Search and Retrieve URL Service (SRU) and the

Search and Retrieve Web Service (SRW) (The Library of Congress [LC], 2005d). At the heart, both the SRU and SRW use the Common Query Language for representing queries. The query syntax used by the Z39.50 protocol is the Reverse Polish Notation.

When requesting digital materials, methods for controlling who has access to these materials are very important. When accessing a digital repository, a two-step process of identification usually takes place. The first is authentication which establishes the identity of a consumer or business application. The second is authorization which determines what a consumer or business application is authorized to do.

Most digital libraries and repositories of established scholarly publishers implement access policies at the level of the digital repository. This can easily be achieved by restricting access permissions to IP addresses, IP address ranges, or connection-specific DNS suffixes. Or one could employ cross-repository authentication and authorization solutions, such as Shibboleth, that enable organizations to build single sign-on environments (Erdos & Cantor, 2002). In such environments, an organization remains responsible for authenticating the consumer, that is, for checking that the credentials the consumer presents are correct. The decision to authorize access to data is the responsibility of the actual owner of the data, and is based on the consumer's information.

As described by Arms (2001), if access management is implemented at the repository or organizational level, access is effectively controlled locally. Once the data leaves the repository environment, it is hard for the original manager of the digital repository to retain effective control. Numerous copies of the data are generated in networked computers, including caches, mirrors, and other servers, and are distributed beyond the control of the local repository environment. To date, most digital libraries have been satisfied to provide access management at the cross-repository level, while relying on social and legal pressure to control subsequent use. However, in an era of regulatory restrictions, and particularly in market drive information communities, content providers are often concerned that the distributed nature of current information environment lessens their control on the data and this lack of control could damage their revenues.

Therefore, more recently, techniques are being developed and standardized that enable managing data after the data has left the secured and controlled digital repository environment. A technique that has received quite some approval is the practice of secure packages. In this technique, digital data is delivered to an end-user in a package that contains secondary data about access policies. In addition, some or all of the datastreams in the package can be encrypted. In order to access the information, a user must meet the access policies and acquire a digital key which might be received from an electronic payment system or other method of authentication is needed.

Presently, several rights expression languages and intellectual property management tools are under development that enable defining and enforcing such package-level restrictions. Such technologies include rights expression languages as the MPEG-21 REL (and its predecessor XrML), Creative Commons and ODRL, intellectual property enforcement tools as the ones proposed by MPEG-21 IPMP and encryption tools as W3C XML Encryption and Processing. Yet, until recently, the impracticability to tentatively explore digital datastreams embedded in such packages, and the immaturity and technical complexity of such technologies proved to be a barrier for widespread adoption.

5. Scope of this dissertation

This dissertation focuses on the design and definition of persistent, standards-based *read-only* interfaces to repositories. Two interfaces are considered: An interface to harvest batches of digital assets (i.e., harvest interfaces) and an interface to obtain disseminations of digital assets and their constituent datastreams on a one-by-one basis (i.e., obtain interfaces). This dissertation does neither address *write* interfaces, that is, interfaces for the deposit of digital assets in a repository, nor interfaces for deleting digital assets. Also, the exploration of *discovery* interfaces to browse and search digital repositories is beyond the scope of this study. It was felt by the author that interfaces to browse and search should not be regarded as core service interfaces that need to be supported by a digital repository. Rather, browse and search discovery solutions can be implemented as autonomous services that overlay one or more digital repositories and that are created through interaction with truly core repository interfaces for harvesting and obtaining. Once harvested, digital assets can be stored into a remote aggregator so that a search or browse interface can be built on top of the retrieved assets, or so that – once value has been added – the harvested material can be re-exposed for browsing, harvesting or searching.

For completeness, it should be mentioned that, in the setting of this doctoral study, little or no attention will be given to access control. While access control is believed to be a crucial facet of the overall access function of a digital repository, it is felt that access control can be implemented as a separate layer that operates on top of the proposed interfaces. Access policies could be implemented either at the level of the repository itself (e.g., by using such cross-repository access middleware as Shibboleth), or at the level of the individual packages themselves, by inserting proper access rights and access authorization information.

The harvest interfaces and obtain interfaces advanced in this study have been designed with the following requirements in mind:

- The interfaces should be protocol-oriented as they are to be used in a distributed networked environment.
- The interfaces should operate in manner that is neutral to the way in which a digital repository internally stores, manages, and preserves its information. Each repository system – including legacy systems – should be able to join the interoperability environment enabled by the interfaces.

  First, the interfaces should operate in a manner that is neutral to the data models used by the repositories to store digital assets.

  Second, the interfaces should operate in a manner that is neutral with regard to the type of identifier that is used for the identification of interchanged digital assets. This allows the interfaces to be implemented for a broad variety of repository systems, in which, for cultural, political, organizational and/or technical reasons, different choices are made regarding identification schemes.

Third, the interfaces should also uphold the distinction between various versions of data, insofar as versioning strategies are supported by the digital repository that exploits the interface (for example as part of a repository's preservation strategy).

- The interfaces should be defined in a generic manner, that is, concepts underlying the interfaces should persist over time, while their implementation may change over time as technologies evolve. The introduction of interfaces that are conceptually persistent over time resonates well with the long-term perspective of digital repositories.
- The interfaces should preferably build on existing standards and technologies. The use of existing technology may lower the barrier for adoption, as implementers may have acquired experience in working with those technologies; and the availability of stable reference software can be guaranteed.
- The interfaces should be expressible as lightweight interoperability specifications. A lightweight specification is easy-to-implement and hence cost-effective, and as a result has a high potential for broad acceptance. Examples of such specifications include RSS (Berkman Center at Harvard Law, n.d.), OAI-PMH (Lagoze et al., 2003b) and OpenURL 0.1 (Van de Sompel, Hochstenbach & Beit-Arie, 2000).

As will be shown, the interfaces proposed in this thesis are based upon a set of technologies that have recently emerged from and have successfully been adopted by the digital library and scholarly information communities. These technologies are the OAI-PMH, the OpenURL Framework for Context-Sensitive Services and several XML-based techniques that can be employed for the representation and packaging of compound digital assets. A framework built on the *combination* of such new technologies may bring us closer to having the ability to devise a set of public repository interfaces that holds great potential for the deployment of a federated content framework that allows for the use, reuse and transfer of content.

6. Structure of this dissertation

This dissertation describes both experimental and theoretical aspects of interfaces for harvesting and obtaining assets from digital repositories. Earlier versions of the chapters of this dissertation have been published, in whole or in part, over recent years, and were also delivered as lectures. While the author has modified, often substantially, all of the texts that reappear here from earlier publications, he nonetheless notes their significance in the development of his work. Among the most important, in the context of this thesis, are (Bekaert, Hochstenbach & Van de Sompel, 2003; Bekaert, Balakireva, Hochstenbach & Van de Sompel, 2004; Bekaert & Van de Sompel, 2005a; Bekaert & Van de Sompel, 2005b; Bekaert, Liu & Van de Sompel, 2005; Van de Sompel, Bekaert, Liu, Balakireva & Schwander, 2005; Bekaert, De Kooning & Van de Sompel; 2006).

This dissertation is divided into three Chapters. A first Chapter provides a brief overview and description of the building blocks needed to the devise the respective repository interfaces. It also highlights and discusses such technologies as the OAIS Reference Model, MPEG-21 Digital Item Declaration (MPEG-21 DID), the OAI-PMH, and NISO's OpenURL Framework for Context-Sensitive Services. Chapter 2 reports on two elaborate experiments that have been conducted over the past three years. Both experiments lay the foundation for the thinking on interfaces that underlies the theoretical interfaces presented in Chapter 3. A first experiment focuses on the multifaceted use of the OAI-PMH and NISO's OpenURL as information access protocols in a repository architecture designed and implemented for ingesting, storing, and accessing a vast collection of digital assets at the Los Alamos National Laboratory (LANL). A second experiment aims at designing and implementing a robust solution for the recurrent and accurate transfer of newly added and updated digital assets from the collection of the American Physical Society to the LANL digital repository environment. Again the OAI-PMH plays an important role. Based on the expertise on interfaces resulting from both experiments, a set of generic read-only interfaces is presented in Chapter 3. The study concludes with a summary of contributions, a discussion on the possible impact, and an exploration of the potentials it offers for future research.

Ch.1. Basic concepts and enabling technologies

Ch.1. Basic concepts and enabling technologies

Finding the right terms to describe topics in the fields of digital library and information science is complicated. The library and information science domain borrows concepts and technologies from many different disciplines, each of which brings its own jargon. Therefore, throughout this dissertation, terms are used carefully and attempts are made to clarify their meaning to avoid ambiguity.

This first Chapter introduces the building blocks that are of importance for devising the harvest and obtain interfaces explored in this doctoral study. The building blocks are listed below and depicted in Figure 1.

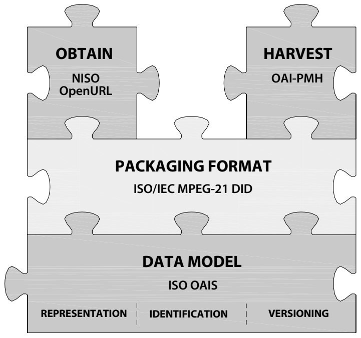


Figure 1. Core building blocks of the harvest and obtain interface

- ***A data model***. There is need for an abstract model that offers a set of well-articulated concepts and vocabularies that help framing the interface problem in a larger repository context. In addition, this model should provide guidance on such concepts as content representation, content identification and content versioning.
- ***A packaging format***. There is need for a packaging format to represent digital assets exposed by the repository interfaces by means of a wrapper document. A packaging format has the ability to convey datastreams and secondary data – including descriptive data, rights data, technical data – pertaining to a datastream. It has the ability to represent both simple digital assets (consisting of a single datastream), and complex digital assets (consisting of multiple datastreams). This packaging format should be tailored in accordance with the (representation, identification and versioning) concepts of the above data model.
- ***A harvest protocol framework.*** There is need for a protocol framework to construct interfaces that expose packaged digital assets for incremental harvesting.
- ***An obtain protocol framework.*** There is need for a protocol framework to define interfaces that enable requesting disseminations of a digital asset (and/or constituent datastreams thereof) on an individual basis. A dissemination is a view of a digital asset generated by a service operation. Both the digital asset (or constituent datastream thereof) and the service operation are uniquely identified.

For each building block, a choice of technologies may be available. The technologies chosen for this dissertation have been selected with the following two criteria in mind:

- ***Persistence***. The technologies should preferably be defined in a manner that supports persistence. That is, a technology should consist of an abstract model that persists over time, while implementations of the model may change as the technological environment evolves. Again, this is dictated by the long-term perspective of digital repositories in general, and the postulated interfaces in particular.
- ***Buy-in***. The technologies should preferably already have received buy-in from the concerned information communities. The use of broadly used technologies as the basis for the creation of the interfaces may help leverage the adoption of such interfaces. Also, the existence of software and the fact that many implementers have acquired experience in working with those technologies may lower the threshold for initial implementation of the interfaces.

For each of the building blocks, a technology has been selected. It is important to note that the reasoning applied in this study allows for the use and insertion of many alternative technologies in the jigsaw of Figure 1, as long as these technologies support the core concepts – with regard to content representation, content identification and content versioning – upheld by the data model. By building on the combination of these technologies, standards-based repository interfaces for harvesting and obtaining assets can be devised. An overview of selected technologies and possible alternatives is provided below. Each of the selected technologies is described in detail in the remainder of this Chapter.

- ***A data model: ISO OAIS Information Model***. The author has opted for the Open Archival Information System (OAIS) Reference Model as the theoretical data model. Of particular importance in this study is the way how the OAIS Reference Model deals with such concepts as content representation, content identification and content versioning. The Model will be used as frame of reference for reconciling heterogeneous and disparate informational and functional concepts that are common in digital library research and practice. Such a frame of reference will prove to be essential when testing implementations of the theoretical repository interfaces in actual repository systems (See Section 2 of Chapter 3).

  The OAIS Reference Model is an ISO standard that is unique in its kind. The OAIS Reference Model is very comprehensive: It defines a Functional Model that outlines the range of functions that need to be undertaken by a repository, and it describes an Information Model that offers guidance on data modeling in all of its facets. Although the initial focus of the OAIS Reference Model is on archives tasked with the long-term preservation of digital information, the Model has proven to be fruitful for discussing matters related to all kinds of digital repositories. The OAIS Reference Model is briefly discussed in Section 1 of this Chapter.
- ***A packaging format: ISO/IEC MPEG-21 Digital Item Declaration.*** The author has selected the MPEG-21 Digital Item Declaration (MPEG-21 DID) as the packaging format. MPEG-21 DID is very suitable because of the existence of a well-specified Abstract Model, and the capability this yields for mapping and representing digital assets in various

evolving technological environments while maintaining the basic architectural concepts. In this doctoral study, this Abstract model provides a helpful bridge between the theoretical (representation, identification and versioning) concepts of the OAIS Information Model and the MPEG-21 DIDL XML syntax used to implement the actual packages. Also, MPEG-21 DID is appealing because it is part of the MPEG suite of ISO standards which is likely to receive strong industry backing.

It should be noted that in recent years, several other packaging formats have emerged from various information communities. Such packaging formats are METS (LC, 2005a), IMS-CP XML Binding (IMS Global Learning Consortium, 2003b), and XFDU (CCSDS, 2004b). Some of these technologies, like METS, have been warmly embraced by the digital library community. The MPEG-21 work has clearly received less attention in the digital library community. This lack of attention is possibly due to lack of clarity on patent statements and the little if any participation of members of the digital library communities in market driven international standardization bodies like MPEG and JPEG. Yet, it is felt by the author that the technical quality and straightforwardness of MPEG-21 DID outweighs this lack of current buy-in. The author also records that established alternative approaches, such as METS, did neither succeed in stabilizing their technology – the most recent published version dates from December 2005, and a new version is again in the making – nor in providing solid reference software. It is the opinion of the author that this lack of stability should be resolved by capturing and endorsing core concepts of the METS packaging format by means of an abstract model.

MPEG-21 DID is described in Section 2 of this Chapter. A mapping of the OAIS Information Model to entities of the MPEG-21 Abstract Model is provided in Section 2.6.

- ***A harvest protocol framework: Open Archives Initiative Protocol for Metadata Harvesting.*** A well-established framework for harvesting materials in the digital library community is the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) (Lagoze et al., 2003b). The OAI-PMH allows an aggregator to harvest materials from compliant repositories using a datestamp-based harvesting strategy. While typically, the OAI-PMH is used to harvest XML-based descriptive metadata records, like Dublin Core or MARCXML, in this dissertation, the OAI-PMH will be employed as a means to harvest XML-based packages of digital assets. These packages must be conformant with the aforementioned XML-based packaging format (that is MPEG-21 DIDL). Also, as will be shown later on, the OAI-PMH can be readily combined with the packaging, identification and versioning concepts of the OAIS Reference Model.

  It should be observed that the OAI-PMH framework is not specified in a persistent manner. As will be described in Section 3 of this Chapter, the OAI-PMH heavily relies on the HTTP protocol and the XML syntax for transporting and serializing the interchanged materials. While this approach proves to be satisfactory in the current technological environment, it may prove to be inadequate as technologies evolve. To remedy this shortcoming, an abstract model would have to be defined in a manner that is neutral to the OAI-PMH transport protocol (HTTP) and OAI-PMH response format (XML).

  However, the OAI-PMH has received a high level of buy-in from the digital library community and has established itself as an important solution for exchanging descriptive metadata among a large and varied group of digital repositories. It is thus intriguing to consider using existing installations of the OAI-PMH (that expose descriptive metadata) as a leverage to promote the widespread adoption of content harvesting interfaces.

Another recent XML-oriented harvesting protocol is the Really Simple Syndication 2.0 (RSS) (Berkman Center at Harvard Law, n.d.). RSS originated in the weblogging community and is a format for syndicating content of news-like web sites. Such web sites include major news sites like BBC – RSS feed available at [http://newsrss.bbc.co.uk/rss/newsonline_world_edition/front_page/rss.xml], community-oriented sites like Slashdot – RSS feed available at [http://rss.slashdot.org/Slashdot/slashdot] – and personal weblogs. An RSS aware aggregator is populated by checking the RSS feeds for newly added or updated items, and typically displays those items on the browser of a consumer.

Both the OAI-PMH and RSS 2.0 provide selective harvesting functionality. Though, in contrast with the OAI-PMH, RSS does not support harvesting items within a pre-defined datetime range; nor does it support grouping items in sets. However, at the time of writing this dissertation, Microsoft has developed an extension, called Simple Sharing Extension (Ozzie, Moromisato & Suthar, 2006), that aims at meeting such synchronization requirements and that enables loosely structured applications to cooperate. Given the growing popularity of RSS, and assuming the successful adoption of Microsoft's plug-in, one could easily imagine building a harvesting framework by replacing OAI-PMH with RSS 2.0.

- ***An obtain protocol framework: OpenURL Framework for Context-Sensitive Services***. The author has selected the NISO OpenURL Framework for Context-Sensitive Services (NISO, 2005) as the basis for the obtain interface. The OpenURL Framework enables defining a service environment, in which packages of information are transported over a network. These packages have an identifier of a referenced digital asset at their core, and they are transported with the intent of obtaining context-sensitive services pertaining to the referenced asset. To enable the recipients of these packages to deliver such context-sensitive services, each package describes the referenced asset itself, the type of service, the network context in which the asset is referenced, and the context in which the service request takes place.

  The NISO OpenURL Framework is a recent NISO standard that is clearly unique in its kind. The Framework has been defined in a persistent manner: A clear distinction has been made between abstract definitions of OpenURL concepts and their concrete implementation. The OpenURL 0.1 specification (Van de Sompel, Hochstenbach & Beit-Arie, 2000) – the precursor of the current NISO OpenURL standard – has received a high level of buy-in from the scholarly information community. The technology has been adopted by information providers as Elsevier, PubMed, JSTOR, and ISI and content aggregators such as Google Scholar, and Microsoft's Windows Life Academic Search. A detailed description of the OpenURL Framework is provided in Section 4 of Chapter 1.

  Again, it could be argued that alternative technologies can be employed as the basis for the obtain interface. Among the most important is the HTTP/1.1 extension for Web-based Distributed Authoring and Versioning (WebDAV). Yet, it is believed by the author that technologies like WebDAV, do not provide an adequate level of persistency and functionality compared to the one offered by the NISO OpenURL Framework. The following two consideration can be made regarding the WebDAV technology:

First, WebDAV relies on identifiers from the '`http`' namespace to identify its Web resources. As a consequence, the WebDAV interface cannot operate in a manner that is neutral to the type of identifier being employed by a repository, and hence, ignores the existence of resources that have been appointed identifiers of other (that is, non-'`http`') namespaces. For example, one identifier namespace that has been well-integrated in the scholarly information domain is '`hdl`', the namespace for Handles. As such, it is interesting to observe that, in the context of the WebDAV Lightweight Interface (Stone, 2006) of the DSpace repository system, to work around this problem, persistent identifiers of the '`hdl`' namespace are mapped to WebDAV resource URIs from the '`http`' namespace using a proprietary process. While such an approach may be feasible in the setting of a DSpace repository environment, it is clearly inadequate in a repository environment that uses an unlimited and diverse set of identifier namespaces. Moreover, the mere fact that WebDAV relies on (mapped) identifiers that are different from the native identifiers of the Web resource may jeopardize the overall access interoperability across repositories. Indeed, a repository system may support different access interfaces. One interface could be based on WebDAV, another on NISO's OpenURL Framework, and a third on the SRU/W search protocol. It order to sustain a basic level of cross-repository interoperability, no matter which interface being employed, the same unique identifier should be used for requesting a digital asset. Preferably, this identifier should be natively attached to the asset and be interface-neutral.

Second, the WebDAV is an extension of the HTTP/1.1 protocol that adds methods to perform remote web content authoring operations. Such operations include the ability to create, remove, and query information about Web resources. WebDAV does not offer off-the-shelf methods that enable requesting rich services pertaining to a digital asset. But, WebDAV allows for such an environment to be created by extending the current specification using proprietary methods, and for example, by adding proprietary arguments on the WebDAV query structure. Also, in contrast with the OpenURL Framework, WebDAV does not define a pre-defined set of concepts that enable conveying context-sensitive information, such as information on the network context and the context in which the request has taken place.

1. A Reference Model for an Open Archival Information System

One of the greatest challenges facing the information scientists in the twenty-first century will be the long-term preservation of digital data. While there has been an awareness of digital preservation problems for some years, their significance have recently been magnified by the rapid growth in the use of the internet and the creation of millions of (online available) transitory digital assets. A well-described list of pressing preservation issues can be found in a recent survey by Rosenthal et al. (2005).

In response to the preservation problem, several initiatives have been established. In December 2000, the US Congress appropriated $100 million for a national digital-strategy effort (viz. NDIIPP), led by the Library of Congress. Also, in January 2001, The Digital

Preservation Coalition was established to foster joint action to address the urgent challenges of securing the preservation of digital resources in the UK. Most likely, many other investments in modeling and testing various technical solutions will take place over the next years, resulting in recommendations about the most viable and sustainable options for long-term preservation.

One very relevant piece of prior work for thinking about long-term preservation problems is the Reference Model for an Open Archival Information System (ISO, 2003a). This Reference Model is an attempt to provide a high-level framework for the development and comparison of digital archives that have accepted the responsibility to preserve data. Its development was coordinated by CCSDS and guided by NASA as part of an International Organization for Standardization (ISO) initiative to develop standards that support of the long-term preservation of satellite and GIS data. Despite its origins in the space data community and its initial application to satellite and GIS data, the OAIS Reference Model has attracted widespread interest in the digital library community.

The Reference Model for an OAIS defines a common framework that can be used to help understand archival challenges, especially those that relate to digital information. Its value in the context of this dissertation, is that it offers a set of well-articulated concepts and a comprehensive, if daunting, vocabulary that facilitates communication and discussion across the different communities interested in digital preservation. The OAIS model defines a Functional Model and an Information Model. The Functional Model outlines the range of functions that would need to be undertaken by an archive, such as access, archival storage, and ingest. The Information Model defines broad types of information that would be required in order to preserve and access the information stored in an archive. Both the Functional Model and the Information Model define useful abstract concepts but not a blueprint for an implementation of an archival system. The organizational and technical choices of how to implement the abstract OAIS concepts in actual concrete environments is left to the communities involved.

1.1. Information Model for an OAIS

The Information Model for an OAIS describes the different types of information that are exchanged and managed within an OAIS. In order to preserve information, an OAIS must store significantly more than just the datastreams of the digital asset it is expected to preserve. This subsection describes the various types of information required for long-term preservation; some of them are depicted in Figure 2. Again it is worth repeating that each of those information types are conceptual and do not imply any specific implementation.

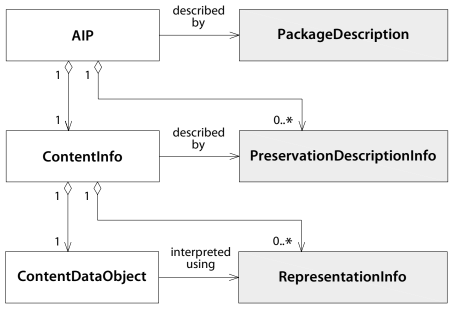


Figure 2. Archival Information Package, Content Information, and Content Data Object

The *Content Information* is defined to be that information that is the original target of preservation in an OAIS environment. In the library and information science domain, Content Information is referred to as a digital asset or a digital object.

Content Information binds *Representation Information* to a *Content Data Object*. The former is secondary information that supports the actual representation and rendering of the Content Data Object. The latter represents the actual data files constituting the Content Information. It is fair to state that these files are the actual datastreams of the Content Information.

The *Preservation Description Information* is information necessary to adequately preserve Content Information. The Preservation Description Information can be further broken down into four sub-categories:

- Reference Information provides one or more assigned identifiers for the Content Information, each of which is named a Content Information Identifier.
- Context Information documents the relationships of the Content Information to other Information Objects that reside elsewhere in the environment.
- Provenance Information documents the history of the Content Information, including changes that may have taken place since the Content Information was originated, and who has had custody of it since it was originated. This provides some assurance as to the likely reliability, so that consumers of the Content Information can better judge how much to trust the data.
- Fixity Information provides data integrity checks used to ensure that the particular Content Information has not been altered over time in an undocumented manner. This includes the provision of digests and digital signatures.

It is worthwhile noting that, in March 2000, the Online Computer Library Center and the Research Libraries Group sponsored the creation of a working group to explore consensus building in the area of preservation metadata and to develop a more detailed model of Preservation Description Information that would accommodate the needs of the library community. The results of this working group's efforts in that regard are reported in *Preservation metadata and the OAIS information model* (OCLC/RLG Working Group on Preservation Metadata, 2002). In June 2003, a second working group – called Preservation Metadata Implementation Strategies (PREMIS) – was formed to address implementation issues associated with preservation metadata. PREMIS completed its activities in May 2005 with the release of the PREMIS Data Dictionary 1.0 (Preservation Metadata Implementation Strategies Working Group [PREMIS], 2005) and corresponding XML Schema serializations [http://www.loc.gov/standards/premis/schemas.html].

Another key concept of the OAIS Information Model is the *Information Package*. An Information Package is a container that binds Content Information to Preservation Description Information. The Information Package also holds local hooks (often called 'Fragment Identifiers') to allow accessing each datastream of which the Content Information consists. The physical syntax of an Information Package is called *Packaging Information*. The latter conforms to with a *Packaging Format*.

An important attribute of the Information Package is the *Information Package Identifier*; an identifier pertaining to the Information Package itself. The value of this identifier must be unique within a (federation of) archive(s). In contrast with the Content Information Identifier (which resides under the Reference Information sub-category of the Preservation Description Information), the Information Package Identifier is an internal property of the archival system, and hence, is typically not exposed to businesses or consumers. An overview of the various levels of identification recognized by the OAIS Information Model is depicted in Figure 3.

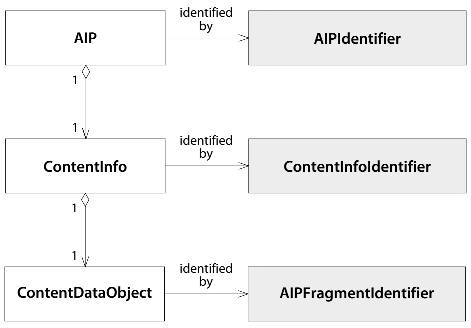


Figure 3. AIP Identifier, Content Information Identifier and AIP Fragment Identifiers

The Information Model for an OAIS recognizes three subtypes of the Information Package: The *Archival Information Package* (AIP), the *Submission Information Package* (SIP), and the *Dissemination Information Package* (DIP). The definitions of these Information Package types are based on the function of the archival process, which uses the Information Package, and the translation from one Information Package to another as it passes through the archival processes. A distinction is made between Information Packages that are submitted to an archive (i.e., SIPs), Information Packages that are subsequently stored and preserved by an archive (i.e., AIPs), and those that are disseminated from an archive (i.e., DIPs). This distinction is needed to reflect the reality that some submissions to an archive will have insufficient information to meet final requirements of that archive. In addition, different archives may organize their data very differently, and hence, may warrant different environment-specific information to be contained within the AIPs being stored. In order for archives in a federation to be able to exchange Information Packages they must support at least one common DIP/SIP Format.

According to the OAIS Reference Model, whenever the Content Information or its Preservation Description Information is updated, a new AIP must be created. Dependent on the nature of the update, a distinction is made between several types of AIPs. Of special interest for this study are the notions of Version and Edition. A *Version* is an AIP that results from applying a transformation, for example induced by a preservation strategy, on the Content Information of a source AIP. An *Edition* is an AIP that results from increasing or improving the Content Information of a source AIP; for example by removing a few typographic errors from one of the constituents of the Content Information. Both an AIP Version and an AIP Edition are candidates to replace the source AIP from which they are derived. No matter which type of update the Content Information or the Preservation Description Information of a source AIP undergoes, the result is a new AIP that receives a new, unique AIP Identifier. Note that in both cases, the Preservation Description Information needs to be updated to provide information about the source AIP, and to describe what was done and why. The Content Information Identifier could remain untouched.

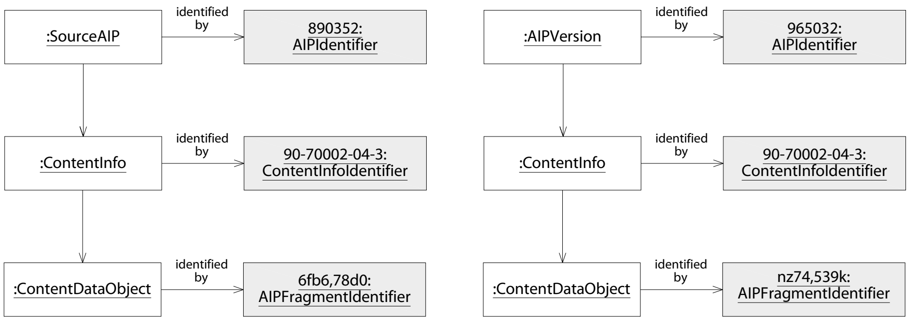


Figure 4. Archival Information Package Versions

Because it may be of historical interest, especially in the context of digital preservation, to retain previous Versions or Editions of AIPs, it results that, within a single archive, several versions of specific Content Information may exist. All these versions share a Content Information Identifier, yet have a different AIP Identifier. Each version of specific Content Information can be retrieved using the AIP Identifier of the AIP that encapsulates the particular version. An example of a source AIP and an AIP Version are depicted in Figure 4. As can be seen, both AIPs share the same Content Information Identifier (viz. the ISBN number '`90-70002-04-3`'); yet each AIP is identified using a different AIP Identifier (viz. the identifiers '`890352`' and '`965032`', respectively). The use of AIP Identifiers to address various versions of content will prove to be vital in the setting of this dissertation.

For completeness, it should be noted that the OAIS Reference Model also recognizes digital migration processes that do not involve creating new AIPs. Such processes include Refreshment, Replication, and Repackaging. These processes do not affect the Content Information or Preservation Description Information. This does not mean that an OAIS does not track such migrations; rather it is not required to update the Preservation Description Information as part of such tracking. If these processes are carried out entirely within Archival Storage, the AIP Identifiers remain the same and there is no implied impact for accessing the migrated AIPs.

1.2. Functional Model for an OAIS

Within the OAIS Functional Model, six primary functional components are identified. The components manage the flow of data from content producers to an archive, and from an archive to consumers. Taken together, they identify the key processes endemic to most systems dedicated to preserving digital information. It is likely that a digital archive will contain functional components similar to those described below, although the specific implementation will differ from archive to archive.

- *Ingest*: This component provides the interface between the archive and the producer. It accepts SIPs from producers during a Data Submission Session, and prepares the retrieved data for storage and management within the archive. The OAIS SIPs will conform to agreements reached between the producer and the archive as defined in a Submission Agreement. Ingest functions include receiving OAIS SIPs, performing quality assurance on OAIS SIPs, and generating OAIS AIPs that comply with the archive's data formatting practice.
- *Archival Storage*: This component handles the storage, maintenance and retrieval of OAIS AIPs held by the archive. Archival Storage functions include receiving OAIS AIPs from the Ingest component, adding them to permanent storage, and performing error checking.
- *Data Management*: This component provides the services and functions for populating, maintaining, and accessing a wide variety of administrative data used to manage the archive. Such data include catalog, indexes, and inventories extracted from the archived data, consumer logs, security controls, policies and procedures, and so forth.
- *Administration*: This component provides the services and functions for the overall operation of the archive system.
- *Preservation Planning*: This component monitors the environment of the archive and provides recommendations to ensure that the data stored in the archive remain accessible over the long term even if the original computing environment becomes obsolete.
- *Access*: This component supports identifying, locating, and accessing the data of interest. This includes the provision of interfaces to the archive's holdings for both search and retrieval purposes. Several pre-defined categories of access requests are distinguished, including requests aimed at returning Dissemination Information Packages (DIPs).

2. The MPEG-21 Digital Item Declaration

2.1. From MPEG-1 to MPEG-21

The Moving Picture Experts Group (MPEG) is a working group of ISO in charge of the development of standards for coded representation of digital audio and video. So far, several

MPEG standards have had a significant impact on the multimedia landscape (Chiariglione, 1998; Koenen, 2001; Burnett, Van de Walle, Hill, Bormans & Pereira, 2003), including:

- The MPEG-1 (1993) and MPEG-2 (1996) standards have enabled the production of widely adopted commercial products, such as Video CD, MP3, Digital Audio Broadcasting, DVD, and digital television (Digital Video Broadcasting) (Watkinson, 2004).
- MPEG-4 standardized audiovisual coding solutions that address the needs of communication, interactive and broadcasting services. MPEG-4 is currently used on the Internet (e.g., as QuickTime 6) (Pereira & Ebrahimi, 2002).
- More recently, MPEG-7, formally named 'Multimedia Content Description Interface', concentrated on the description of multimedia content. While previous MPEG standards have focused on the coded representation of audio-visual content, MPEG-7 is primarily concerned with secondary textual information about the audio-visual content (Hunter, 1999; Hunter, 2001).

Together, the MPEG-1, -2, -4, and -7 standards provide a powerful set of specifications for multimedia representation. Many other high-quality multimedia standards (and proprietary solutions), such as JPEG and JPEG 2000, are being created to meet the needs of different communities. However, widespread deployment of interoperable multimedia applications requires more than just this array of standards that are mainly oriented towards file-based media types. standards are needed that focus on the representation of content consisting of multiple files; standards that specify how applications can interact with such content in an interoperable way; standards that facilitate the adaptation of content; standards for identifying and protecting digital content and the rights of the rights holders; and so forth.

MPEG-21, formally called the 'Multimedia Framework', seeks to fill these gaps. Its vision is '*to define a normative set of tools for multimedia delivery and consumption for use by all the players in the delivery and consumption chain*' (ISO, 2004b). In order to facilitate interoperability within a domain or between domains, those tools may be used selectively in combination. The envisioned techniques endeavor to cover a large part of the content delivery chain, encompassing content creation, protection, adaptation, dissemination and consumption.

2.2. MPEG-21 principles

MPEG-21 introduces the Digital Item as a '*structured digital object with a standard representation, identification and metadata*' (ISO, 2004b). The Digital Item is the digital representation of an asset. It is the unit that is acted upon within the MPEG-21 framework. The Digital Item of MPEG-21 can roughly be considered the equivalent of the digital asset in the library and information science domain, and can also be put on par with the concept of digital object – as defined by the Kahn/Wilensky Framework (Kahn & Wilensky, 1995) – or OAIS Content Information – following the OAIS Reference Model (ISO, 2003a).

Parties that interact within the MPEG-21 environment are categorized as Users. The User roles include creators, consumers, rights holders, content providers, distributors, and so on. There is no technical distinction between providers and consumers. All Users interact with Digital Items. The goal of MPEG-21 can thus be regarded as to provide a set of tools which enables a User to interact with another User and the object of that interaction is a Digital Item. Possible interactions include providing content, modifying content, archiving content, delivering content, consuming content, subscribing to content, and so forth.

2.3. MPEG-21: A suite of standards

MPEG-21 is organized into several parts (currently eighteen), primarily to allow various slices of the technology to be used autonomously. Although the various parts can be used separately, they were developed to give optimal results when used together. The MPEG-21 parts that are of importance in the setting of this dissertation are:

- MPEG-21 Part 2 – Digital Item Declaration (henceforth referred to as MPEG-21 DID), detailing the representation of Digital Items (ISO, 2003b; ISO, 2005a).
- MPEG-21 Part 3 – Digital Item Identification (henceforth referred to as MPEG-21 DII), detailing the identification of Digital Items and their contained entities (ISO, 2003c; Bekaert & Rump, 2005).
- MPEG-21 Part 4 – Intellectual Property Management and Protection (henceforth referred to as MPEG-21 IPMP), detailing a framework to enforce licenses, expressed using MPEG-21 REL (Watt, Huang, Rodriguez & Lauf, 2005).
- MPEG-21 Part 5 – Rights Expression Language (henceforth referred to as MPEG-21 REL), detailing a language to express rights pertaining to Digital Items and/or parts thereof (ISO, 2004c).
- MPEG-21 Part 6 – Rights Data Dictionary (henceforth referred to as MPEG-21 RDD), detailing a set of clear, structured, and uniquely identified terms that can be used to support rights expression languages, such as the MPEG-21 REL (ISO, 2004d). Many terms of the MPEG-21 RDD are borrowed from the aforementioned <in*d*ecs> Metadata Framework. (Rust & Bide, 2000; <indecs> 2rdd Consortium, 2001).
- MPEG-21 Part 7 – Digital Item Adaptation (henceforth referred to as MPEG-21 DIA), detailing the adaptation and transcoding of datastreams based on contextual information such as device capabilities, network characteristics and User preferences (ISO, 2004e).
- MPEG-21 Part 10 – Digital Item Processing (henceforth referred to as MPEG-21 DIP), detailing the association of processing information with Digital Items and/or parts thereof (ISO, 2006).

Although MPEG-21 originates in a community that focuses on the coding of audio and video, there is a clear overlap between the problem domain addressed by the MPEG-21 effort and ongoing efforts regarding the interoperable representation, management, and dissemination of digital assets in the digital library community (Nelson, Argue, Efron, Denn & Pattuelli, 2001). For example, MPEG-21 DID and MPEG-21 DII directly relate to aforementioned XML-based representational approaches, such as XFDU, METS and IMS-CP. Also, MPEG-21 DIP and MPEG-21 DIA reveal a parallel with sophisticated repository architectures that emerged from the digital library community, specifically Fedora (Payette & Lagoze, 1998; Staples et al., 2003; Lagoze et al., in press).

2.4. Declaring Digital Items

In the MPEG-21 Framework, Digital Items are modeled according to the second Part of MPEG-21: the Digital Item Declaration (MPEG-21 DID) standard (ISO, 2005a). MPEG-21 DID specifies the declaration of Digital Items in three distinct sections:

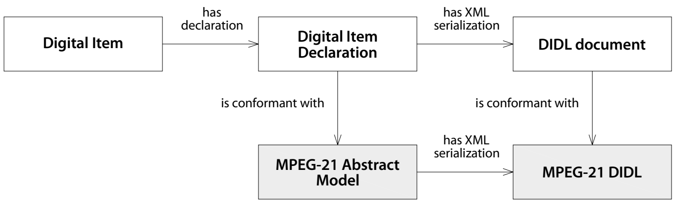


Figure 5. Relationship between MPEG-21 Abstract Model and MPEG-21 DIDL

- *Abstract Model.* A set of abstract terms and concepts that, together, form a well-defined data model for declaring Digital Items. The Abstract Model allows for Digital Items to be represented in many ways. So far, MPEG-21 Part 2 defines a normative XML representation of Digital Items based on the Abstract Model. But the Model anticipates the emergence of many other types of representation formats, such as a format based on the Resource Description Framework (RDF) (McBride, Manola & Miller, 2004) or a binary syntax. The declaration of a Digital Item compliant with the Abstract Model is referred to as a Digital Item Declaration (DID). See also Figure 5 and Section 2.4.1.
- *Representation of the Model in XML.* The description of an XML syntax for each of the entities defined in the Abstract Model. This XML-based syntax is referred to as the MPEG-21 Digital Item Declaration Language (MPEG-21 DIDL). A DID represented according to the MPEG-21 DIDL syntax is referred to as a DIDL document. See also Figure 5 and Section 2.4.2. In the OAIS reference Model, a DIDL document is considered an Information Package (see Section 1).
- *XML Schema.* The W3C XML Schema's (Fallside, 2002) specifying the MPEG-21 DIDL syntax and constrains for the structure of DIDL documents.

The first edition of MPEG-21 Part 2 was published as an ISO standard in March 2003 (ISO, 2003b). A second edition of the standard (ISO, 2005a) has been finalized mid 2005 and mainly enhances the functionality of the MPEG-21 DIDL. The lion's share of those enhancements is the result of amendments and suggestions proposed by the author, and are driven by a digital library use case of MPEG-21 technologies. The differences between both editions are emphasized in the text. The second edition of MPEG-21 DID is freely available from the ISO Information Technology Task Force website [http://standards.iso.org/ittf/PubliclyAvailable Standards/c041112_ISO_IEC_21000-2_2005(E).zip].

2.4.1. Abstract Model

The introduction of a data model is a characteristic that distinguishes MPEG-21 DID from related techniques aimed at representing digital assets. The existence of a data model provides flexibility for deployment of compatible digital assets representations in various technical environments. The MPEG-21 Abstract Model defines several constituent entities of a DID. This section provides a simplified explanation of each of those entities. When a reference to an entity of the Abstract Model is made, the *italic* font style is used. Interested readers are referred to the standard for full details.

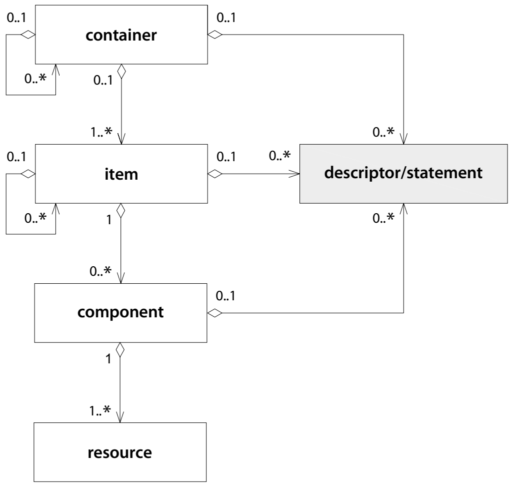


Figure 6. Core entities of the MPEG-21 DID Abstract Model

A first set of entities consists of the core building blocks for declaring Digital Items. Some are shown in Figure 6. These entities make up the backbone of a DID and are presented in a bottom-up approach, starting at the leaf of the tree with the *resource* entity, representing an actual datastream of a digital asset:

- *resource*. A *resource* is an individual datastream such as a video file, image, audio clip, or textual asset.
- *component*. A *component* is the binding of one or more equivalent *resources* to a (set of) *descriptor/statement* construct(s). These *descriptor/statement* constructs(s) contain secondary data related to all the *resource* entities bound by the *component*. A *component* is considered a dummy entity that is merely used to group *resources* and secondary data conveyed in one or more associated *descriptor/statement* construct(s).
- *item* or Digital Item. An *item* is the binding of one or more *items* and/or *components* to a (set of) *descriptor/statement* construct(s). These *descriptor/statement* construct(s) contain information about the represented *item*. In the Abstract Model, an *item* entity is equivalent to a Digital Item. Hence, *items* are the first point of entry to the content for a User. If *items* contain other *items*, then, the outermost *item* represents the composite *item*, and the inner *items* represent the individual *items* that make up the composite.
- *container*. A container is the binding of one or more *items* and/or *containers* to a (set of) *descriptor/statement* construct(s). The *containers* can be used to form groupings of Digital Items. The *descriptor/statement* constructs contain information about the represented *container*.

- *descriptor/statement*. A *descriptor/statement* construct, shown in Figure 6, introduces an extensible mechanism that can be used to associate textual secondary data with other entities of the Abstract Model. A *descriptor/statement* construct typically associates the information with its enclosing entity. For example, a *descriptor/statement* construct attached as a child entity to an *item* provides secondary data about that *item*. Similarly, a *descriptor/statement* construct attached as a child entity to a *container* provides secondary data about that *container*. The usage of *descriptor/statement* constructs attached to a *component* deviates from the default rule in that they provide information about the *resources* bound by the *component*, and not about the enclosing *component* entity. Examples of likely secondary information (or 'statements') include information supporting discovery, digital preservation and rights expressions. *Statements* are non-identifiable entities, and, as a result, cannot have licenses associated with them.

A next set of entities, shown in Figure 7 refines the description of the *resource* entity:

- *fragment*. A *fragment* designates a specific point or range within a *resource*. *Fragments* may be media type-specific.
- *anchor*. An *anchor* binds one or more *descriptor/statement* constructs to a *fragment*. These *descriptor/statement* constructs contain information related to the *fragments* bound by the *anchor*. Similarly to the *component* entity, an *anchor* is a dummy entity that is used merely to group *fragment* and *descriptor/statement* constructs.

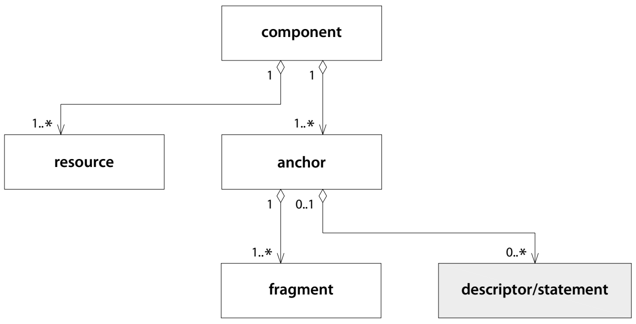


Figure 7. *fragment* and *anchor* entities

For example, a *fragment* may specify a polygonal area within an image *resource* or a specific point in time of an audio track. The *anchor* entity binds secondary data to the polygonal area of the image or the time point of the audio track, respectively.

Another entity of the DID Abstract Model is *annotation*. An *annotation* adds information to an entity of the model without altering or adding something to that entity. The initial source entity remains intact. The information conveyed by an *annotation*, can take the form of an *assertion*, *descriptor/statement*, or *anchor*.

A last set of entities, containing the *choice*, *selection*, *condition* and *assertion* entities, allows describing a Digital Item (or a part thereof) as being optional or available under specific conditions. For example, a *choice* in a DID may represent the choice between a high bandwidth internet connection and a dial-up connection. Based on the *selection* made by a User, the *conditions* attached to (an entity of) a Digital Item may be fulfilled and (the entity of) the Digital Item may become available. Dependent on the nature of the *conditions*, the entity could contain a high resolution datastream or a compressed file, respectively. The details of these conditional mechanisms are beyond the scope of this doctoral study. Interested readers are referred to the MPEG-21 DID specification for full details.

2.4.2. Representation of the Abstract Model in XML

The Digital Item Declaration Language (MPEG-21 DIDL) is the normative XML Representation of the Abstract Model. The entities defined in the Abstract Model are each represented in MPEG-21 DIDL by a like-named XML element. In what follows, a `monospaced` font is used to refer to these XML elements. For example, the *component* entity of the Abstract Model is represented in MPEG-21 DIDL by the `Component` XML element. The semantics and structural positions of the DIDL elements correspond with those of the entities of the Abstract Model. The MPEG-21 DIDL also defines some special elements that do not correspond to any of the Abstract Model entities. These elements can only exist in the XML Representation of the Abstract Model:

- The `DIDL` element is the root element of a DIDL XML document. The second edition of MPEG-21 DID allows for the association of secondary data pertaining to the DIDL document itself by adding attributes to the `DIDL` root element or by including XML elements in the `DIDLInfo` element that itself is a child of the `DIDL` root element. Also, the `DIDL` root element may have an optional `DIDLDocumentId` attribute. This attribute can be used to convey the identifier of the DIDL document. The identifier of the DIDL document corresponds to what the OAIS Information Model (ISO, 2003a) categorizes as an OAIS Information Package Identifier. Identifiers of the digital asset(s) or Digital Item(s) declared within the DIDL document are not conveyed by this attribute, but rather by a `Descriptor/Statement` construct as explained in Section 2.5.3. Both the `DIDLInfo` element and the `DIDLDocumentId` attribute have resulted from proposals submitted by the author (Bekaert, DeMartini, Van de Walle & Van de Sompel, 2004; Bekaert et al., 2005) and are largely inspired by a digital library and digital archive use case of MPEG-21 DID.
- The `Reference` element represents a link to another XML element of a DIDL document, and virtually embeds the contents of the referenced element into the referring element. References can be made to elements within the same DIDL document or to elements in another DIDL document. The former type of reference is known as an internal reference; the latter is known as an external reference. An internal reference allows a single source to be maintained for an element that occurs in more than one place in a DIDL document. An external reference allows a DIDL document to be split up into multiple linked discrete DIDL documents. In the second edition of MPEG-21 DID, the `Reference` element has been removed as similar functionality can be achieved by using XML Inclusion 1.0 (XInclude) (Marsh & Orchard, 2004). The latter has been published as a W3C Recommendation in December 2004.
- The `Declarations` element is used to define DIDL elements in a DIDL document without actually instantiating them. A declared element can then be instantiated by using a `Reference` element or XInclude construct.

The grammar and syntax of the MPEG-21 DIDL is captured in a W3C XML Schema (Fallside, 2002). The XML Schema of the second edition of MPEG-21 DID is available from the ISO website [http://Standards.iso.org/ittf/PubliclyAvailableStandards/MPEG-21_schema_files/]. To be conformant with the MPEG-21 DIDL syntax, validation against the grammar of the XML Schema is necessary, yet not sufficient. Some mechanisms, such as Schematron (ISO Joint Technical Committee 1 [JTC1] Sub-Committee 34 [SC34], 2004), may be required to express XPath (Clark & DeRose, 1999) based rules that cannot be expressed by means of the XML Schema. The full specifics of these rules go beyond the scope of this introduction. Interested readers are referred to the standard for full details of the MPEG-21 DIDL syntax.

2.5. A step-by-step example of a DIDL document

In this Section, XML excerpts of a sample DIDL document are provided to illustrate some of the core properties of MPEG-21 DIDL. The DIDL document is shown in full in Table 9. The document is the MPEG-21 DIDL-based XML representation of a digital asset consisting of:

- A scholarly publication with identifier '`info:doi/10.1045/july95-arms`'. Two variants of the publication are provided as separate datastreams: the original high resolution PostScript file, and a derived PDF file.
- Secondary information about the publication expressed in the Dublin Core format (NISO, 2001).

2.5.1. Providing constituent datastreams

Each constituent datastream of a Digital Item is conveyed in a separate `Resource` element, representing a *resource*. As shown in Table 1, MPEG-21 DIDL allows those *resources* to be physically embedded within a DIDL document – also called By Value provision – or to be pointed at from within a DIDL document – also called By Reference provision. Table 1 also shows how the nature of those assets relates to the way in which they can be included By Value. MPEG-21 DIDL allows for the inclusion of unencoded (mixed) XML elements not defined by the MPEG-21 DIDL XML Schema. In addition, binary datastreams may be provided By Value using a base64 encoding as specified in IETF RFC 3548 (Josefsson, 2003).

DIDL element	**By Value**	**By Value**	**By Reference**
	(mixed) XML elements	**base 64 encoded data**	
`Resource`	□	■	■
`Statement`	■	■	□

Table 1. By Value and By Reference provision of datastreams and secondary data in a DIDL document

The white rectangles in Table 1 indicate provision techniques that became available in the second edition of MPEG-21 DID as a result of amendments proposed by the author (Bekaert, Hochstenbach, De Neve, Van de Sompel & Van de Walle, 2003; Bekaert, Hochstenbach, Van de Sompel & Van de Walle, 2003).

Table 2 shows a By Reference provision of the PDF datastream of the sample digital asset. The `ref` attribute contains the actual network location. A By Value provision of datastreams can be accomplished by base64 encoding the binary data and wrapping the outcome in the `Resource` element. As a result of the same amendments, the second edition of MPEG-21 DID also requires the use of the `encoding` attribute (with a value set to '`base64`'), whenever the `Resource` element contains base64 encoded data. If the `encoding` attribute is omitted, the data must be un-encoded. This removes an ambiguity in the first edition of MPEG-21 DID, which does not support an explicit way to determine whether embedded character data is base64 encoded or not. Table 3 shows an abbreviated By Value provision of the PDF datastream. In addition to the `ref` and `encoding` attributes, a mandatory `mimeType` attribute indicates the MIME media and subtype (Freed & Borenstein, 1996) of the datastreams.

```
<didl:Resource mimeType="application/pdf"
               ref="http://purl.lanl.gov/tech/pdf/015997845.pdf"/>
```

Table 2. By Reference provision of the PDF datastream

```
<didl:Resource mimeType="application/pdf" encoding="base64">
   JVBERi0xLjMKJeLjz9MNCjEgMCBvYmoKPDwgCi9TdWJqZWN0ICgpCi9LZXl3b3JkcyAoKQov
   YXRvciAoWFBQKQovVGl0bGUgKCkKL1Byb2R1Y2VyICgpCi9Nb2REYXRlIChEOjIwMDQwNTE0
   MzE2KQovQ3JlYXRpb25EYXRlICgyMDA0MDUxNDEyMjMwMSkKL0F1dGhvciAoKQo+PiAKZW5k
   CjIgMCBvYmoKPDwgCi9UeXBlIC9QYWdlIAovUGFyZW50IDExIDAgUiAKL1Jlc291cmNlcyA0
   ZWUgdGVjaG5pcXVlcyBhcmUgZGlzY3Vzc2VkOiAoMSkgZGVzaWduIHR3byBleHBlcmltZW50
   aGF0IHdvdWxkIG1lYXN1cmUgdGhlIHF1YW50aXRpZXMgb24gZWl0aGVyIHNpZGUgb2YgdGhl
   dWFsaXR5OyAoMikgZXhhbWluZSBzcGVjaWFsIGNhc2VzOyAoMykgY29uc2lkZXIgdGhlIGNv
   ...
</didl:Resource>
```

Table 3. By Value provision of the PDF datastream

It may be important to note that the second edition of MPEG-21 DID also allows expressing additional content-encodings that have been applied to a *resource* of a DIDL document and thus to indicate which decoding mechanisms need to be applied to the *resource* in order to obtain the MIME media-type identified by the `mimeType` attribute (Bekaert, 2005). This is achieved using a `contentEncoding` attribute. This functionality is primarily used to allow a *resource* to be compressed without losing the knowledge regarding its underlying MIME media-type. If multiple content-encodings have been applied to the *resource*, all content-encoding values are concatenated in space-delimited list, in the order in which they were applied. Accepted tokens for use in the value of the `contentEncoding` attribute are the content-encoding tokens registered by the IANA.

2.5.2. Providing secondary data

Secondary textual information, or *statements*, are contained inside a `Statement` element. Whereas *resources* are considered identifiable entities, *statements* cannot be identified, and hence, should not be treated as individual entities in their own right. `Statement` elements are embedded inside `Descriptor` elements, forming so-called *descriptor/statement* constructs.

As indicated in the previous section, a *statement* can be provided By Value and/or By Reference. Just like a `Resource` element, a `Statement` element has a required `mimeType` attribute, and optional `ref`, `encoding`, and `contentEncoding` attributes. Table 4 shows a `Statement` element containing a By Value provision of secondary data about the sample scholarly publication, expressed in Dublin Core format.

```
<didl:Statement mimeType="application/xml; charset=utf-8">
  <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:dcterms="http://purl.org/dc/terms/">
    <dc:title>Key Concepts in the Architecture of the Digital Library</dc:title>
    <dc:creator>William Y. Arms</dc:creator>
  </oai_dc:dc>
</didl:Statement>
```

Table 4. By Value provision of secondary data expressed in Dublin Core format

MPEG-21 DID does neither provide categorizations nor presumptions about the types of information that can be conveyed using `Statement` elements. While METS (LC, 2005a) and XFDU (CCSDS, 2004b) impose some kind of categorization by pre-defining placeholders for descriptive, administrative, rights, source and provenance information, MPEG-21 DIDL relies on the XML namespaces of the information provided in `Statement` elements to determine the semantics of the conveyed information. For example, according to the MPEG-21 Rights Expression Language (MPEG-21 REL), XML elements in the MPEG-21 REL XML namespace convey rights related information. This approach also allows for application domains to profile DIDL documents by restricting the XML namespaces that can be used, and by specifying the semantics of the use of elements of those namespaces.

2.5.3. Associating secondary data

`Descriptor/Statement` elements, representing a *descriptor/statement* construct, provide an extensible mechanism to associate textual secondary data with entities of the Abstract Model. For example, in order to associate – say – an identifier with an *item*, a *descriptor* containing the identifier *statement* can be created as a child element of the `Item` element.

As will be shown, the MPEG-21 framework itself defines ways to use *descriptor/statement* constructs as a means to convey – among other things – identifiers, rights information, and processing information. This approach is illustrated in Section 2.5.3.1. To enable the provision of domain or application-specific information, *descriptor/statement* constructs may also be defined by third parties. This approach is illustrated in Section 2.5.3.2.

2.5.3.1. MPEG-21 defined descriptor/statement constructs

A. MPEG-21 DII: Using descriptor/statement constructs to identify Digital Items

Through the introduction of a special part, the MPEG-21 Digital Item Identification (MPEG-21 DII), MPEG-21 recognizes the importance of identifiers in network-based applications. MPEG-21 DII specifies the usage of `Descriptor/Statement` constructs for the identification of Digital Items and parts thereof. To that end, it introduces a DII XML namespace with elements that can be used to associate identifiers and typification information with *container*, *item*, *component*, and *anchor* entities. For example, Table 5 shows the use of the `Identifier` element from the MPEG-21 DII XML namespace to associate the URI '`info:doi/10.1045/july95-arms`' with the Digital Item; where '`doi:10.1045/july95-arms`' is a unique DOI (NISO, 2000) number for the publication, and '`info`' is the namespace of the aforedescribed info URI Scheme (Van de Sompel, Hammond, Neylon & Weibel, 2004). This identifier corresponds to what the OAIS Information Model (ISO, 2003a) categorizes as an OAIS Content Information Identifier.

```
<didl:Item>
  <didl:Descriptor>
    <didl:Statement mimeType="application/xml; charset=utf-8">
      <dii:Identifier xmlns:dii="urn:mpeg:mpeg21:2002:01-DII-NS">
      info:doi/10.1045/10.1045/july95-arms</dii:Identifier>
    </didl:Statement>
  </didl:Descriptor>
  ...
</didl:Item>
```

Table 5. A `Descriptor/Statement` construct conveying an identifier of a digital asset

MPEG-21 DII also defines a `RelatedIdentifier` element that carries identifiers that are 'related' to a Digital Item or a part thereof. The MPEG-21 DII specification, however, remains silent on the nature of the relationship of the thing identified by a `RelatedIdentifier` to the Digital Item (or constituent thereof) that carries the `RelatedIdentifier`. At the time of writing, the author is involved in an amendment (Bekaert & Rump, 2005) aimed at qualifying the nature of this relationship by means of a `relationshipType` attribute. The value of this attribute is in the form of a URI and corresponds to a verb from the MPEG-21 Rights Data Dictionary (MPEG-21 RDD) (ISO, 2004d), which, in turn, is based on the <in*d*ecs> Metadata Framework (Rust & Bide, 2000). According to the latter, different intellectual variants bring along different types of relationships. For example, the relationship between a book, and a translation of that book, could be categorized as 'isTranslationOf', the relationship between an article and an abstract of that article could be described as 'isDerivationOf', and so forth.

B. MPEG-21 REL: Using descriptor/statement constructs to associate rights expressions with Digital Items

The MPEG-21 Rights Expression Language (MPEG-21 REL) specifies the use of `Descriptor/Statement` constructs to associate rights expressions with a Digital Item or parts thereof. This is achieved through the introduction of a language inspired by XrML (ContentGuard, n.d.) with elements and attributes in a REL XML namespace. Table 6 shows the use of the `license` element of the REL XML namespace to associate very basic copyright information with the sample digital asset. MPEG-21 IPMP, currently under development, will provide tools that assist in enforcing rights expressions declared by the REL.

```
<didl:Item>
  <didl:Descriptor>
    <didl:Statement mimeType="application/xml; charset=utf-8">
      <r:license xmlns:r="urn:mpeg:mpeg21:2003:01-REL-R-NS">
        <!-- MPEG-21 REL Licenses can be added here -->
```

```
          <r:otherInfo>
            <dc:rights xmlns:dc="http://purl.org/dc/elements/1.1/">
            Copyright 1995; CNRI</dc:rights>
          </r:otherInfo>
        </r:license>
    </didl:Statement>
  </didl:Descriptor>
  ...
</didl:Item>
```

Table 6. A `Descriptor/Statement` construct conveying a simple rights expression

C. MPEG-21 DIP: Using descriptor/statement constructs to associate processing information with Digital Items

MPEG-21 Digital Item Processing (MPEG-21 DIP) specifies an architecture pertaining to the dissemination of Digital Items. This MPEG-21 part introduces the concept of a Digital Item Method, as a way to allow a Digital Item author to provide suggested interactions of a consumer with a Digital Item. A Digital Item Method is physically contained in the same DIDL document as the Digital Item with which it is associated. In the current practice, a Digital Item Method is associated with an element of a DIDL document using a special-purpose `Descriptor/Statement` construct, containing elements of the MPEG-21 DIP XML namespace. The Digital Item Methods (and the associated Digital Items parts) can be extracted from a DIDL document and processed upon request of a consumer by a special component, called an MPEG-21 DIP Engine.

2.5.3.2. Non-MPEG-21 defined descriptor/statement constructs

`Descriptor/Statements` constructs may also be used by third parties to provide secondary data that is specific to the nature of content, the application and the community. Table 7 shows the use of a `Descriptor/Statement` construct to associate the statement represented in Table 4 with an `Item`.

```
<didl:Item>
  <didl:Descriptor>
    <didl:Statement mimeType="application/xml; charset=utf-8">
      <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
        xmlns:dc="http://purl.org/dc/elements/1.1/">
        <dc:title>Key Concepts in the Architecture of the Digital Library</dc:title>
        <dc:creator>William Y. Arms</dc:creator>
      </oai_dc:dc>
     </didl:Statement>
  </didl:Descriptor>
  ...
</didl:Item>
```

Table 7. A `Descriptor/Statement` construct conveying domain-specific information

The second edition of MPEG-21 DID also allows the association of third party information with MPEG-21 DIDL elements via XML attributes. For instance, an `Item` element may have attributes from other XML namespaces providing information about that *item*. Note that, as is the case with the `Descriptor/Statement` construct, these attributes are considered XML serializations of the abstract *descriptor/statement* entities.

2.5.4. Putting the DIDL document together

So far, the `Resource` and `Statement` elements have been introduced, and the use of `Descriptor/Statement` constructs has been shown to enable associating secondary data with a Digital Item or its constituents. This section uses these building blocks to compile a DIDL document that represents the sample digital asset. The explanation works its way up from the `Resource` and `Component` elements to the `Item` element, and then all the way up to the `DIDL` root element. The complete DIDL document, representing the sample asset, is shown in Table 9.

A `Component` element, representing a *component*, is introduced to bind one or more `Resource` elements to a (set of) `Descriptor` elements. In MPEG-21 DID first edition, the *resources* contained inside a single *component* are considered 'equivalent'. In MPEG-21 DID second edition, based on input from the author, 'equivalence' has been further qualified to mean 'bit-equivalent' (Bekaert, De Keukelaere, Van de Sompel & Van de Walle, 2004). As a result, MPEG-21 DID allows for the provision of multiple bit-equivalent *resources* inside a single *component*, using both By Value and By Reference provision techniques. This feature is appealing for expressing that the same data is available at multiple mirrored repositories.

Table 8 shows a `Component` element wrapping the PDF *resources* represented in Table 2 and Table 3. The first *resource* is provided By Reference through the inclusion of a reference to its network-location; the second, bit-equivalent, *resource* is provided By Value, and is base64 encoded.

```
<didl:Component>
  <didl:Resource mimeType="application/pdf"
    ref="http://purl.lanl.gov/tech/pdf/015997845.pdf"/>
  <didl:Resource mimeType="application/pdf" encoding="base64">
    JVBERi0xLjMKJeLjz9MNCjEgMCBvYmoKPDwgCi9TdWJqZWN0ICgpCi9LZXl3b3JkcyAoKQov
    YXRvciAoWFBQKQovVGl0bGUgKCkKL1Byb2R1Y2VyICgpCi9Nb2REYXRlIChEOjIwMDQwNTE0
    MzE2KQovQ3JlYXRpb25EYXRlICgyMDA0MDUxNDEyMjMwMSkKL0F1dGhvciAoKQo+PiAKZW5k
    CjIgMCBvYmoKPDwgCi9UeXBlIC9QYWdlIAovUGFyZW50IDExIDAgUiAKL1Jlc291cmNlcyA0
    ZWUgdGVjaG5pcXVlcyBhcmUgZGlzY3Vzc2VkOiAoMSkgZGVzaWduIHR3byBleHBlcmltZW50
    aGF0IHdvdWxkIG11YXN1cmUgdGhlIHF1YW50aXRpZXMgb24gZWl0aGVyIHNpZGUgb2YgdGhl
    dWFsaXR5OyAoMikgZXhhbWluZSBzcGVjaWFsIGNhc2VzOyAoMykgY29uc2lkZXIgdGhlIGNv
    ...
  </didl:Resource>
</didl:Component>
```

Table 8. A `Component` element grouping bit-equivalent *resources*

The sample digital asset consists of two datastreams: The PDF datastream, and a datastream in PostScript format. As the PostScript datastream is not bit-equivalent with the one in PDF format, it is provided in a separate *component*. And, because the PostScript *resource* and the PDF *resource* share an identifier, both *components* are provided in the same *item*. This *item* is considered to be the declarative representation of the sample digital asset. An `Item` element is introduced that contains both *components*, and binds them to a set of *descriptor/statement* constructs. The `Descriptor/Statement` constructs associated with the `Item` convey secondary data about the Digital Item. A first `Descriptor/Statement` construct conveys the DOI of the Digital Item using the `Identifier` element from the DII XML namespace. A second `Descriptor/Statement` construct conveys secondary data about the Digital Item expressed in the Dublin Core format. The DIDL document itself opens with the `DIDL` root element containing the MPEG-21 DIDL XML namespace declaration, and an identifier for the DIDL document.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<didl:DIDL DIDLDocumentId="info:lanl-repo/i/ b236-6498-000629ba5445"
  xmlns:didl="urn:mpeg:mpeg21:2002:02-DIDL-NS">
  <didl:Item>
    <didl:Descriptor>
      <didl:Statement mimeType="application/xml; charset=utf-8">
        <dii:Identifier xmlns:dii="urn:mpeg:mpeg21:2002:01-DII-NS">
        info:doi/10.1045/july95-arms</dii:Identifier>
      </didl:Statement>
    </didl:Descriptor>
    <didl:Descriptor>
      <didl:Statement mimeType="application/xml; charset=utf-8">
        <oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
          xmlns:dc="http://purl.org/dc/elements/1.1/"
          xmlns:dcterms="http://purl.org/dc/terms/">
          <dc:title>Key Concepts in the Architecture of the Digital
            Library</dc:title>
          <dc:creator>William Y. Arms</dc:creator>
        </oai_dc:dc>
      </didl:Statement>
    </didl:Descriptor>
    <didl:Component>
      <didl:Resource mimeType="application/pdf"
        ref="http://purl.lanl.gov/tech/pdf/015997845.pdf"/>
      <didl:Resource mimeType="application/pdf" encoding="base64">
        JVBERi0xLjMKJeLjz9MNCjEgMCBvYmoKPDwgCi9TdWJqZWN0ICgpCi9LZXl3b3JkcyAoKQov
        YXRvciAoWFBQKQovVGl0bGUgKCkKL1Byb2R1Y2VyICgpCi9Nb2REYXRlIChEOjIwMDQwNTE0
        MzE2KQovQ3JlYXRpb25EYXRlICgyMDA0MDUxNDEyMjMwMSkKL0F1dGhvciAoKQo+PiAKZW5k
        CjIgMCBvYmoKPDwgCi9UeXBlIC9QYWdlIAovUGFyZW50IDExIDAgUiAKL1Jlc291cmNlcyA0
        ZWUgdGVjaG5pcXVlcyBhcmUgZGlzY3Vzc2VkOiAoMSkgZGVzaWduIHR3byBleHBlcmltZW50
        aGF0IHdvdWxkIG1lYXN1cmUgdGhlIHF1YW50aXRpZXMgb24gZWl0aGVyIHNpZGUgb2YgdGhl
        dWFsaXR5OyAoMikgZXhhbWluZSBzcGVjaWFsIGNhc2VzOyAoMykgY29uc2lkZXIgdGhlIGNv
        cXVlbmNlcyBpZiB0aGUgZXF1YWxpdHkgZmFpbGVkIHRvIGhvbGReXjVeX0EwMTUwXl9BMDE1
        X0EwNTcwXl9BMDUyMF5eNF5fZWR1Y2F0aW9uYWwgY291cnNlc15fc3RhdGlzdGljYWwgbWVj
        ...
      </didl:Resource>
    </didl:Component>
    <didl:Component>
      <didl:Resource mimeType="application/ps"
        ref="http://purl.lanl.gov/tech/ps/015997845.ps"/>
    </didl:Component>
  </didl:Item>
</didl:DIDL>
```

Table 9. A DIDL document representing the sample digital asset

2.6. MPEG-21 DID and the OAIS Information Model

This section briefly explores the potential of the MPEG-21 DID in a digital preservation context, by looking at the core entities of the OAIS Information Model (ISO, 2003a) (See also Section 1.1) and the way in which these entities correspond with entities of the MPEG-21 DID Abstract Model, and elements from the MPEG-21 DIDL syntax.

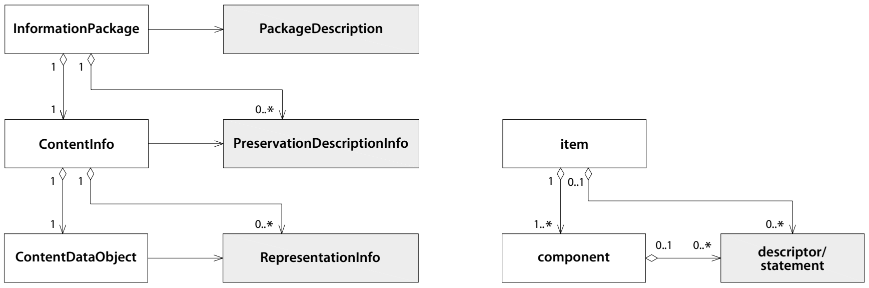


Figure 8. Comparing concepts from the OAIS Information Model (left) and the MPEG-21 DID Abstract Model (right)

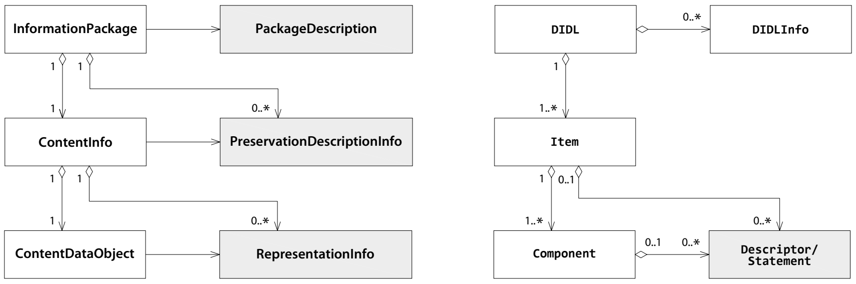


Figure 9. Comparing concepts from the OAIS Information Model (left) and the MPEG-21 DIDL XML syntax (right)

The following mappings follow easily – several properties of the mapping process are depicted in Figure 8 and Figure 9:

- OAIS Content Information can be put on par with an MPEG-21 Digital Item. Both are considered the main point of focus in their respective environments.
- OAIS Preservation Information is needed to adequately preserve the OAIS Content Information with which it is associated. In MPEG-21 DID, such information can be provided by appending *descriptor/statement* constructs to the *item*-level. One such *descriptor/statement* construct could convey the Identifier of the OAIS Content Information, by wrapping the identifier in an `Identifier` element from the MPEG-21 DII namespace. Other *descriptor/statement* constructs could convey preservation information by – for example – using elements from the PREMIS effort (PREMIS, 2005).

- Content Information, as specified by the Information Model for an OAIS, is comprised of a Content Data Object combined with secondary information related to the Representation of the Content Data Object. A Content Data Object can be mapped to one or more *component/resource* constructs, depending on whether the Content Data Object is implemented using one or more files. Representation Information about the Content Data Object can be conveyed using *descriptor/statement* constructs appended at the *component*-level. Also, each *component* entity may hold a Fragment Identifier for accessing the actual datastream. In XML, Fragment Identifiers are expressed using the syntax of the XPointer Framework (DeRose, 2002; Grosso, 2003). One of the key XPointer methods for identifying XML elements is the use of XML IDs.
- In MPEG-21 DIDL, an OAIS Information Package is represented in XML and is referred to as a DIDL document. According to the OAIS Information Model, Package Descriptions may be added to the Information Package to convey secondary information about the Information Package. In MPEG-21 DIDL, such information can be conveyed using `DIDLInfo` elements. The Identifier of the Information Package is conveyed using the `DIDLDocumentId` attribute.

MPEG-21 DID takes an approach that enforces cross-community interoperability, while allowing the flexibility for the emergence of compliant, domain or application-specific design choices. These design choices are typically inspired by the requirements of the target community or application, and are bundled in so-called application or community 'profiles'. One such profile could focus on the representation of digital assets for long-term preservation as per definition of the OAIS Information Model. A detailed description of such a profile has been published in the conference proceedings of PV-2005 (Bekaert, Liu & Van de Sompel, 2005).

3. The Open Archives Initiative Protocol for Metadata Harvesting

The Open Archives Initiative (OAI) was formed to drive the process of developing low-threshold solutions that aim to facilitate the efficient retrieval of content. The OAI arose out of the e-print community in an effort to enhance access to scholarly communication across fairly heterogeneous repositories. The primary product of experimentation and development was the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) (Lagoze et al., 2003b). Simply put, the OAI-PMH is a framework that allows descriptive metadata to be harvested from one place to another. Therefore, in the OAI-PMH world, a separation is made between 'data providers' and 'service providers'. A data provider hosts a collection of descriptive metadata records through a so-called OAI-PMH repository. A service provider may provide value services (such as indexing, browsing or searching) on the descriptive metadata harvested or aggregated – by an OAI-PMH harvester – from one or more data providers. The interaction between the data provider and the service provider is the basis of the protocol.

The OAI-PMH directly builds on existing standards, notably W3C XML (Bray et al., 2004) for encoding the exchanged metadata, and IETF HTTP (Fielding et al., 1999) for transporting the metadata. Service providers may request information from data providers using a standard set of six OAI-PMH verbs. OAI-PMH requests are transmitted according the rules of HTTP 1.0, with requests specified using URL encoded parameters and responses delivered in strictly valid XML.

The OAI-PMH 2.0 solution is based on a data model – depicted in Figure 10 – that helps specifying the semantics of the six protocol requests. In what follows, OAI-PMH entities of the data model are written in *italic*, while OAI-PMH protocol requests are written in a `monospaced font`.

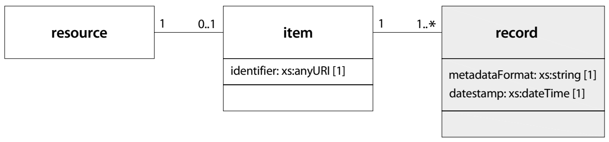

resource
1
0..1
item
identifier: xs:anyURI [1]
1
1..*
record
metadataFormat: xs:string [1]
datestamp: xs:dateTime [1]

Figure 10. The OAI-PMH data model

- At the very left is a *resource*, about which an OAI-PMH repository exposes *metadata*. By definition, the nature of the *resources* themselves is outside of the scope of the OAI-PMH.
- To the right of the *re*source is the *item*. The *item* is the highest-level entity within the scope of the OAI-PMH. The *item* is the entry point to all available *metadata* pertaining to a *resource*. That *metadata* may be disseminated on-the-fly from the associated *resource*, cross-walked from some canonical form, or actually stored in the repository. In the protocol, the *item* is uniquely identified by a unique OAI-PMH *identifier*. All possible *records* available from a single *item* share the same identifier.
- Next to the *item*, several *records* are shown. A *record* is *metadata* in a specific *metadata format*. A specific *record* in the OAI-PMH is unambiguously identified by means of the combination of the *OAI-PMH identifier* (of the *item*), the *metadata format* used for the dissemination of the *metadata* and the OAI-PMH *datestamp* of the *metadata*. In OAI-PMH 2.0, the *datestamp* is a property of the *metadata record*, not of the *item* as used to be the case in OAI-PMH version 1.x. This reflects the fact that *metadata* of various *metadata formats* may be made available and may be modified independently, thus having different OAI-PMH *datestamps*.
- The OAI-PMH also defines a *set* – not depicted in Figure 10 – as an optional construct for grouping *items* for the purpose of selective harvesting (see below). Repositories may organize *items* into *sets*. A *set* organization may be a simple list or have a more hierarchical structure. Multiple, parallel, *set* structures may exist.

The OAI-PMH defines six verbs, three of which reveal the characteristics of the repository (`ListMetadataFormats`, `ListSets`, and `Identify`) and three verbs for extracting *metadata* from the repository (`GetRecord`, `ListRecords`, `ListIdentifiers`). Each OAI-PMH verb requires and/or allows the use of certain parameters that further restrict the exact nature and details of the request.

The OAI-PMH defines three supporting protocol requests that help a harvester understand the nature of an OAI-PMH repository:

- `Identify`: This verb is used to retrieve information about a repository; an important information element returned in the response to the `Identify` request is the time granularity of the *datestamp* supported by the repository (in particular day-level or seconds-level precision).
- `ListMetadataFormats`: This verb is used to retrieve the *metadata formats* available from a repository.
- `ListSets`: This verb is used to retrieve the *set* structure of a repository. This information is useful for selective harvesting.

The OAI-PMH defines three further protocol requests that support the actual harvesting of *metadata*:

- `ListRecords`: This verb is used to harvest *records* from a repository. Optional arguments permit selective harvesting of *records* based on *set* membership and/or *datestamp*.
- `GetRecord`: This verb is used to retrieve an individual *record* from a repository. The verb has two required arguments; one argument specifies the OAI-PMH *identifier* of the *item* from which the *record* is requested; the other conveys the *metadata format* of the *metadata* that should be included in the *record*. The datestamp of the identified *metadata record* is the most recent date and time of the creation, modification, or deletion of the *metadata record*.
- `ListIdentifiers`: This verb is an abbreviated form of `ListRecords`, retrieving only *identifiers*, *datestamps* and *set* information.

In response to an OAI-PMH request, data providers process the received request and reply with an appropriate OAI-PMH response, which is always in the form of valid XML (including the *metadata records* themselves) conforming to the top-level XML Schema defined by the OAI-PMH. In this way, the service provider can learn who the *metadata* provider is (`Identify` request), what *metadata formats* it supports (`ListMetadataFormats` request), and how it has divided its *metadata* (`ListSets` request). The service provider can also request the *metadata* itself (`GetRecord`, `ListIdentifiers`, `ListRecord` requests).

In addition, so-called ‘selective harvesting’ strategies allows harvesters to limit harvest requests to portions of the *metadata* available from a repository. The OAI-PMH supports selective harvesting with two types of harvesting criteria that may be combined in an OAI-PMH request: *datestamps* and *set* membership. For example, in order to obtain *metadata* in Dublin Core format (NISO, 2001) for all *items* that have been modified or added to a repository – exposed at base URL ‘`baseURL`’ – from datetime T1 to datetime T2, the OAI-PMH protocol request will look as follows:

```
[ baseURL?verb=Listrecords&
      metadataPrefix=oai_dc&
      from=T1&
      until=T2 ]
```

Dates and times are uniformly encoded using ISO 8601 and are expressed in Coordinated Universal Time (ISO, 2004a; Wolf & Wicksteed, 1997).

The OAI-PMH uses an opaque data structure called a `resumptionToken` to separate long responses into many shorter responses. For example, if a `ListRecords` response contains ten thousand records, neither the repository nor the harvester could likely handle that response. The repository might choose to separate the complete list into one hundred incomplete lists of one hundred records each. A `resumptionToken` is simply a way for OAI-PMH harvesters to ask for the next chunk. The distinguishing characteristic is that the repository chooses the size of the `resumptionToken`, not the harvester. This allows repositories to throttle the load placed on them by harvesters.

Due to its origins in the realm of *resource* discovery, the OAI-PMH mandates the support of the Dublin Core *metadata format* as its lingua franca, but strongly encourages supporting more expressive *metadata formats*. As a result, any *metadata format* can be used as long as it is defined by means of an XML Schema. In this doctoral study, I introduce the use of *metadata formats* that are more complex, expressive, and accurate in their description of digital *resources*. These formats have specifically been defined to represent digital assets, and are typically referred to as packaging formats. One example of such a packaging format is MPEG-21 DID (See Section 2 of this Chapter).

The combination of packaging formats and the OAI-PMH is an attractive option to help addressing the content harvesting issues that is subject of this dissertation (Van de Sompel, Nelson, Lagoze & Warner, 2004). Attractive features of this approach include:

- Packaging formats represent a *resource* by means of a wrapper XML document, and therefore, a representation can natively be conveyed as *metadata* (inside the `metadata` element) of OAI-PMH responses.
- Using packaging formats as a separate *metadata format* within the OAI-PMH framework yields an unambiguous trigger mechanism for harvesting *resources*. Per definition, the OAI-PMH *datestamp* is the date of creation or modification of *metadata*. When using a packaging format, that *metadata* is a representation of the *resource* that covers all its constituents including its multiple datastreams, its secondary data, and so on. As a result, as soon as a change occurs in one of these constituents, the associated OAI-PMH *date-stamp* must change. It should be noted that such changes do not necessarily result in a change of the wrapper XML document. Indeed, if changes occur to a datastream that is provided By Reference, its network location may remain unchanged, and hence, the wrapper XML document may remain unchanged. However, because the By Value and By Reference provision of datastreams are considered bit equivalent in packaging formats, the OAI-PMH *datestamp* must change in the By Reference approach, as it would in a By Value approach. As a result, the addition of new *resources* or the modification of existing *resources* can be detected using the OAI-PMH *datestamp* of the package representing the *resource*.
- Packaging formats provide a uniform *resource* harvesting solution both when the *resource* is simple (a single datastream) and when it is compound (consisting of multiple datastreams).

- Packaging formats provide the ability to disambiguate between identifiers and locators of *resources* (or datastreams), and even to disambiguate between identifiers and locators of *resources* and those of secondary information.
- When packaging formats are used within OAI-PMH, properties such as *set*-membership and the use of `about` containers (to convey secondary data pertaining to the OAI-PMH *metadata*) apply with consistent semantics to *metadata records* that are packaged digital asset representations just as they do to other *metadata records*.

Since its inception, the OAI-PMH has established itself as an important solution for exchanging descriptive metadata among a large and varied group of digital repositories. Perhaps one of the most underrated reasons of this success is its simple model of *metadata* harvesting. The OAI-PMH does not provide (distributed) services across various repositories; it simply allows bringing descriptive metadata together in one place. In order to provide services, the harvesting approach must be combined with other mechanisms.

As broached in the introduction of this Chapter, it should be mentioned that, in contrast with standards such as XML Information Set (Cowin & Tobin, 2004) and NISO OpenURL (NISO, 2005), the OAI-PMH framework is not specified in a generic way. The OAI-PMH is very dependent on the HTTP protocol and the XML syntax for transporting and serializing the interchanged *metadata*, respectively. While this approach proves to be satisfactory in the current technological environment, it may prove to be inadequate as technologies evolve. Providing an abstract model of OAI-PMH interactions, in a manner that is neutral to the transport protocol and response format, would be a change for the better. New and updated protocols and representation techniques could be employed, while the concepts underlying the OAI-PMH would persist over time.

4. The OpenURL Framework for Context-Sensitive Services

The NISO OpenURL standard originated from the scholarly information community. Within that community, the initial OpenURL 0.1 specification (Van de Sompel, Hochstenbach & Beit-Arie, 2000) – the precursor of the NISO OpenURL standard – was introduced for the specific purpose of reference-linking and was targeted at facilitating the provision of context-sensitive service links for popular types of scholarly works such as journal articles and books (Van de Sompel & Beit-Arie, 2001b).

The result of a consumer activating such an OpenURL 0.1 link is the transport of a description of a scholarly publication, for example a journal article, to a user-specific linking server. Hereby, identifiers and metadata describing the resource are conveyed using a controlled-vocabulary and are transported inline as the query string of an HTTP GET request. The linking server uses a knowledge base – that records holdings, subscriptions and preference information specific to the organization hosting the server – to provide a user with appropriate services and 'appropriate copies' (Beit-Arie et al., 2001; Van de Sompel & Beit-Arie, 2001a; Paskin, 2003) pertaining to the requested resource.

A generalization of the essential components of the initial OpenURL 0.1 solution, beyond the scholarly information field, inspired the very nature of the OpenURL Framework. This ANSI/NISO standard (NISO, 2005) defines a generalized framework for creating community and application-oriented *OpenURL Framework Applications*. Simply put, an *OpenURL Framework Application* is a networked service environment, in which packages of information are transported over a network. The main purpose of the transportation of these packages is to request and obtain context-sensitive services pertaining to a referenced resource. In order to do so, each package describes the referenced resource itself, the network context in which the resource is referenced, and the context in which the service request takes place.

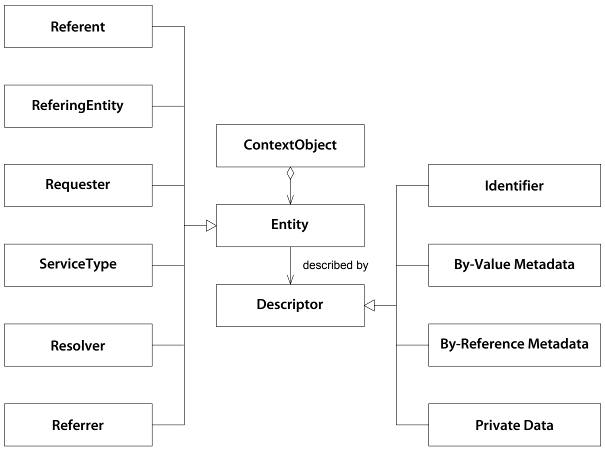


Figure 11. Each *Entity* of a *ContextObject* is specified using a *Descriptor*; four *Descriptor* types can be used simultaneously.

To that end, the NISO OpenURL standard introduces the notion of a *ContextObject*; an abstract information construct that contains descriptions of various *Entities* involved in the process of providing context-sensitive services. The different *Entities* are shown in Table 10 and are also depicted in Figure 11. In what follows, pre-defined terms provided by the NISO OpenURL standard are written in *italic* font.

A *ContextObject* can be transported to a networked system, named a *Resolver*, in order to request services pertaining to the *Referent* described in it. To decide upon the nature of such services, the *Resolver* may take *Entities* other than the *Referent* into account. These other *Entities* are *ReferringEntity*, *Requester*, *Referrer*, and *ServiceType*. Each *Entity* of a *Context-Object* can be described by means of so-called *Descriptors*. The standard distinguishes between four types of *Descriptors*: *Identifier Descriptors*, *Metadata Descriptors* and *Private Data Descriptors*. All types of *Descriptors* can be used simultaneously. An overview is provided in Table 11.

Entity	Definition
Referent	The *Entity* that is referenced in a networked environment and about which the *ContextObject* is created
ReferringEntity	The *Entity* that references the *Referent*
Requester	The *Entity* that requests services pertaining to the *Referent*
ServiceType	The *Entity* that defines the type of service requested
Resolver	The *Entity* at which a request for services is targeted
Referrer	The *Entity* that generates the *ContextObject*

Table 10. Six possible *Entities* of a *ContextObject* in the NISO OpenURL Framework

Descriptor	Definition
Identifier	This *Descriptor* unambiguously specifies the *Entity* by means of a URI. This URI either points to the *Entity* itself or to descriptive metadata that specify the *Entity*.
By-Value Metadata	This *Descriptor* specifies properties of the *Entity* by the combination of 1) a URI reference to a *Metadata Format*; and 2) a particular instance of descriptive metadata about the *Entity* expressed according to this *Metadata Format*.
By-Reference Metadata	This *Descriptor* specifies properties of the *Entity* by the combination of 1) a URI reference to a *Metadata Format*; and 2) the network location of a particular instance of descriptive metadata about the *Entity* expressed according to this *Metadata Format*.
Private Data	This *Descriptor* specifies information about the *Entity* using a method not defined in the OpenURL Framework standard. The *Resolver* and the *Referrer* have a common understanding of the *Descriptor* based on a bilateral agreement.

Table 11. Four possible methods to specify information about an *Entity*

The NISO OpenURL standard makes a clear distinction between the abstract definition of the above concepts and their concrete representation and the protocol by which such representations are transported. The OpenURL Framework allows for a *ContextObject* to be represented in many different *Formats* and transported using many different *Transports*, as technologies evolve. Yet, the concepts underlying the OpenURL Framework persist over time.

To address the issue of open-endedness, and allow for the creation of highly interoperable solutions, an OpenURL Framework *Registry* is introduced [http://www.openurl.info/registry], providing a mechanism for the public disclosure of specific selections of technologies for the representation and transportation of *ContextObjects*. In general, a community or application domain defines an instantiation of the OpenURL Framework by selecting entries from the *Registry* to represent and transport *ContextObjects*. If necessary and/or desired, the community may define and register new entries. The selections are bundled in a so-called *Community Profile*. Based on such a *Community Profile*, *OpenURL Framework Applications* can be built.

Currently, a Key/Encoded-Value (KEV) *ContextObject Format* and an XML *ContextObject Format* have been defined and are registered. The KEV *ContextObject Format* defines how to represent a *ContextObject* as a concatenation of ampersand delimited Key/Encoded-Value pairs. The XML *ContextObject Format* defines how to represent one or more *ContextObjects* as an XML (Bray et al., 2004) document.

Also, the transport of a *ContextObject* can occur using various network protocols. The OpenURL *Registry* already defines several methods to convey *ContextObject* over a network. All methods use the HTTP (Fielding et al., 1999) and HTTPS protocols (Rescorla, 2000). Again, communities may use these registered *Transports*, and/or may choose to create and register new instances of *Transports*. For example, a community could consider defining a SOAP-based (Gudgin et al., 2003) *Transport* for *ContextObjects* in XML *Format*.

The example below illustrates a service request conformant with the San Antonio Level 1 *Community Profile* [http://www.openurl.info/registry/docs/pro/info:ofi/pro:sap1-2004]. This *Community Profile* targets the deployment of *OpenURL Framework Applications* in the scholarly-information community and provides an elegant migration path from the OpenURL 0.1 specification to the NISO OpenURL standard. The San Antonio Level 1 *Community Profile* builds on the KEV *ContextObject Format* and the HTTP(S) GET *Transports*. The sample service request below describes a journal article. KEV pairs are concatenated with the ampersand character to form the query string of the HTTP GET request. For readability, the query string is not URL encoded.

```
[ baseURL?url_ver=Z39.88-2004&
      url_ctx_fmt=info:ofi/fmt:kev:mtx:ctx&
      rft_val_fmt=info:ofi/fmt:kev:mtx:journal&
      rft.genre=article&
      rft.atitle=aDORe: a Standards-based Digital Object Repository&
      rft.jtitle=The Computer Journal&
      rft.aulast=Van de Sompel&
      rft.auinit=H ]
```

The base URL of the *Transport* specifies the target of the transportation or the network location of the target *Resolver*. The query string (carrying the *ContextObject*) is appended to the base URL, and separated from it by a question mark. The query string starts with the KEV pair '`url_ver=Z39.88-2004`', indicating the version of the OpenURL standard used to design the query string. Next, the key '`url_ctx_fmt`' specifies the *Format* of the transported *ContextObject* representation. In the above sample, the value assigned to the '`url_ctx_fmt`' key is '`info:ofi/fmt:kev:mtx:ctx`', or the *Registry* identifier of the KEV *ContextObject Format*. Next, the *Referent* of the *ContextObject* is described using a *By-Value Metadata Descriptor*. The latter consists of two parts (see also Row 2 of Table 11). The first part is a KEV pair that declares the *Metadata Format* '`rft_val_fmt=info:ofi/ fmt:kev:mtx:journal`'. The second part is a set of KEV pairs that specify the actual metadata in the specified *Metadata Format*. These KEV pairs have keys with the '`rft.`' prefix to indicate that they describe the *Referent*.

The *Referent Entity* may also be described using an *Identifier Descriptor*. Assuming the referent journal article has identifier '`info:doi/10.1093/comjnl/bxh114`', the above HTTP GET method could be written as:

```
[ baseURL?url_ver=Z39.88-2004&
      url_ctx_fmt=info:ofi/fmt:kev:mtx:ctx&
      rft_id=info:doi/10.1093/comjnl/bxh114 ]
```

Many of today's information providers can generate and output OpenURL links, including Elsevier, PubMed, JSTOR and ISI. OpenURL linking servers are available from such companies as ExLibris (SFX), Endeavor (LinkFinderPlus) and EBSCO (LinkSource). An important illustration of the ongoing penetration of the OpenURL Framework is the adoption of the OpenURL Framework by Google and Microsoft. Instead of relying on the links to publishers' websites, Google and Microsoft provide OpenURL links to locate the full-text of scholarly publications found in Google Scholar and Windows Life Academic Search, respectively. The former can be accessed at [http://scholar.google.com]. The latter can be accessed at [http://academic.live.com/].

5. Conclusion

In the preceding pages, several technologies have been described: A data model (viz. the OAIS Reference Model), a packaging format (viz. MPEG-21 DID), a harvest protocol (viz. the OAI-PMH) and an obtain framework (viz. the OpenURL Framework for Context-Sensitive Services). As will be shown later on, when used selectively in combination, these technologies can be employed for the creation of persistent repository interfaces for harvesting and obtaining digital assets.

Section 1 of this Chapter described several facets of the OAIS Reference Model. The author has shown that the OAIS Reference Model recognizes two levels of identification: The identification of the actual content, and the identification of the package that represents and wraps that content. The author has also shown how both levels of identification can be employed as a means to manage and identify different versions of content.

The succeeding section focused on the MPEG-21 DID standard. The MPEG-21 DID Abstract Model was introduced, and a step-by-step example of its serialization in XML has been provided. The author has shown how terminology and concepts of the MPEG-21 DID standard can be joined and mapped to the OAIS Information Model. Also, several important contributions made by the author to the MPEG-21 standardization process have been highlighted. The greater part of these contribution are directly related to the applicability of MPEG-21 technology in the information domain, in general, and the digital repository experiments described in the next Chapter, in particular. Such contributions include the extension of datastream-level provision techniques for MPEG-21 DIDL, the addition of the `DIDLInfo` element and the `DIDLDocumentId` attribute, and the introduction of a technique to express and qualify the functional granularity of the relationships between various intellectual variants by combining MPEG-21 DII and MPEG-21 RDD. It is also worth mentioning that based on those contributions, a second Edition of MPEG-21 DID was initiated, as well as an amendment on MPEG-21 DII and an amendment on MPEG-21 RDD.

Generally, it is fair to say that, MPEG-21 DID as a format for packaging compound digital assets is a feasible, and actually attractive option. From a strategic perspective, MPEG-21 DID is appealing because it is part of the MPEG suite of ISO standards and has been developed by major players in the content and technology industry, which provides some guarantees regarding its adoption and the emergence of compliant tools. From a functional perspective, MPEG-21 DID is attractive because of the existence of a well-specified Abstract Model, and the capability this yields for maintaining the basic architectural concepts in the long term. Also, MPEG-21 DID takes an approach that enforces cross-community interoperability, while allowing the flexibility for the emergence of compliant, community-specific profiles. The XML Schema of MPEG-21 DIDL is elegant and simple, making the development of tools quite straightforward. Especially appealing is the flexibility and extensibility provided by the *descriptor/statement* approach, represented in MPEG-21 DIDL by means of the `Descriptor/Statement` elements and XML attributes. The MPEG-21 standard itself makes use of those *descriptor/statement* constructs to provide a fundamental solution for the identification of Digital Items, to associate processing methods with Digital Items, and to express rights related to Digital Items. *Descriptor/statement* constructs can also be used to address community-specific interoperability requirements.

The chapter continued with a brief explanation of the OAI-PMH. The OAI-PMH presents itself as an interoperability framework that allows XML-based descriptive metadata to be harvested on a recurrent basis. The author has argued that also digital *resources*, and not just *metadata* about those *resources*, can be harvested using the OAI-PMH. To this end, he expanded the scope of descriptive metadata to be more than just Dublin Core, MARCXML and similar bibliographic formats. He used *metadata formats* that are more complex, expressive, and accurate in their description of digital *resources* by introducing packaging formats such as MPEG-21 DIDL. The combination of these packaging formats with the OAI-PMH results in an unambiguous mechanism for harvesting *resources*. By introducing a new OAI-PMH *datestamp* whenever a constituent of the represented *resource* changes, the technique also yields an unambiguous trigger for harvesting *resources* that were created or modified within a specified date range.

The chapter ended with a brief description of NISO's OpenURL Framework for Context-Sensitive Harvesting. The author explained how the OpenURL standard enables devising an environment for obtaining context-sensitive services pertaining to an identified digital asset. The OpenURL standard defines an abstract data model that persists over time, while implementations of the model may change as the technological environment evolves. Also, the OpenURL standard allows for the creation of a so-called rich service environment, by defining a pre-defined set of concepts that enable conveying context-sensitive information, such as the context in which the service request has taken place. This is a feature that is almost non-existent in related technologies, such as WebDAV. The OpenURL technology has already received a high level of buy-in from the scholarly information community, and is being used by such companies as Elsevier, Google and Microsoft.

Ch.2. Experiments conducted at the Los Alamos National Laboratory

Ch.2. Experiments conducted at the Los Alamos National Laboratory

So far, a brief overview of important concepts and technologies has been given. This chapter introduces two experiments that explore the use of standards-based interfaces for harvesting and obtaining assets from digital repositories, by building on the combination of the technologies presented in Chapter 1. In the experiments, two sets of technical interfaces are devised. A first set focuses on the harvest of XML-based packages of digital assets, by building on the combination of the OAI-PMH framework and the MPEG-21 DID standard. A second set is centered on NISO's OpenURL Framework for Context-Sensitive Services and enables the dissemination of digital assets, as well as of their constituent datastreams, on a one-by-one basis. The thinking underlying both sets of interfaces lays the foundation for the theoretical interface descriptions presented in Chapter 3. Several facets of the experiments discussed in this Chapter may not be of direct importance for the actual functioning of the harvest and obtain interfaces that are subject of this study; yet, these parts may prove to be helpful to set the scene of the actual experiments.

- A first experiment, described in Section 1, focuses on the multifaceted use of the OAI-PMH and NISO's OpenURL as information access protocols in a repository architecture designed and implemented for ingesting, storing, and accessing a vast collection of digital assets at the Research Library of the Los Alamos National Laboratory (LANL). The repository architecture is named aDORe and its design and deployment have been carried out by the LANL Research Library. The aDORe architecture is highly modular and standards-based. In the architecture, the MPEG-21 Digital Item Declaration Language is used as the XML-based format to represent and package compound digital assets. Upon ingestion, the DIDL documents that package those assets are stored and exposed in a multitude of Autonomous OAI-PMH Repositories. An OAI-PMH compliant Repository Index keeps track of the creation and location of all those repositories, whereas an OAI-PMH compliant Identifier Locator keeps track of the location of individual (versions of) digital assets. In addition, two read-only interfaces to the complete environment are introduced: The OAI-PMH Federator and the aDORe OpenURL *Resolver*. The OAI-PMH Federator is introduced as a single-point-of-access to downstream harvesters. It hides the complexity of the aDORe environment to those harvesters, by providing a single interface through which batches of stored DIDL documents can be requested. In terms of the OAIS Reference Model, this interface enables requesting batches of OAIS Dissemination Information Packages (OAIS DIPs). A second interface, the aDORe OpenURL *Resolver*, provides an interface to the aDORe environment from which various disseminations of an individual digital asset (again packaged as a DIDL document) or its constituent datastreams can be obtained using service requests that are compliant with NISO's OpenURL standard. The aDORe OpenURL *Resolver* interacts with other components of the environment mainly using the OAI-PMH. Both the OAI-PMH Federator and the aDORe OpenURL *Resolver* may call upon a Transform Engine, when crosswalks of stored DIDL documents or transformations of stored datastreams are requested. In terms of the

OAIS Reference Model, this interface supports obtaining OAIS DIPs and contained datastreams on a one-by-one basis.

- Section 2 of this Chapter describes the results of a second experiment, conducted by the LANL Research Library and the American Physical Society (APS), aimed at designing and implementing a robust solution for the recurrent and accurate transfer of newly added and updated digital assets from the APS collection to the LANL aDORe environment. Again, various recent standards are combined to obtain a framework that should be attractive as a means to optimize content transfer in environments beyond the specific APS/LANL project. Of specific importance is the MPEG-21 DIDL, which is employed for the application-neutral representation of compound digital assets of all sorts. The solution also uses the OAI-PMH as a REST-based protocol that allows incrementally collecting new and updated assets, represented as DIDL documents, from the APS. Through periodic OAI-PMH harvesting, LANL collects updated and added content from the APS, thereby relying on the semantics of the OAI-PMH *datestamp* for exposed compound digital assets to ensure timely duplication of content. Also, it builds on an XML-specific technique – the W3C XML Signatures and Processing standard – to provide guarantees regarding accuracy and authenticity of the transferred assets. Results from this experiment suggest that the tested approach can eventually be successfully brought into production. Although the project has its origin in a specific publisher-to-library use case, it aims to explore a broadly applicable solution for the transfer of assets between a content provider and a content consumer.

Both experiments have been conducted at the Research Library of the Los Alamos National Laboratory and were supervised by dr. Herbert Van de Sompel who I would like to thank for his invaluable guidance.

1. The multi-faceted use of the OAI-PMH and the NISO OpenURL standard in the aDORe environment

When compared with most academic and research libraries, the Research Library of LANL follows a rather unique strategy with respect to providing access to digital scholarly information. The general trend in digital library services is to have users access externally hosted materials through third party services, federated through a locally hosted Web Portal. In order to be self-supporting with respect to mission-critical scholarly information, the LANL library acquires or licenses a vast collection of scholarly digital assets, hosts those digital assets locally, and makes them accessible through locally developed user services. The locally hosted digital assets include secondary data feeds from Biosciences Information Service (BIOSIS), INSPEC, Thomson Scientific, and primary information feeds from major scholarly publishers such as the American Physical Society, the Institute of Physics, Elsevier and John Wiley & Sons. In addition to that, the LANL Research Library is actively investigating the deployment of institutional repository capabilities to host locally created materials such as technical reports, datasets, videotaped presentations, and so on. Also, research is underway to augment the locally hosted collection with materials gathered by focused Web crawling and to include logs detailing the usage of repository assets in the repository as assets in their own right (Bollen & Luce, 2002; Van de Sompel, Young & Hickley, 2003).

At the time of writing this dissertation, the collection amounts to around one hundred million locally hosted assets. In many cases, these digital assets are complex in the sense that they consist of multiple constituent datastreams that jointly form a single logical unit. That logical unit can, for example, be a scholarly publication comprising a research paper in both PDF and ASCII format, secondary data describing the paper, references made in the paper expressed in XML format, auxiliary datastreams such as images and videos in various formats, including TIFF, JPEG, MPEG-4, and so forth.

Over the last years, the author has been involved in an effort guided by the Digital Library Research and Prototyping Team of the LANL Research Library to design the aDORe repository architecture: A repository system aimed at ingesting, storing, and making accessible to downstream applications an ever growing heterogeneous digital collection. While the aDORe design was inspired by requirements imposed by LANL, the architecture has properties that appear to be attractive for other repository projects, including repository federations. Such properties are:

- Standards-based design: Throughout the architecture, standards or de facto standards are used. These include the MPEG-21 Digital Item Declaration (MPEG-21 DID), the MPEG-21 Digital Item Identification (MPEG-21 DII), the Open Archives Protocol for Metadata Harvesting (OAI-PMH), the NISO OpenURL Framework for Context-Sensitive Services (NISO OpenURL), the info URI Scheme, and the Internet Archive ARCfile format.
- Natively component-based, distributed design: The architecture operates on the basis of various autonomous components; interaction with those components is protocol-based.
- The dynamic binding of dissemination services to stored digital assets and constituent datastreams.

Figure 12 introduces the major components of the aDORe architecture. All components are explained in detail in what follows and a summary is provided here. The ingestion process is positioned at the extreme right hand side, while downstream applications that interact with the repository are situated at the extreme left hand side.

- At the right hand side, Figure 12 shows an 'expose' box containing a multitude of *Autonomous OAI-PMH Repositories*. Each of these autonomous repositories stores a collection of DIDL documents each of which represents and packages a digital asset (Digital Item in MPEG-21 terminology) in accordance with the MPEG-21 DID specification (ISO, 2005a). Section 1.1 is dedicated to detailing aspects related to this packaging process and these Autonomous OAI-PMH Repositories.
- The *Repository Index* is sited in the 'locate' box. It is a registry that keeps track of the creation and location of Autonomous OAI-PMH Repositories in the aDORe environment. This component, which is also accessible through the OAI-PMH, is detailed in Section 1.2.
- The *Identifier Locator* is shown below the Repository Index. For each DIDL document stored in aDORe, this component contains the identifiers associated with the DIDL document itself and with the Digital Item it represents. It also contains the location of the Autonomous OAI-PMH Repository in which the DIDL document and, hence, the digital asset reside. When multiple versions of the same digital asset exist, the Identifier Locator

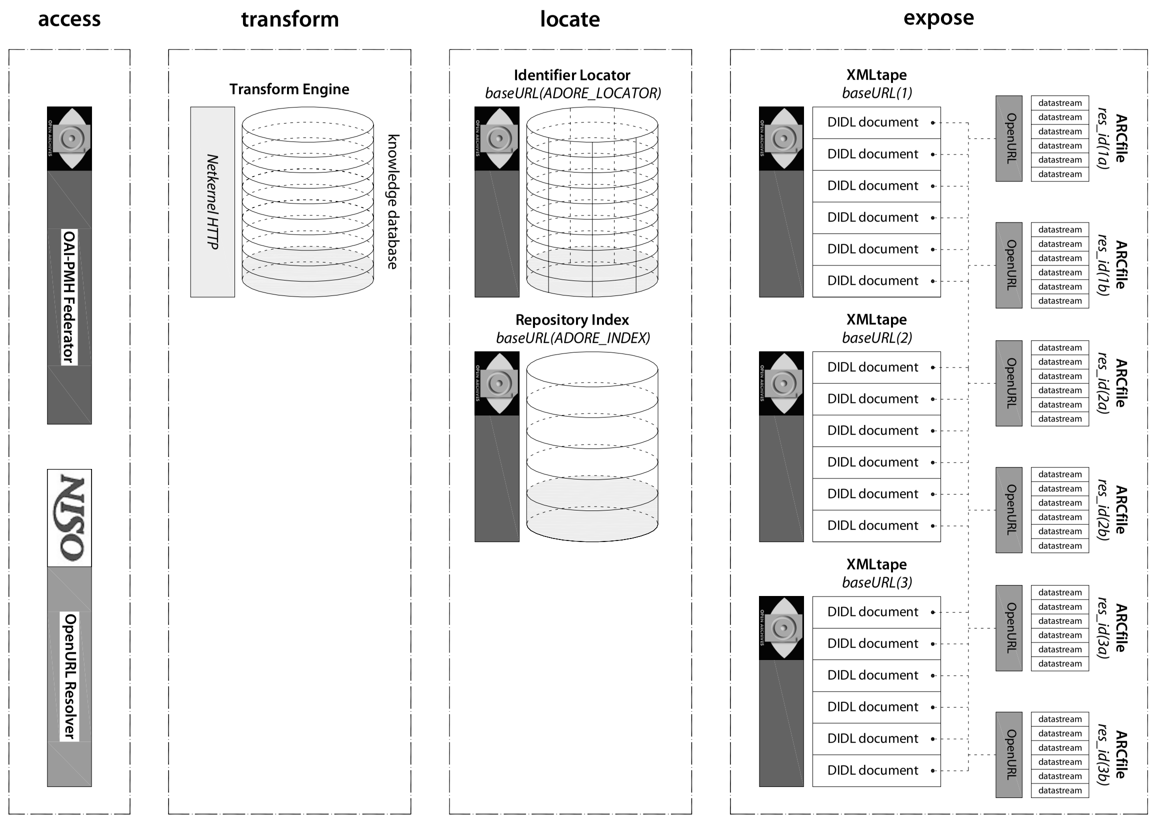


Figure 12. Building blocks of the aDORe architecture

- keeps track of all locations. The Identifier Locator can be populated through batch loading or OAI-PMH harvesting. It can be queried in a variety of ways, including the Handle System Protocol (Sun et al., 2003). This component is described in Section 1.3.
- To the left of the Repository Index, Figure 12 shows the *Transform Engine* and its associated *Knowledge Database*. These components are introduced to allow the transformation of stored DIDL documents, digital assets and their constituent datastreams. Section 1.4 provides a brief insight into their operation.
- At the top of the 'access' box, Figure 12 shows the *OAI-PMH Federator*. This component turns the complete aDORe environment into a single OAI-PMH repository, thereby hiding all architectural details and complexities from downstream harvesters. The OAI-PMH Federator becomes the single point of access for off-the-shelf OAI-PMH harvesters aiming to collect batches of DIDL documents from the aDORe environment. The OAI-PMH Federator interacts with other components of the environment mainly using the OAI-PMH. It calls upon the Transform Engine when transformations of stored DIDL documents are requested, rather than the stored DIDL documents themselves. Details are provided in Section 1.5.1.
- Finally, below the OAI-PMH Federator, Figure 12 introduces the *aDORe OpenURL Resolver*. This component provides an interface to the aDORe environment from which various disseminations of an individual digital asset or its constituent datastreams can be obtained using service requests that are compliant with the NISO OpenURL standard. The aDORe OpenURL *Resolver* interacts with other components of the environment mainly using the OAI-PMH. It calls upon the Transform Engine to deliver the requested disseminations. The aDORe OpenURL *Resolver* is described in Section 1.5.2.

1.1. Ingesting digital assets in the aDORe environment.

1.1.1. Using MPEG-21 DID for representing digital assets

The complex nature of the digital assets to be ingested into the aDORe environment led to an investigation regarding existing approaches to represent and package those compound assets as OAIS Archival Information Packages (AIPs). Given the incontestable dominance of XML in the current technological environment, this quickly led to an interest in XML-based packaging formats, which itself resulted in the selection of the MPEG-21 DID (ISO, 2005a) as the sole technique to model and serialize OAIS AIPs stored in aDORe. An overview of strategical and technical qualifications of MPEG-21 DID is provided in the previous chapter.

component	genre	MIME media type	identifier
digital asset	scholarly publication	n/a	info:doi/10.1103/ PhysRevB.69.174413
datastream 1	Abstract & Indexes	application/xml (MARCXML)	-
datastream 2	full-text file	application/pdf	-

Table 12. A list of components of the sample digital asset

For comprehensibility, in the remainder of this Chapter, a sample digital asset will be used to illustrate several design choices made. The main characteristics of the sample asset are listed in Table 12. The DIDL document representing and packaging the sample asset is shown in Table 13.

At LANL, digital assets to be ingested in aDORe can, in principle, be obtained in a variety of ways including FTP, OAI-PMH resource harvesting (see also Section 2 of this Chapter), Web crawling and delivery on physical media. An ingestion process has been devised that represents each digital asset according to the MPEG-21 DID specification. Hereby, a DIDL document representing and packaging the digital asset is created. In the MPEG-21 world, the digital asset itself is equated with a Digital Item. The core characteristics of a DIDL document created by the ingestion process are explained below. The explanation is given in terms of the XML elements of the DIDL document. As shown in Section 2.4.2 of Chapter 1, each such XML element corresponds to an entity of the Abstract Model defined by MPEG-21 DID.

□ In accordance with MPEG-21 DIDL, a digital asset (or Digital Item) is represented by an `Item` element. Constituents of the digital asset are provided as child elements of this `Item`. Hence, the scholarly publication, shown in Table 12, is represented as an `Item`.

□ In accordance with MPEG-21 DIDL, the structure of a Digital Item can be conveyed by providing (nested) sub-`Item` elements as child elements of the top-level `Item`. These sub-`Item` elements represent the individual Digital Items that make up the composite asset. In aDORe, sub-`Item` elements are used whenever the constituent of the Digital Item has a Digital Item Identifier in its own right, whereas `Component` elements are used when the constituent does not. As can be seen from Table 12, only the scholarly publication itself has an identifier. As a consequence, all constituents of the scholarly publication are provided as `Component` elements that are direct children of the top-level `Item`. Note that also the Abstract and Indexes are embedded as an autonomous `Component` element in its own right. While this might be contrary to the mainstream MPEG-21 DID approach – where auxiliary information is conveyed using `Descriptor/Statement` constructs – a case in its favour can be made. First, descriptive metadata records (or abstract & Indexes) should also be the subject of digital preservation, and as a result be treated as potentially endangered datastreams in their own right. Second, as technologies evolve and datastreams of varying media-types become directly searchable, the special status of descriptive metadata as the sole point of entry to those datastreams might eventually weaken, turning descriptive metadata and datastreams into peers. As will be shown, the relationship between the descriptive metadata record and the content it describes is represented by means of a special-purpose `Descriptor/Statement` construct.

□ In accordance with the MPEG-21 DIDL, a constituent datastream of a Digital Item is provided in a `Resource` element. As demonstrated in Section 2.5.1 of Chapter 1, MPEG-21 DIDL allows those datastreams to be physically embedded within a DIDL document (i.e., By Value provision) or to be pointed at from within a DIDL document (i.e., By Reference provision).

□ Apart from the primary constituent datastreams, a DIDL document also contains secondary data pertaining to the Digital Item and/or parts thereof. Some of this secondary data is directly taken from the digital asset as it was obtained from the information provider. Other secondary data is added by the ingestion process and is crucial for the functioning of aDORe and its directly associated processes. This secondary data is provided by means of `Descriptor/Statement` constructs or XML attributes. In accordance with MPEG-21 DID, application-specific `Descriptor/Statement` constructs are used to convey information about a digital asset and its constituent datastreams, while the `DIDLInfo` child element of the `DIDL` root element, as well as attributes attached to that root element, are used to convey information about the DIDL document itself.

  - As will be described in Section 1.1.1.1, each `Item` element has a `Descriptor/Statement` construct conveying the Digital Item Identifier of the digital asset. In the OAIS Information Model (ISO, 2003a), this identifier corresponds with an OAIS Content Information Identifier. In contrast, OAIS AIP Identifiers are conveyed by the `DIDL-documentId` attribute of the `DIDL` root element.
  - As will be described in Section 1.1.1.2, `Component` elements may have a `Descriptor/Statement` construct conveying an XML signature computed over the datastream provided by the `Resource` child element of that `Component`. In addition, enveloped XML signatures calculated over the DIDL document itself are embedded in a `DIDLInfo` element of that DIDL document.
  - Other `Descriptor/Statement` constructs are inserted, as required. For example, domain-specific `Descriptor/Statement` constructs are introduced to express relationships within and among digital assets (See Section 1.1.1.3). Also, different creation datetimes and media format information are associated with digital assets, constituent datastreams and DIDL documents. (See Section 1.1.1.4).

- The top-level `Item` is embedded in the `DIDL` root element to obtain a DIDL XML document that is the OAIS Information Package for the digital asset.

```
<?xml version="1.0" encoding="UTF-8"?>
<didl:DIDL DIDLDocumentId="info:lanl-repo/i/58f202ac"
  diext:DIDLDocumentCreated="2005-12-03T11:57:12Z"
  xmlns:didl="urn:mpeg:mpeg21:2002:02-DIDL-NS"
  xmlns:diext=http://library.lanl.gov/2005-08/aDORe/DIDLextension/">
  <didl:DIDLInfo>
    <dsig:Signature
      xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
      <!-- XML signature of DIDL document -->
    </dsig:Signature>
  </didl:DIDLInfo>
  <!--  Item representing the scholarly publication -->
  <didl:Item id="uuid-00005e90">
    <didl:Descriptor>
      <didl:Statement mimeType="application/xml; charset=utf-8">
        <!--  Native identifier of the scholarly publication -->
        <dii:Identifier xmlns:dii="urn:mpeg:mpeg21:2002:01-DII-NS">
          info:doi/10.1103/PhysRevB.69.174413</dii:Identifier>
      </didl:Statement>
    </didl:Descriptor>
    <didl:Component id="uuid-0000a01c">
      <didl:Descriptor>
        <didl:Statement mimeType="application/xml; charset=utf-8">
          <dsig:Signature
          xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
          <!-- XML signature calculated over the descriptive metadata -->
        </dsig:Signature>
      </didl:Statement>
    </didl:Descriptor>
    <didl:Descriptor>
      <didl:Statement mimeType="application/xml; charset=utf-8">
        <diadm:Admin xmlns:diadm="http://library.lanl.gov/2004-01/STB-RL/DIADM">
          <!--  Media format identifier of the MARCXML metadata -->
          <dc:format xmlns:dc="http://purl.org/dc/elements/1.1/">
            info:lanl-repo/fmt/3</dc:format>
          <!--  datetime of creation of the MARCXML metadata -->
          <dcterms:created xmlns:dcterms="http://purl.org/dc/terms/">
            2004-01-15T18:42:28Z</dcterms:created>
        </diadm:Admin>
      </didl:Statement>
    </didl:Descriptor>
    <didl:Resource mimeType="application/xml; charset=utf-8">
        <record xmlns="http://www.loc.gov/MARC21/slim">
          <leader>01142cam 2200301 a 4500</leader>
            <controlfield tag="005">19930521155141.9</controlfield>
            <datafield tag="010" ind1=" " ind2=" ">
              <subfield code="a">92005291</subfield>
            </datafield>
            <datafield tag="042" ind1="" ind2="">
              <subfield code="a">lcac</subfield>
            </datafield>
            <datafield tag="050" ind1="0" ind2="0">
              <subfield code="a">PS3537.A618</subfield>
              <subfield code="b">A88 1993</subfield>
            </datafield>
            <datafield tag="082" ind1="0" ind2="0">
              <subfield code="a">811/.52</subfield>
              <subfield code="2">20</subfield>
            </datafield>
            ...
    </didl:Resource>
  </didl:Component>
  <didl:Component id="uuid-00004a42">
    <didl:Descriptor>
      <didl:Statement mimeType="application/xml; charset=utf-8">
        <dsig:Signature
          xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
          <!-- XML signature calculated over PDF datastream -->
        </dsig:Signature>
      </didl:Statement>
    </didl:Descriptor>
    <didl:Descriptor>
      <didl:Statement mimeType="application/xml; charset=utf-8">
        <diadm:Admin xmlns:diadm="http://library.lanl.gov/2004-01/STB-RL/DIADM">
          <!--  Media format identifier of the PDF datastream -->
          <dc:format xmlns:dc="http://purl.org/dc/elements/1.1/">
            info:lanl-repo/fmt/3</dc:format>
          <!--  datetime of creation of the PDF datastream -->
          <dcterms:created xmlns:dcterms="http://purl.org/dc/terms/">
            2004-01-29T11:12:33Z</dcterms:created>
        </diadm:Admin>
      </didl:Statement>
    </didl:Descriptor>
    <didl:Resource mimeType="application/pdf"
      ref="http://arc.lanl.gov:9000/arc-servlet/aps_20040506160150?
```

```
          rfr_id=info:sid/library.lanl.gov&
          url_ver=z39.88-2004&
          rft_id=info:lanl-repo/arc/0309024.pdf"/>
    </didl:Component>
  </didl:Item>
</didl:DIDL>
```

Table 13. A DIDL document representing the sample digital asset listed in Table 12.

1.1.1.1. Identifiers

A clear comprehension of the meaning and use of identifiers in the aDORe environment is crucial to understand its very design. aDORe uses two parallel identification mechanisms. Both are described below and illustrated in Figure 13 by means of the sample digital asset.

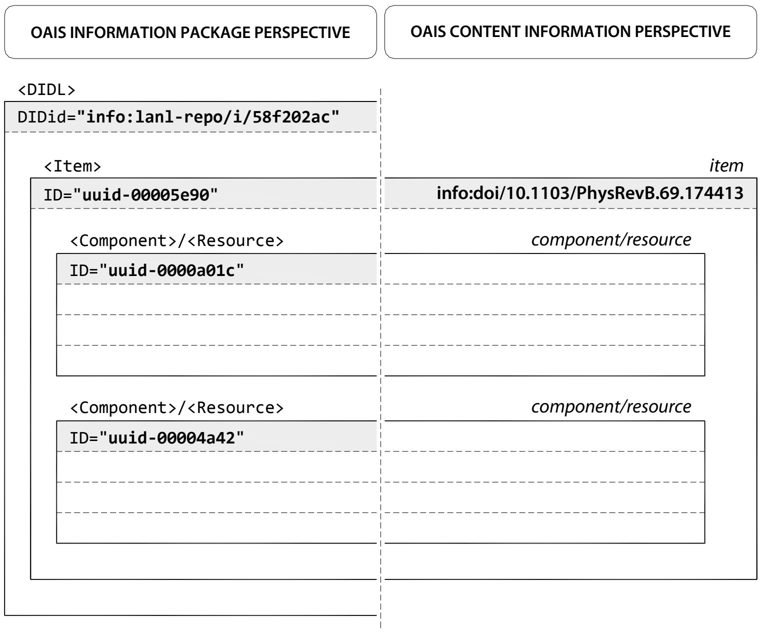


Figure 13. An OAIS perspective on Digital Item Identifiers, DIDL document Identifiers and fragment identifiers.

A. Digital Item Identifiers

The identifier(s) of the Digital Item correspond to what the OAIS Information Model (ISO, 2003a) categorizes as OAIS Content Information Identifiers. These identifiers are directly related to identifiers that are natively attached to digital assets before their ingestion into aDORe. In many cases such digital assets, or their constituent datastreams, have identifiers that were associated with them when they were created or published. According to MPEG-21 DII, Digital Item Identifiers are expressed as URIs. The Digital Item Identifier of the sample digital asset is '`info:doi/10.1103/PhysRevB.69.174413`'.

During the ingestion process, a `Descriptor/Statement` construct attached to the `Item` element is used to convey the Digital Item Identifier of the Digital Item, using the `Identifier` element of the MPEG-21 DII XML namespace (ISO, 2003c; Bekaert & Rump, 2005) (See also Section 2.5.3.1 of Chapter 1). This can be seen in Table 13, where the Item element has identifier '`info:doi/10.1103/PhysRevB.69.174413`'. As mentioned previously, the ingestion process will create sub-`Items` for all constituent datastreams of a digital asset that have Digital Item Identifiers. Because the constituent PDF and XML datastreams of the sample digital asset do not have a Digital Item Identifier, the ingestion process maps both datastreams to a `Component` element for which no Identifier element is provided. That `Component` element is a child of the `Item` element.

B. DIDL document Identifiers

The aforementioned `Item` representing the digital asset, is part of a DIDL document, which functions as an OAIS AIP in aDORe. During ingestion, this DIDL document itself is accorded a globally unique identifier, which the OAIS Information Model would categorize as an OAIS AIP Identifier (as explained in Section 1.1 of Chapter 1).

In aDORe, these DIDL document Identifiers are constructed using the Universally Unique Identifier (UUID) algorithm (Leach, Mealling & Salz, 2004), and are expressed as URIs in the '`info:lanl-repo`' namespace, which the LANL Research Library has registered under the 'info' URI Scheme (Van de Sompel, Hammond, Neylon & Weibel, 2005). As explained in Section 2.4.2 of Chapter 1, the identifier of each DIDL document is conveyed as the value of the `DIDLDocumentId` attribute of the `DIDL` root element. The value of the identifier of the sample DIDL document is '`info:lanl-repo/i/58f202ac`'.

Also, during the ingestion process, `Item` and `Component` elements receive globally unique XML IDs, again created using the UUID algorithm. As a result, these XML elements become globally addressable using a combination of the DIDL document Identifier of the DIDL document in which they are contained, and their own XML ID. In our sample digital asset, this addressing mechanism is as follows (for readability, UUID values have been shortened):

- The identifier of the DIDL document which represents the sample digital asset is '`info:lanl-repo/i/58f202ac`'.
- The `Item` that represents the digital asset has XML ID '`uuid-00005e90`'. As a result, it can be addressed as '`info:lanl-repo/i/58f202ac#uuid-00005e90`'. As was shown, using Digital Item Identifiers, it can also be addressed as '`info:doi/10.1103/PhysRevB.69.174413`'.
- The `Component` containing the descriptive metadata record has XML ID '`uuid-00004a42`'. As a result, it can be addressed as '`info:lanl-repo/i/58f202ac#uuid-00004a42`'. It cannot be addressed using a Digital Item Identifier.
- The `Component` containing the PDF datastream has XML ID '`uuid-0000a01c`'. As a result, it can be addressed as '`info:lanl-repo/i/58f202ac#uuid-0000a01c`'. It cannot be addressed using a Digital Item Identifier.

An important, OAIS-inspired, characteristic of the aDORe environment is that whenever a new version of a previously ingested digital asset needs to be ingested, a new DIDL document is created for it. Because of archiving reasons, and following the OAIS versioning strategy (see also Section 1.1 of Chapter 1), existing DIDL documents are never updated or edited. A new version of a digital asset may, for example, become available because an information provider delivers an updated version, or because a digital preservation strategy requires the migration of a file format of a datastream of the digital asset. As will be shown in Section 1.3, the Identifier Locator keeps track of all versions of a digital asset, by relying on the above described identification mechanisms.

All versions of a digital asset share a single (OAIS) Content Information Identifier, yet have a different Archival Package Information (AIP) Identifier. As such, each version of specific Content Information can be uniquely retrieved using the AIP Identifier of the AIP that encapsulates the particular version. The OAIS Content Information Identifier of the digital asset is conveyed by the `Identifier` element of the MPEG-21 DII XML namespace, which is typically inserted using a `Descriptor/Statement` construct into the `Item` that represents the Digital Item. The OAIS AIP Identifier is conveyed by the `DIDLDocumentId` attribute of the `DIDL` root element to identify a DIDL document, and this same OAIS AIP Identifier extended with a Fragment Identifier (expressed as XML IDs) is used to identify XML elements of the DIDL document.

1.1.1.2. Digests and signatures

XML signatures are used to allow verifying the accuracy and authenticity of assets stored in aDORe. The XML signatures are compliant with the W3C XML Signature and Processing standard (Bartel et al., 2002).

- To allow verifying the DIDL document itself, a `Signature` element from the W3C Signature XML namespace is provided as a child of the `DIDLInfo` element, which itself is a child of the `DIDL` root element.
- To allow verifying a datastream, a `Signature` element from the W3C Signature XML namespace is provided in a `Descriptor/Statement` construct attached to the `Component` that contains the datastream as a *resource*.

XML signatures are also used in conjunction with DIDL documents in the collaborative experiment between APS and the LANL Research Library, described in Section 2 of this Chapter. In this setup, aimed at permanently mirroring the collection of the APS at LANL, the OAI-PMH is used as a protocol to recurrently harvest assets represented as DIDL documents from the APS. In order to allow for the verification of the accurate transfer of the assets, XML signatures are included in the exposed DIDL documents.

1.1.1.3. Relationships

Investigations are ongoing aimed at introducing special-purpose `Descriptor/Statement` constructs to express relationships within and among Digital Items. These `Descriptor/Statement` constructs contain RDF (McBride, Manola & Miller, 2004) statements expressing relationships such as '`isMemberOf`' and '`isTranslationOf`'. The RDF statements are based on an experimental ontology of relationships that will be evolved towards interoperability in collaboration with colleagues from Cornell University, in the context of the Pathways project ("Pathways," n.d.; Bekaert, Liu, Van de Sompel, Lagoze, Payette, & Warner, in press).

It should be noted that the relationships provided in these `Descriptor/Statement` constructs are different from the structural relationships that are inherent to the structure of the DIDL document itself. Relationships, such as '`hasResource`', '`isPartOfItem`', and '`hasIdentifier`' can be dynamically derived from the structure of the DIDL document, and hence, do not necessarily need to be explicitly included as RDF statements within the DIDL documents.

1.1.1.4. Creation datetimes and media formats

Different creation datetimes are associated with Digital Items that are ingested in the LANL Repository:

- The creation date of a DIDL document is conveyed by means of a `DIDLDocumentCreated` attribute that is attached to the `DIDL` root element. This attribute is from an XML namespace defined as part of the aDORe effort. The value of this attribute is used as the datestamp for OAI-PMH access to aDORe. The creation datetime of the sample DIDL document shown in Table 13 is '`2004-11-22T18:07:18Z`'.
- When known, the creation datetime of a datastream of a Digital Item is conveyed using a `Descriptor/Statement` construct attached to the `Component` element that contains the datastream as a *resource*. This construct conveys the creation time using the `created` element from the Dublin Core Element Set (NISO, 2001), which was imported into an aDORe XML namespace. The creation dates of the PDF and Postscript files of the sample digital asset are '`2003-10-29T18:07:18Z`' and '`2003-10-26T10:03:12Z`', respectively.

Also, each `Component` element may have a `Descriptor/Statement` construct that conveys a unique media format identifier, capturing fine-grained media format information, of the constituent datastream contained inside the `Component`. The sample document in Table 13 show the use of `format` elements from the Dublin Core Element Set (NISO, 2001) to convey format identifiers of the PDF and XML datastreams of the sample asset.

At the time of writing, investigations are ongoing aimed at expressing these media formats by elements from the PREMIS effort (PREMIS, 2005). Also, the values of these elements could be expressed using controlled media format registries, such as the Global Digital Format Registry (Abrams & Seaman, 2003) and the PRONOM (Darlington, 2003) File Format registry. The latter can be accessed via the Web at [http://www.nationalarchives.gov.uk/ pronom/].

1.1.2. Storing digital assets in Autonomous OAI-PMH Repositories

Once a delivered asset has been turned into a DIDL document by the ingestion process, the DIDL document is stored in an OAI-PMH repository. Digital assets at LANL are typically received in large batches, since secondary or primary publishers that account for the bulk of the digital assets to be stored in the aDORe repository deliver weekly or annual feeds. In those cases, an Autonomous OAI-PMH Repository is created per delivered batch. As a result, many Autonomous OAI-PMH Repositories exist in the aDORe environment, each of which exposes DIDL documents.

While, for reasons of interoperability, the OAI-PMH mandates support for Simple Dublin Core, it strongly encourages supporting more expressive *metadata formats* that are defined through an XML Schema (Fallside, 2002). The Autonomous OAI-PMH Repositories in aDORe support *metadata* that are highly expressive and accurate in their representation of digital assets by using MPEG-21 DIDL as a *metadata format*, and hence by exposing DIDL documents as OAI-PMH *metadata* to harvesters.

Introducing packaging formats as highly expressive and accurate metadata in the OAI-PMH framework yields a robust and general solution to the resource-harvesting problem. Especially, as reported by Van de Sompel, Nelson, Lagoze and Warner (2004), unlike other approaches aimed at gathering *resources* (and not just *metadata*) based on OAI-PMH harvesting, an approach based on packaged digital assets guarantees that a change to any constituent of a *resource* will result in a change of the OAI-PMH *datestamp* of the package representing the *resource*. As a result, the OAI-PMH *datestamp* becomes a fully reliable trigger to incrementally harvest updated and added OAI-PMH *resources* when those *resources* are represented and encapsulated using an XML-based packaging format.

In the described solution, each Autonomous OAI-PMH Repository has the following characteristics:

□ It has a unique, persistent repository base URL, the HTTP address '`baseURL(N)`'.

□ Contained *records* are DIDL documents only

□ The *identifier* used by the OAI-PMH is the identifier of the DIDL document.

□ The *datestamp* used by the OAI-PMH is the datetime of creation of the DIDL document.

□ The only supported *metadata format* is MPEG-21 DIDL, with `metadataPrefix` '`didl`'.

□ The supported OAI-PMH time granularity is seconds-level.

□ *Set* structures may be supported, but will not be discussed in this dissertation.

As a result, harvesters operated by downstream service providers – such as indexing engines – can use the selective harvesting capabilities of the OAI-PMH (*datestamp* and *set*) to recurrently collect DIDL documents from the Autonomous OAI-PMH Repositories. Because DIDL documents are never updated, incremental harvesting will yield newly added DIDL documents only.

Because of this write-once/read-many process, a special storage solution, called the XMLtape/ARCfile, has been devised (Liu, Balakireva, Hochstenbach, and Van de Sompel, 2005). An XMLtape is an XML file that concatenates the XML-based representation of multiple digital assets. In the aDORe implementation of the XMLtape, the XML-based packages of digital assets are the DIDL documents. In order to keep these DIDL documents small and hence easy to process, datastreams of the digital asset are provided By Reference and are physically stored in Internet Archive ARCfiles. The technology underlying ARCfile (Burner & Kahle, 1996) is developed by Internet Archive to store data resulting from large-scale web crawling.

Both XMLtapes and ARCfiles can be accessed using a persistent standards-based protocol. Each XMLtape is exposed as an Autonomous OAI-PMH Repository with the above described characteristics. Each ARCfile is exposed as an OpenURL *Resolver*. The OpenURL *Resolver* has a base URL '`baseURL(ARC_FILE)`', which is an HTTP address that contains the identifier of the ARCfile to ensure its uniqueness. References embedded in the DIDL documents are

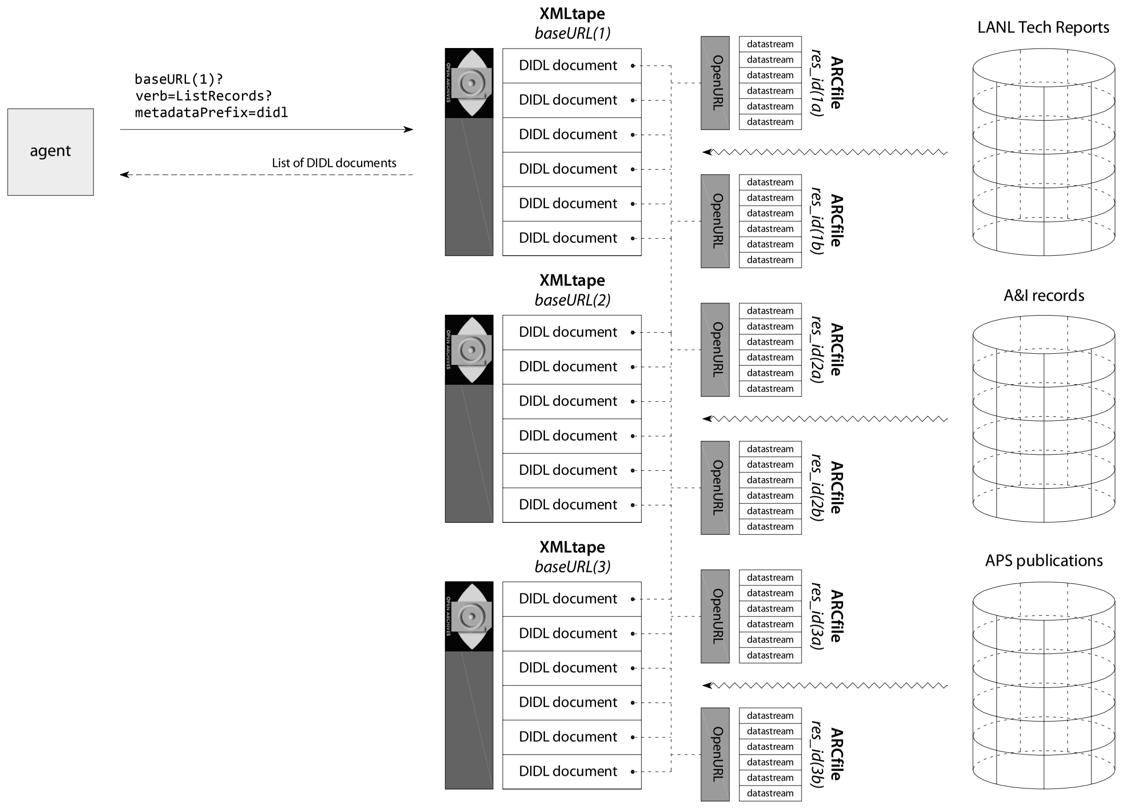


Figure 14. File-based storage of digital assets and constituent datastreams: XMLtapes and Internet Archive ARCfiles

compliant with the NISO OpenURL standard (NISO, 2005) and point to the datastreams stored in the ARCfiles. The *ContextObject* of the OpenURL link is expressed using the Key/Encoded-Value *ContextObject Format* and transported using the By Value HTTP GET method. The *Referent* of the *ContextObject* is a datastream stored in an ARCfile. The latter is described on the OpenURL by means of a unique datastream identifier. The sole service provided by the OpenURL *Resolver* is delivery of the datastream referenced on the OpenURL. The OpenURL link looks as follows:

```
[ baseURL(ARC_FILE)?
      url_ver=Z39.88-2004&
      rft_id=DatastreamKey ]
```

Interested readers are referred to (Liu et al., 2005) for a detailed explanation of the XMLtape/ARCfile solution. The aDORe implementation of the XMLtape/ARCfile solution is depicted in Figure 14

1.2. The Repository Index: A registry of Autonomous OAI-PMH Repositories

Using OAI-PMH selective harvesting techniques, updates can be harvested from the Autonomous OAI-PMH Repositories. However, the question remains unanswered as to how harvesters learn about the existence, addition and location of these Autonomous OAI-PMH Repositories. In order to provide this crucial intelligence the Repository Index is introduced. The Repository Index contains an entry for each Autonomous OAI-PMH Repository in the aDORe environment, with the following information:

- The repository base URL: The base URL of an Autonomous OAI-PMH Repository, which is a unique and persistent URI.
- The repository registration datetime: The time when the Autonomous OAI-PMH Repository became harvestable, by its very appearance in the Repository Index. This time is expressed as an ISO 8601 (ISO, 2004a; Wolf & Wicksteed, 1997) datetime with a time precision of seconds.
- Information pertaining to the creation of the Autonomous OAI-PMH Repository, such as the nature of its content.

The fact that the first two information elements of the Repository Index map directly to the notions of the *identifier* and the *datestamp* of the OAI-PMH, respectively, cannot be overlooked. Indeed, in aDORe, the Repository Index is exposed as an OAI-PMH repository in its own right, with the following properties:

- It has a unique, persistent base URL, the HTTP address ‘`baseURL(ADORE_INDEX)`’.
- Contained *records* comply with a locally defined XML-based *metadata format*, identified by the `metadataPrefix` ‘`index`’, which expresses *metadata* about the Autonomous OAI-PMH Repositories. An example *record* is shown in Table 14.
- The *identifier* used by the OAI-PMH is the repository base URL ‘`baseURL(N)`’.

- The *datestamp* used by the OAI-PMH is the repository registration datetime. There are no updates to *metadata* contained in the Repository Index, and hence this *datestamp* will never change and always remain equal to the time the Autonomous OAI-PMH Repository became available for harvesting in the aDORe environment.
- The supported OAI-PMH time granularity is seconds-level.
- *Set* structures may be supported. Typically, *set* structures would be used in the Repository Index to broadly categorize the nature or content of Autonomous OAI-PMH Repositories.

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/">
  <responseDate>2006-01-02T15:31:28Z</responseDate>
  <request identifier="http://purl.lanl.gov/aDORe/XMLtape/2005_4acb6e28"
    verb="GetRecord" metadataPrefix="didl">
    http://purl.lanl.gov/aDORe/RepoIndex/</request>
  <GetRecord>
    <record>
      <header>
        <identifier>http://purl.lanl.gov/aDORe/XMLtape/2005_4acb6e28</identifier>
        <datestamp>2005-12-03T11:58:14Z</datestamp>
        <setSpec>collection:info*3Asid*2Flibrary.lanl.gov*3Asci</setSpec>
      </header>
      <metadata>
        <idx:oaipmhRepository
          xmlns:idx="http://library.lanl.gov/2005-08/aDORe/RepoIndex/">
          <idx:oaipmhBaseURL>
            http://purl.lanl.gov/aDORe/XMLtape/2005_4acb6e28</idx:oaipmhBaseURL >
          <dcterms:created xmlns:dcterms="http://purl.org/dc/terms/">
            2005-12-03T11:58:14Z</dcterms:created>
          <dcterms:isPartOf xmlns:dcterms="http://purl.org/dc/terms/">
            info:sid/library.lanl.gov:sci</dcterms:isPartOf>
        </idx:oaipmhRepository>
      </metadata>
    </record>
  </GetRecord>
</OAI-PMH>
```

Table 14. A sample *OAI-PMH metadata record* exposed by the Repository Index.

As a result of the introduction of the Repository Index, OAI-PMH harvesters operated by downstream applications can use a *datestamp*-based harvesting strategy to gather newly added DIDL documents from the aDORe environment. Also the harvester contained in the OAI-PMH Federator which is a special type of downstream application described in Section 1.5, will interact with the environment in the manner described below and depicted in Figure 15.

Presume that T1 and T2 are datetimes with a time precision of seconds, with T2 > T1, and that a harvester wants to collect DIDL documents added to the aDORe environment since the last harvest, which was conducted at T1. These are the steps involved:

- The harvester issues a `ListIdentifiers` request against the Repository Index, using the until parameter with a value of T2. In response, the harvester receives a list of repository base URLs and associated repository registration datetimes.

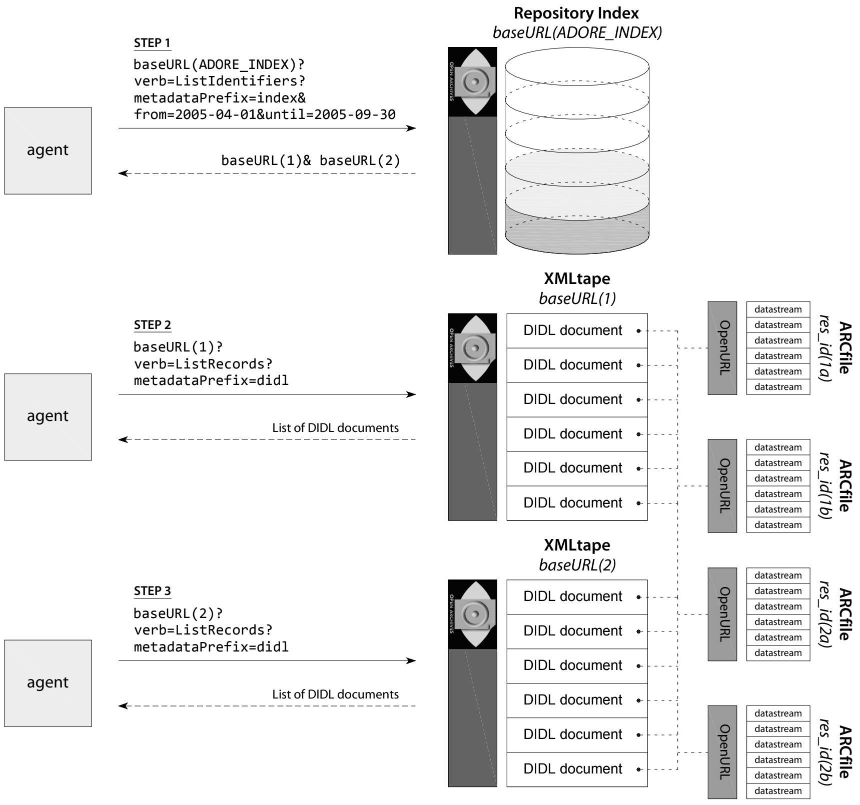


Figure 15. The Repository Index: A registry of autonomous OAI-PMH repositories

```
[ baseURL(ADORE_INDEX)?
      verb=ListIdentifiers&until=T2&
      metadataPrefix=index ]
```

- For each repository base URL that has a repository registration datetime larger than T1, the harvester issues a `ListRecords` request against the actual repository base URL. Since OAI-PMH repositories that meet this condition have become available for harvesting after the last harvest – all DIDL documents need to be collected from it – the harvester does not use the `from` argument in this request. However, it does use an `until` parameter conveying the value T2.

```
[ baseURL(N)?
      verb=ListRecords&until=T2&
      metadataPrefix=didl ]
```

- For each repository base URL that has a repository registration datetime smaller than or equal to T1, the harvester issues a `ListRecords` request against the actual repository base URL. Since these repositories were already available for harvesting at the time of the last harvest – only new or updated DIDL documents need to be collected – the harvester issues a `ListRecords` with a value of T1 for the `from` argument and a value of T2 for the `until` argument.

```
[ baseURL(M)?
      verb=ListRecords&from=T1&until=T2&
      metadataPrefix=didl ]
```

In order for these harvesting operations not to miss out on any updates or additions made to this highly distributed and dynamic environment, the following are crucial:

- Usage of the `until` parameter in the aforementioned harvesting requests.
- Synchronization of the clocks on all machines operating the autonomous OAI-PMH repositories and the Repository Index. This can be achieved using the Network Time Protocol (Mills, 1996).
- The content of the Repository Index must perfectly reflect the collection of harvestable OAI-PMH repositories in the environment. This tight synchronization of the Repository Index and the collection of harvestable repositories can be achieved in various ways. For example, the process of populating the Repository Index can be integrated with that part of the ingestion process where new OAI-PMH repositories are created.

1.3. The Identifier Locator: Locating DIDL documents, Digital Items and constituent datastreams

Harvesters, working on behalf of service providers, collect DIDL documents from the aDORe environment and build services with the represented digital assets. As a result, identifiers contained in the harvested DIDL documents become available in applications. As explained in Section 1.1.1.1, these identifiers can either be identifiers identifying DIDL documents or DIDL XML elements thereof, or identifiers identifying Digital Items. It is essential that, when such identifiers show up in downstream applications, the identified assets can be retrieved from the aDORe environment. For this purpose the Identifier Locator is introduced. The Identifier Locator collects the following identifier-related information from the aDORe environment:

DIDL document Identifier	OAI-PMH repository	datetime of insertion
info:lanl-repo/i/58f202ac	baseURL(1)	2005-12-03T11:59:18Z
info:lanl-repo/i/002035b2	baseURL(2)	2005-12-06T11:12:12Z

Table 15. DIDL document Identifier module of the Identifier Locator

Digital Item Identifier	DIDL document Identifier	XML ID
info:doi/10.1103/ PhysRevB.69.174413	info:lanl-repo/i/58f202ac	uuid-00005e90
info:lanl-repo/biosis/abcdef	info:lanl-repo/i/002035b2	uuid-0000356d
info:lanl-repo/tech/klmnop	info:lanl-repo/i/003276k7	uuid-000076f4
info:lanl-repo/tech/klmnop	info:lanl-repo/i/12e303be	uuid-00008k44

Table 16. Digital Item Identifier module of the Identifier Locator

- For DIDL documents (OAIS AIPs): The identifier of each DIDL document in the aDORe environment and for each DIDL document the base URL of the Autonomous OAI-PMH Repository in which the DIDL document with the given DIDL document Identifier resides. In Table 15, '`info:lanlrepo/i/58f202ac`' and '`info:lanl-repo/i/002035b2`' are identifiers of DIDL documents located in the OAI-PMH repository with base URL '`baseURL(1)`' and '`baseURL(2)`', respectively. The identifiers are added to the Identifier Locator on '`2005-12-03T11:59:18Z`' and '`2005-12-06T11:12:12Z`', respectively. The first entry of Table 15 is obtained from our sample DIDL document shown in Table 13.
- For Digital Items (OAIS Content Information): The identifier of each digital asset and constituent stored in the aDORe environment and for each such identifier, the DIDL document Identifier (including the Fragment Identifier of the corresponding `Item` element) of all DIDL documents that contain a Digital Item with the given Digital Item Identifier. In Table 16, OAIS Content Information Identifiers '`info:doi/10.1103/PhysRevB.69.174413`' and '`info:lanl-repo/biosis/abcdef`' identify digital assets contained in the DIDL documents with DIDL document Identifier '`info:lanlrepo/i/58f202ac`' and '`info:lanl-repo/i/002035b2`', respectively. Within these DIDL documents, the XML elements representing the Digital Items with the aforementioned Digital Item Identifiers have XML IDs '`uuid-00005e90`', and '`uuid-0000356d`', respectively. Again, from Table 15 one can infer that the DIDL documents are located in the OAI-PMH repository with base URL '`baseURL(1)`' and '`baseURL(2)`' and are added to the Identifier Locator on '`2005-12-01T12:00:53Z`' and '`2005-12-06T11:12:12Z`', respectively. Table 16 also shows that the digital asset with OAIS Content Information Identifier '`info:lanl-repo/tech/klmnop`' is represented by two OAIS AIPs with OAIS AIP Identifiers '`info:lanl-repo/i/003276k7`'and '`info:lanl-repo/i/12e303be`', indicating the existence of two different versions. The first entry of Table 16 is obtained from our sample DIDL document.

Similarly to the Repository Index, the Identifier Locator can be exposed as an OAI-PMH repository. The first columns of Table 15 and Table 16 map directly to the notion of OAI-PMH *identifier*; the last column of Table 16 corresponds with the OAI-PMH *datestamp*. The OAI-PMH repository has the following properties:

- It has a unique, persistent base URL, the HTTP address '`baseURL(ADORE_LOCATOR)`'.
- Contained records comply with a locally defined XML-based *metadata format*, identified by the `metadataPrefix` '`locator`' which allows expressing the information shown in Table 15 and Table 16. A sample metadata record is shown in Table 17.
- The *identifier* used by the OAI-PMH can be both a DIDL document Identifier or a Digital Item Identifier.
    - If a DIDL document Identifier, the following information is returned by the Identifier Locator (See Table 16): 1) The DIDL document Identifier sent in the original request, 2) the base URL of the Autonomous OAI-PMH Repository in which the document with the given DIDL document Identifier resides, and 3) the datetime of creating the DIDL document.
    - If a Digital Item Identifier, the following information is returned by the Identifier Locator (See Table 16 in combination with Table 15): 1) The Digital Item Identifier sent in the original request, 2) The DIDL document Identifier of all DIDL documents that contain a Digital Item with the given Digital Item Identifier, 3) for each DIDL document Identifier, the base URL of the Autonomous OAI-PMH Repository in which the document with the given DIDL document Identifier resides, and 4) for each DIDL document, the datetime of creating the DIDL document.
- The *datestamp* used by the OAI-PMH is the datetime of inserting the DIDL document Identifier (and associated information) in the Identifier Locator. Information in the Identifier Locator is never updated. An updated digital asset will always result in a new *metadata record*. Hence this *datestamp* will never change and always remain equal to the time the identifier of the DIDL document was collected by the Identifier Locator.
- The supported OAI-PMH time granularity is seconds-level.
- Similarly to the Repository Index, *set* structures may be supported.

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/">
  <responseDate>2006-01-02T15:33:28Z</responseDate>
  <request identifier="http://purl.lanl.gov/aDORe/XMLtape/2005_4acb6e28"
    verb="GetRecord" metadataPrefix="didl">
    http://purl.lanl.gov/aDORe/IdLocator/</request>
  <GetRecord>
  <record>
    <header>
      <identifier>info:doi/10.1103/PhysRevB.69.174413</identifier>
      <datestamp>2005-12-03T11:59:18Z</datestamp>
      <setSpec>collection:info*3Asid*2Flibrary.lanl.gov*3Aaps</setSpec>
    </header>
    <metadata>
      <IdLocations xmlns="http://library.lanl.gov/2005-08/aDORe/IdLocator/">
```

```
        <IdLocation>
          <dc:identifier xmlns:dc="http://purl.org/dc/elements/1.1/
            xmlns:diext=http://library.lanl.gov/2005-08/aDORe/DIDLextension/"
            DIDLDocumentId="info:lanl-repo/i/58f202ac"
            diext:DIDLDocumentCreated="2005-12-03T11:57:12Z"
            DIDLxmlID="uuid-00005e90">
            info:doi/10.1103/PhysRevB.69.174413</dc:identifier>
          <dcterms:created xmlns:dcterms="http://purl.org/dc/terms/">
            2005-12-03T11:59:18Z</dcterms:created>
          <idx:oaipmhBaseURL
            xmlns:idx="http://library.lanl.gov/2005-08/aDORe/RepoIndex/">
            http://purl.lanl.gov/aDORe/XMLtape/2005_4acb6e28</idx:oaipmhBaseURL>
          <oaipmhRequest>
            http://barracuda.lanl.gov:8080/project/moai2/abc?verb=GetRecord&
            identifier=info:lanl-repo/i/58f202&
            metadataPrefix=didl</oaipmhRequest>
        </IdLocation>
      </IdLocations>
    </metadata>
  </record>
 </GetRecord>
</OAI-PMH>
```

Table 17. A sample *OAI-PMH metadata record* exposed by the Identifier Locator.

By default, the Identifier Locator is populated through recurrent OAI-PMH harvesting from the Autonomous OAI-PMH Repositories, but for cases that involve massive ingestion of digital assets, a faster batch loading mechanism has been implemented. Through consultation with the Identifier Locator, an application can obtain the information necessary to use the OAI-PMH to collect the DIDL document with a specified DIDL document Identifier, or the DIDL document in which a Digital Item with a specified Digital Item Identifier resides. The process is depicted in Figure 14 and described below.

□ If the resource with identifier '`info:lanl-repo/i/58f202ac`' is requested, a look-up in the Identifier Locator will learn that it is a DIDL document with identifier '`info:lanl-repo/i/58f202ac`' located at '`baseURL(1)`'. This DIDL document can be obtained by issuing the OAI-PMH request written below.

```
[ baseURL(1)?verb=GetRecord&
      identifier=info:lanlrepo/i/58f202ac&
      metadataPrefix=didl ]
```

□ If the resource with identifier '`info:lanl-repo/i/58f202ac#uuid-00005e90`' is requested, a look-up in the Identifier Locator will learn that the identified resource is an XML element contained in the DIDL document with DIDL document Identifier '`info:lanl-repo/i/58f202ac`'. Furthermore, the lookup will learn that this DIDL document is located at '`baseURL(1)`'. Again, that DIDL document can be obtained by issuing the above OAI-PMH request. From the resulting DIDL document, the XML element with XML ID '`uuid-00005e90`' can be extracted.

□ If the resource with identifier '`info:lanlrepo/biosis/abcdef`' is requested, a look-up in the Identifier Locator will learn that it is available from a DIDL document with DIDL document Identifier '`info:lanlrepo/i/002035b2`', and that – in this DIDL document – it has the XML ID ‘`uuid-0000356d`’. A look-up of this DIDL document Identifier learns that the corresponding DIDL document is located at ‘`baseURL(2)`’. This DIDL document can be obtained by issuing the OAI-PMH request written below.

```
[ baseURL(6)?verb=GetRecord&
      identifier=info:lanl-repo/i/002035b2&
      metadataPrefix=didl ]
```

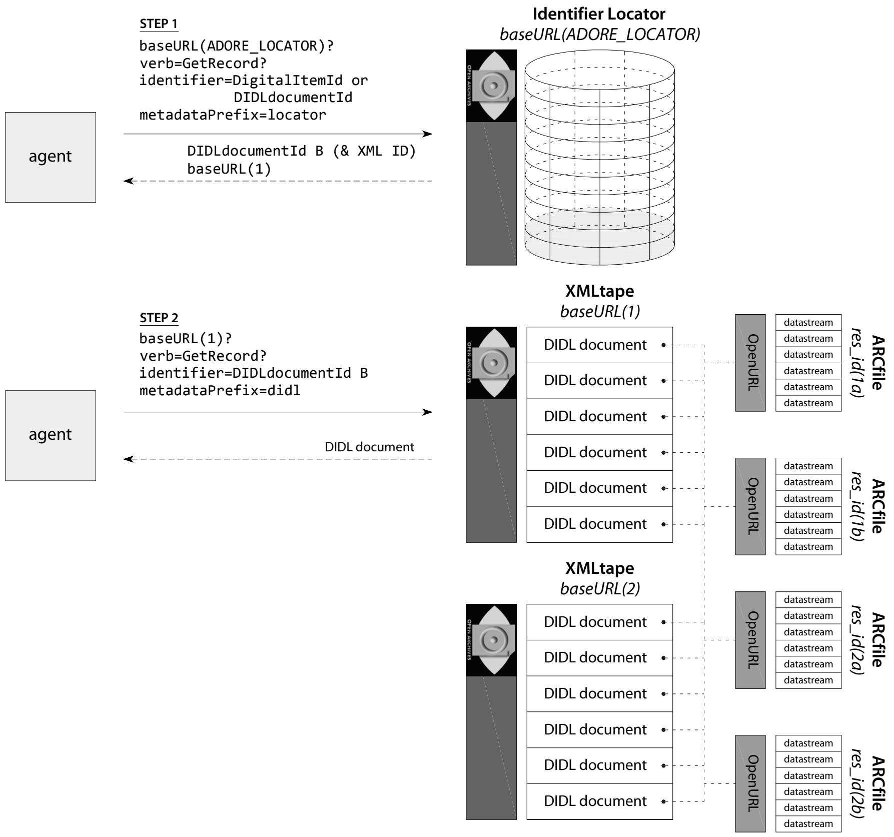


Figure 16. The Identifier Locator: Locating DIDL documents, digital assets, and constituent datastreams

From the resulting DIDL document, the `Item` element representing the Digital Item can be extracted. This approach allows for a mapping of Digital Item Identifiers (also known as OAIS Content Information Identifiers) to DIDL document Identifiers (OAIS AIP Identifiers).

It is worth reporting that, at the time of writing, efforts are ongoing aimed at building the Identifier Locator on Handle System infrastructure. Note that the Handle Systems can be employed without having to use Handles for identifying stored digital assets or DIDL documents. Once more, scalability is a critical design criterion, as presently, the Identifier Locator hosts around two hundred million identifiers, and hence will be the largest Handle System implementation ever built.

1.4. The Transform Engine: Applying services to DIDL documents, Digital Items, and constituent datastreams

So far, a set of components have been introduced that – when used selectively in combination – allow for the harvest of DIDL documents from the aDORe repository environment. However, because of various reasons, downstream applications or consumers may wish to receive information in formats different than the ones stored in aDORe. To this end, a separate component is introduced in the aDORe environment: The Transform Engine.

The Transform Engine can generate various transformations of stored materials. A distinction is made between two levels of transformations:

- *DIDL document transformations*. Such transformations are also referred to as 'crosswalks'. Transformations of practical use include the transcoding of DIDL documents to documents that are compliant with such packaging formats as the MPEG-21 File Format (ISO, 2005b), METS (LC, 2005a), XFDU (CCSDS, 2004b) and IMS-CP (IMS Global Learning Consortium, 2003b). Also the transformation of DIDL documents to XML documents that contain identifier-related information only is supported. The latter can be used to populate the Identifier Locator.
- *Datastream transformations*. Such transformations are produced by a service operation that takes as input a datastream of a digital asset. Examples include the transcoding of a TIFF formatted still image to a JPEG2000 encoded image. Or the transcoding of a plain text file to an MPEG-4 natural speech stream.

It is worthwhile noting that several concepts underlying the aDORe dissemination techniques have been pioneered and explored by other sophisticated service-oriented architectures that emerged from the digital library domain, including Fedora (Payette & Lagoze, 1998; Staples et al, 2003; Lagoze et. al, in press), SODA (Nelson & Maly, 2001a; Nelson & Maly, 2001b; Neslon, 2002) and the context broker mechanism described in (Dushay, 2001; Dushay, 2002). At the time of writing, a pragmatic implementation of the Transform Engine has been introduced, awaiting the development of a more robust version of the engine in the context of the aforementioned Pathways project.

1.4.1. Dynamically associating services with DIDL documents, and constituents

In contrast with such systems as Fedora and SODA, service operations in aDORe are not embedded in the DIDL documents that are stored in the Autonomous OAI-PMH Repositories. Embedding services would yield the need to frequently update the stored DIDL documents to add new or to edit existing services as new services emerge, or as existing ones are updated. Given the anticipated size of aDORe, the administrative overhead in doing so would be extensive if not forbidding. Also, the need to regularly 'touch' stored DIDL documents is a poor fit with the rather static nature of the content currently stored in aDORe, and with the previously described strategy to create new DIDL documents – instead of updating existing ones – when some form of editing has been performed. Therefore aDORe handles the linking of services dynamically, based on structure, type and content of the DIDL documents.

Connecting services to a DIDL document is achieved through a look-up in an underlying special-purpose registry, known as the Knowledge Database. The Knowledge Database stores a list of rules that binds services to a set of patterns. The availability of proper patterns determines whether or not a service can be applied on that data. A pattern might be as simple as the presence of a MIME type for a requested datastream. But many services require more detailed information than mere MIME types. For example, some services may impose limitations on the nature of the data being requested, others may take into account contextual information about the network characteristics or terminal capabilities of the agent requesting the service, and so on. As will be shown later on, such parameters can be readily conveyed using the NISO OpenURL Framework.

At the time of writing, in the context of the Pathways project ("Pathways," n.d.), investigations are ongoing that aim at defining several facets of the knowledge database using the Web Ontology Language (OWL) (Smith, Welty & McGuinness, 2004); OWL-S (The OWL Services Coalition, n.d.) and MPEG-21 DIA (ISO, 2004e) technologies.

1.5. The OAI-PMH Federator and the aDORe OpenURL Resolver: Accessing the aDORe environment

To facilitate the access of stored materials, two interfaces to the complete aDORe environment are introduced. The interfaces hide the complexity of the aDORe environment to client-applications, and fulfill specific functions as summarized in Table 18.

aDORe interface	standard	request type	items in response
OAI-PMH Federator	OAI-PMH	OAIS DIP crosswalks	1 or more
aDORe OpenURL *Resolver*	NISO OpenURL	OAIS DIP crosswalk datastream-level diss.	1

aDORe interface	MPEG-21 identifier	OAIS identifier
OAI-PMH Federator	DIDL document Identifier	AIP Identifier
aDORe OpenURL *Resolver*	DIDL document Identifier Digital Item Identifier	AIP Identifier Content Information Identifier

Table 18. Characteristics of the aDORe interfaces

- The OAI-PMH Federator is an interface through which (batches of) stored DIDL documents (or crosswalks thereof) can be requested. This interface is typically used by machines and responses are expressed in XML. In terms of the OAIS Reference Model, this interface enables requesting OAIS Dissemination Information Packages (OAIS DIPs). Available OAIS DIPs include the DIDL documents themselves as well as transformations thereof, such as the representation of a stored DIDL document using packaging formats other than MPEG-21 DID. The OAI-PMH Federator exposes the aDORe environment as a single OAI-PMH repository, in which the DIDL document Identifiers (also known as OAIS AIP Identifiers) of stored DIDL documents act as the OAI-PMH *identifier*. The

OAI-PMH Federator allows to harvest batches of DIDL document (using the OAI-PMH `ListRecords` verb) or to obtain DIDL documents with a given DIDL document Identifier on an individual basis (using the OAI-PMH `GetRecord` verb).

- The aDORe OpenURL *Resolver* is an interface that supports two types of service requests. A first type of request aims at obtaining a DIDL document (or crosswalk thereof) on an individual basis. This type of request is similar to the `GetRecord` request of the OAI-PMH Federator. A second type of request aims at obtaining disseminations of constituent datastreams of a digital asset. Hereby, a dissemination could be the datastream itself or a (computed) transformation thereof. A dissemination is typically produced by a service operation. Such service operations include 'get PDF', 'show abstract', 'play video', and so on. The result of applying a service is a MIME typed bit stream, and is typically provided to a consumer for immediate consumption. The aDORe OpenURL *Resolver* provides an interface to the complete aDORe environment that is compliant with the NISO OpenURL standard (NISO, 2005), and in which both Digital Item Identifiers (also known as OAIS Content Information Identifiers) and DIDL document Identifiers (also known as OAIS AIP Identifiers), including the XML ID fragment, can be used to identify the requested data resource.

1.5.1. The OAI-PMH federator: An OAI-PMH Interface for harvesting DIDL documents

The OAI-PMH Federator is introduced in the environment for two reasons:

- As demonstrated above, harvesters need to be aware of the structure of the aDORe environment in order to be able to collect DIDL documents. Required knowledge includes the existence of the Repository Index and the Identifier Locator, as well as the protocol to interface with them. This need for this prior know-how hinders the use of off-the-shelf OAI-PMH harvesting tools and requires the use of special-purpose clients.
- In addition to being able to disseminate stored DIDL documents from aDORe, there is a need for disseminations of various transformations thereof. For example, for reasons of interoperability, it is desirable to disseminate stored DIDL documents rendered according to various profiles of MPEG-21 DID, or compound asset formats such as XFDU, IMS-CP, and METS. It is also attractive to be able to feed the Identifier Locator with the bare essentials of a stored DIDL document (the contained identifiers) rather than with the detailed DIDL document itself. Also, there is a need to be able to request a table of contents of services that are applicable on a stored DIDL document. Supporting such transforms at the level of each Autonomous OAI-PMH Repository would significantly increase the complexity of the tools required for their implementation. Again, no off-the-shelf OAI-PMH repository tools could be used.

The OAI-PMH Federator addresses both problems:

- It exposes the whole aDORe environment as a single OAI-PMH repository by translating incoming OAI-PMH requests into appropriate requests targeted at the Repository Index, Identifier Locator and Autonomous OAI-PMH Repositories. Since many of these components are themselves OAI-PMH Repositories, the OAI-PMH Federator operates its private OAI-PMH harvester. Logic built into the OAI-PMH Federator ensures that the responses received from the various components are interpreted correctly, and, whenever appropriate, handed over to downstream harvesters as valid OAI-PMH responses.

- It supports requesting transforms of stored DIDL documents by handing off transformation requests to the previously described Transform Engine.

The OAI-PMH Federator is implemented as an OAI-PMH repository with the following characteristics:

- It has a unique, persistent base URL, the http address '`baseURL(FEDERATOR)`'.
- The *identifier* used by the OAI-PMH is the Identifier of the stored DIDL document (or AIP Identifier).
- The *datestamp* used by the OAI-PMH is the datetime of creation of the DIDL document.
- MPEG-21 DIDL is the natively supported *metadata format*, but, through dynamic processing of DIDL documents by the Transform Engine, potentially many other descriptive metadata formats can be supported. The term *metadata format* must be interpreted broadly, as the `metadataPrefix` argument in harvesting requests issued against the OAI-PMH Federator can be used to express several types of transformations that can be applied to stored DIDL documents, including transformations that map MPEG-21 DIDL to another compound asset model such as IMS-CP. In this case, the value for the `metadataPrefix` argument in harvesting requests could be '`ims-cp`', and the IMS-CP XML Schema would define the *metadata format.*
- The supported time granularity is seconds-level.
- In order to support harvesting from a specific Autonomous OAI-PMH Repository, if this would be required, the OAI-PMH Federator can expose an OAI-PMH *set* structure according to their base URLs. Other *set* structures implemented by all Autonomous OAI-PMH Repositories can also be exposed by the OAI-PMH Federator. In the current aDORe implementation a collection-oriented and media format-oriented *set* structure is available.

1.5.1.1. The OAI-PMH Federator: A step-by-step example

The interaction of an off-the-shelf OAI-PMH harvester with the aDORe environment via the OAI-PMH Federator is illustrated by detailing the manner in which the response to the `GetRecord` and `ListRecords` requests is provided.

A. OAI-PMH GetRecord request

```
[ baseURL(Federator)?
      verb=GetRecord&
      identifier=info:lanl-repo/i/58f202ac&
      metadataPrefix=didl ]
```

and

```
[ baseURL(Federator)?
      verb=GetRecord&
      identifier=info:lanl-repo/i/58f202ac&
      metadataPrefix=ims-cp ]
```

The steps involved in generating the appropriate GetRecord response are depicted in Figure 17 and listed below.

- □ Through interaction with the Identifier Locator, the OAI-PMH Federator discovers the location of the DIDL document with DIDL document Identifier 'info:lanlrepo/i/58f202ac', namely 'baseURL(1)'. If no entry for 'info:lanl-repo/i/58f202ac' exists in the Identifier Locator, the OAI-PMH Federator generates an 'IdDoesNotExist' error response.
- □ The OAI-PMH Federator obtains the stored DIDL document by issuing the following GetRecord request:

  ```
  [ baseURL(1)?
        verb=GetRecord&
        identifier=info:lanlrepo/i/58f202ac&
        metadataPrefix=didl ]
  ```

  - □ If the metadataPrefix requested in the original GetRecord request was 'didl', no special action needs to be taken.
  - □ If the metadataPrefix requested in the original GetRecord request was 'ims-cp', via the knowledge database, the OAI-PMH Federator can determine whether the requested metadataPrefix (namely 'ims-cp') is supported. If yes, the OAI-PMH Federator passes the DIDL document on to the Transform Engine, which applies the service. If the requested metadataPrefix is not supported, a 'cannotDisseminateFormat' error response is generated.
- □ The OAI-PMH Federator embeds the record that results from the previous step in a correct OAI-PMH response. This may include inserting set membership information in the headers of the responses.

B. OAI-PMH ListRecords request

```
[ baseURL(ADORE_FEDERATOR)?
      verb=ListRecords&
      from=T1&until=T2&
      metadataPrefix=didl ]
```

and

```
[ baseURL(ADORE_FEDERATOR)?
      verb=ListRecords&
      from=T1&until=T2&
      metadataPrefix=ims-cp ]
```

These are the steps involved in generating the appropriate ListRecords response:

- □ Identification of the Autonomous OAI-PMH Repositories that meet the harvesting criteria by interacting with the Repository Index.

  ```
  [ baseURL(ADORE_INDEX)?
        verb=ListIdentifiers&until=T2&
        metadataPrefix=index ]
  ```

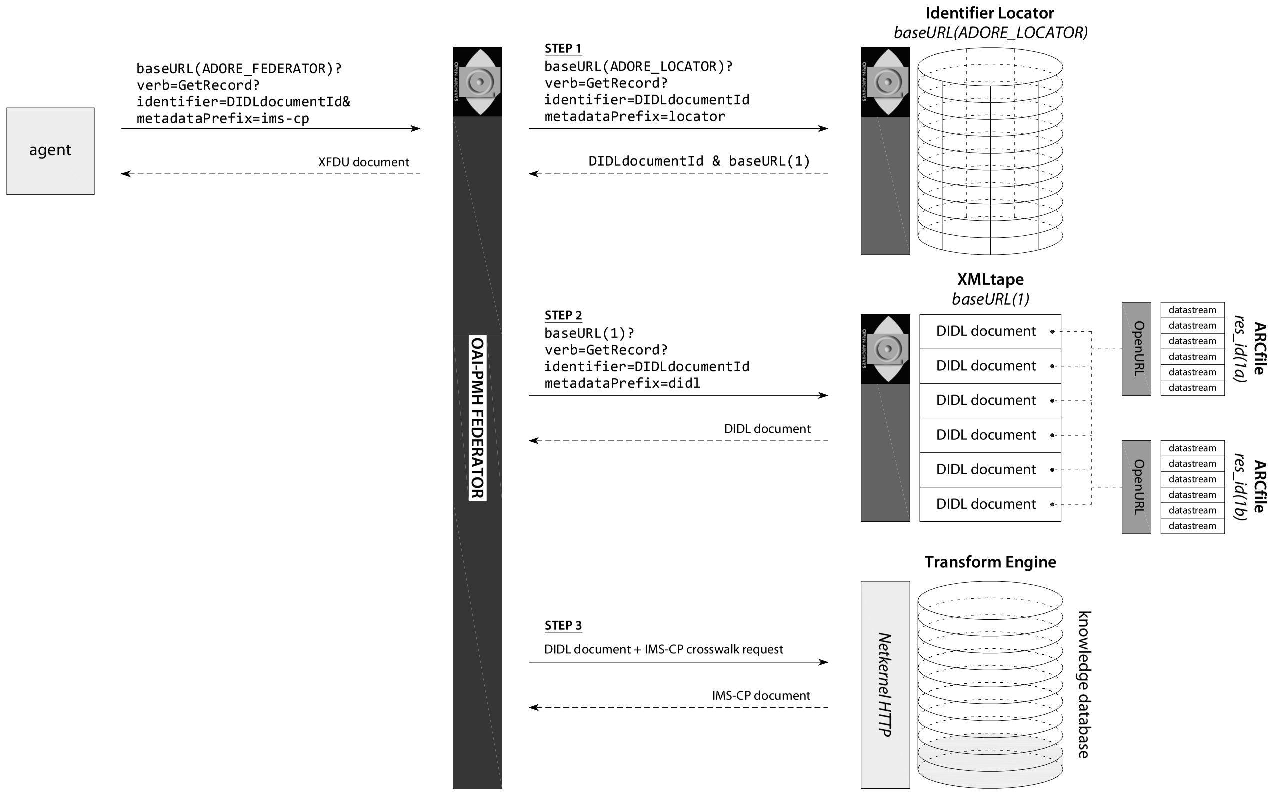


Figure 17. The OAI-PMH Federator: An interface for requesting OAIS DIPs

- For each Autonomous OAI-PMH Repository identified through interaction with the Repository Index one of the following actions is taken:
  - If the OAI-PMH repository was created after the last harvest, collect all DIDL documents

    ```
    [ baseURL(N)?
          verb=ListRecords&until=T2&
          metadataPrefix=didl ]
    ```
  - If the OAI-PMH repository already existed at the last harvest, collect the added or updated DIDL documents

    ```
    [ baseURL(N)?
          verb=ListRecords&from=T1&until=T2&
          metadataPrefix=didl ]
    ```

  If the `metadataPrefix` requested in the original `ListRecords` request was '`didl`', no special action needs to be taken. If a *metadata format* other than '`didl`' is requested, via the knowledge database, the OAI-PMH Federator can determine whether the requested `metadataPrefix` (e.g., '`ims-cp`') is supported. If yes, the OAI-PMH Federator passes the DIDL document on to the Transform Engine, which applies the service. Since support of a given package format is a Repository-wide attribute, a '`cannotDisseminateFormat`' error response is generated by the OAI-PMH Federator in case the requested `metadataPrefix` is not supported.
- The OAI-PMH Federator returns the responses to harvesting requests issued against the individual Autonomous OAI-PMH Repositories as valid OAI-PMH responses to the downstream harvester. As was the case with the `GetRecord` response, this might include editing the headers of the responses to insert *set* membership information. The following are important with respect to this step in the process:
  - It is – in theory – possible that the Repository Index returns a '`noRecordsMatch`' error response. The OAI-PMH Federator must return such a response to the downstream harvester.
  - It is possible that Autonomous OAI-PMH Repositories respond with a '`noRecords-Match`' error response. The OAI-PMH Federator must not pass on such responses to the downstream harvester but rather interpret them as a command to start harvesting from the next autonomous OAI-PMH repository that was returned by the Repository Index. Only if no meaningful responses have been received from any of the individual OAIPMH repositories must the OAI-PMH Federator itself return a '`noRecordsMatch`' error response.
  - Care must be taken to appropriately handle `resumptionTokens` delivered by an Autonomous OAI-PMH Repository. As a matter of fact, the OAI-PMH Federator will need to adapt such `resumptionToken` by adding the following information to it: 1) the base URL of the autonomous OAI-PMH repository from which the `resumptionToken` was received, and 2) the requested `metadataPrefix`. Also, the OAI-PMH Federator may need to create `resumptionTokens` of its own, to make the transition between harvesting from one Autonomous OAI-PMH Repository to the next easier.

- If one of the Autonomous OAI-PMH Repositories to be harvested from returns a ‘badResumptionToken’ error message, the OAI-PMH Federator must pass this on to the downstream harvester. If one of the OAI-PMH repositories fails to respond, the OAI-PMH Federator must generate an appropriate HTTP error indicating this ‘internal error’. The HTTP status-code 503 ‘service unavailable’ is suitable for that purpose. In both cases, the responses indicate to the downstream harvester that its ongoing harvest can not be completed successfully, and that the intended harvest should be restarted at some later time.

Through a process similar as the one described above, the response to a ListRecords request without from and/or until arguments can be generated.

1.5.2. The aDORe OpenURL Resolver: A NISO OpenURL Interface for obtaining DIDL documents and their datastreams.

Through OAI-PMH harvesting, downstream applications obtain DIDL documents and use those documents and their constituents in the creation of their services. For example, a search engine might extract all textual information from harvested DIDL documents and make it available to consumers for searching. In this case, brief search results will point back to the corresponding digital asset containing the textual information stored in aDORe. Obviously, in such pointers the identification of the digital asset will play a crucial role, and both DIDL document Identifiers (including the associated XML ID Fragment Identifiers) and Digital Item Identifiers can be involved. Also, using the Transform Engine, the aDORe environment allows delivery of various disseminations of stored datastreams through the application of services to those datastreams. Hence, requests to obtain a dissemination of a stored datastream should not only identify the digital asset containing the requested datastream but also the service that needs to be applied to it. The NISO OpenURL standard (NISO, 2005), as described in Section 4 of Chapter 1, provides a perfect framework to convey such service requests.

Two mappings follow easily:

- The resource for which a dissemination is requested is the *Referent* of the OpenURL *ContextObject*. In aDORe, the *Referent* is described by an *Identifier Descripto*r and an optional *By-Value Metadata Descriptor*. The value of the former is the DIDL document Identifier of the DIDL document (containing the datastream) for which a service is requested. The latter conveys an optional Fragment Identifier, which corresponds with a datastream (of the obtained DIDL document) that is used as input to a service operation. As will be shown, Fragment Identifiers are expressed using an arg key, from the aDORe *Metadata Format*. If no argument is provided, the service operation is applied upon the DIDL document itself.

  The aDORe OpenURL *Resolver* also supports obtaining disseminations using Digital Item Identifiers (OAIS Content Information Identifiers). Indeed, for example, if the resource with Digital Item Identifier ‘info:lanlrepo/biosis/abcdef’ is targeted, through interaction with the Identifier Locator, we will learn that the digital asset is available as a DIDL document with DIDL document Identifier ‘info:lanl-repo/i/002035b2’. From the resulting DIDL document, the Item element containing the Descriptor/Statement construct with the given Digital Item Identifier can be extracted. Remember that multiple versions of a single digital asset may exist. In aDORe, each version is represented as a separate DIDL document. As a result, given a single Digital Item Identifier, the Identifier Locator may return multiple DIDL document Identifiers. In this case, by default, the aDORe OpenURL *Resolver* will use the most recent DIDL document available.

□ The service that needs to be applied to obtain the requested dissemination corresponds to the *ServiceType* of the OpenURL *ContextObject*. In aDORe, the *ServiceType* is typically described by an *Identifier Descriptor*.

The aDORe OpenURL *Resolver* is introduced at base URL '`baseURL(ADORE_RESOLVER)`' to which all requests for the dissemination of stored DIDL documents or constituent datastreams are targeted. A list of sample OpenURL requests is provided below. The *ContextObjects* are represented using the KEV *Format* and transported using the HTTP protocol.

□ *Request No 1. Obtain an IMS-CP document*. The resource for which a dissemination is requested, is a DIDL document with identifier '`info:lanl-repo/i/58f202ac`'. The service is identified by the KEV pair '`svc_id=info:lanl-repo/svc/ims-cp`' and asks that the DIDL document would be repackaged according to the IMS-CP specification.

```
[ baseURL(ADORE_RESOLVER)?
      url_ver=Z39.88-2004&
      rft_id=info:lanl-repo/i/58f202ac&
      svc_id=info:lanl-repo/svc/ims-cp ]
```

The same type of request can also be formulated using the identifier of the Digital Item. Remember that for a given Digital Item, multiple versions may exist. In aDORe, each version of a Digital Item is ingested as a new DIDL document, and can be uniquely addressed using the identifier of the DIDL document. A lookup in the Identifier Locator may learn that the Digital Item with a given Digital Item Identifier is available from multiple DIDL document – one DIDL document for each version. As a result, when requesting a service using the identifier of a Digital Item, dependent on the DIDL document being processed, a different result could be returned. In aDORe, by default, the aDORe OpenURL *Resolver* will always use the DIDL document representing the most recent version of the digital asset, available in the aDORe environment.

```
[ baseURL(ADORE_RESOLVER)?
      url_ver=Z39.88-2004&
      rft_id= info:doi/10.1103/PhysRevB.69.174413&
      svc_id=info:lanl-repo/svc/ims-cp ]
```

□ *Request No 2. Obtain a Table of Contents*. The resource for which a service is requested is a Digital Item with Digital Item Identifier '`info:doi/10.123/PhysRevB.69.174413`'. The service is the display of the Table of Contents as an XHTML page, listing all constituent datastreams pertaining to the asset as well as the services that are available for them. The KEV pair '`rft_id=info:doi/10.1103/PhysRevB.69.174413`' expresses the identifier of the *Referent*. The KEV pair '`svc_id=info:lanl-repo/svc/table_of_contents`' conveys the identifier of a service that displays a Table of Contents of the digital asset identified by the *Referent*.

```
[ baseURL(ADORE_RESOLVER)?
      url_ver=Z39.88-2004&
      rft_id=info:doi/10.1103/PhysRevB.69.174413&
      svc_id=info:lanl-repo/svc/table_of_contents ]
```

The same request can also be formulated using a DIDL document Identifier. The *Referent* is described by the combination of an *Identifier Descriptor* and a *By-Value Metadata Descriptor*. The value of the former corresponds with the DIDL document Identifier of the most recent DIDL document containing the digital asset in question. The latter uses the `arg` key to convey the Fragment Identifier of the `Item` element representing the digital asset; that is, the `Item` element that carries a `Descriptor/Statement` construct with the Digital Item Identifier. The *Metadata Format* used to described the *Referent* is identified by the KEV pair '`rft_val_fmt= info:lanl-repo/fmt/adore`'.

```
[ baseURL(ADORE_RESOLVER)?
      url_ver=Z39.88-2004&
      rft_id=info:lanl-repo/i/58f202ac&
      rft_val_fmt=info:lanl-repo/fmt/adore&
      rft.arg=uuid-00005e90&
      svc_id=info:lanl-repo/svc/table_of_contents ]
```

- *Request No 3. Obtain the MARCXML (LC, 2005b) descriptive metadata, as a MODS (LC, 2005c) metadata record.* As no Digital Item Identifier exists for this datastream, the request must be formulated using the DIDL document Identifier. The identifier of the *ServiceType* that transforms MARXCML to MODS is '`info:lanl-repo/svc/marc_2_mods`'.

```
[ baseURL(ADORE_RESOLVER)?
      url_ver=Z39.88-2004&
      rft_id=info:lanl-repo/i/58f202ac&
      rft_val_fmt=info:lanl-repo/fmt/adore&
      rft.arg=uuid-0000a01c&
      svc_id=info:lanl-repo/svc/marc_2_mods ]
```

Currently, the *Referent* and *ServiceType* are the only *Entities* of the *ContextObject* that are used in aDORe OpenURL requests. But, NISO OpenURL requests may convey other *Entities* that can be taken into account to deliver context-sensitive service requests. Of particular interest is the *Requester*. This *Entity* allows adapting the actual dissemination to the agent requesting it. *Requester* information could convey identity, and this would allow responding differently to the same service request depending on whether the requesting agent is a human or machine. Or different humans could receive different disseminations based on recorded preferences or access rights. But the NISO OpenURL specification is purposely very generic and extensible, and would also support to convey the characteristics of a user's terminal and/or the user's location via the *Requester* entity. Such information could be passed on to the Transform Engine and be taken into account in the delivery of an actual dissemination. As a matter of fact, the expressiveness of NISO OpenURL seems to resonate nicely with the nature of the MPEG-21 DIA (ISO, 2004e) effort that focuses on the description of usage environment characteristics such as terminal capabilities, network conditions and other contextual information.

1.5.2.1. The aDORe OpenURL Resolver: A step-by-step example

The steps involved in responding to the above 'marc_2_mods' OpenURL request are depicted in Figure 18 and listed below.

```
[ baseURL(ADORE_RESOLVER)?
      url_ver=Z39.88-2004&
      rft_id=info:lanl-repo/i/58f202ac&
      rft_val_fmt=info:lanl-repo/fmt/adore&
      rft.arg=uuid-0000a01c&
      svc_id=info:lanl-repo/svc/marc_2_mods ]
```

- First, the aDORe OpenURL *Resolver* obtains the *Referent*. In the above *ContextObject*, the *Referent* is described by the combination of an *Identifier Descripto*r and a *By-Value Metadata Descriptor*. Through interaction with the Identifier Locator, the aDORe OpenURL *Resolver* learns that the DIDL document with identifier '`info:lanl-repo/i/58f202ac`' is located at base URL '`baseURL(1)`'. The aDORe OpenURL *Resolver* retrieves the DIDL document from its OAI-PMH repository by issuing the OAI-PMH `GetRecord` request written below:

  ```
  [ baseURL(1)?verb=GetRecord&
        identifier=info:lanlrepo/i/58f202ac&
        metadataPrefix=didl ]
  ```

  Next, from the obtained DIDL document, the element with XML ID '`uuid-0000a01c`' can be extracted. As shown in Table 13, the resulting fragment is a `Component` element. By definition of MPEG-21 DIDL, each `Component` element groups bit-equivalent datastreams (or *resources*). In aDORe, datastreams are provided By Reference; and following MPEG-21 DIDL, network locations are conveyed in the `ref` attribute of a `Resource` child element. In aDORe, the network locations themselves are compliant with the NISO OpenURL Framework and point to datastreams that are stored in ARCfiles (See also Section 1.1.2). The *ContextObject* of the OpenURL is expressed using the KEV *Format* and transported using the By Value HTTP GET method, and is written as:

  ```
  [ baseURL(ARC_FILE)?
        url_ver=Z39.88-2004&
        rft_id=DatastreamKey ]
  ```

  Issuing the above OpenURL request results in a look-up of the datastream key in a URI-based index that was created for the targeted ARCfile. This look-up reveals the byte-offset and byte-count of the required datastream in the ARCfile. Given this information, a process can access the datastream in the ARCfile and return it to the aDORe OpenURL *Resolver*.

  The aDORe OpenURL *Resolver* passes the datastream (as well as associated secondary data), and the value of the OpenURL *ServiceType* on to the Transform Engine. Through inspection of the received information, and a look-up in the knowledge database, the Transform Engine determines whether the requested service with identifier '`info:lanl-repo/svc/marc_2_mods`' can at all be applied to the datastream. Assuming that the service can indeed be applied, the appropriate service is called, and executed by the Transform Engine. The Transform Engine returns the MODS record to the aDORe OpenURL *Resolver*, which delivers it to the requesting agent.

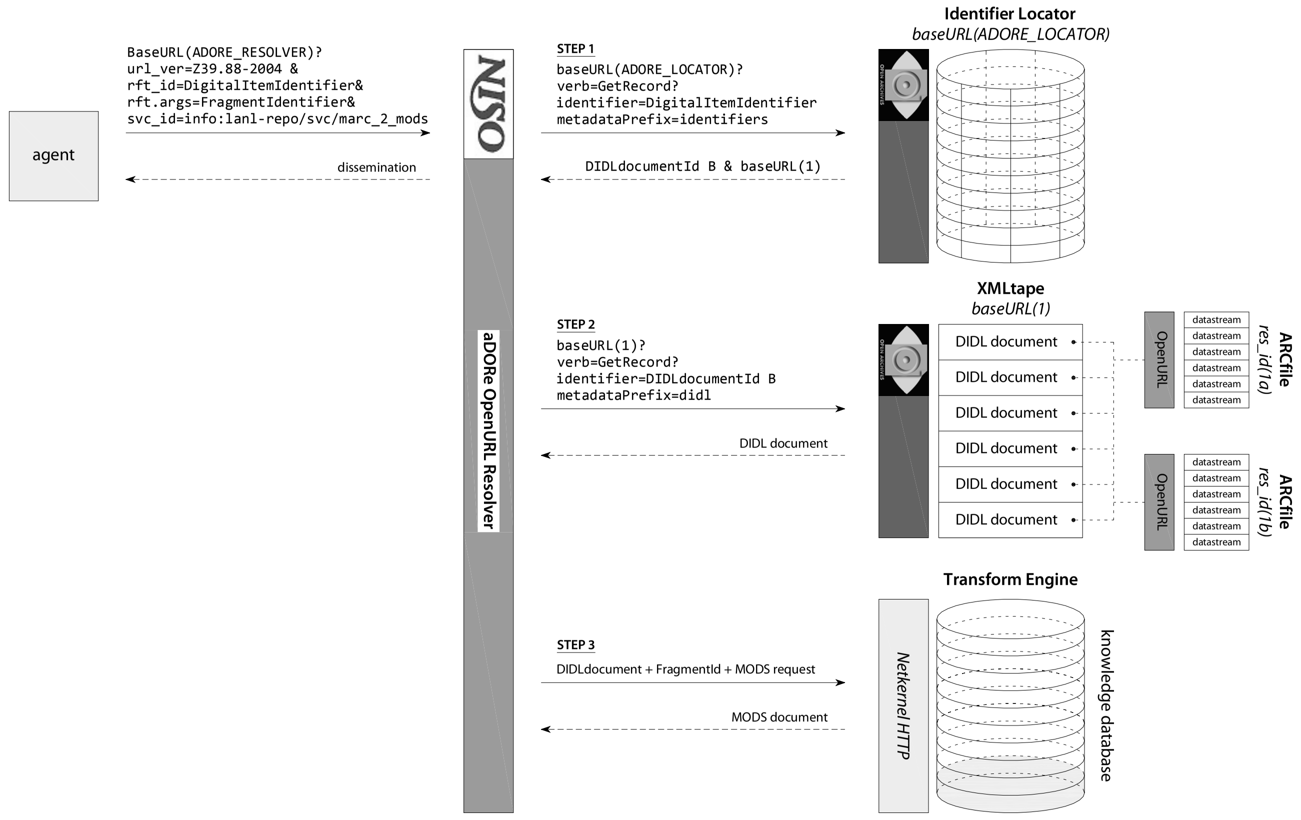


Figure 18. The aDORe OpenURL *Resolver*: An interface for requesting datastream-level services

1.6. Conclusion

So far, this Chapter described an approach to uniformly make a vast and, ever growing collection of digital assets available to downstream applications. To this end, several interacting components have been introduced: The Autonomous OAI-PMH Repositories (i.e., the XMLtapes), the ARCfile OpenURL *Resolvers*, the Repository Index, the Identifier Locator, the Transform Engine, the OAI-PMH Federator and the aDORe OpenURL *Resolver*. Each of those components is tasked with a specific sub-function of the repository environment. For most of the interactions between the components, the lightweight OAI-PMH protocol plays a prominent role.

Of specific importance, in the context of this study, are aDORe's interfaces to harvest and obtain XML-based packages of digital assets, and obtain constituent datastreams of digital assets. A first set of interfaces has been provided at the level of the individual repositories that store the actual assets. Two storage mechanisms are combined and made accessible via protocol-based interfaces.

- □ The Autonomous OAI-PMH Repositories (in aDORe implemented as XMLtapes) host the XML-based representations of the digital assets, and are made accessible through the OAI-PMH. Contained records are DIDL documents. The *identifier* used by the OAI-PMH is the identifier of the DIDL document.
- □ ARCfiles concatenate constituent datastreams of digital assets, and are made accessible through an OpenURL *Resolver*. The *Referent* of the *ContextObject* is a datastream stored in an ARCfile. The sole service provided by the OpenURL *Resolver* is delivery of the datastream referenced on the OpenURL.
- □ The interconnection between both is provided by including OpenURL links to constituent datastreams of a digital asset in DIDL document representing the digital asset.

The protocol-based nature of both components further increases the flexibility in light of evolving technologies as it introduces a layer of abstraction between the access method and the technology by which both components are implemented. Also, because of the use of existing standards, it allows the use off-the-shelf tools. Note that neither the Autonomous OAI-PMH Repositories, nor their XMLtape implementation, are tied to aDORe's choice of MPEG-21 DIDL for representing digital assets. The approach can also be used when digital assets are represented using other packaging formats such as METS or IMS-CP.

A second set of interfaces consists of the OAI-PMH Federator and the aDORe OpenURL *Resolver*; two interfaces through which downstream applications can harvest and obtain stored material from the aDORe environment.

- □ The OAI-PMH Federator provides an interface through which batches of stored DIDL documents can be harvested. This interface is centered on the OAI-PMH and exposes DIDL documents stored in the aDORe environment as OAI-PMH *metadata*. The OAI-PMH *identifier* is put on par with the identifier of the DIDL document. Because of this, and because of the fact that, in aDORe, a new DIDL document is created whenever a new version of a previously ingested digital asset needs to be ingested, the OAI-PMH Federator allows to harvest any version available in the aDORe environment. To this end, the existence of a unique DIDL document Identifier ensures that different versions remain distinguishable.

- The aDORe OpenURL *Resolver* provides an interface through which DIDL documents and datastream-level disseminations can be obtained on a one-by-one basis. This interface is compliant with NISO's OpenURL Framework for Context-Sensitive Services, and both Digital Item Identifiers and DIDL document Identifiers, including the XML ID Fragment Identifiers, can be used to identify the material of interest. The experiment introduces a novel use of the NISO OpenURL standard which begs reflection on its use as an interoperable interface to request disseminations of resources stored in heterogeneous repositories. The fact that NISO's OpenURL allows to convey information that is relevant for the delivery of a context-sensitive response to a service request makes the idea all the more appealing.

The modular and protocol-based nature of the aDORe design suggests the possibility of the use of the aDORe interface concepts in a distributed Web environment, even though they were conceived and implemented for a local environment. A particular area of interest is a federation of Institutional Repositories (Jerez, Liu, Hochstenbach & Van de Sompel, 2004). A federation of community-based OAI-PMH repositories containing compound digital assets can be imagined for which a Repository Index, an Identifier Locator and an OAI-PMH Federator are deployed. The population of a Repository Index and Identifier Locator in this scenario would be different from the approaches described in the context of the aDORe repository architecture but its OAI-PMH functionality would remain identical. Again, an OAI-PMH Federator would hide the complexity of the multitude of repositories to harvesters, and through a component similar to the Transform Engine, it could support crosswalks between multiple compound asset format(s) used by the federation.

Other attractive characteristics of the aDORe environment are listed below.

- The modular nature of the design, especially the ability to store DIDL documents in multiple Autonomous OAI-PMH Repositories provides reassurances regarding the scalability of the solution.
- Handling the binding of dissemination methods to stored digital assets in a dynamic manner avoids what would likely become a significant workload in editing stored DIDL documents to update embedded methods.
- Avoiding editing of stored DIDL documents is further realized by the creation of a new DIDL document whenever a new version of a digital asset needs to be added. As all such versions share the same OAIS Content Information Identifier, the parallel identification mechanism (OAIS Content Information Identifier, OAIS AIP Identifier) ensures that the different versions remain distinguishable. And, through the Identifier Locator, all versions can be located. The parallel identification approach also allows addressing resources that have no OAIS Content Information Identifier in a direct manner.
- The standards-based nature of the design allows the use of off-the-shelf, open source tools in all areas of the aDORe environment. This has resulted in significant time savings for the development, but also allows inspection, improvement modification of code if required. For example, various free and open source OAI-PMH and OpenURL tools are used,

including OCLC's OAICat (OCLC, 2005a), OAIHarvester (OCLC, 2005b), OAI Viewer (OCLC, 2004) and OpenURL 1.0 (OCLC, 2005c). Also, it is expected that reference implementations of MPEG-21 DIDL parsers will become available for integration into the environment.

- Furthermore, the protocol-based design provides the flexibility to choose from or migrate between different software implementations for aDORe components, without the overall design and functionality being affected.
- As far as can be verified, at the time of writing, aDORe is the first environment in which various MPEG-21 technologies are used at a significant scale. If nothing else, aDORe strongly suggests the significance of the MPEG-21 standardization effort to the digital library, archiving and e-learning communities.

An initial implementation of the aDORe repository architecture has been brought into production at the end of 2003. A new release of the aDORe software has been finalized mid 2005. The current aDORe implementation has been developed in Java and Perl and has been tested in Linux and Solaris environments. The modular and protocol-based nature of the design enables the distribution of aDORe components across multiple machines; as a matter of fact, the Autonomous OAI-PMH Repositories, the OAI-PMH Federator, the OpenURL *Resolver*, the Repository Index and the Identifier Locator can all be running on separate machines.

2. The use of the OAI-PMH for the accurate transfer of digital assets between the APS and aDORe environments

Systematic and ongoing transfer of published content in a networked environment is a challenging task. Many degrees of freedom exist for devising a solution, including the choice between a push and a pull model, the choice of a method to package content for transport, and the choice of a method to transport the packaged content over the network. Typically a different solution to the same problem exists per content producer and per content type.

Being a large-scale aggregator of published content, the Research Library of the Los Alamos National Laboratory is directly confronted with the significant overhead caused by the lack of standards in existing content transfer solutions. Therefore, inspired by its work on the aDORe repository infrastructure, the Research Library of LANL opted to explore the establishment of a standards-based content transfer framework, in the context of an agreement between LANL and the American Physical Society (APS) [http://www.aps.org/]. Under the terms of that agreement, LANL will mirror content from the APS collection, both for the purposes of creating discovery services and preserving digital content. Over the last year, LANL has worked with the APS on the design and implementation of a solution aimed at replicating the assets of the APS collection at LANL. Although the experiment has its origin in a specific publisher-to-library use case, it aims to explore a broadly applicable solution for the transfer of assets between a content provider and a content consumer. Two sets of requirements were considered.

First, requirements of the content transfer mechanism. These requirements include:

- Transfer of assets must be achieved in a timely manner: The consuming archive (LANL) must remain tightly synchronized with the producing archive (APS).
- The solution must be neutral to the repository architectures at the producing archive and at the consuming archive: In order for the solution to be applicable beyond the specific APS/LANL use case, it should be neutral to specific repository architectures.
- Reciprocity of the content transfer solution must exist: It must be possible for the producing archive to harvest assets back from the consuming archive.

Second, a requirement about accuracy and authenticity of the transferred content in light of digital preservation:

- The assets harvested by the consuming archive must be accurate: Guarantees must be provided that the copies of the assets stored by both the producing archive and the consuming archive are identical.

This article describes the design and characteristics of the solution that has emerged in response to the aforementioned requirements. The solution is based on the combination of standards that have emerged recently. The standards that play a core role in the design include the MPEG-21 DID (ISO, 2005a), the MPEG-21 DII (ISO, 2003c; Bekaert & Rump, 2005), the W3C XML Signature Syntax and Processing standard (Bartel et al., 2002), and the OAI-PMH (Lagoze et al., 2003b). The core characteristics of the solution are described below.

To meet the requirements regarding the content transfer mechanism:

□ Using the MPEG-21 DID Abstract Model as the 'crosswalk data model' for the transfer of digital assets. Digital assets from the producing archive that conform to the producing archive's internal data model are mapped to the application-neutral MPEG-21 DID Abstract Model before transfer. Upon receipt, the consuming archive maps from this application-neutral data model to its internal data model.

□ Serializing the result of the aforementioned mapping of a digital asset to the MPEG-21 DID Abstract Model as an XML document that is compliant with the MPEG-21 DIDL. This XML document contains (pointers to) constituent datastreams of the asset, secondary data pertaining to the digital asset, and the identifier of the digital asset. The latter is conveyed in a manner compliant with MPEG-21 DII.

□ Exposing those XML documents via an OAI-PMH repository at the end of the producing archive. This allows an OAI-PMH harvester at the end of the consuming archive to incrementally harvest updated and added XML documents that represent the assets from the producing archive.

To meet the requirement regarding accuracy and authenticity:

□ Including XML signatures in the OAI-PMH responses. This allows verification of authenticity and integrity of the XML documents once they have been harvested by the OAI-PMH harvester at the end of the consuming archive. The XML signatures are in accordance with the W3C XML Signature Syntax and Processing standard.

□ Including XML signatures in the XML documents. These XML signatures allow the verification of the authenticity and integrity of the constituent datastreams of the assets once they are collected by the consuming archive. Again, the XML signatures are in accordance with the W3C XML Signature Syntax and Processing standard.

2.1. An OAIS perspective on the content transfer solution

Each asset of the APS collection coincides with a single publication and is complex, in the sense that it may consist of multiple datastreams of a variety of MIME media types. Each asset also holds secondary information, such as an identifier and descriptive information about the publication. Based on the OAIS Information Model presented in Chapter 1, we can say that the APS publication corresponds with the concept of Content Information in the OAIS framework. An identifier assigned to OAIS Content Information object is referred to as an OAIS Content Information Identifier.

In accordance with the OAIS Functional Model, in what follows, the terms 'producing archive' and 'consuming archive' will be used to refer to the archive providing the information and the archive requesting and receiving the information, respectively. The terms 'LANL' and 'APS' will be used when describing application-specific design choices.

In the proposed solution, the manner in which the producing archive and the consuming archive internally represent and package the assets is of no importance. What matters is that both archives understand the application-neutral Information Packages holding the assets that are transferred between the archives during an OAI-PMH-based OAIS Data Submission Session (CCSDS, 2004a).

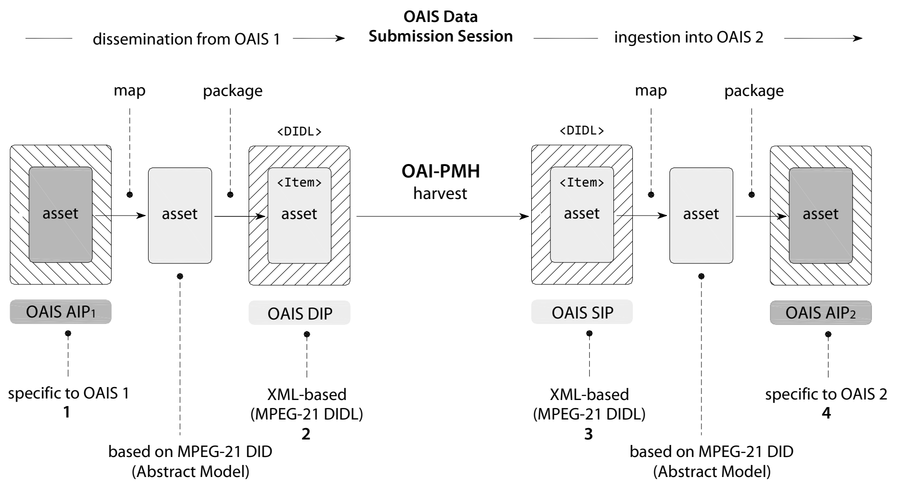


Figure 19. An OAIS perspective on content transfer using the OAI-PMH

- As shown in Figure 19, the producing archive exposes OAIS DIPs through an OAI-PMH interface (number 2 in Figure 19). Those exposed OAIS DIPs result from – dynamically – mapping the assets in the producing archive (packaged in AIP_1 in Figure 19) to the Abstract Model for Digital Items as defined by the MPEG-21 DID Standard. Following this mapping, an XML-based representation of the Digital Item is provided and embedded in a package (or DIDL document) in a manner that is also compliant with the MPEG-21 DID Standard. The resulting OAIS DIPs are application-neutral in the sense that they do not reflect the characteristics of the technical and architectural environment at either the producing or the consuming archive.
- When transferred through the OAI-PMH, the OAIS DIPs (number 2 in Figure 19) disseminated by the producing archive become OAIS SIPs to the consuming archive (number 3 in Figure 19). Once transferred, the consuming archive can map the asset packaged in the application-neutral OAIS SIP to the MPEG-21 DID Abstract Model. The asset can then be represented and packaged as an OAIS AIP, compliant with all other OAIS AIPs stored in the consuming archive (AIP_2 in Figure 19).
- The consuming archive itself may re-expose the harvested assets using the same technique, thereby allowing the producing archive, as well as other archives to incrementally collect digital assets from the producing archive.

2.2. Exposing OAIS DIPs from the producing archive

This section describes the approach taken to expose the digital assets stored in the producing archive to the consuming archive. The approach is standards-based.

To meet the requirements regarding the content transfer mechanism the approach uses:

- □ MPEG-21 DID to package a digital asset of the producing archive as an application-neutral, XML-based OAIS DIP of that digital asset. The OAIS Content Information Identifier of the asset is conveyed in a manner compliant with MPEG-21 DII.
- □ OAI-PMH to expose the XML-based OAIS DIPs from the producing archive. The notion of the OAI-PMH datestamp applied to these XML packages guarantees synchronicity between the producing and the consuming archives.

To meet the requirement regarding accuracy and authenticity the approach uses:

- □ XML signatures embedded in an OAIS DIP to allow verifying the integrity and authenticity of constituent datastreams of a represented asset. The XML signatures are compliant with the W3C XML Signature Syntax and Processing standard.
- □ XML signatures computed over the complete XML-based OAIS DIP, provided in the 'about' container of an OAI-PMH response, to allow verifying the authenticity and integrity of the XML-based OAIS DIP itself. The XML signatures are compliant with the W3C XML Signature Syntax and Processing standard.

2.2.1. Using MPEG-21 DID to create XML-based, application-neutral OAIS DIPs

A digital asset created by the APS typically coincides with a single publication. In the APS archive, an asset has multiple constituent datastreams, including expressive descriptive metadata, a research paper in various formats (PDF, SGML, etc.), and auxiliary content such as datasets and video recordings. Moreover, each such asset has a globally unique DOI (NISO, 2000), which the OAIS Information Model categorizes as a Content Information Identifier.

In order to use the OAI-PMH to transfer digital assets between the producing and the consuming archives, the digital assets must be represented as XML. Therefore, once more, digital assets are represented and packaged using the MPEG-21 DID standard. Several reasons motivated the choice for MPEG-21 DID. A subjective motivator was the established expertise at LANL, resulting from using MPEG-21 DID in the aforementioned aDORe repository, and from being actively involved in its ISO/MPEG standardization process. Another motivator of importance to the described data transfer problem is the existence of an abstract data model (viz. the MPEG-21 DID Abstract Model) that underlies the MPEG-21 DIDL packaging format. Indeed, this Abstract Model provides valuable guidance regarding how to map a digital asset from the producing archive to a Digital Item compliant with the standardized MPEG-21 DID Abstract Model, and how to map from that abstract intermediate representation to a digital asset in the consuming archive. The MPEG-21 DID Abstract Model can function as the standards-based crosswalk between the data models for digital assets as used by various archives.

In the proposed solution, each asset of the producing archive is packaged in a DIDL document that wraps the constituent datastream(s) of that asset. The DIDL document also contains one or more identifiers, as well as secondary information such as media format of constituent datastreams. In what follows, the sample asset given in Table 12 is retaken to illustrate several design choices made. An example of a DIDL document resulting from the mapping and packaging of the sample APS asset is provided in Table 19. The mapping and packaging process is similar to the one described in Section 1.1.1. The core properties are summarized below.

- In accordance with the MPEG-21 DID Abstract Model, a digital asset from the producing archive is mapped to a top-level `Item` element. Constituents of the asset are provided in child elements of this top-level `Item`.
- The structure of the digital asset can be conveyed by providing (nested) sub-`Item` elements as child elements of the top-level `Item`. In the applied mapping, sub-`Item` elements are used whenever the constituent of the asset has an OAIS Content Information Identifier in its own right, whereas `Component/Resource` constructs are used when the constituent does not. Typically, in the case of the APS archive, only the digital asset itself has a Content Information Identifier, and constituents don't. Hence, as can be seen in the sample DIDL document in Table 19, all constituents of the asset are provided as `Component/Resource` constructs that are direct children of the top-level `Item`.
- A constituent datastream can be provided By Reference and/or By Value. In both approaches, the `mimeType` attribute of the `Component` element specifies the MIME media type and subtype of the datastream. Combining the `mimeType` attribute and the `content-Encoding` attribute allows conveying both the MIME type of the datastream as well as the compression that was applied to it, respectively. In the context of the APS experiment, because of performance reasons, several datastreams are being compressed using the 'GNU Zip Compression' algorithm before transfer.
- As can be seen from Table 19, the constituent datastreams of the sample APS asset are provided By Reference and have a MIME type '`application/xml`' and '`application/pdf`' respectively. The network location of the descriptive metadata record is '`http://oai.aps.org/filefetch?id=PhysRevB.69.174413&description=apsmeta`'. The PDF file can be found at '`http://oai.aps.org/filefetch?id=PhysRevB.69.174413&description=print`'.
- Secondary data pertaining to an entity of the MPEG-21 DID Abstract Model are conveyed using `Descriptor/Statement` constructs attached to the entity. The OAIS Content Information is expressed in a manner compliant with MPEG-21 DII. Also, as will be described in Section 2.2.2, to meet the accuracy and authenticity requirement, each `Component/Resource` construct has a `Descriptor/Statement` construct that conveys an XML signature computed over the datastream it provides.
- The top-level `Item` is embedded in the `DIDL` root element to obtain a DIDL XML document. This DIDL document is the OAIS Information Package that packages the digital asset.

```
<?xml version="1.0" encoding="UTF-8"?>
<didl:DIDL xmlns:didl="urn:mpeg:mpeg21:2002:02-DIDL-NS">
  <!--  +++++++++++++++++++++++++++++++++++++++ -->
  <!--  Item representing the APS digital asset -->
  <!--  +++++++++++++++++++++++++++++++++++++++ -->
  <didl:Item id="uuid-279775c8">
    <!-- Content Information Identifier of the APS digital asset -->
    <didl:Descriptor>
```

```
      <didl:Statement mimeType="application/xml; charset=utf-8">
        <dii:Identifier xmlns:dii="urn:mpeg:mpeg21:2002:01-DII-NS">
          info:doi/10.1103/PhysRevB.69.174413</dii:Identifier>
      </didl:Statement>
    </didl:Descriptor>
    <!--  ++++++++++++++++++++++++++++++++++++++++++++++ -->
    <!--  Constituent datastream of the APS digital asset -->
    <!--  ++++++++++++++++++++++++++++++++++++++++++++++ -->
    <didl:Component id="uuid-2798efaa">
      <!-- Creation datetime of the datastream -->
      <didl:Descriptor>
        <didl:Statement mimeType="application/xml; charset=utf-8">
          <dcterms:created
            xmlns:dcterms="http://purl.org/dc/terms/">
          2004-05-20T11:40:31Z</dcterms:created>
        </didl:Statement>
      </didl:Descriptor>
      <!-- XML signature computed over the datastream -->
      <didl:Descriptor>
        <didl:Statement mimeType="application/xml; charset=utf-8">
          <dsig:Signature
            xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
              ...
          </dsig:Signature>
        </didl:Statement>
      </didl:Descriptor>
      <!-- By Reference provision of the datastream -->
      <didl:Resource mimeType="application/xml; charset=utf-8"
        ref="http://oai.aps.org/filefetch?id=PhysRevB.69.174413&
        description=apsmeta"/>
    </didl:Component>
    <!--  ++++++++++++++++++++++++++++++++++++++++++++++ -->
    <!--  Constituent datastream of the APS digital asset -->
    <!--  ++++++++++++++++++++++++++++++++++++++++++++++ -->
    <didl:Component id="uuid-279ee2d4">
      <!-- Creation datetime of the datastream -->
      <didl:Descriptor>
        <didl:Statement mimeType="application/xml; charset=utf-8">
          <dcterms:created
            xmlns:dcterms="http://purl.org/dc/terms/">
          2004-05-18T15:43:33Z</dcterms:created>
        </didl:Statement>
      </didl:Descriptor>
      <!-- XML signature computed over the datastream -->
      <didl:Descriptor>
        <didl:Statement mimeType="application/xml; charset=utf-8">
          <dsig:Signature
            xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
              ...
          </dsig:Signature>
        </didl:Statement>
      </didl:Descriptor>
      <!-- By Reference provision of the datastream -->
      <didl:Resource mimeType="application/pdf"
        ref="http://oai.aps.org/filefetch?id=PhysRevB.69.174413&
        description=print"/>
    </didl:Component>
  </didl:Item>
</didl:DIDL>
```

Table 19. A sample DIDL document exposed by the APS archival system.

2.2.2. Using W3C XML Signature and Processing to enable verification of integrity and authenticity

When transferring DIDL documents from the producing archive to the consuming archive, there may be a requirement to guarantee the integrity of the transferred content and the authenticity of the sender. This is the case in the APS/LANL experiment. I next explore the technologies used to achieve this requirement.

2.2.2.1. Digests, digital signatures, certificates, and XML signatures

This section provides a crash-course in issues related to data security, in order to allow an understanding of the approach that is used to enable verification of integrity and authenticity in the context of the proposed data transfer solution. The following concepts from the domain of data security play a fundamental role:

- *Digests*: A digest value, often referred to as a hash value, provides a unique fingerprint of the data to be transferred. The message digest depends upon the input data as well as a digest algorithm. Since it is assumed that it is computationally infeasible to produce two messages having the same message digest, the digest value can be used to verify the integrity of the data – that is, to ensure that the data has not been altered while on its way from the sender to the receiver. The sender sends the message digest value along with the message. On receipt of the message, the recipient repeats the digest calculation. If the message has been altered, the digest value will not match and the alteration will be detected. Note, however, that both the digest value and the message could have been altered. This kind of change may not be detectable at the recipient end. As such, a message digest is necessary yet not sufficient to ensure the integrity of the data. A digital signature will be needed to remedy this shortcoming.
- *Digital signatures*: An asymmetric encryption algorithm could be used to generate a pair of keys consisting of a public and a private key. A private key (which is kept confidential) is used by the sender to produce a digital signature over the digest value. The recipient of the data first checks the integrity of the digest value by repeating the digest calculation. The recipient then uses the public key of the sender (which is open for use by anyone who wishes to securely communicate with the sender) to verify the signature. If the digest value has been altered, the signature will not verify at the recipient end. If both the digest value and signature verification steps succeed, one can conclude that the data has not been altered after digest calculation (data integrity) and the message is coming from the owner of the public key (user authentication).
- *Certificates*: A certificate is a data structure that holds the identification information (such as name and contact address) and the public key of the certificate owner. A certificate issuing authority issues certificates to people or organizations. The certificate issuing authority will also sign the certificate using its own private key; any interested party can verify the integrity of the certificate by verifying the signature (using the authority's public key).

Because in the proposed data transfer solution, XML documents are transferred, usage of the XML signature & Processing specification (Bartel et al., 2002) has been explored and adopted. This W3C specification defines an XML-compliant digital signature syntax that adds authentication, data integrity, and support for non-repudiation to the data that is signed. It builds on the previously described digest, signature and certificate concepts. A detailed walk-through is provided in Section 2.2.2.2. Dependent on the relative position of the XML signature and the signed data, three different types of XML signatures can be distinguished:

- *Detached*: A detached signature is an XML signature in which the signed data and XML signature exist separately from each other. Detached XML signatures can sign content that is external to the XML document itself, or they can be applied within the same XML document where the XML signature and the signed data are sibling elements within that document.
- *Enveloped*: An enveloped signature is an XML signature of a document in which the XML signature itself is embedded within the signed document.
- *Enveloping*: An enveloping signature is an XML signature in which the signed data is embedded within the XML signature.

2.2.2.2. XML signatures for constituent datastreams of an asset

In the proposed solution, an XML signature is provided for each constituent datastream of an asset of the producing archive. Following MPEG-21 DIDL, each such XML signature is provided as secondary data using a `Descriptor/Statement` construct attached to the `Component/Resource` construct that contains the datastream. These datastream-level XML signatures will allow checking whether or not a datastream provided by the producing archive is unchanged at the time that the consuming archive has gathered it. This type of XML signature falls under the OAIS category of Fixity Information.

An XML signature is represented by a `Signature` element of the XML DSIG namespace, and it typically consists of three parts. The first part of the XML signature, represented by the `Reference` element, holds various bits of information related to the calculation of the digest of the data being signed, including:

- A URI reference identifying the data that is used as input to the digest calculation process. As mentioned above, a datastream can be included inside the DIDL document (By Value) or can reside outside the DIDL document (By Reference). In the first scenario, one may point to the datastream using an XML Fragment Identifier (XPointer). If the datastream resides outside the DIDL document, a URI reference should be used. This information is conveyed via the `URI` attribute of the `Reference` element (of the DSIG XML namespace). If no `URI` attribute is provided, the receiving application is expected to know the identity of the data.
- An ordered list of transforms (conveyed by the `Transforms` element, which itself is a child of the `Reference` element) describing how the signer obtained the data that was digested. When transforms are applied, the digest is not calculated over the datastream as obtained from the `URI` attribute, but over the result of applying the specified transforms to that obtained datastream. The order in which the transforms are applied is important. The input to the first transform is the data obtained from the `URI` attribute. The output of each transform serves as input to the next transform. Dependent on the nature of the datastreams and the technique used to provide the datastream (By Value or By Reference),

different transforms on the datastreams may be required prior to digest calculation. Four types of transforms are of interest in the context of the APS/LANL experiment (see columns 3 to 6 of Table 20):

- *XPath Transform* (Boyer, Hughes & Reagle, 2002) (Column 3): The main purpose of this transform is to provide a technique for extracting a portion of an XML document that is to be used as input to the digest calculation process. It should be noted that, in many cases, a similar functionality can be achieved using a URI with a Fragment Identifier in the URI attribute (see above). However, the W3C XML Signature Syntax and Processing specification does advise against the use of URI Fragment Identifiers different than the same-document XPointer `#xpointer(id('ID'))`. In those cases, for the sake of interoperability, fragments of an XML resource should be obtained using an XPath transform.
- *Canonicalization Transform* (Boyer, 2001; Boyer, Eastlake & Reagle, 2002) (Column 4): This transform describes a method for generating a physical representation, the canonical form, of an XML document. Canonicalization algorithms are important in XML signature applications because message digest algorithms treat XML data as octet streams. In addition, two different octet streams can represent the same XML resource. For example, they may differ in their entity structure, attribute ordering, and character encoding.
- *Base64-decoding Transform* (Column 5): The input data of this transform is decoded by means of the base64 algorithm. This transform is useful if an application needs to sign the raw data obtained after base64-decoding the content of an XML element. For example, in this experiment, a base64-encoding algorithm is used to embed a binary datastream (Rows 2 and 3 of Table 20) in a `Resource` element. The base64-decoding transform allows checking if problems occurred during this base64-encoding process.
- *Content-decoding Transform* (Column 6): This transform specifies a content-decoding algorithm, such as GNU zip decompression, that is used as a modifier to the datastream that is being transferred (Rows 2 and 4 of Table 22). When present, its value indicates which content-decoding algorithm must be applied in order to obtain the datastream for which a digest is calculated. Possible content-decoding values are listed in RFC 2616 (Fielding et al., 1999) and registered by IANA.

- The digest algorithm and the digest value itself. The digest value results from applying the digest algorithm to the identified and transformed datastream.

A second part of the XML signature is related to the generation of the signature value. It identifies an algorithm – conveyed by the `SignatureMethod` element – used to calculate the signature value, and the signature value itself – conveyed by the `SignatureValue` element – that results from applying the algorithm to the `SignedInfo` element containing the digest (as described in the previous step). Also, it identifies a canonicalization transform – conveyed by the `CanonicalizationMethod` element – that is applied to the `SignedInfo` element prior to

	1	2	3	4	5	6	7
	provision type of signed data	**media type of signed data**	**XPath transform**	**canonicalize transform**	**base64-decoding transform**	**content-decoding transform**	**signature type**
1	By Value	XML (UTF-8)	Y	Y	N	N	detached
2	By Value	Content-encoded data	Y	N	Y	Y	detached
3	By Value	All data not covered in 1 and 2	Y	N	Y	N	detached
4	By Reference	Content-encoded data	N	N	N	Y	detached
5	By Reference	All data not covered in 4	N	N	N	N	detached

Table 20. Using W3C XML Signatures and Processing to sign datastreams referenced or embedded in a DIDL document

	1	2	3	4	5	6	7
	provision type of signed data	**media type of signed data**	**XPath transform**	**canonicalize transform**	**base64-decoding transform**	**content-decoding transform**	**signature type**
1	(By Value) DIDL document	XML (UTF-8)	Y	Y	N	N	detached

Table 21. Using W3C XML Signatures and Processing to sign DIDL documents

signature calculation. It is important to note that in order to generate the signature value, the signature algorithm is used in combination with the signer's private key.

An optional third part of the XML signature, represented by the `KeyInfo` element, contains an X.509 certificate that conveys the public key needed for signature verification.

Table 22 shows an XML signature pertaining to the PDF datastream of the sample APS asset. The XML signature is conveyed using a `Descriptor/Statement` construct that, in turn, is encapsulated in a `Component` element. The datastream (represented by the `Resource` element) is provided By Reference, and no content-encodings have been applied. As such, the syntax of the XML signature follows the scenario outlined in Row 5 of Table 20.

```
<didl:Component id="uuid-279ee2d4">
  <!-- Descriptor/Statement containing an XML signature computed over the datastream -->
  <didl:Descriptor>
    <didl:Statement mimeType="application/xml; charset=utf-8">
      <!-- Start of the XML signature construct -->
      <dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
        <dsig:SignedInfo>
          <dsig:CanonicalizationMethod
            Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
          <dsig:SignatureMethod
            Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
          <dsig:Reference URI="http://oai.aps.org/filefetch?
            id=PhysRevB.69.174413&description=print">
            <dsig:DigestMethod
              Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
            <dsig:DigestValue>2jmj7l5rSw0yVb/vlWAYkK/YBwk=</dsig:DigestValue>
          </dsig:Reference>
        </dsig:SignedInfo>
        <dsig:SignatureValue>LzPzUgO6uF870gXs20F4r2pYD...</dsig:SignatureValue>
        <dsig:KeyInfo>
          <dsig:X509Data>
            <X509Certificate>MIICwzCCAiygAwIBAgIGA...</X509Certificate>
          </dsig:X509Data>
        </dsig:KeyInfo>
      </dsig:Signature>
    <!-- End of the XML signature construct -->
    </didl:Statement>
  </didl:Descriptor>
  <!-- By Reference provision of the datastream -->
  <didl:Resource mimeType="application/pdf"
    ref="http://oai.aps.org/filefetch?id=PhysRevB.69..."/>
</didl:Component>
```

Table 22. A W3C XML signature computed over the PDF datastream of the sample asset

2.2.3. Exposing OAIS DIPs via the OAI-PMH

2.2.3.1. Use of packaging formats with the OAI-PMH

In the proposed content transfer solution, the OAI-PMH is used as the interoperability protocol to harvest DIDL documents from the producing archive to the consuming archive. Indeed, as shown in the aDORe experiment, introducing packaging formats as *metadata formats* guarantees that a change to any constituent of a resource will result in a change of the

OAI-PMH *datestamp* of the package representing the resource. As a result, the OAI-PMH *datestamp* offers a fully reliable trigger to incrementally harvest updated and added *resources* when those *resources* are represented using a packaging format. In light of the proposed content transfer solution, this feature is essential to trigger harvesting of assets that were added to or updated in the producing archive.

The OAI-PMH repository operated by the producing archive has the following properties:

- There is a one-by-one correspondence between a digital asset in the producing archive and an OAI-PMH *item*; the digital asset is the OAI-PMH *resource*.
- *metadata* in various formats can be disseminated from each OAI-PMH *item*. Of specific interest to the proposed solution is the dissemination based on MPEG-21 DIDL. A DIDL document that packages an asset is provided as *metadata*.
- In order to remain tightly synchronized with the producing archive, the time granularity of the OAI-PMH repository is seconds-level. If tight synchronization is not required, a day-level precision may be sufficient.
- In the APS/LANL implementation, the OAI-PMH *identifier* of the OAI-PMH *item* is derived from the Content Information Identifier of the digital asset. As a matter of fact, the OAI-PMH *identifier* uses the value of the Content Information Identifier value as part of an identifier that is compliant with the OAI-PMH *identifier* Format (Lagoze, Van de Sompel, Nelson & Warner, 2002a).
- The OAI-PMH *datestamp* of the OAI-PMH *record* that contains the DIDL document as *metadata* is the datetime of creation of a digital asset, or the datetime of the most recent change of any constituent of the asset, including datastreams, and secondary data. This property, which is a direct result of the application of the notion of the OAI-PMH *datestamp* to the representation of an asset using a packaging format, is essential to allow the consuming archive to remain permanently synchronized with the producing archive.
- As will be explained in Section 2.2.3.2, in order to meet the accuracy and authenticity requirement, an '`about`' container conveying an XML signature for a complete DIDL document is provided per OAI-PMH *record* in an OAI-PMH response.

2.2.3.2. XML signatures for DIDL documents

In addition to the XML signatures provided at the level of each constituent datastream of a digital asset of the producing archive, an XML signature is also created for the complete DIDL document that packages the digital asset, and that is exposed through the OAI-PMH repository of the producing archive. This XML signature will allow checking the integrity of the transferred Information Package as a whole, after it has been harvested. The DIDL-level XML signature provides guarantees that are not provided by the datastream-level XML signatures. Indeed, data-corruption may, for example, occur in secondary data (`Descriptor/ Statement` constructs) that is not covered by datastream-level XML signatures. Of particular concern is corruption that might occur at the level of the OAIS Content Information Identifier of the asset.

This DIDL-level XML signature is provided in the ‘about’ container of the OAI-PMH *record* that contains the DIDL document as *metadata*. Doing so is in accordance with the OAI-PMH, which specifies that an ‘about’ container provides secondary data pertaining to the *metadata* provided in an OAI-PMH *record*. Table 21 summarizes the properties of this XML signature. Table 22 shows an example.

The OAI-PMH framework allows for batch retrieval of DIDL documents (using the OAI-PMH `ListRecords` request), as well as for access to individual DIDL documents (using the OAI-PMH `GetRecord` request). Dependent on the request being issued, an OAI-PMH response may contain one or more OAI-PMH *records*. We can distinguish two approaches for identifying the DIDL document that is to be digested/signed. Both approaches function independently of whether an OAI-PMH response contains one or more DIDL documents.

- Using an XPath transform: A `URI` attribute is appended to the `Reference` element, and the value of the attribute is kept blank (‘’). This empty value indicates that the data being identified is the complete XML document containing the signature. As such, the `URI` attribute refers to the XML document representing the OAI-PMH response. In addition, an XPath Filter transform 2.0 is provided (See column 3 of Table 21) with a value ‘`here()/ancestor::oai-pmh:record/oai-pmh:metadata/*`’, to filter out the DIDL document to which the ‘about’ container is attached from the complete OAI-PMH response. Also, a canonicalization transform is provided to generate the canonical form of the DIDL document that results from applying the XPath transform. The result is used as input to the digest calculation process.
- Relying on the semantics of the OAI-PMH: The `URI` attribute and the XPath transform could be omitted altogether. In this scenario, it is expected that the receiving application understands the identity of the data being digested and signed. Both archives would have to rely on the semantics of the OAI-PMH ‘about’ container to identify the data to be signed. While this solution is compliant with the W3C Signature Syntax and Processing specification, and is far more processing efficient, it would not allow the use of off-the-shelf XML Signature and Processing tools (as these tools are not aware of the semantics of the OAI-PMH model).

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<responseDate>2004-09-01T15:33:28Z</responseDate>
<request identifier="oai:doi/10.1103/PhysRevB.69.174413" verb="GetRecord"
  metadataPrefix="didl">http://oai.test.ridge.aps.org/oai</request>
<GetRecord>
  <record>
    <header>
      <identifier>oai:aps.org:PhysRevB.69.174413</identifier>
      <datestamp>2004-05-18T15:43:33Z</datestamp>
      <setSpec>journal:PRB</setSpec>
      <setSpec>journal:PRB:69</setSpec>
    </header>
    <metadata>
      <!-- DIDL document packaging an APS asset -->
      <didl:DIDL xmlns:didl="urn:mpeg:mpeg21:2002:02-DIDL-NS"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

```
          ...
        </didl:DIDL>
      </metadata>
      <about>
        <!-- XML signature computed over the DIDL document -->
        <dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#">
          <dsig:SignedInfo>
            <dsig:CanonicalizationMethod
              Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#"/>
            <dsig:SignatureMethod
              Algorithm="http://www.w3.org/2002/06/xmldsig#rsa-sha1"/>
            <dsig:Reference URI="">
              <dsig:Transforms>
                <dsig:Transform Algorithm="http://www.w3.org/.../xmldsig-filter2">
                  <xpath:XPath
                    xmlns:xpath="http://www.w3.org/.../xmldsig-filter2"
                    xmlns:oai-pmh="http://www.openarchives.org/OAI/2.0/"
                    Filter="intersect">
                    here()/ancestor::oai-pmh:record/oai-pmh:metadata/*
                  </xpath:XPath>
                </dsig:Transform>
                <dsig:Transform Algorithm="http://www.w3.org/.../xml-exc-c14n#"/>
              </dsig:Transforms>
              <dsig:DigestMethod
                Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
              <dsig:DigestValue>a1YkQDH/XGdccOaiyxOrP6AQBeM=</dsig:DigestValue>
            </dsig:Reference>
          </dsig:SignedInfo>
          <dsig:SignatureValue>dRM9axVQYPMd0vfzkbsta...</dsig:SignatureValue>
          <dsig:KeyInfo>
            <X509Data xmlns="http://www.w3.org/2000/09/xmldsig#">
              <X509Certificate>MIICwzCCAiygAwIBAgIGA...</X509Certificate>
            </X509Data>
          </dsig:KeyInfo>
        </dsig:Signature>
      </about>
    </record>
  </GetRecord>
</OAI-PMH>
```

Table 23. An OAI-PMH `GetRecord` response containing an APS DIDL document

2.3. Ingesting OAIS SIPs in the consuming archive

This section describes the process that runs at the end of the consuming archive, and that is devised to recurrently collect new and updated assets from the producing archive and to store those in the pre-ingest area of the consuming archive. The process is depicted in Figure 20 and consists of the actions described below.

Related to the transfer mechanism part of the solution:

- Harvesting, via OAI-PMH, DIDL documents that are XML-based packages of assets from the producing archive.
- Collecting constituent datastreams of the assets from the harvested DIDL documents, by extracting embedded base64-encoded data in case a datastream is delivered By Value, or through resolving a URI in case a datastream is provided By Reference.

- Recording the results of these actions in control files.

Related to the accuracy and authenticity part of the solution:

- Verifying authenticity and integrity by checking the XML signatures of both the DIDL document and the constituent datastreams.
- Recording the results of these actions in control files.

Once this process has been concluded, and the resulting materials have been collected into the pre-ingest area of the consuming archive, they can be further processed to meet the criteria for ingestion into the consuming archive and to be ingested consecutively.

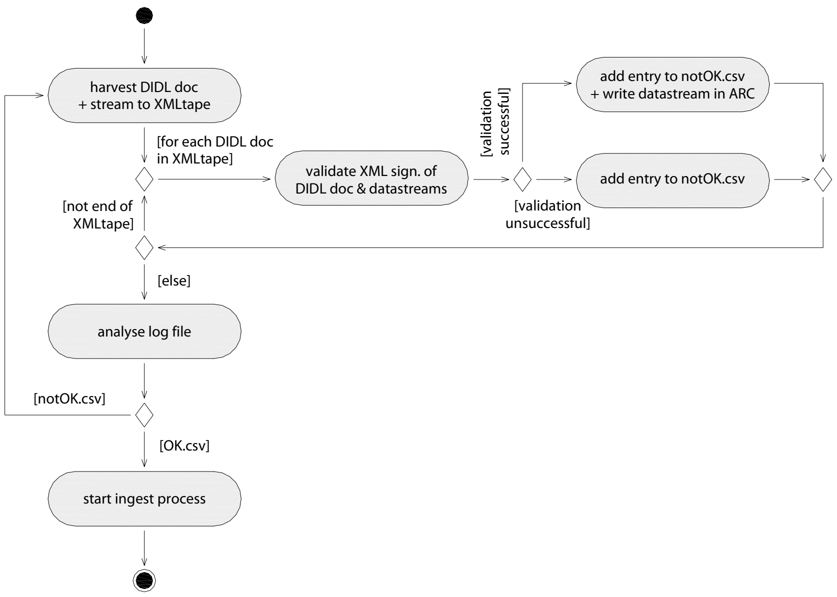

harvest DIDL doc + stream to XMLtape
[for each DIDL doc in XMLtape]
validate XML sign. of DIDL doc & datastreams
[validation successful]
add entry to notOK.csv + write datastream in ARC
add entry to notOK.csv
[validation unsuccessful]
[not end of XMLtape]
[else]
analyse log file
[notOK.csv]
[OK.csv]
start ingest process

Figure 20. UML Activity Diagram of the pre-ingest area of the aDORe environment

2.3.1. Harvesting of DIDL documents via the OAI-PMH

Through recurrent OAI-PMH harvesting, the consuming archive can collect DIDL documents from the OAI-PMH repository at the end of the producing archive. These DIDL documents are XML-based packages of digital assets from the producing archive. The semantics of the OAI-PMH *datestamp* for the exposed DIDL documents ensures that all DIDL documents that package digital assets that have been added or updated since the previous harvesting session will be harvested. The harvested DIDL documents are stored in the pre-ingest area of the consuming archive.

The specifics of this part of the OAI-PMH-based *resource* harvesting process are as follows:

- Periodically, an off-the-shelf OAI-PMH harvester operated by the consuming archive issues the following incremental harvesting request:

  ```
  [ baseURL(PRODUCING_ARCHIVE)?
  ```

```
      verb=ListRecords&
      from=T1&
      metadataPrefix=didl ]
```

Hereby, T1 is the datetime (with a precision of seconds) of the previous harvest. This datetime is expressed according to ISO 8601 (ISO, 2004a; Wolf & Wicksteed, 1997). The harvesting frequency is determined by how tightly the consuming archive needs to be synchronized with the producing archive. It is anticipated that LANL will poll APS on a three to four hour basis.

- Once harvested, the OAI-PMH *records* (including OAI-PMH header information, the DIDL document, and the '`about`' containers) are streamed into XMLtapes (Liu et al., 2005). In addition, provenance information pertaining to the harvest is being added to the tape record. This provenance information is expressed using the OAI-PMH provenance XML Schema (Lagoze, Van de Sompel, Nelson & Warner, 2002b), and enables a repository to know when a DIDL document has been collected. As described in Section 1.1.2, XMLtapes are used as a persistent storage mechanism in LANL's aDORe repository. But in aDORe, XMLtapes are also used as a temporary storage medium for OAI-PMH harvesting processes, and are used as such in the described digital asset transfer process.

2.3.2. Gathering constituent datastreams of assets

Once the harvesting of DIDL documents is completed, a separate process run by the consuming archive is tasked with:

- Collecting all the constituent datastreams of the assets for which DIDL documents were harvested, and,
- Verifying the authenticity and integrity of both the harvested DIDL document and the collected constituent datastreams.

In the LANL implementation, this process starts by parsing the XMLtape(s) resulting from the harvesting process, and by passing, one-by-one, each DIDL document contained in the XMLtape on to a sub-process tasked with collecting datastreams and verifying authenticity and integrity. That sub-process operates as follows (see also Figure 20):

- First, if the content transfer mechanism has requirements regarding authenticity and accuracy that were met by the producing archive through the inclusion of XML signatures, then the XML signature of the DIDL document contained in the '`about`' container is verified. Details are provided in the Section 2.3.3. If the verification is unsuccessful, information about the faulty DIDL document is written to a log file – notOK.csv – that summarizes failures in the resource harvesting process. The sub-process stops, and control is handed back to the process that parses the XMLtapes. The next DIDL document is handed over to the sub-process.
- Next, the sub-process continues by collecting, one-by-one, all constituent datastreams of the asset. The following actions are undertaken:
  - If no XML signatures are provided to ensure authenticity and accuracy of datastreams, collecting the datastream from a DIDL document is achieved by processing the

`Resource` elements, each of which contains a constituent datastream provided either By Value or By Reference, or both.

- If an XML signature is included in a `Descriptor/Statement` construct attached to the `Component` element that contains the datastream, verification of the authenticity and integrity of a datastream requires processing this XML signature. This process requires collecting the datastream. As a result, in order to avoid duplicate work, collecting a datastream and verifying its authenticity and integrity are integrated in the processing of the XML signature. For example, when verifying the XML signature of the PDF datastream of the sample asset, the datastream is obtained by dereferencing the URI provided in the `URI` attribute appended to the `Reference` element of the XML signature (i.e., the value '`http://oai.aps.org/filefetch?id=PhysRevB.69.174413&description=print`'). The PDF datastream is obtained as part of the XML signature validation process to obtain the datastream to be digested.

The following scenarios are possible:

- Collecting (and verifying) *all* datastreams of the DIDL document is successful. In this case, all datastreams are committed to an Internet Archive ARCfile (Burner & Kahle, 1999). Multiple ARCfiles may be created during the described sub-process, depending on the size of the collected datastreams. Furthermore, for each datastream, an entry is created in a log file – OK.csv – that summarizes successfully processed DIDL documents.
- Collecting (and verifying) *one or more* of the datastreams of the DIDL document is unsuccessful. Reasons include a failure to dereference a datastream that is provided By Reference, and a failure to verify the authenticity and integrity of the datastream based on the XML signature. In this case, no datastreams are written into an Internet Archive ARCfile. For each datastream for which verification is unsuccessful, an entry is created in the failure log – notOK.csv – indicating the DIDL document in which the failure occurred as well as the reason of the failure.

In both cases, the sub-process stops, and control is handed back to the process that parses the XMLtapes. The next DIDL document is handed over to the sub-process.

2.3.3. The verification of XML signatures

In the proposed data transfer technique, XML signatures are provided for both the DIDL document as a whole, and for all the constituent datastreams of the asset represented by the DIDL document. Only the successful validation of the XML signatures at both levels guarantees the faultless transfer of a packaged asset. The validation of an XML signature requires the verification of the digest value by repeating the digest calculation over the transferred data and by confirming the signature value by using the signer's public key. Below, the process and the interpretation of its possible outcomes are discussed for both types of XML signatures.

Checking the DIDL-level XML signatures. A DIDL-level XML signature is extracted from the '`about`' container of a harvested DIDL document and is processed to check its validity. The following scenarios may occur:

- If the validation of the XML signature completes successfully, then the consuming archive knows that the harvested DIDL document was indeed created by the producing archive and that it is identical to the DIDL document that was exposed by the producing archive.
- If the digest value checks invalid, the consuming archive knows that the DIDL document was corrupted during the harvesting process. The consuming archive has to re-harvest the faulty DIDL document. It can do so by issuing an OAI-PMH `GetRecord` request.
- If the signature value checks invalid, the consuming archive cannot verify that the harvested DIDL document was indeed exposed by the producing archive. Again, the consuming archive must try to re-harvest the DIDL document.

Checking the datastream-level XML signatures. A datastream-level XML signature is extracted from a `Descriptor/Statement` construct associated with the `Component` element that contains the datastream. As described in the above, dependent on the nature of the datastream and on the method by which it is provided (By Value or By Reference), different transforms may be required before the re-calculation of the digest value can occur. The following scenarios may occur:

- If the validation of the XML signature completes successfully, then the consuming archive knows that the collected datastream was indeed delivered by the producing archive and that it is identical to the datastream that is available at the producing archive.
- If the digest value checks invalid, one of the following problems has occurred:
  - An error occurred in the process of embedding the datastream inside the DIDL document (in case of By Value provision), in retrieving the datastream from its network location (in case of By Reference provision), or in content-encoding (compressing) of the original datastream prior to collecting. The consuming archive must re-collect the problematic datastream.
  - The datastream was updated in the timeframe between the OAI-PMH harvesting and the collecting of the datastream. Such an event must be reflected by a changed OAI-PMH *datestamp* of the DIDL document, and can be corrected by re-harvesting it.
  - The digest created by the producing archive does not match the datastream over which the digest has been calculated. This can occur when the digest was pre-computed and stored by the producing archive. This problem may indicate bit rot of the datastream stored in the producing archive. The consuming archive needs to inform the producing archive (APS) of the problem.
- If the signature value checks invalid, the consuming archive cannot verify that the collected datastream originates from the producing archive. Again, the consuming archive must re-collect the problematic datastream.

2.3.4. The pre-ingest area of the consuming archive

As a result of the processes described in Sections 2.3.1 to 2.3.2, the pre-ingest area of the consuming archive now contains a collection of harvested DIDL documents, datastreams associated with those DIDL documents, information on the authenticity and integrity of both the DIDL documents and the datastreams, and information on the successful or unsuccessful dereferencing of datastreams.

In the LANL implementation of the process, this translates to the availability of:

- One or more XMLtapes that contain DIDL documents that resulted from OAI-PMH harvesting from the producing archive.
- Per XMLtape, one or more Internet Archive ARCfiles that contain datastreams that resulted from successfully processing complete DIDL documents. In such an ARCfile, all datastreams will be available for those DIDL documents for which:
    - Verification of the DIDL-level XML signature was successful.
    - Collection of all datastreams of the DIDL document was successful, as was the verification of their associated XML signature.
- An OK.csv file listing information for all DIDL documents for which processing was successful. The log has an entry per datastream that was collected for a specific DIDL document; a sample entry is shown in Table 24.
- A notOK.csv file listing DIDL documents for which processing was unsuccessful. Table 25 shows a sample entry from such a log.

information elements in the OK.csv entry	**sample value**
OAI-PMH identifier of harvested DIDL document	`info:doi/10.1103/PhysRevB.69.174413`
XPath pointing to the XML node (of the harvested DIDL document) that was used for collecting a constituent datastream	`//didl:Component[0]/didl:Resource[0] /@ref`
URI from which the constituent datastream was collected	`http://oai.aps.org/filefetch? id=PhysRevB.69.174413& description=apsmeta`
Datetime of collecting the constituent data-stream	`2006-01-06T23:42:16Z`
Name of the ARCfile in which the constituent datastream is stored	`aps_test_2005_d236cd58--9d84-c6450d3ebb7a`
Identifier of the constituent datastream stored in the ARCfile	`info:lanl-repo/arc/881848de-9b6b-4401-924d-ad528877d34e`
Digest calculated over constituent datastream	`urn:sha1:tKbU9kVeJsnvh2JfMXIbwTecO+Y`

Table 24. OK.csv: Log information for all DIDL documents for which processing was successful.

information elements in the notOK.csv entry	sample value
OAI-PMH identifier of harvested DIDL document	`info:doi/10.1103/PhysRevB.69.174413`
XPath pointing to the XML node (of the harvested DIDL document) that was used for collecting a constituent datastream	`//didl:Component[2]/didl:Resource[0] / @ref`
URI from which the constituent datastream was collected	`http://oai.aps.org/filefetch? id=PhysRevB.69.174413& description=print`
Datetime of collecting the constituent datastream	`2006-01-06T23:56:09Z`
Error status	`Reference for URI has no XMLSignatureInput`

Table 25. notOK.csv: Log information for all DIDL documents for which processing was unsuccessful.

The notOK.csv file provides a starting point for undertaking actions regarding harvested DIDL documents that were processed unsuccessfully. It is expected that acting upon information in this file will remain a manual task until the moment enough knowledge has been acquired about the possible error scenarios; once such knowledge is available, certain follow-up actions could be automated. Based on the analysis provided in Section 2.3.3, it was decided that all actions aimed at correcting problems start by re-harvesting the DIDL document in which the error was detected, and by consecutively repeating the sub-process described in Section 2.3.2. As will be explained in Section 2.3.5, the OK.csv file is crucial for ingesting the obtained assets into the consuming archive.

2.3.5. Ingesting assets into the consuming archive

The pre-ingest area of the consuming archive now effectively contains all information required to (re)construct an asset from the producing archive; all datastreams and secondary data that were shared by the producing archive are available locally. This means that an application-neutral representation, based on the MPEG-21 DID Abstract Model, of an asset from the producing archive can be recreated in the pre-ingest area of the consuming archive. Using knowledge of both the MPEG-21 DID Abstract Model, the data model used by the consuming archive, and the structure of Archival Information Packages in the consuming archive, an ingestion process can be devised that processes this information and turns it into an OAIS AIP that can be stored by the consuming archive. It can be seen that conceptually this process is very similar to the map/package process that occurred at the producing archive when it exposed its assets as XML-based packages (see Figure 19).

In the LANL implementation, the OK.csv file (Table 24) unambiguously ties a DIDL document stored in an XMLtape to its constituent datastreams stored in one or more ARCfiles. As a result, the file allows for the (re)construction, at the end of the consuming archive, of an MPEG-21 DID-based representation of the asset that was exposed by the producing archive. In this representation, all constituent datastreams of an asset are provided

By-Reference, with the references being pointers into the ARCfiles stored in the pre-ingest area of the LANL aDORe repository.

As a result of the above described processes, an OAIS AIP exists in both the producing archive and the consuming archive. Both OAIS AIPs package the same asset. The actual packaging of the asset as an AIP in both archives can very well be different, because those packagings are based on the data models used by the repository architectures of the respective archives. It should be noted that, especially in the context of use of the described solution for preservation purposes, the following information should be available in the OAIS AIPs in the consuming archive:

- Content Information Identifier of the asset as provided by the producing archive: This is essential for all further communication about the asset between the producing archive and the consuming archive.
- Digests for all constituent datastreams of the asset: Storing digests is considered good practice in any repository environment, but it takes on a special meaning in a data transfer scenario. Indeed, when exchanging data, digests are provided by the producing archive to the consuming archive. If those digests check out successfully, it means that the data transfer was successful. But, when storing those same digests in the consuming archive, both archives end up with the same digest for the same datastream. This enables archives to compare notes by exchanging digests.
- OAI-PMH Provenance information (Lagoze et al., 2002b): The inclusion of OAI-PMH Provenance information enables the automatic re-harvest of an asset based on information contained in the OAIS AIP. Such re-harvesting may, for example, be required in case a deficient constituent datastream is detected when recurrently controlling the integrity of datastreams stored in the consuming archive.

  To this end, it is worth observing that using the Content Information Identifier directly as the OAI-PMH *identifier* may simplify the content transfer framework, especially if multiple nodes in a content transfer network are considered. First, different OAI-PMH nodes may employ different OAI-PMH *identifier* schemes. As a consequence, interchanging digital assets using the OAI-PMH requires keeping track of which nodes refer to which *resources* by means of which OAI-PMH *identifier*. Next, repositories may use interfaces different than the one proposed in this experiment. In order for a content transfer network to readily interchange information between such a diverse set of interfaces, the identifier should preferably be the same no matter which repository interface is being used.

2.4. Discussion

The actual size of the complete data collection of the American Physical Society is about 700 GB. For rather obvious reasons the APS/LANL content transfer technique does not intend to transmit this complete collection in the manner described. Rather, an initial batch of the APS collection covering all materials up to a specific moment in time is being delivered on tapes. All materials that are added to the collection beyond that moment in time are collected using the described approach. An estimation of the size of the dataset that will be collected by LANL is obtained by considering that, in 2004, the APS published 16,500 papers, corresponding with an archival dataset of approximately 44 GB. The APS expects a 5-10% annual increase in the amount of publications over the coming years. A quick calculation learns that, on a daily basis, around 120 MB of archival data will be collected by LANL and ingested into the aDORe repository. Such an amount fits well within the limits of the capabilities of the described solution and its underlying technologies. In the current implementation, all datastreams of an updated asset will be collected, irrespective of which actual datastreams were updated. Given that the amount of APS assets that were updated after initial publication was only around 500 in 2004, this approach does not seem to cause significant overhead for the given solution. However, scenarios can be imagined that would require optimization in this respect. An obvious optimization would build on the comparison of the digests of the datastreams of the harvested DIDL document that represents the updated asset (available in the DIDL document) with the digests of constituent datastreams of the previously stored version of the asset (stored in the consuming archive).

In the course of the experiment, a significant lesson has been learned regarding the manner in which to deliver datastreams in DIDL documents. Initial implementations made use of both the By Value and By Reference capabilities available in MPEG-21 DIDL. However, use of the By Value technique, by which binary datastreams are base64-encoded before they are embedded in a DIDL document, rapidly leads to memory problems at both the producing and consuming archives. This is, among other factors, because large XML documents must be constructed and processed, a task that is typically achieved by building an in-memory copy first. As a result, it was decided to implement an approach whereby data is only provided By Value up to a threshold size for a DIDL document that met the hardware restrictions at the APS end. Once that threshold was reached, all datastreams of a specific asset were delivered By Reference. Interestingly enough, setting a safe threshold turned out not to be the easiest of tasks, and consecutive iterations kept leading to memory problems. The memory problems also inspired an implementation whereby DIDL documents were streamed out of the APS OAI-PMH repository rather than being completely built before being transferred. In such an implementation, it becomes impossible to verify the validity of exposed XML documents. Moreover, that implementation is typically not supported by off-the-shelf OAI-PMH repository tools. Above all, none of the described approaches addressed possible restrictions at the end of the LANL consuming archive.

The problems discovered in all these implementation iterations led to the decision to deliver datastreams By Reference only. Such a solution can be deployed on the basis of off-the-shelf OAI-PMH tools, imposes hardware requirements on the OAI-PMH implementations of both the producing and consuming archives that are very similar to those imposed by a scenario in which – say – MARCXML is harvested, and, as a result, is generic. The By-Reference-only approach has a significant additional advantage in cases where (some) datastreams need to be delivered from near-line or off-line storage. Indeed, in such cases, the OAI-PMH harvesting process can be conducted without interruption, while the waiting times to obtain those datastreams can be postponed to the dereferencing sub-process described in Section 2.3.2, and can be throttled by a dedicated process at the producing archive end. The By Reference approach also has advantages from the perspective of access rights. Indeed, a DIDL document without embedded datastreams could be exposed to all downstream harvesters without limitations, while access to specific datastreams could be controlled by a dedicated interface to the producing archive.

For completeness, it should be mentioned that the By Reference approach may introduce certain complexities because it requires the provision of a dereferencable URI per individual datastream of the producing archive. The URI should be neutral to the actual storage layout of the producing archive as URIs are expected to be dereferencable into the indefinite future. It is envisioned that they will be used when cross-checking the integrity of the copy of the producing archive. Because of this, URIs have to be created that combine the Content Information Identifier with secondary data aimed at unambiguously typing individual datastreams of the identified asset. Note that the NISO OpenURL Framework (NISO, 2005) provides a perfect framework to convey such locations.

Another design choice that requires further attention is the use of the content-decoding transform indicating the use of compression techniques as described in Section 2.2.2.2. Such a transform specifies which content-decoding algorithms must be applied to the data before a digest can be calculated. In the APS/LANL experiment, to improve performance of the content transfer mechanism, large datastreams are compressed using the 'GNU Zip Compression' algorithm. MPEG-21 DIDL allows expressing both the MIME type of the original, uncompressed data and the compression that was applied to it. Hence, it is possible to transfer such datastreams unambiguously and to provide an XML signature for the compressed datastream, requiring no indication of a content-decoding transform at the level of the XML signature. However, such an approach does not enable the detection of problems that might occur during the process of compressing/decompressing the datastream. Such detection is only possible when the digest is computed over the original, uncompressed datastream. In the APS/LANL experiment, this has been achieved through the introduction of a transform that indicates the need to unzip the identified datastream before computing the digest. This transform is not supported by the W3C XML Signature and Processing specification, and hence requires additions to the XML Signature and Processing software, but it was felt that the introduction of this transform was important with regard to the digital preservation goal of the APS/LANL experiment.

2.5. Conclusion

Technologies that have emerged since the BIBLINK and NEDLIB projects reached their conclusions, provide capabilities that were previously not available to address the content transfer problem. When combined, those technologies facilitate devising a standards-based solution to the content transfer problem. The proposed solution, as designed and tested in the APS/LANL project, uses:

- An XML-based packaging format (MPEG-21 DIDL) that allows for the application-neutral representation of compound digital assets of all sorts,
- A pull-oriented HTTP-based protocol (OAI-PMH) that allows incremental harvesting of new and updated assets, represented as XML documents, from a producing archive,

- An XML-specific technique (W3C XML Signatures Syntax and Processing) to provide guarantees regarding authenticity and accuracy of the transferred assets if such guarantees are required by the content transfer framework.

Because the proposed solution is standards-based, it is largely deployable using off-the-shelf tools, and it is well-suited for cross-archive and cross-community content transfer. The proposed solution also has interesting characteristics that are not available in typical deployed solutions. The following characteristics that derive from the use of the OAI-PMH as a synchronization protocol are especially noteworthy:

- An asset-level synchronization capability, via the OAI-PMH *datestamp*,
- A built-in, unambiguous manifest of transferred assets, via the `ListRecords` response,
- A means for the consuming archive to record when it received an asset and from where it was received, via the OAI-PMH Provenance concept,
- A means for the consuming archive to recollect previously collected digital assets from the producing archive without the need for human interaction.

Clearly, the solution addresses only part of the content transfer problem, namely the recurrent and accurate transfer of content between a producing archive and a consuming archive. Content-level problems such as the processing of received content by the consuming archive to meet the requirements of a service remain unaddressed. A typical example is the normalization of descriptive metadata and/or content from a variety of origins to a single format suitable for use in a search engine. While such processing is typically compute-intensive, it is mainly the intellectual effort required to devise accurate cross-walks between formats that is forbidding. It can only be hoped that content producers will increasingly converge towards the use of a limited amount of XML-based formats. Meanwhile, it is felt that the proposed content transfer framework can result in a significant optimization of the process of exchanging content between nodes in our networked information environment.

Ch.3. Generalizing the aDORe and APS experiments

Ch.3. Generalizing the aDORe and APS experiments

In previous chapter, two experiments have been described. A first experiment introduces the aDORe repository environment at the Los Alamos National Laboratory; a second experiment explores the transfer of digital assets from the American Physical Society to the aDORe repository environment. Both experiments define read-only interfaces to digital repositories: Interfaces to harvest batches of compound digital assets (i.e., harvest interfaces) and interfaces to obtain individual disseminations of digital assets and their constituent datastreams (i.e., obtain interfaces). Between those repository interfaces, the following points of similarities can be observed.

First, digital assets exposed by a repository interface are serialized and packaged using an application-neutral XML-based packaging format.

- Digital assets exposed by the aDORe interfaces are represented and packaged in accordance with the MPEG-21 DID standard. Some interfaces also expose transforms of stored DIDL documents by handing off crosswalk requests to the aDORe Transform Engine.
- Digital assets exposed by the APS interface are dynamically mapped to the MPEG-21 Abstract Model. Following this mapping, an XML-based representation of the Digital Item is provided and embedded in an XML container, in a manner that again is compliant with MPEG-21 DID.

Second, the interfaces are protocol-based. In the experiments, two technologies are employed:

- The OAI-PMH is used to harvest XML-based packages of digital assets. For example, in the aDORe environment, a harvest interface is introduced per Autonomous OAI-PMH Repository. Also, a harvest interface – called the OAI-PMH Federator – is introduced that turns the complete environment into a single OAI-PMH repository, thereby hiding all aDORe's architectural details and complexities from downstream harvesters. In the APS environment, a public interface is introduced that enables the incremental OAI-PMH-based harvest of new and updated digital assets from the APS collection.
- The NISO OpenURL Framework is used to obtain disseminations of a digital asset and its constituent datastreams on an individual basis. In the aDORe environment, an OpenURL *Resolver* is introduced per Internet Archive ARCfile. The sole service provided by such an OpenURL *Resolver* is straightforward delivery of a datastream stored in the ARCfile. Also, an overlaying OpenURL-based interface is introduced to the complete aDORe environment – called the aDORe OpenURL *Resolver* – from which various disseminations of an individual digital asset or its constituent datastreams can be obtained.

Third, the interfaces are centered on one of two levels of identification:

- One level is directly related to the identification of digital assets (also known as OAIS Content Information), by using the identifiers of the digital assets (also known as OAIS Content Information Identifiers). For example, digital assets exposed by the OAI-PMH interface of the APS repository are represented as DIDL documents and exposed using their (native) content identifier; that is, the DOI identifiers of the APS articles. In FRBR

(IFLA, 1998), this type of identifier is aligned with the identifier of a Work, Expression or Manifestation (ProductType).

- Another level is directly related to the identification of the XML container (also known as OAIS AIP) that physically represents and packages the digital asset. In the OAIS Reference Model, this type of identifier is called OAIS AIP Identifier. For example, in the aDORe environment, the *identifiers* used by the Autonomous OAI-PMH Repositories and the OAI-PMH Federator are the identifiers of the stored DIDL documents. In FRBR, this type of identifier is an attribute of an Item.

Dependent on the protocol and identification-level being chosen, different harvest- and obtain interfaces, can be devised. Core properties of each of the repository interfaces will be described using concepts and terms of the OAIS Information Model. The focus hereby is on the handling of identifiers and versioning of content.

1. An OAIS perspective on interfaces for harvesting and obtaining digital assets

Four theoretical interfaces can be distinguished and grouped into two sets (See Table 20). A first set of interfaces (Interface No 1 and Interface No 2) uses OAIS AIP Identifiers. A second set of interfaces (Interface No 3 and Interface No 4) uses OAIS Content Information Identifiers.

	OAIS AIP Identifiers	**OAIS Content Information Identifiers**
harvest	Interface No 1	Interface No 3
obtain	Interface No 2	Interface No 4

Table 26. An OAIS perspective on repository interfaces

Interface No 1 and Interface No 3 are based upon the OAI-PMH and support the harvesting of batches of XML-based packages of digital assets. In OAIS terminology, those XML representations are referred to as OAIS Dissemination Information Packages (DIP).

Interface No 2 and Interface No 4 are based upon NISO's OpenURL Framework for Context-Sensitive Services. For each such interface, two levels of conformance are defined. A first level of conformance supports requesting XML-based representations of a digital asset on a one-by-one basis. A second level of conformance supports requesting datastream-level disseminations of a digital asset.

Interface No 1 and Interface No 2 are described in Section 1.1; Interface No 3 and Interface No 4 are explored in Section 1.2. A brief summary of results and a reflection on the properties of the repository interfaces is provided in Section 1.3.

1.1. Harvest- and obtain interfaces using OAIS AIP Identifiers

1.1.1. Interface No 1: Harvesting OAIS DIPs through OAI-PMH, using OAIS AIP Identifiers

A first interface (henceforth referred to as 'Interface No 1') allows for the harvest of (batches of) OAIS DIPs from a digital repository using the OAI-PMH protocol. The identifier of the OAIS AIP, from which an OAIS DIP is derived, serves as the OAI-PMH *identifier*. The response returned by the interface is an OAI-PMH *record*. Each *record* physically embeds the requested OAIS DIP as OAI-PMH *metadata*. Following the OAI-PMH specification, the OAIS DIP must be delivered in strictly valid XML. The specifics of this interface and the manner in which downstream harvesters interact with it are described below.

Interface No 1 has the following OAI-PMH characteristics:

- The base URL of the OAI-PMH interface is the HTTP address '`baseURL(OAIPMH_AIPID)`'.
- The OAI-PMH *identifiers* used by Interface No 1 are the OAIS AIP Identifiers of the OAIS AIPs stored in the repository.
- The OAI-PMH *datestamps* used by Interface No 1 are the datetime of creation of the OAIS AIPs. Note that, because the OAIS Reference Model requires the creation of a new AIP (instead of updating an existing OAIS AIP) for every update of its constituents, the OAI-PMH *datestamp* of a given OAIS AIP will never change once the AIP has been created.
- The natively supported *metadata format* is an XML-based OAIS DIP Packaging Format. Various XML-based Packaging formats – some of which have been standardized – can be employed. Examples include, the ISO MPEG-21 DIDL (ISO, 2005a), METS (METS) (LC, 2005a), IMS-CP (IMS Global Learning Consortium, 2003), and XFDU, a pre-standard developed by CCSDS Panel 2 (CCSDS, 2004b). It is important to observe that an OAIS DIP is derived from an OAIS AIP, and assuming that all OAIS AIPs stored in an OAIS share a common format, support of a given OAIS DIP Packaging Format can be regarded a repository-wide property.
- The supported time granularity of Interface No 1 can be day or seconds-level.
- *Set* structures may be supported for grouping OAIS AIPs for the purpose of selective harvesting.

The interaction of a downstream OAI-PMH harvester with the repository through the above OAI-PMH interface is illustrated in Figure 21 and explained below. Typically, the interaction is a two-step approach:

- First, the OAI-PMH harvester issues a `ListMetadataFormats` request against the repository interface. In response, the OAI-PMH harvester receives a list of supported XML-based OAIS DIP Packaging Formats. The request looks as follows:

```
[ baseURL(OAIPMH_AIPID)?
      verb=ListMetadataFormats ]
```

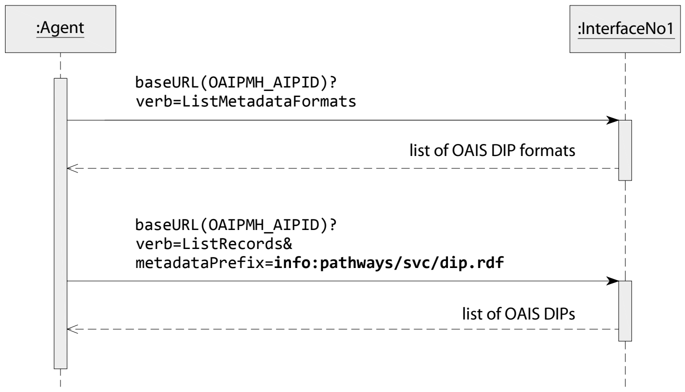


Figure 21. UML Sequence Diagram of Interface No 1

- Once a list of OAIS DIP Packaging Formats has been obtained, the OAI-PMH harvester can collect OAIS DIPs from the OAI-PMH interface of the repository by issuing an OAI-PMH `ListRecords` request, thereby specifying one of the OAIS DIP Packaging Formats retrieved from the repository by issuing the `ListMetadataFormats` request. Each OAIS DIP is embedded in an OAI-PMH *record* and is uniquely identified by the combination of an OAIS AIP Identifier of the OAIS AIP (from which the OAIS DIP will be derived) as the OAI-PMH *identifier*, and an OAIS DIP Packaging Format as the OAI-PMH *metadata format*. Note that in contrast with a typical OAI-PMH scenario the OAI-PMH *datestamp* is not essential as a third information element to uniquely identify an OAIS AIP. The rationale stems from the OAIS versioning strategy. Whenever an existing set of OAIS Content Information is updated, a new OAIS AIP (with a new OAIS AIP Identifier, and a new datetime of creation) is created. As such, within the context of a given OAIS Content Information Identifier, both the AIP Identifier and the OAIS AIP creation datetime can be employed to uniquely address an OAIS AIP.

  The format of the OAIS DIP must correspond with the one specified in the OAI-PMH `ListRecords` request; it is one of the OAIS DIP Packaging Formats retrieved from the repository by issuing the `ListMetadataFormats` request.

  An example of a `ListRecords` request is given below. The `ListRecords` request results in a list of OAIS DIPs, in which each OAIS DIP has been created within the bounds of the (optional) `from` and `until` arguments. '`info:pathways/dip.rdf`' identifies an application-neutral RDF-based OAIS DIP Packaging Format (developed in the context of the Pathways Project ("Pathways," n.d.)).

```
[ baseURL(OAIPMH_AIPID)?
      verb=ListRecords&
      from=T1&until=T2&
      metadataPrefix=info:pathways/svc/dip.rdf ]
```

1.1.2. Interface No 2: Obtaining OAIS DIPs and datastream-level disseminations through NISO's OpenURL, using OAIS AIP Identifiers

A second interface (henceforth referred to as Interface No 2) employs NISO's OpenURL Framework for Context-Sensitive Services. As described in Section 4 of Chapter 1, the NISO OpenURL Framework allows a community or application domain to build *OpenURL Framework Applications* by unambiguously defining the restrictions on, and representation and transportation techniques of OpenURL *ContextObjects* in a so-called *Community Profile*. The *Community Profile* underlying Interface No 2, initiates a delivery process by providing an interface through which disseminations can be requested on a one-by-one basis. Hereby, two levels of conformance are defined.

- □ Conformance Level 1 supports obtaining instances of a digital asset (represented and packaged using an XML-based package format). In contrast with interface No 1, this interface does not support batch harvesting of Information Packages, rather it allows obtaining Information Packages on a one-by-one basis. A detailed description is provided in Section 1.1.2.1; a sequence diagram of the interaction is shown in Figure 22.
- □ Conformance Level 2 supports obtaining datastream-level disseminations of a digital asset. The result is a MIME typed bit stream representing the datastream itself, or a transformation thereof. A detailed description is provided in Section 0; a sequence diagram of the interaction is given in Figure 23.

In what follows, *ContextObjects* are represented using the KEV *ContextObject Format* and are transported to an OpenURL *Resolver* using the HTTP(S) GET mode of the *By-Value OpenURL Transport*. It should be noted however that, because the OpenURL Framework is specified in a generic manner, the same interface can be implemented in many different ways as technologies evolve. For example, based on the concepts underlying Interface No 2, a *Community Profile* could be defined in which *ContextObjects* are represented by means of the XML *ContextObject Format* and transported using an XML-based protocol such as SOAP. In essence, the concepts underlying Interface No 2 persistent over time; their actual implementation can change over time.

1.1.2.1. Interface No 2, Level 1: Obtaining OAIS DIPs through NISO's OpenURL, using OAIS AIP Identifiers

The interface of an OpenURL *Resolver* compliant with Level 1 of the proposed *Community Profile* accepts two types of service requests. Both types of requests are expressed by means of a *ContextObject* that is transported towards the OpenURL *Resolver* at base URL '`baseURL(OPENURL_AIPID)`'.

- □ **The OAIS DIP bootstrap service request**. An OpenURL *Resolver* compliant with Level 1 supports the 'OAIS DIP bootstrap' request. This request is conveyed as a *ContextObject* with the following characteristics:
  - □ The *Referent* of the *ContextObject* is an OAIS AIP stored in the digital repository. The *Referent* is described by means of an *Identifier Descriptor*. Its value is the identifier of the OAIS AIP in question.

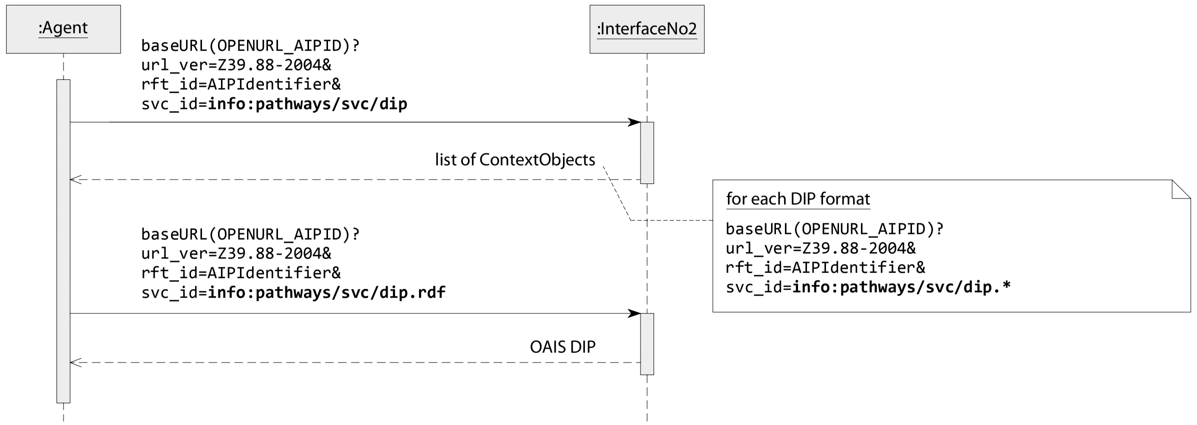


Figure 22. UML Sequence Diagram of Interface No 2, Level 1

- ◻ The *ServiceType* of the *ContextObject* is a service requesting a list of all OAIS DIP Packaging Formats that can be provided for the given *Referent*. The *ServiceType* is described by means of an *Identifier Descriptor* with the fixed value '`info:pathways/svc/dip`'.
- ◻ The *ContextObject* may contain *Entities* other than *Referent* and *ServiceType*. These *Entities* offer the potential for describing context-related properties, for example pertaining to the agent that issues the request. This allows tailoring the response to those properties.

The response to the above request is a list of requests (targeted at the OpenURL *Resolver*) for the dissemination of OAIS DIPs that can be provided for the *Referent* specified in the bootstrap request. The list contains one request per supported OAIS DIP Packaging Format, and each request is again compliant with the OpenURL Framework. In the proposed implementation, the list is expressed as an XHTML container of *ContextObjects*, in which each individual *ContextObject* is expressed in the KEV *ContextObject Format*; and is ready to be transported using the HTTP(S) GET mode of the *By-Value OpenURL Transport* towards the OpenURL *Resolver* at base URL '`baseURL(OPENURL_AIPID)`'. For each OAIS DIP Packaging Format available from the digital repository, a separate *ContextObject* is provided. Each such *ContextObject* has the following characteristics:

- ◻ The *Referent* of the *ContextObject* is the OAIS AIP for which the initial OAIS DIP bootstrap service has been requested, and is described by an *Identifier Descriptor* conveying the OAIS AIP Identifier.
- ◻ The *ServiceType* of the *ContextObject* conveys an OAIS DIP Format supported by the digital repository. The *ServiceType* is provided using an *Identifier Descriptor*. The value of this *Descriptor* is a property of the digital repository system hosting the OAIS AIPs. Though it should be noted that, in order for digital repositories to transfer content in an interoperable manner, a set of standardized OAIS DIP Packaging Formats will be required.
- ◻ Figure 22 shows the use of an *Identifier Descriptor* to convey a *ServiceType* of the form '`info:pathways/dip.*`', in which the asterix represents a placeholder for a specific OAIS DIP Packaging Format.
- ◻ Similar to the OAIS DIP bootstrap request, *Entities* other than *Referent* and *ServiceType* may be provided, and the OpenURL *Resolver* may use such information to provide context-sensitive responses.

◻ **A specific OAIS DIP requests supported by the digital repository**. Once the list of *ContextObjects* has been received by the agent, the agent may choose the *ContextObject* of interest and send it back as a service request to the OpenURL *Resolver*. Note that in some applications this extra step of interaction with the agent could be by-passed; for instance by allowing the OpenURL *Resolver* to decide on an OAIS DIP Packaging Format based on context-related information provided by the agent in the initial OAIS DIP bootstrap request. An OpenURL *Resolver* compliant with Level 1 supports the service requests that it listed in the XML container of *ContextObjects* in response to the initial OAIS DIP

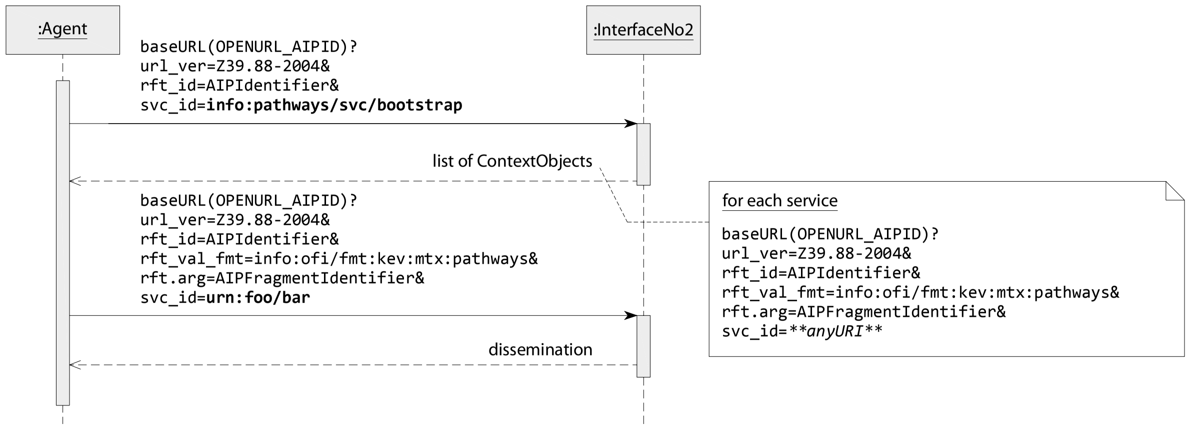


Figure 23. UML Sequence Diagram of Interface No 2, Level 2

bootstrap service request. Each such service requests results in the response of an OAIS DIP. The Packaging Format of the OAIS DIP is not defined by the *Community Profile*.

1.1.2.2. Interface No 2, Level 2: Obtaining datastream-level disseminations through NISO's OpenURL, using OAIS AIP Identifiers

The interface of an OpenURL *Resolver* compliant with Level 2 of the proposed *Community Profile* accepts two types of service requests. Similarly to Level 1, both types of requests are expressed by means of a *ContextObject* that is transported towards the OpenURL *Resolver* at base URL '`baseURL(OPENURL_AIPID)`'.

- □ **The datastream bootstrap service request.** The OpenURL *Resolver* of a digital repository compliant with Level 2 of this *OpenURL Framework Application* supports the 'datastream bootstrap' request. This request is conveyed as a *ContextObject* with the following characteristics:
  - □ The *Referent* of the *ContextObject* is an OAIS AIP stored in the digital repository. The *Referent* is described by means of an *Identifier Descriptor*. Its value is the OAIS AIP Identifier.
  - □ The *ServiceType* of the *ContextObject* is a service requesting a list of all dissemination services that can be provided for datastreams constituting the *Referent*. The *Service-Type* is described by means of an *Identifier Descriptor* with the value '`info:pathways/svc/bootstrap`'.
  - □ The *ContextObject* may contain *Entities* other than *Referent* and *ServiceType*, such as the *Requester Entity*.

  The response to a request with the datastream bootstrap *ServiceType* is a list of all dissemination services that can be provided for the datastreams or constituents packaged by the referenced OAIS AIP. This list is expressed as an XHTML container of *Context-Objects* in which each individual *ContextObject* details a specific datastream-level service request. For each dissemination service, a new *ContextObject* is provided. Each such *ContextObject* has the following characteristics:
  - □ The *Referent* of the *ContextObject* is a constituent datastream of the OAIS AIP for which the initial datastream bootstrap service has been requested. The *Referent* is described by the combination of two *Descriptors*: 1) an *Identifier Descriptor* conveying the OAIS AIP Identifier of the OAIS AIP for which the initial datastream bootstrap service was requested and 2) a *By-Value Metadata Descriptor* that carries an OAIS Fragment Identifier pertaining to that OAIS AIP. This Fragment Identifier is conveyed using the `arg` key and it identifies a datastream packaged by the OAIS AIP. The syntax of this key is unambiguously defined by the *Metadata Format* with identifier '`info:ofi/fmt:kev:mtx:pathways`'.
  - □ The *ServiceType* of the *ContextObject* conveys an available datastream-level service supported by the digital repository. The *ServiceType* should be conveyed using an *Identifier Descriptor* (see 'anyURI' in

  - Figure 23) and optional *By-Value* and *By-Reference Metadata Descriptors* could be used to convey arguments for the identified service. The values of all these *Descriptors* are a property of the digital repository and need not necessarily be interoperable.
  - The *ContextObject* may contain *Entities* other than *Referent* and *ServiceType*. Again, these *Entities* offer the potential for expressing context-related information. This allows tailoring the response to those properties.
- **A specific datastream-level service requests supported by the digital repository.** Once the list of *ContextObjects* has been received by the agent, the agent may choose the service request of interest and send it back to the OpenURL *Resolver*. The OpenURL *Resolver* of a digital repository compliant with Level 2 of the proposed *Community Profile* supports the service requests that it listed in the container of *ContextObjects* in response to the initial datastream bootstrap service request. Each such service requests results in the response of a dissemination of a datastream packaged by the referenced OAIS AIP; the result is returned as a MIME-typed stream.

1.2. Harvest- and obtain interfaces using OAIS Content Information Identifiers

1.2.1. Interface No 3: Harvesting OAIS DIPs through OAI-PMH, using OAIS Content Information Identifiers

A third interface (henceforth referred to as Interface No 3) takes an approach similar as Interface No 1. Again, OAIS DIPs are gathered using OAI-PMH harvesting. However, in contrast with Interface No 1, the identifiers used by Interface No 3 are not the identifiers of the AIPs stored in the repository. Rather, Interface No 3 equates an OAI-PMH *identifier* with the concept of an identifier of the OAIS Content Information packaged by an OAIS AIP. Again, following the OAI-PMH specification, the response returned by the interface is an XML-based OAI-PMH *record*. Each *record* physically embeds the requested OAIS DIP as OAI-PMH *metadata*. The specifics of Interface No 3 and the manner in which downstream harvesters interact with it are described below.

Interface No 3 has the following OAI-PMH characteristics:

- The base URL of the OAI-PMH interface is the HTTP address '`baseURL(OAIPMH_CIID)`'.
- The OAI-PMH *identifiers* used by Interface No 3 are the OAIS Content Information Identifiers of the OAIS Content Information packaged by the OAIS AIPs stored in the repository.
- The OAI-PMH *datestamp* used by the OAI-PMH interface is the datetime of creation of the OAIS AIP(s) that package(s) the Content Information with the same Content Information Identifier. Again remember, because the OAIS Reference Model requires the creation of a new OAIS AIP for every new version of OAIS Content Information, many AIPs may exist, each with its own creation datetime. Because of the nature of the OAI-PMH, a request for an OAIS DIP containing OAIS Content Information with a specific OAIS Content Information Identifier will result in an OAIS DIP derived from the most recent OAIS AIP that packages the identified OAIS Content Information; that is from the most recent version of the OAIS Content Information.

- Similarly to Interface No 1, the natively supported *metadata format* is an XML-based OAIS DIP Packaging Format. Again, several OAIS DIP Packaging Formats could be supported by Interface No 3.
- The supported time granularity of Interface No 3 can be day or seconds-level.
- *Set* structures may be supported for grouping OAIS AIPs for the purpose of selective harvesting.

The interaction of a downstream OAI-PMH harvester with the repository through the OAI-PMH interface is illustrated in Figure 24 and explained below. Similarly to Interface No 1, the interaction is a two-step process.

- First, the OAI-PMH harvester issues a `ListMetadataFormats` request against the interface. In response, the OAI-PMH harvester receives a list of supported XML-based OAIS DIP Packaging Formats. The request looks as follows:

```
[ baseURL(OAIPMH_CIID)?
      verb=ListMetadataFormats ]
```

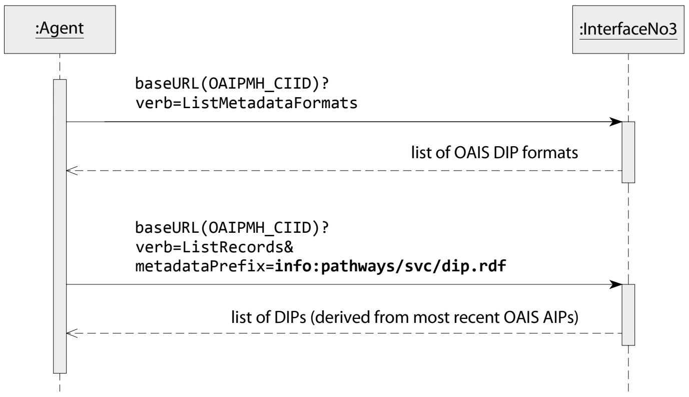


Figure 24. UML Sequence Diagram of Interface No 3

- Once a list of OAIS DIP Packaging Formats has been obtained, OAI-PMH harvesters can collect OAIS DIPs from the OAI-PMH interface of the repository by issuing an OAI-PMH `ListRecords` request, thereby specifying one of the OAIS DIP Packaging Formats retrieved from the repository by issuing the `ListMetadataFormats` request. Each OAIS DIP is embedded in an OAI-PMH *record* and is uniquely identified by the combination of an OAIS Content Information Identifier of the OAIS Content Information as the OAI-PMH *identifier*, an OAIS DIP Format as the OAI-PMH *metadata format*, and the datetime at which the OAIS AIP containing the OAIS Content Information (and from which the OAIS DIP will be derived) has been created.

  The format of the OAIS DIP must correspond with the one specified in the OAI-PMH `ListRecords` request; it is one of the OAIS DIP Packaging Formats retrieved from the repository by issuing the `ListMetadataFormats` request.

An example of a `ListRecords` request is shown below. A `ListRecords` request results in a list of OAIS DIPs, in which each OAIS DIP is derived from an OAIS AIP that has been created within the bounds of the (optional) `from` and `until` arguments. If for specific OAIS Content Information multiple OAIS AIPs have been created within the bounds of the (optional) `from` and `until` arguments, only the most recent version is provided. '`info:pathways/ dip.rdf`' identifies an RDF-based OAIS DIP Packaging Format.

```
[ baseURL(OAIPMH_CIID)?
      verb=ListRecords&
      from=T1&until=T2&
      metadataPrefix=info:pathways/svc/dip.rdf ]
```

For reasons of completeness, it should be noted that the proposed interface equates the *identifier* of the OAI-PMH *item* with the *identifier* of the OAI-PMH *resource*. While the *item* and *resource* are two different entities in the OAI-PMH data model, the OAI-PMH does neither specify the nature of the OAI-PMH *resource* nor of its identifier. Indeed, the nature of the *resource* and its identifier are outside the scope of the OAI-PMH specification and hence, may vary dependent on the application domain. In the context of this application domain, the identifier of the OAI-PMH *item* is set to match the OAI-PMH *resource*.

The motivation for this design choice is threefold.

□ In the proposed application domain, the *records* exposed by the OAI-PMH repository are more than just descriptive metadata about an OAI-PMH *resource*. They are, in fact, XML serializations of an OAI-PMH *resource*. These XML serializations convey a variety of secondary information pertaining to the *resource*, including descriptive, rights, technical, structural, and provenance information. But, they also unambiguously convey *resource* identifiers, and include the datastreams pertaining to the *resource* itself, either By Reference or By Value.

□ The OAIS DIPs exposed by OAI-PMH repositories will, eventually, show up in downstream applications. Those applications use those Information Packages and their constituent datastreams in the creation of services. For example, a search service may extract textual information from harvested Information Packages and make it available for searching by consumers. In this case, the search results returned to a user will point back to the corresponding digital asset stored in the digital repository, using the identifier of that digital asset. Because of this, it is important that the identifiers returned by the search service – that is, the identifier of the assets that match the query request of the consumer – and the identifiers resolved by the digital repository that holds the digital assets – through an OAI-PMH interface – are the same. This need is even more pressing, in case of a digital repository supporting multiple interfaces, each of which is centered around a different interface protocol. For example, one such interface could be based on the OAI-PMH, another on NISO's OpenURL Framework, and a third on the SRU/W search protocol. It goes without saying that – given a digital asset – the identifier should preferably be the same no matter which repository interface is being used.

□ As can be observed in the APS experiment, the use of an OAIS Content Information Identifier directly as the OAI-PMH *identifier* may simplify the content transfer framework, especially if multiple nodes in a content transfer network are considered. Indeed, different OAI-PMH nodes might employ different (OAI-PMH) *identifier* schemes for the same resources, resulting in a Tower of Bable of identifiers that all refer to the same *resource*. This creates the need to do complex bookkeeping to track which nodes refer to which *resources* by means of which OAI-PMH *identifier*. Putting the identifier of the digital asset on par with OAI-PMH *identifier* avoids this overhead.

1.2.2. Interface No 4: Obtaining OAIS DIPs and datastream-level disseminations through NISO's OpenURL, using OAIS Content Information Identifiers

A fourth interface (henceforth referred to as Interface No 4) employs NISO's OpenURL Framework for Context-Sensitive Services. Similar to Interface No 2, the *Community Profile* underlying this interface initiates a delivery process by providing an interface through which disseminations of a digital asset can be requested on a one-by-one basis. However, as opposed to Interface No 2, the identifiers resolved by Interface No 4 are not the identifiers of the OAIS AIPs stored in the digital repository, but the identifiers of the OAIS Content Information packaged by those AIPs. Again, a distinction is made between two levels of conformance. Both levels can be implemented independently.

- Conformance Level 1 supports obtaining instances of a digital asset (represented and packaged using an XML-based package format) on a one-by-one basis. A detailed description is provided in Section 1.2.2.1 and illustrated in Figure 25.
- Conformance Level 2 supports obtaining datastream-level disseminations of a digital asset. A detailed description is provided in Section 1.2.2.2 and depicted in Figure 26.

ContextObjects of the proposed interface are represented using the KEV *ContextObject Format* and are transported to an OpenURL *Resolver* using the HTTP(S) GET mode of the *By-Value OpenURL Transport*. Again, it should be noted that the OpenURL Framework allows for the same interface to be implemented in many different ways as technologies evolve.

1.2.2.1. Interface No 4, Level 1: Obtaining OAIS DIPs through NISO's OpenURL, using OAIS Content Information Identifiers

The interface of an OpenURL *Resolver* compliant with Level 1 of the proposed *Community Profile* accepts two types of service requests. Both types of requests are expressed by means of a *ContextObject* that is transported towards the OpenURL *Resolver* at base URL '`baseURL(OPENURL_CIID)`'.

- **The OAIS DIP bootstrap service request.** The OpenURL *Resolver* of a digital repository compliant with Level 1 of this *Community profile* supports the 'OAIS DIP bootstrap' request. This request is conveyed as a *ContextObject* with the following characteristics:
    - The *Referent* of the *ContextObject* is OAIS Content Information stored (as an OAIS AIP) in the repository. The *Referent* is described by means of an *Identifier Descriptor*. Its value is the OAIS Content Information Identifier of the OAIS Content Information in question.
    - The *ServiceType* of the *ContextObject* is a service requesting a list of all OAIS DIP formats that can be provided for the *Referent*. The *ServiceType* is described by means of an *Identifier Descriptor* with the fixed value '`info:pathways/svc/dip`'.

- The *ContextObject* may contain *Entities* other than *Referent* and *ServiceType*. These *Entities* offer the potential for describing, for example, properties of the agent that issues the request. Again, these properties allow tailoring the response to those properties.

Given an OAIS Content Information Identifier, multiple OAIS AIPs may exist that package content with that identifier. The first task of this *OpenURL Framework Application*, in response to the initial 'OAIS DIP bootstrap' request, is to disambiguate between the various OAIS AIPs available from the digital repository. Therefore, a separate process is started in which the *OpenURL Framework Application* generates a list of all OAIS AIPs that can be provided for a given OAIS Content Information Identifier, and presents the list to the agent. This list is expressed as an XHTML container of *ContextObjects*, in which each individual *ContextObject* is expressed in the KEV *ContextObject Format*; and is ready to be transported using the HTTP(S) GET mode of the *By-Value OpenURL Transport* towards the OpenURL *Resolver* at base URL '`baseURL(OPENURL_CIID)`. Each such *ContextObject* has the following characteristics:

- The *Referent* of the *ContextObject* is an OAIS AIP containing the OAIS Content Information for which the initial OAIS DIP bootstrap service has been requested. The *Referent* is described by the combination of an *Identifier Descriptor* and a *By-Value Metadata Descriptor*. The value of the former is the OAIS Content Information Identifier as conveyed by the initial OAIS DIP bootstrap service request. The latter conveys the OAIS AIP Identifier of the *Referent*. The *Metadata Format* used to described the *Referent* is identified by the KEV pair '`rft_val_fmt=info:ofi/fmt:kev:mtx:pathways`'. The `aip` key from the identified *By-Value Metadata Format* expresses the OAIS AIP Identifier. The syntax of the OAIS AIP Identifier itself is a property of the digital repository.
- Other *Entities* of the *ContextObject* are copied from the initial OAIS DIP bootstrap request.

Once the list of *ContextObjects* has been received by the agent, the agent may choose one and send it back to the OpenURL *Resolver* at base URL '`baseURL(OPENURL_CIID)`'. It should be noted that in some applications this extra step of interaction with the agent could be bypassed; for instance by allowing the OpenURL *Resolver* to pick a *ContextObject* based on context-related information provided by the agent in the initial OAIS DIP bootstrap request.

The response to this request, is a generated list of all OAIS DIPs that can be provided for the *Referent*. Again, this list is expressed as an XHTML container of *ContextObjects*. Each individual *ContextObject* details a specific OAIS DIP request. For each OAIS DIP Packaging Format available from the digital repository, a separate *ContextObject* is provided. Each such *ContextObject* has the following characteristics:

- The *Referent* of the *ContextObject* is the OAIS AIP that has been selected by the agent. Again, the *Referent* of the *ContextObject* is described using the combination of an *Identifier Descriptor* and a *By-Value Metadata Descriptor*. The latter conveys the OAIS AIP Identifier of the OAIS AIP being requested by means of the `aip` key. The former

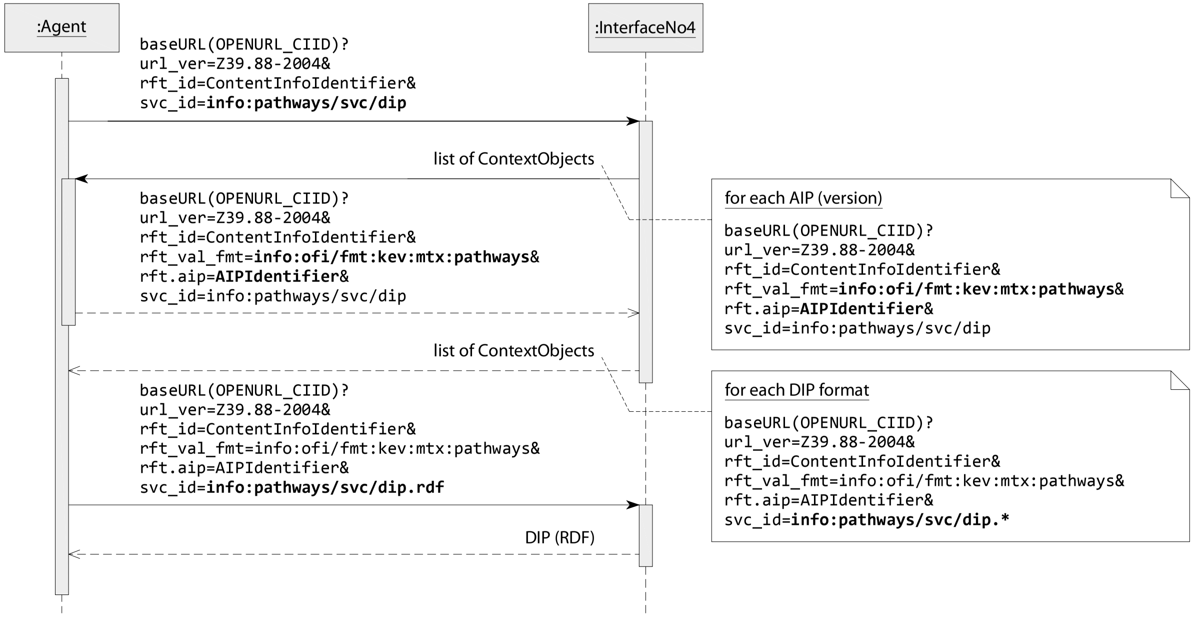

:Agent
:InterfaceNo4
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
svc_id=**info:pathways/svc/dip**
list of ContextObjects
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=**info:ofi/fmt:kev:mtx:pathways**&
rft.aip=**AIPIdentifier**&
svc_id=info:pathways/svc/dip
list of ContextObjects
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.aip=AIPIdentifier&
svc_id=**info:pathways/svc/dip.rdf**
DIP (RDF)
for each AIP (version)
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=**info:ofi/fmt:kev:mtx:pathways**&
rft.aip=**AIPIdentifier**&
svc_id=info:pathways/svc/dip
for each DIP format
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.aip=AIPIdentifier&
svc_id=**info:pathways/svc/dip.***

Figure 25. UML Sequence Diagram of Interface No 4, Level 1

expresses the OAIS Content Information Identifier of the OAIS Content Information packaged by that OAIS AIP and is the OAIS Content Information for which the initial OAIS DIP bootstrap service has been requested.

  - The *ServiceType* of the *ContextObject* conveys an available OAIS DIP Packaging Format supported by the digital repository. The *ServiceType* is provided using an *Identifier Descriptor*. The value of this *Descriptor* is a property of the digital repository hosting the OAIS AIPs. Though it should be noted that, in order for digital repositories to transfer content in an interoperable manner, a set of standardized OAIS DIP Packaging Format – each of which is identified using an *Identifier Descriptor* – will be required.
  - Figure 25 shows the use of an *Identifier Descriptor* to convey a *ServiceType* of the form '`info:pathways/dip.*`' in which the asterix represents a placeholder for a specific OAIS DIP Packaging Format.
  - Similar to the OAIS DIP bootstrap request, *Entities* other than *Referent* and *ServiceType* may be provided, and the OpenURL *Resolver* may use such information to provide context-sensitive responses.

- **A specific OAIS DIP requests supported by the digital repository.** Once the list of *ContextObjects* has been received by the agent, the agent may choose the *ContextObject* of interest and send it back as a service request to the *OpenURL Resolver*. The *OpenURL Resolver* of a digital repository compliant with Level 1 of the proposed *Community Profile* supports the service requests that it listed in the container of *ContextObjects* in response to the initial OAIS DIP bootstrap service request. Each such service requests results in the response of an OAIS DIP. The Format of the OAIS DIP is not defined by this *Community Profile*.

1.2.2.2. Interface No 4, Level 2: Obtaining datastream-level disseminations through NISO's OpenURL, using OAIS Content Information Identifiers

The interface of an OpenURL *Resolver* compliant with Level 2 accepts two types of service requests. Similarly to Level 1, both types of requests are expressed by means of a *ContextObject* that is transported towards the OpenURL *Resolver* at base URL '`baseURL(OPENURL_CIID)`'.

- **The datastream bootstrap service request.** The OpenURL *Resolver* of a digital repository compliant with Level 2 of this *Community Profile* supports the 'datastream bootstrap' request. This request is conveyed as a *ContextObject* with the following characteristics:
  - Similarly to Level 1 of this *OpenURL Framework Application*, the *Referent* of the *ContextObject* is a set of OAIS Content Information stored (as an OAIS AIP) in the digital repository. The *Referent* is described by means of an *Identifier Descriptor*. Its value is the OAIS Content Information Identifier.
  - The *ServiceType* of the *ContextObject* is a service requesting a list of all dissemination services that can be provided for datastreams constituting the *Referent*. The *ServiceType* is described by means of an *Identifier Descriptor* with the value '`info:pathways/svc/bootstrap`'.

- The *ContextObject* may contain *Entities* other than *Referent* and *ServiceType*, including the *Requester Entity*.

Again, multiple versions of the OAIS Content Information may exist. In accordance with the OAIS Reference Model, for each such version a new OAIS AIP is created. Therefore, similarly to Level 1 of this *OpenURL Framework Application*, a separate process is started in which the *OpenURL Framework Application* generates a list of all OAIS AIPs that can be provided for that given OAIS Content Information Identifier, and presents the list to the agent. The list is expressed as an XML container of *ContextObjects*. Each such *ContextObject* has the following characteristics:

- The *Referent* of the *ContextObject* is an OAIS AIP containing the set of OAIS Content Information for which the initial datastream bootstrap service has been requested. The *Referent* is described by the combination of an *Identifier Descriptor* and a *By-Value Metadata Descriptor*. The value of the former is the OAIS Content Information Identifier as has been conveyed by the initial datastream bootstrap service. The latter uses the `aip` key to convey the OAIS AIP Identifier of the *Referent*. Again, the *Metadata Format* used to described the *Referent* is identified by the KEV pair '`rft_val_fmt=info:ofi/fmt:kev:mtx:pathways`'.
- Other *Entities* of the *ContextObject* are copied from the initial datastream bootstrap request.

Once the list of *ContextObjects* has been received by the agent (or otherwise), the agent may choose the OAIS AIP of interest and send the corresponding *ContextObject* back to the OpenURL *Resolver* at base URL '`baseURL(OPENURL_CIID)`'.

The response to this request, is a list of all dissemination services that can be provided for the (constituents of the) selected OAIS AIP. This list is expressed as an XML container of *ContextObjects* in which each individual *ContextObject* details a specific datastream-level service request. The XML syntax of the container is again conformant with the official XML Schema for the XML *ContextObject Format*. For each Dissemination service available, a new *ContextObject* is provided. Each such *ContextObject* has the following characteristics:

- The *Referent* of the *ContextObject* is a (set of) datastream(s) of the OAIS AIP for which the initial datastream bootstrap service has been requested. The *Referent* is described by the combination of two *Descriptors*: 1) an *Identifier Descriptor* conveying the OAIS Content Information Identifier for which the initial datastream bootstrap service was requested 2) a *By-Value Metadata Descriptor* consisting of 2 keys. A first key (`aip`) conveys the OAIS AIP Identifier of the OAIS AIP that has been selected by the agent in the aforementioned process. A second key (`arg`) carries a Fragment Identifier identifying a datastream packaged by the OAIS AIP. The syntax of both keys is unambiguously defined by the *Metadata Format* with identifier '`info:ofi/fmt:kev:mtx:pathways`'.

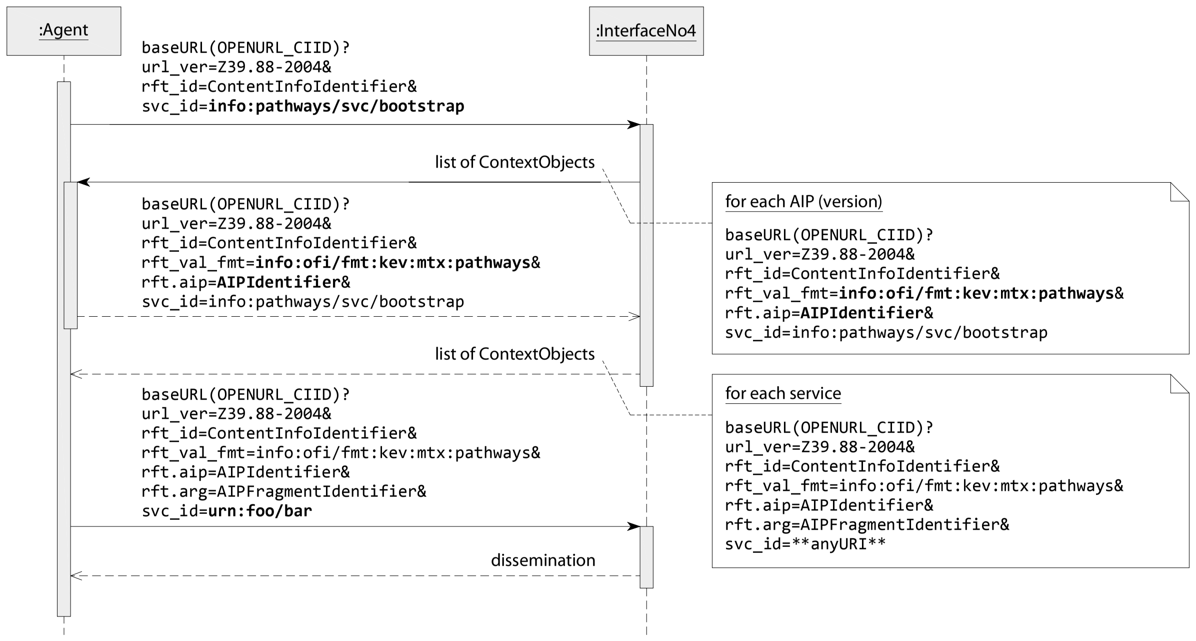

:Agent
:InterfaceNo4
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
svc_id=info:pathways/svc/bootstrap
list of ContextObjects
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.aip=AIPIdentifier&
svc_id=info:pathways/svc/bootstrap
list of ContextObjects
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.aip=AIPIdentifier&
rft.arg=AIPFragmentIdentifier&
svc_id=urn:foo/bar
dissemination
for each AIP (version)
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.aip=AIPIdentifier&
svc_id=info:pathways/svc/bootstrap
for each service
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.aip=AIPIdentifier&
rft.arg=AIPFragmentIdentifier&
svc_id=**anyURI**

Figure 26. UML Sequence Diagram of Interface No 4, Level 2

    - The *ServiceType* of the *ContextObject* conveys a datastream-level service supported by the repository. The *ServiceType* should be conveyed using an *Identifier Descriptor* and optional *By-Value* and *By-Reference Metadata Descriptors* could be used to convey arguments for the identified service. The values of all these *Descriptors* are a property of the repository and need not necessarily be interoperable.
    - The *ContextObject* may contain *Entities* other than *Referent* and *ServiceType*. Again, these *Entities* offer the potential for expressing context-related information allowing for the request of context-sensitive service requests.
- **A specific datastream-level service requests supported by the digital repository.** Once the list of *ContextObjects* has been received by the agent, the agent may choose the *ContextObject* that describes the service request of interest and send it back to the OpenURL *Resolver*. The OpenURL *Resolver* of a digital repository compliant with Level 2 of the proposed *Community profile* supports the service requests that it listed in the container of *ContextObjects* in response to the initial datastream bootstrap service request. Each such service requests results in the response of a dissemination of a datastream packaged by the referenced OAIS AIP; the result is returned as a MIME-typed stream.

1.3. Conclusion

So far, four generic read-only interfaces to digital repositories were introduced. The core characteristics of each of those interfaces are listed in Table 27 and summarized below.

interface	**interaction**	**standards**	**OAIS identifier**
Interface No 1	harvest	OAIS, OAI-PMH, …	AIP Identifier
Interface No 2	obtain	OAIS, NISO OpenURL, …	AIP Identifier
Interface No 3	harvest	OAIS, OAI-PMH, …	Content Info Identifier
Interface No 4	obtain	OAIS, NISO OpenURL, …	Content Info Identifier

interface	**exposure**	**version support**	**items in response**
Interface No 1	internal	all versions	1 or more
Interface No 2	internal	all versions	1
Interface No 3	public	most recent	1 or more
Interface No 4	public	all versions	1

Table 27. Repository Interfaces and their properties

Core properties of Interface No 1:

- Interface No 1 enables downstream applications to harvest batches of XML-based packages of digital assets using the OAI-PMH. To this end, Interface No 1 uses OAIS AIP Identifiers.
- Interface No 1 enables the harvesting of all versions of content available in a digital repository. This characteristic is a direct result of the OAIS versioning strategy to create a

new OAIS AIP, whenever a new version of content becomes available. Also, by using a *datestamp*-based harvesting strategy, newly added and updated versions can be harvested using a `ListRecords` request, by setting the value of the optional `from` parameter to the datetime of the last harvest that was conducted.

- Because of its use of OAIS AIP Identifiers, Interface No 1 is preferably used within the boundaries of a repository environment because OAIS AIP Identifiers are inherent to the way in which a repository internally manages its information, and are typically not exposed to the public. Hence, this interface might, for example, be used to create a complete (i.e., all versions) off-site mirror of an archive that remains under the custody of the repository itself.
- Interface No 1 proves to be particularly effective for the interchange of data collections to facilities that mirror the content to guarantee safety copies, or backups. By issuing *datestamp*-based harvesting requests, both the digital repository and its mirror can remain tightly in sync.
- Interface No 1 has been tested in the aDORe experiment. aDORe's Autonomous OAI-PMH Repositories and the aDORe OAI-PMH Federator expose DIDL documents by means of their DIDL document Identifier, allowing internal applications to collect all versions of stored digital assets.

Core properties of Interface No 3:

- Similar to Interface No 1, Interface No 3 enables downstream applications to harvest batches XML-based packages of digital assets using the OAI-PMH. In contrast with Interface No 1, the identifiers used by Interface No 3 are not the identifiers of the OAIS AIPs, rather the identifiers of the OAIS Content Information packaged by the OAIS AIPs.
- Interface No 3 enables harvesting the most recent versions of content available in a repository. This 'most recent' property stems from combining OAIS Content Information Identifiers with the OAI-PMH framework. The latter restricts harvesting to the most recent *datestamp* of an identified metadata record. Harvested content can be kept up to date, using a *datestamp*-based selective harvesting strategy.
- Interface No 3 exposes digital content using their natively attached OAIS Content Information Identifiers. Since, OAIS Content Information Identifiers can be shared beyond the repository, this interface qualifies for public use.
- Interface No 3 proves to be particularly effective for the interchange of assets from digital repositories to applications that provide value services (such as searching, browsing, and indexing) on the collected data. As shown in the APS experiment, this type of interface can also be used to mirror content, insofar as 1) the repository hosting the interface does not retain multiple versions (or packages) of the same content or 2) the mirroring process is considered adequate if only 'a few' versions (instead of all versions) of the content are mirrored. Even if a time granularity with a precision of seconds is supported, versions can be missed out.

- Interface No 3 has been tested in the APS experiment. In that experiment, an OAI-PMH transfer technique is introduced that allows for the incremental harvest of new and updated digital assets from the APS collection. Each digital asset is packaged in a DIDL document and resolved using an OAI-PMH *identifier* that is derived from the (native) content identifier of the digital asset (i.e., the APS article). As new versions of content trickle in slowly into the APS repository, and because the LANL OAI-PMH harvester polls the APS repository on a three to four hour basis, the chance of missing out versions is minimal (and in fact admissible in the setting of this application).

Core properties of Interface No 2 and Interface No 4:

- Interfaces No 2 and No 4 are based upon NISO's OpenURL Framework and allow to obtain disseminations of digital assets and constituent datastreams on a one-by-one basis. For this purpose, Interface No 2 uses OAIS AIP Identifiers; Interface No 4 uses OAIS Content Information Identifiers.
- Both interfaces support obtaining specific versions of content: Interface No 2 by directly using OAIS AIP Identifiers: Interface No 4 by returning a list of possible versions per given OAIS Content Information Identifier.
- Because of the types of identifiers they use, Interface No 3 is preferably used in intra-repository environments, while Interface No 4 lends itself for public exposure.
- Both interfaces prove to be particularly effective for requesting services pertaining to the identified data. The NISO OpenURL specification is purposely very generic and extensible, and also supports conveying context-sensitive information. For example, conveying *Requester* information may be of particular interest, as this would allow adapting the actual dissemination to the agent requesting it. *Requester* information could convey identity, and this would allow responding differently to the same service request depending on whether the requesting agent is a human or machine. The *Requester* entity could also convey information about the consumer's location or terminal.
- Both interfaces have been explored and tested in the aDORe experiment. The aDORe OpenURL *Resolver* provides an interface through which DIDL documents and data-stream-level disseminations can be obtained. This interface uses both Digital Item Identifiers and DIDL document Identifiers, including the XML ID Fragment Identifiers. The use of NISO's OpenURL Framework has also been explored to facilitate the delivery of individual datastreams stored in Internet ARCfiles.

2. A practical perspective on interfaces for harvesting and obtaining digital assets

In the preceding pages, sustained attention has been given to the description of theoretical interfaces for harvesting and obtaining assets from digital repositories. In what follows, I will explore the practicality of these interfaces by investigating how the concepts underlying those interfaces can be implemented in the aDORe repository environment and such institutional repository systems as DSpace and Fedora.

2.1. The representation, identification and versioning of digital assets in actual repository systems

This section explores how aDORe, DSpace and Fedora deal with the representation, versioning and identification of digital assets. The data models that underlie each of these repositories will be mapped to concepts of the OAIS Information Model. Next, based on this mapping the practicality of the aforementioned interfaces in these actual repository systems will be discussed.

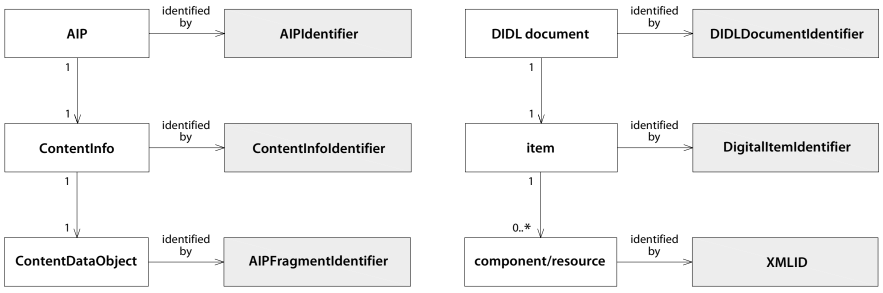


Figure 27. The representation, identification and versioning of aDORe Digital Items

- *aDORe* (See also Section 1 of Chapter 2): Compound digital assets stored in the aDORe environment are represented according to the MPEG-21 DID Abstract Model and serialized using the MPEG-21 DIDL syntax. In MPEG-21, the digital asset itself is called a Digital Item; the XML document that packages and serializes the Digital Item is referred to as a DIDL document. The aDORe Digital Items map to OAIS Content Information; the DIDL XML documents that package the Digital Items map to OAIS Archival Information Packages. The MPEG-21 DIDL syntax is the Packaging Information.

  aDORe recognizes two parallel identification mechanisms. The first identification mechanism pertains to Digital Items. This type of identifier is typically associated with a digital asset at the moment of its creation by a publisher, and, hence already exists when the asset is ingested into aDORe. If it does not yet exist, it is created upon ingestion. Clearly, this type of identifier maps to the concept of OAIS Content Information Identifiers. A second identification mechanism pertains to the DIDL XML documents that package the Digital Items. These identifiers are minted during ingestion into aDORe. They are unique within the aDORe repository, and even globally unique through the use of the info URI Scheme. This type of identifiers is mainly used for repository management purposes; it directly maps to the concept of OAIS AIP Identifiers.

During creation of a DIDL XML document, each constituent datastream of a Digital Item is conveyed as an MPEG-21 DID *component/resource* constructs, and is accorded a Fragment Identifier (XML ID). As a result, each constituent datastream becomes addressable in the aDORe repository using a combination of the AIP Identifier of the DIDL document in which it is contained, and its own Fragment Identifier. Figure 27 summarizes the above mapping principles.

In the aDORe repository, whenever a new version of a previously ingested digital asset is ingested, a new DIDL XML document is created for it; existing DIDL documents are never updated or edited. A version of a digital asset can be directly obtained using the identifier of the DIDL document that packages the transformed or modified content. All these versions of a specific digital asset share the same Digital Item Identifier. This approach closely resembles the versioning strategies defined by the OAIS Reference Model.

- *DSpace* (Smith et al., 2003; Tansley et al., 2003; Tansley et al., 2005): Compound digital assets stored in the DSpace repository are organized using the DSpace Data Model. A digital asset is typically represented as a DSpace Item; datastreams aggregated by the digital asset are called DSpace Bitstreams. The current DSpace digital repository system (release 1.3.2 – October 2005) instantiates the DSpace Data Model using linked tables in a relational database. DSpace Bitstreams are stored in a file system. Every DSpace Item receives a persistent unique identifier; DSpace uses the Handle System for minting, managing and resolving these identifiers. It is important to note that DSpace treats the identifiers that were assigned to digital assets before their ingestion into DSpace (e.g., URNs, info URIs, etc.) as descriptive metadata (`dc:identifier`), not as identifiers that can easily be mapped to identifier concepts in the OAIS Information Model. This is in contrast with aDORe that treats these identifiers as OAIS Content Information Identifiers. Also, the current version of the DSpace system does not seem to provide an unambiguous solution for the versioning of stored digital assets. While it is argued that a separate DSpace Item could be created for each distinct version of a digital asset (Smith, 2004), this approach does not allow for multiple versions of a digital asset to share the same content identifier. Overall, it seems difficult to unambiguously map the manner in which the current version of DSpace handles identifiers and versions to corresponding concepts of the OAIS Information Model.

  However, at the time of writing, plans exist to create a new version of the DSpace repository system (version 2.0) (Tansley et al., 2003). In the planned version, the DSpace Data Model would remain untouched, but it would introduce a flat file storage mechanism in which each DSpace Item is represented as an XML document conformant with the METS syntax (LC, 2005a). The main reason for replacing the current relational database structure with the METS-based solution is to ease various preservation-related tasks, including disaster recovery, versioning control, and data replication. In this revised approach, the DSpace Items that represent the actual content can be considered Content Information. A METS XML document that represents and serializes the content as a storable package can be considered the Archival Information Package. The METS syntax is the Packaging Information. In the planned release, different versions of a stored digital asset would be represented by different METS documents, one per version. Following this approach, several versions of a DSpace Item may share the same handle identifier; they will be distinguished by a different METS document. From this, it follows that in the planned version 2.0 of the DSpace system, the handle that is assigned to each DSpace Item maps to the concept of the OAIS Content Identifier. A unique identifier for a METS file would map to the concept of an OAIS AIP Identifier. In DSpace 1.x, a DSpace Bitstream within a DSpace Item has a numerical ID that is unique to a DSpace instance; in Dspace 2.0, most probably, such a value would be conveyed using an XML ID. The exact details with this respect remain a topic of further study by the DSpace group (Tansley et al., 2003). Figure 28 depicts the findings related to DSpace 2.0.

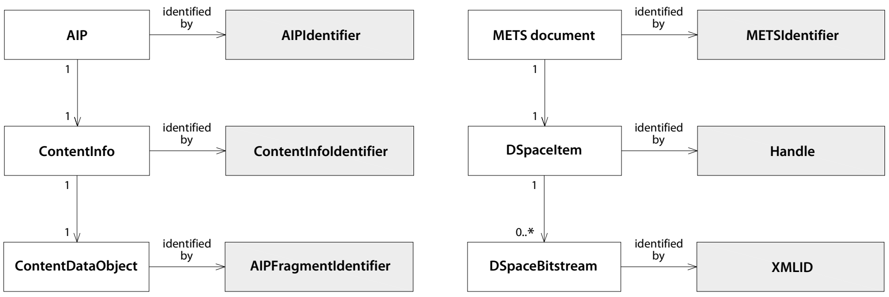


Figure 28. The representation, identification and versioning of DSpace Items

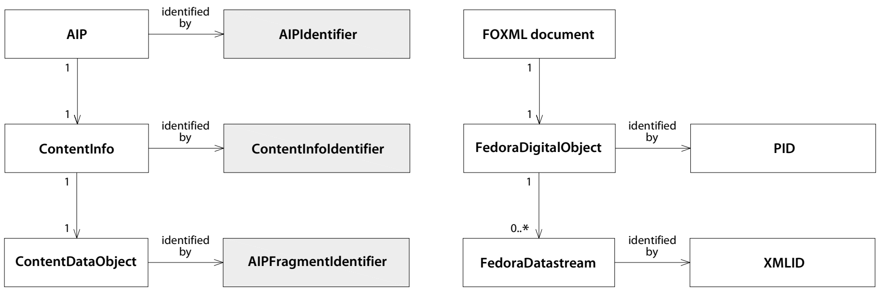


Figure 29. The representation, identification and versioning of Fedora Digital Objects

- *Fedora* (Payette & Lagoze, 1998; Lagoze et al., in press): Compound digital assets stored in the Fedora repository system (version 2.0) are represented in accordance with the Fedora Digital Object Model (Fedora Project, 2005b) and encoded using the FOXML syntax (Fedora Project, 2005a). The digital asset itself is referred to as a Fedora Digital Object; it maps to the OAIS concept of Content Information. The serialization of a Fedora Digital Object in FOXML is called a FOXML document. A FOXML document maps to the OAIS concept of Archival Information Package. The FOXML syntax can be considered the Packaging Information.

The Fedora system documentation does not make an explicit distinction between identifiers accorded to the actual content (i.e., Content Information Identifiers), and identifiers pertaining to stored packagings of the content (i.e., Information Package Identifiers). Upon ingestion, a unique persistent identifier, called 'PID', is assigned to the Fedora Digital Object and included in the FOXML document. PIDs may be minted by a Fedora repository or may be user-defined; the latter allows for the use of identifiers that were assigned to digital assets prior to their ingestion into Fedora. During the creation of a FOXML document, each datastream constituting a Fedora Digital Object is accorded an XML Fragment Identifier. In Fedora, these constituent datastreams become addressable using a rather obscure combination of the PID and their own Fragment Identifier.

In the Fedora repository, a modification made to a constituent of a Fedora Digital Object results in the creation of a new version of that constituent. Fedora does not create a new FOXML document when a new version (of a constituent) of a Digital Object becomes available. Instead, each specific constituent is versioned in the source FOXML document through the assignment of a local property – conveyed via an attribute in the FOXML document – that conveys a datetime of creation. In this versioning approach, the PID of the Fedora Digital Object remains constant (Fedora Project, 2005c). Because multiple versions of a Fedora Digital Object may share the same PID, and because a PID is considered the primary identification mechanism of a Fedora Digital Object for down-stream application, the PID maps to the OAIS concept of Content Information Identifier. Another objective motivator for this equation is the fact that a Fedora Digital Object, with a given PID, can be ingested and exported in different XML-based packaging formats, including FOXML, METS and MPEG-21 DIDL. Fedora's PIDs can also be expressed as URIs by prepending '`info:fedora`', a namespace that resides under the info URI umbrella.

In addition, a version of a Fedora Digital Object can be addressed using a local datetime property. In order for this datetime property to be used in a repository-global manner, it has to be combined with the PID of the Fedora Digital Object. This way of working is in contrast with the versioning strategy postulated by the OAIS Reference Model. The latter recommends creating a separate package per version. Each package, and hence, each version, is distinguished using a repository-unique AIP Identifier. The above findings are depicted in Figure 29.

2.2. Interfaces for harvesting and obtaining digital assets from actual repository systems

In aDORe and DSpace 2.0, content is versioned through the creation of new DIDL and METS documents, respectively. All versions (of a set of content) share the same OAIS Content Information Identifier (that is, a Digital Item Identifier in aDORe or a Handle in DSpace 2.0). Each version is distinguishable by a repository-unique OAIS AIP Identifier (DIDL document Identifier in aDORe, and METS Identifier in DSpace 2.0).

In Fedora, content is not versioned through the creation of new FOXML documents. Instead, when a new version of a constituent (of a Fedora Digital Object) becomes available, it is wrapped in the source FOXML document, and assigned a local key that conveys the datetime of creation of that constituent. As a result, a version of a Fedora Digital Object can be uniquely addressed within a Fedora system using the combination of the PID and the local datetime property.

Based on this brief exploration, it is fair to say that not all actual repository systems represent, identify and version their digital materials in a manner that is fully compliant with the OAIS Information Model. In particular, the OAIS-inspired use of repository-unique identifiers to distinguish between various versions of (the same) content, appears to be not well-established. It is obvious that in such cases Interface No 1 and Interface No 2, both of which are centered on OAIS AIP Identifiers, cannot be employed.

In what follows, I will revisit the characteristics of Interface No 3 and Interface No 4, both of which are centered on OAIS Content Information Identifiers, and explore how – in absence of repository-unique OAIS AIP Identifiers – versions of content can be retrieved.

Interface No 3. As described in Section 1.2.1, this interface enables harvesting packaged digital assets through the OAI-PMH by using identifiers of the digital assets (OAIS Content Information Identifiers). Based on the foregoing, it follows that:

- The OAI-PMH *identifiers* used by the interface are the identifiers of the stored digital assets. In aDORe, these identifiers correspond with the identifiers of Digital Items, in DSpace with the handles of DSpace Items, and in Fedora with the PIDs of Fedora Digital Objects.
- The most recent OAI-PMH *datestamp* is the datetime at which the most recent version of the identified OAIS Content Information has been created. In aDORe and DSpace, this *datestamp* corresponds with the datetime of creation of the DIDL document or METS document representing the most recent version of a Digital Item or DSpace Item, respectively. In Fedora, this *datestamp* corresponds with the most recent datetime at which a constituent of a digital asset has been added to the FOXML document.
- The OAI-PMH *metadata format* used by the interface is an XML-based packaging format. The semantics and structure of that format must be known by both the OAI-PMH harvester and the aDORe, DSpace and/or Fedora repositories.

Interface No 4. As described in Section 1.2.2, this interface enables obtaining disseminations of digital assets and their constituents through NISO's OpenURL Framework and by using identifiers of the digital assets (OAIS Content Information Identifiers). Based on the above, the following can be inferred:

- The *Referent Identifier Descriptor* corresponds with an identifier of a digital asset stored in the repository. In aDORe, DSpace and Fedora, this type of identifier is mapped to the identifier of the Digital Item, the Handle of the DSpace Item and the PID of the Fedora Object, respectively.
- The `aip` key of the *By-Value Metadata Descriptor* of the *Referent* conveys a repository-unique identifier of the package representing the digital asset. As shown in the above mapping, in aDORe, such an identifier would correspond with the identifier of a DIDL document and in DSpace with the identifier of a METS document.

In Fedora, however, no OAIS AIP Identifiers are available. Different versions of the same content are distinguished through the use of local datetime keys. A version of a Fedora Digital Object can be uniquely addressed using the combination of the PID of the Object and a datetime at which the version has been created.

In order for Interface No 4 to accommodate such scenarios, it should allow for digital assets to be versioned using any type of (local or global) version parameter. This can readily be met by replacing the `aip` key of the *By-Value Metadata Descriptor* of the *Referent* by a `version` key. While the value of the `aip` key, by definition of the OAIS AIP Identifier, must be unique within a digital repository, the syntax and uniqueness of the `version` key is defined by the repository system itself. A `version` key could convey a value that is globally unique within the context of an information system (notably OAIS AIP Identifier) or could carry a value that is unique within the context of a given OAIS Content Information Identifier (e.g., the Fedora datetime key).

- The `arg` key of the *By-Value Metadata Descriptor* of the *Referent* conveys a fragment identifier pertaining to the package (that has been obtained using the above *Descriptors*). In aDORe, DSpace and Fedora, those fragment identifiers typically correspond with XML IDs of a DIDL document, a METS document and a FOXML document, respectively.

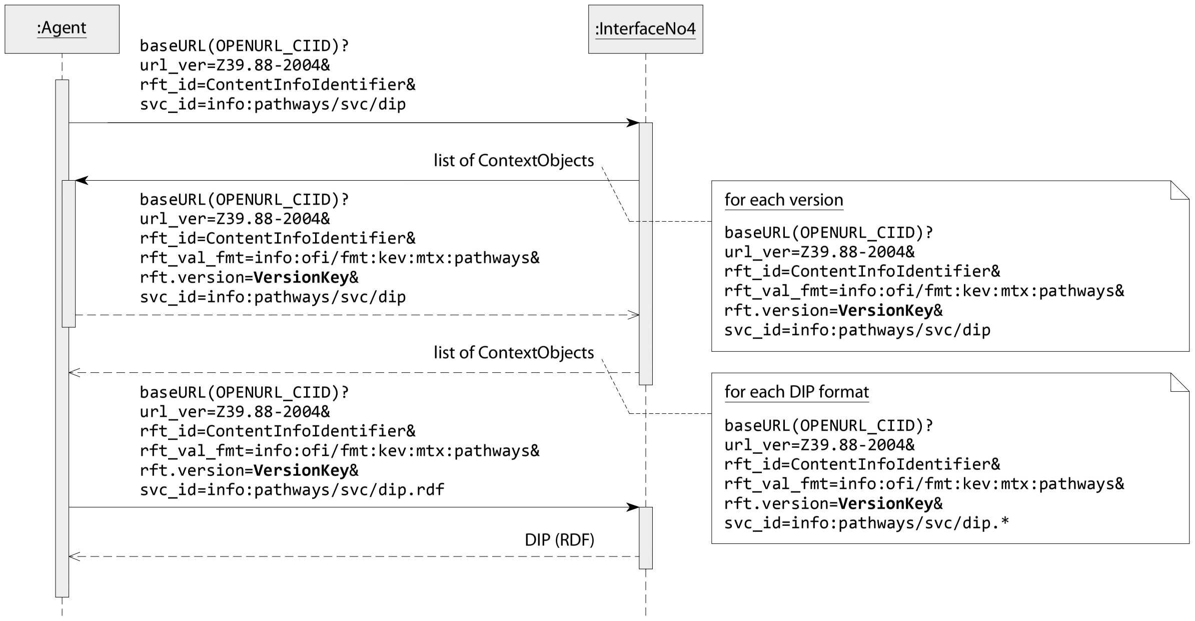
The interaction between a downstream agent and the above repository interface is depicted in the sequence diagrams in Figure 30 (Conformance Level 1) and Figure 31 (Conformance Level 2). Especially note the use of the `version` key, instead of the previously employed `aip` key. The XML document that defines the *Community Profile* of this repository interface is provided in Appendix A. The matrix defining the KEV Metadata Format to represent version- and datastream-level identification of a digital asset is provided in Appendix B.

3. Conclusion

This chapter focused on the design and definition of persistent, standards-based *read-only* interfaces to repositories. A distinction has been made between interfaces that use shareable identifiers of the digital assets themselves (that is OAIS Content Information Identifiers), and interfaces that use the internal, repository-specific identifiers of packages representing these digital assets (that is OAIS AIP Identifiers). As was discussed, the former lend themselves for public exposure and can therefore play a significant role to augment cross-repository interoperability. The latter are especially useful in the context of intra-repository mirroring or preservation scenarios that require full duplication of all versions of digital assets.

The potential of the OAI-PMH has been explored to build interfaces for harvesting packaged digital assets. The use of the OAI-PMH as the protocol framework for a harvest interface has the following attractive features:

- The ability to selectively harvest batches of XML-based packages of digital assets from a digital repository system in a manner that augments cross-system and cross-community interoperability.
- The ease of implementation: OAI-PMH is a lightweight protocol-based framework for which several software tools are readily available.



```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
svc_id=info:pathways/svc/dip
```

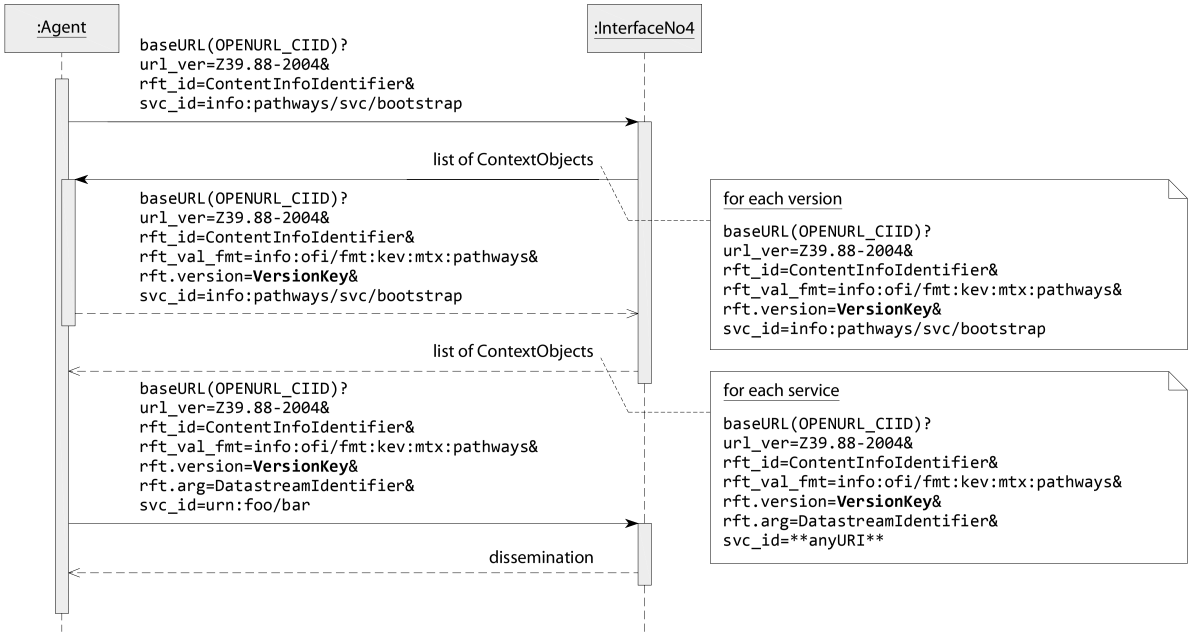
```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.version=VersionKey&
svc_id=info:pathways/svc/dip
```

```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.version=VersionKey&
svc_id=info:pathways/svc/dip.rdf
```

```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.version=VersionKey&
svc_id=info:pathways/svc/dip
```

```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.version=VersionKey&
svc_id=info:pathways/svc/dip.*
```

Figure 30. UML Sequence Diagram of Interface No 4, Level 1 (revisited)



```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
svc_id=info:pathways/svc/bootstrap
```

```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.version=VersionKey&
svc_id=info:pathways/svc/bootstrap
```

```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.version=VersionKey&
rft.arg=DatastreamIdentifier&
svc_id=urn:foo/bar
```

```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.version=VersionKey&
svc_id=info:pathways/svc/bootstrap
```

```
baseURL(OPENURL_CIID)?
url_ver=Z39.88-2004&
rft_id=ContentInfoIdentifier&
rft_val_fmt=info:ofi/fmt:kev:mtx:pathways&
rft.version=VersionKey&
rft.arg=DatastreamIdentifier&
svc_id=**anyURI**
```

Figure 31. UML Sequence Diagram of Interface No 4, Level 2 (revisited)

- The ability to augment the OAI-PMH *record* and its associated *metadata* using third party XML Schemas. For example, as shown in the APS experiment, XML signatures can be included in the OAI-PMH responses to allow verification of authenticity and integrity of the harvested information. Similarly, data providers may associate rights expressions with(in) the returned OAI-PMH *record* to indicate how it may be used, shared and modified after it has been harvested. A practical solution on how to convey such rights expressions is described by Lagoze , Van de Sompel, Nelson and Warner (2003a).

The potential of the NISO OpenURL Framework has been explored to build interfaces capable of responding to two kinds of service requests: First, obtaining packaged digital assets from digital repositories on a one-by-one basis. Second, the request of datastream-level disseminations.

This study also argues that the use of the OpenURL standard for the specification of an obtain interface to a repository has other highly attractive features:

- The ability to obtain various disseminations of digital assets and their constituent datastreams in a manner that augments cross-system and cross-community interoperability.
- Because the OpenURL standard is specified in a generic manner, it allows for the same conceptual interfaces to be implemented in different ways as technologies evolve. The concepts underlying the interface persistent over time.
- Because the *ContextObject* recognizes other *Entities* that are involved in a service request – notably *Requester*, *Referrer* and *Resolver* – it offers the potential for requesting context-sensitive services.

General discussion and conclusion

General discussion and conclusion

This dissertation studied the definition of read-only persistent interfaces to digital repositories. The following scientific contributions can be distilled (per Chapter):

- In the opening Chapter, the author listed the core building blocks that are needed for devising interfaces for harvesting and obtaining assets from digital repositories. A clear distinction has been made between a data model (viz. the OAIS Reference Model), a packaging format (viz. MPEG-21 DID), a harvest protocol (viz. the OAI-PMH) and an obtain framework (viz. the OpenURL Framework for Context-Sensitive Services).

  The author ascertained the different levels of identification that are latently present in the OAIS Reference Model. He also demonstrated how these levels of identification can be employed as a means to manage and identify different versions of content. The author provided a brief description of the MPEG-21 DID standard, and showed how terminology and concepts of the MPEG-21 DID Abstract Model can be joined and mapped to the OAIS Information Model.

  The author provided a brief summary of the OAI-PMH and suggested how a content harvest interface can be devised within the boundaries of the OAI-PMH. The technique builds on the introduction of more expressive *metadata formats*, notably packaging formats, to represent OAI-PMH *resources*. The author also described the recent NISO OpenURL standard, discussed its persistent data model, and emphasized its unique capability to convey context-sensitive service requests.

- In the succeeding Chapter, the author has illustrated the validity and feasibility of repository interfaces to harvest and obtain digital assets by means of two elaborate experiments. In the aDORe experiment, the author has studied the multi-faceted use of the OAI-PMH and NISO's OpenURL Framework for Context-Sensitive Services to harvest and obtain assets from a large collection of digital assets at the Research Library of the Los Alamos National Laboratory. In the APS experiment, the author has proposed a fully standards-based approach to the longstanding problem of content transfer between nodes in the electronic scholarly communication system. Again the OAI-PMH played a significant role in the deployed solution. The author has also demonstrated that the devised content transfer solution can easily be integrated with technologies that provide guarantees regarding authenticity and accuracy of the transferred assets (e.g., W3C XML Signatures Syntax and Processing), if such guarantees are required by downstream businesses or consumers.

  It should be noted that many other contributions can be distilled from these experiments, including the thinking underlying such components as the aDORe Transform Engine and the aDORe XMLtape storage solution. These contributions have been performed in cooperation with team members of the Digital Library Research and Prototyping Team of the Los Alamos National Laboratory.

  It should also be noted that the contributions made by these experiments go beyond the mere illustration of the correctness of the proposed interfaces. The aDORe repository experiment introduced a novel way of thinking about repository systems. aDORe's design is highly distributed, and modular; and all components are accessed through persistent harvest and obtain interfaces. Also, standards are used at every level of the design. Because of these characteristics, the aDORe experiment has attracted significant attention in the realm of repository system design as such. But more importantly, the highly distributed nature of the design has already impacted the thinking regarding cross-repository interoperability, as is illustrated – for example – by the ongoing work on the Chinese DSpace repository federation project ("The China Digital Museum Project," 2005) (see below). In the APS experiment, the transfer solution as such will have an impact on ongoing practice with this respect, and has already impacted the thinking in the realm of digital preservation where content mirroring is considered a core strategy to preserve our digital heritage.

- In the third Chapter, the author introduced a set of theoretical interfaces that can be devised for harvesting and obtaining assets from digital repositories. A distinction has been made between repository interfaces that use identifiers of the digital assets (i.e., OAIS Content Information Identifiers), and repository interfaces that use identifiers of the packages that represent and wrap these digital assets (i.e., AIP Identifiers). The former set of interfaces is used in intra-repository environments, while the latter lends itself for public exposure. The author highlighted the characteristics of the interfaces as they build upon the different levels of identification.

  In the second part of this Chapter, the author revealed how existing repository systems sometimes fail to clearly distinguish between these OAIS-levels of identification. Following on from this, the author reformulated the theoretical interfaces so they can be deployed across actual repository systems; even systems that do not strictly adhere to identifier concepts as proposed by the OAIS Reference Model. By doing so, the author reconciled theory and pragmatism.

- In order for such standards as MPEG-21 DID and METS to be used in digital repository scenarios as the ones envisaged in this dissertation, the author has submitted several proposals to the MPEG Committee and METS Editorial Board. Several of these proposals raised the need for endorsing the difference between OAIS Content Information, on the one hand, and OAIS Information Packages, on the other hand, by introducing two distinct identification mechanisms, that is, one for identifying digital assets, and one for addressing the XML-based packages of these digital assets.

  The MPEG Committee added the capability to identify a DIDL document by means of an optional attribute (`DIDLDocumentId`) conveyed within that DIDL document; and the METS Editorial Board added the capability to express identifiers of the actual content by appending a `CONTENTID` attribute to the `div`, `fptr`, `mptr` and `area` elements. While both additions are seemingly simple at first sight, they caused a great deal of controversy, and required many lengthy discussions, in the respective standardization communities. Eventually, all proposals got adopted, and their importance got acknowledged. Ensuing from this, a second edition of MPEG-21 DID was created, as well as an amendment on MPEG-21 DII, an amendment on MPEG-21 RDD, and a new release of the METS XML Schema. More information on these MPEG proposals can be found in Section 2 of Chapter 1. A list of contributions to MPEG is provided in annex.

The repository interfaces advanced in this study have been designed with a number of requirements in mind. See Section 5 of the Introduction. Based on these requirements, the following reflections upon the proposed interface can be made:

- The interfaces employ a novel use of the OAI-PMH protocol and the NISO OpenURL Framework. Both technologies support the protocol-based interchange of digital information in a networked environment.

- The interfaces operate in a manner that is neutral to the way in which a digital repository stores and manages its assets. This is reflected in several ways:

  First, an interoperable data model is introduced that functions as a crosswalk between the heterogeneous data models for digital assets used by the various autonomous repositories and the (profiled) data model used to interchange digital assets. A digital asset exposed by a repository interface is represented as an XML-based tangible package that, in turn, is compliant with the data model to interchange digital assets.

  Second, the interfaces do not make any presumptions about the identifier namespace that is used for the identification of digital assets, and hence, can be implemented across a broad variety of repository systems.

  Third, the interfaces allow for various version of content to be distinguished, insofar as versioning strategies are supported by the digital repository that exploits the interfaces. Following the OAIS preservation strategy, for each version of content being transferred, a new package is created. As all such versions share the same OAIS Content Information Identifier, the parallel identification mechanism (OAIS Content Information Identifier, OAIS AIP Identifier) ensures that different versions remain distinguishable.

- Concepts underlying the interfaces are – for the greater part – defined in a persistent manner, while their implementation may change over time when technologies evolve. The NISO OpenURL Framework makes a clear distinction between the abstract definition of a *ContextObject* and its concrete representation and the protocol by which such representations are transported. The OpenURL Framework allows for a *ContextObject* to be represented in many different *Formats* and transported using many different *Transports*. Also the MPEG-21 DID makes a clear distinction between an Abstract Data Model, and its serialization in XML (viz. MPEG-21 DIDL).

  For completeness, it should be added that concepts of the OAI-PMH are not defined in a fully abstract manner. The OAI-PMH is highly dependent on the HTTP protocol and the XML syntax for transporting and serializing the exposed materials, respectively. While this approach proves to be satisfactory in the current technological environment, it may prove to be inadequate as technologies evolve. Providing an abstract model of OAI-PMH interactions, neutral to the transport protocol and serialization syntax, would be a change for the better.

- For each of the employed technologies, various – free and open source – tools are available, such as OCLC's OAICat (OCLC, 2005a), OAIHarvester (OCLC, 2005b), OAI Viewer (OCLC, 2004) and OpenURL 1.0 (OCLC, 2005c). Also the choice for XML to represent digital assets is fruitful. A broad choice of XML processing tools is currently available, including tools that validate XML documents against XML Schema definitions that specify compliance with regard to both structure and datatypes. Also, its broad industry support provides some guarantees regarding longevity or, eventually, migration paths.

- Both the OAI-PMH and the OpenURL have received a high level of buy-in from the digital library and scholarly information communities.

Since the publication of the above study, several efforts have been started that aim at exploring new levels of interoperability by building upon the insights gained from this study. Such efforts include the implementation of the proposed interfaces for a variety of repository systems, the exploration of data models and packaging formats for the modeling, representation and packaging of digital assets, the development of federations of scholarly repositories, the study of cross-repository service overlays and automated service workflows, and so forth. A brief summary is provided below.

- ***DSpace and Fedora harvest interfaces*** ("Mpeg21-DIDL distribution plug-in," 2004). In a first attempt to publicly demonstrate the attractiveness and feasibility of the harvest interface, the author was involved in an effort to create experimental plug-ins for the DSpace v.1.2 and Fedora systems. Through these plug-ins, digital assets from DSpace and Fedora repository systems are dynamically represented as MPEG-21 DIDL XML documents, and exposed for harvesting through the existing OAI-PMH interface of both systems. To this end, the OAI-PMH *identifier* is put on a par with the handle of a DSpace Item and the PID (in the form of info URI) of the Fedora Digital Object, respectively. It is also worth mentioning that, inspired by the DSpace DIDL plug-in, the DSpace group has developed a separate module to expose DSpace Items packaged as METS documents. Both modules will be integrated in the second version of the DSpace software.

- ***mod_oai harvest interface*** (Nelson, Van de Sompel, Liu, Harrison & McFarland, 2005). mod_oai, a joint project of the Old Dominion University and the LANL Research Library and funded by the Andrew W. Mellon Foundation, aims to bring the incremental harvesting power offered by combining OAI-PMH and MPEG-21 DID to the Web crawling community. An Apache module, mod_oai, is being developed that automatically responds to OAI-PMH requests on behalf of a web server. Apache is an open-source web server that is used by 63% (approximately 27 million) of the websites in the world. Remember, as broached in the introductory section of this dissertation, a web server is the most common – but also most transitory – form of digital repository. The mod_oai demonstrates that the public harvest interface can be readily integrated in basic web server technology.

  By embedding OAI-PMH capabilities into an Apache server, OAI-PMH enabled web robots can be much more efficient than current web robots, which traverse an entire web site to locate new and updated web pages. Indeed, when using the OAI-PMH to access an Apache web server that is powered by the mod_oai module, new and updated pages become immediately visible. Unnecessary accesses to unchanged pages are eliminated, robots can harvest quicker, and network traffic is considerably reduced for web sites. In order to achieve its goal, the mod_oai module dynamically represents resources available on an Apache Web server as MPEG-21 DIDL XML documents, which can then be transported using the OAI-PMH. The HTTP URI of the web page functions as the OAI-PMH *identifier*.

- ***aDORe Federation***. The creation of the aDORe environment (see also Chapter 2) has led to interesting insights regarding possible new levels of interoperability in a federation of heterogeneous repositories distributed over the Web (Jerez et al., 2004). Generally speaking, a federation of community-based autonomous repositories can be imagined, in which each repository exposes its content through interfaces that are proposed in the setting of this dissertation. An OAI-PMH interface enables harvesting XML-based packages of digital assets and an interface compliant with NISO's OpenURL Framework facilitates obtaining disseminations of digital assets on a one-by-one basis. Packaging formats supported by those repositories could be homogeneous across the federation with each repository supporting, for example, MPEG-21 DIDL. But the federation could also be heterogeneous, with one repository supporting MPEG-21 DIDL, another METS, and yet another IMS-CP.

  Next, following the aDORe design, several infrastructural components can be introduced that support the overall functioning of the federation. Such components include a Repository Index, and a Digital Asset Locator. A Repository Index is a registry that keeps track of the creation and location of the autonomous repositories, and the base URLs of their public interfaces; a Digital Asset Locator keeps track of the identifiers of the digital assets stored in such repositories. The exact details and boundary conditions of these components are subject of further study and prototyping.

- ***China Digital Museum Project*** ("The China Digital Museum Project," 2005). This project, under development by the Chinese Ministry of Education, Hewlett-Packard and several Chinese universities, aims at creating and deploying a large-scale federation of DSpace institutional repositories to host content from Chinese universities' digital museums. It is assumed that each university museum runs a local instance of DSpace to hold the digitized content and associated descriptive metadata from its local holdings. In addition, two *data centers* will be devised, each of which holds a replicated copy of all the digitized content and descriptive metadata from all of the museums. In order to transfer full content and descriptive metadata from the individual DSpace instances to these data centers, content hosted by the individual repositories will be packaged as METS documents and made harvestable via the OAI-PMH *resource* harvesting technique as devised in this study, and by using the aforementioned DSpace METS harvest plug-in. In addition to these repository-level interfaces, several central components will be developed to keep track of the individual repositories and their contained content. The exact details with this respect remain a topic of further study by the respective participants.

- ***Pathways*** ("Pathways," n.d.; Bekaert, Liu, Van de Sompel, Lagoze, Payette, & Warner, in press). This project, guided by Cornell University and the Los Alamos National Laboratory and funded by the National Science Foundation, aims at exploring data models and protocols to create a loosely-coupled, highly distributed, interoperable scholarly communication system.

  In this project, the author is involved in the creation of an interoperable data model, called Pathways Core model, that facilitates digital asset re-use and a broad spectrum of cross-repository services. The Pathways Core model is designed to be lightweight to implement, and to be widely applicable as a shared profile or as an overlay on data models currently used in repository systems and applications. The Pathways Core model can be serialized in a variety of ways, and an RDF serialization as well as a profile of MPEG-21 DIDL have been created as reference implementations.

The author is also involved in an experiment to illustrate the power of the Pathways Core. A number of heterogeneous repositories implemented an obtain interface from which, given the identity of a digital asset, an RDF serialization of the digital asset compliant with the Pathways Core can be retrieved. Using these obtain interfaces, an overlay journal can collect serializations of some digital assets (notably scholarly papers) from the different collaborating repositories, and assemble those into a new issue of the journal. The overlay journal then itself implemented the same obtain interface, and as a result, an RDF serialization of the entire journal, an issue, and an article could be extracted. This interface could then be used by a preservation repository to collect content from the overlay journal for ingest and mirroring.

Pathways illustrates how the creation of cross-repository services and service workflows can be facilitated through support of an interoperable data model (the Pathways Core) and an interoperable service interface (the OpenURL-based obtain interface). But more importantly, Pathways demonstrates how the harvest and obtain repository interfaces can be implemented using technologies other than the once described in this dissertation. The Pathways Core model is introduced as an alternative data model. Its serializations in RDF and MPEG-21 DIDL are introduced as packaging formats. (See also Figure 1)

The research presented in this dissertation touches on a few issues that require further study. The most relevant possibilities for future research are:

- The compilation of full interface specifications, and the implementation and testing of those specifications for widely deployed repository software.
- The definition of an abstract model of the OAI-PMH. The concepts underpinning such a model would remain persistent, while the OAI-PMH response formats and transport protocols may change as technologies evolve.
- The exploration of techniques for access control. As explained in the opening Chapter of this dissertation, access control can be implemented using cross-repository authentication and authorization solutions, such as Shibboleth, or by implementing access policies at the level of the individual packages. The latter may require the use of such tools as MPEG-21 REL, MPEG-21 IPMP, and W3C XML Encryption and Processing. The experience gained from the APS experiment by building in measures for data accuracy and data authenticity may prove to be helpful in such study.
- The exploration of rich, context-sensitive service requests as envisaged by the NISO OpenURL Framework. Such service requests would, for example, take into account information about the *Requester*; that is the resource that requests services pertaining to the *Referent*. Characteristics of the *Requester* could be expressed by means of Usage Environment Description Tools of MPEG-21 DIA (ISO, 2004e). These tools enable

describing user characteristics and preferences, terminal capabilities, network characteristics and natural environment characteristics.

To conclude, it should be noted that this dissertation has mainly focused on technical matters to increase cross-repository interoperability. However, it is important to understand that unless these technical solutions are accompanied by an endorsement at the cultural, political and organizational level, and a renewed commitment from the various parties involved to work together, it is unlikely that efforts to achieve technical interoperability between heterogeneous repositories will be successful.

Therefore, it is interesting to observe that, at the time of writing this dissertation, several leading organizations from the information domain have joined forces and taken initial steps aimed a providing such a level of commitment. In particular, under guidance of the Andrew W. Mellon Foundation, the Coalition for Networked Information, the Digital Library Federation (DLF), the Joint Information Systems Committee (JISC) and Microsoft, and instigated by dr. Herbert Van de Sompel (LANL) and dr. Tony Hey (Vice President of Microsoft Technical Computing), a meeting has been called aimed at identifying concrete steps that could be taken to augment interoperability across heterogeneous scholarly repositories (See also Appendix C). The meeting focuses on a limited set of core, protocol-based repository interfaces that would allow downstream applications to interact with heterogeneous repositories in a consistent manner. In particular, the following discussion topics are proposed:

- Support for a packaging format of digital assets that can be used for interchange between repositories and downstream applications as well as between repositories.
- Support for a harvest interface from which XML-based packages of digital assets can be harvested.
- Support for an obtain interface from which XML-based packages of an identified digital asset can be obtained on a one-by-one basis.
- Support for an obtain interface from which a list of repository-specific services pertaining to an identified digital asset can be obtained; including the definition and identification of a core set of service requests that are understood by all repositories.
- Support for a deposit interface to which a digital asset (likely represented in accordance with an XML-based packaging format) can be put from client applications, as well as from other repositories

The meeting will be attended by repository software representatives from such repository systems as DSpace, EPrints, and Fedora; content repository representatives from major publisher companies, and technology advisors from such projects as aDORe, Pathways, NSDL and the DSpace Chinese Federation Project. The research conducted in the setting of this dissertation was a primary source of inspiration for drafting the agenda, and may prove fruitful in ensuing discussions, and the compilation of required interoperability specifications.

Appendix A

Appendix A: Obtain interface Community Profile

```xml
<?xml version="1.0" encoding="UTF-8"?>
<profile xmlns="info:ofi/pro-2004">
  <registry-identifier>info:ofi/pro:pathways</registry-identifier>
  <name>Obtain interface Community Profile</name>
  <context-object-format>
    <context-object minOccurs="1" maxOccurs="1">
      <referent minOccurs="1" maxOccurs="1">
        <identifier minOccurs="1" maxOccurs="1"/>
        <by-value-metadata minOccurs="0" maxOccurs="1"/>
        <by-reference-metadata minOccurs="0" maxOccurs="0"/>
        <private-data minOccurs="0" maxOccurs="0"/>
      </referent>
      <referring-entity minOccurs="0" maxOccurs="1">
        <identifier minOccurs="0" maxOccurs="1"/>
        <by-value-metadata minOccurs="0" maxOccurs="1"/>
        <by-reference-metadata minOccurs="0" maxOccurs="1"/>
        <private-data minOccurs="0" maxOccurs="1"/>
      </referring-entity>
      <requester minOccurs="0" maxOccurs="1">
        <identifier minOccurs="0" maxOccurs="1"/>
        <by-value-metadata minOccurs="0" maxOccurs="1"/>
        <by-reference-metadata minOccurs="0" maxOccurs="1"/>
        <private-data minOccurs="0" maxOccurs="1"/>
      </requester>
      <service-type minOccurs="1" maxOccurs="1">
        <identifier minOccurs="0" maxOccurs="1"/>
        <by-value-metadata minOccurs="0" maxOccurs="1"/>
        <by-reference-metadata minOccurs="0" maxOccurs="1"/>
        <private-data minOccurs="0" maxOccurs="0"/>
      </service-type>
      <resolver minOccurs="0" maxOccurs="1">
        <identifier minOccurs="0" maxOccurs="1"/>
        <by-value-metadata minOccurs="0" maxOccurs="1"/>
        <by-reference-metadata minOccurs="0" maxOccurs="1"/>
        <private-data minOccurs="0" maxOccurs="1"/>
      </resolver>
      <referrer minOccurs="0" maxOccurs="1">
        <identifier minOccurs="0" maxOccurs="1"/>
        <by-value-metadata minOccurs="0" maxOccurs="1"/>
        <by-reference-metadata minOccurs="0" maxOccurs="1"/>
        <private-data minOccurs="0" maxOccurs="1"/>
      </referrer>
    </context-object>
  </context-object-format>
  <format>
    <registry-identifier>info:ofi/fmt:kev:mtx:ctx</registry-identifier>
    <name>Key/Encoded-Value ContextObject Format</name>
  </format>
  <serialization>
    <registry-identifier>info:ofi/fmt:kev</registry-identifier>
    <name>Key/Encoded-Value Physical Representation</name>
  </serialization>
  <constraint-language>
    <registry-identifier>info:ofi/fmt:kev:mtx</registry-identifier>
    <name>NISO Z39.88-2004 Matrix Constraint Language</name>
  </constraint-language>
  <character-encodings>
    <character-encoding type="default">
```

```
      <registry-identifier>info:ofi/enc:UTF-8</registry-identifier>
      <name>UTF-8 encoded Unicode</name>
    </character-encoding>
    <character-encoding>
      <registry-identifier>info:ofi/enc:ISO-8859-1</registry-identifier>
      <name>ISO Latin 1</name>
    </character-encoding>
  </character-encodings>
  <metadata-formats>
    <metadata-format entityRange="referent">
      <registry-identifier>info:ofi/fmt:kev:mtx:pathways</registry-identifier>
      <name>KEV Metadata Format for info related to the repository interfaces</name>
    </metadata-format>
  </metadata-formats>
  <namespaces>
    <namespace entityRange="referent">
      <registry-identifier>info:ofi/nam:ftp:</registry-identifier>
      <name>Namespace for "ftp" URI Scheme</name>
    </namespace>
    <namespace entityRange="referent">
      <registry-identifier>info:ofi/nam:http:</registry-identifier>
      <name>Namespace for "http" URI Scheme</name>
    </namespace>
    <namespace entityRange="referent">
      <registry-identifier>info:ofi/nam:https:</registry-identifier>
      <name>Namespace for "https" URI Scheme</name>
    </namespace>
    <namespace entityRange="referent">
      <registry-identifier>info:ofi/nam:info:</registry-identifier>
      <name>Namespace for "info" URI Scheme</name>
    </namespace>
    <namespace entityRange="referent">
      <registry-identifier>info:ofi/nam:urn:</registry-identifier>
      <name>Namespace for "urn" URI Scheme</name>
    </namespace>
    <namespace entityRange="service-type">
      <registry-identifier>info:ofi/nam:info:pathways/scv/dip</registry-identifier>
      <name>Id for service requesting all packaging formats</name>
    </namespace>
    <namespace entityRange="service-type">
      <registry-identifier>info:ofi/nam:info:pathways/scv/bootstrap
        </registry-identifier>
      <name>Id for service requesting a list of datastream-level services</name>
    </namespace>
  </namespaces>
  <transports>
    <transport>
      <registry-identifier>info:ofi/tsp:http:openurl-by-val</registry-identifier>
      <name>By-Value OpenURL over HTTP</name>
    </transport>
    <transport>
      <registry-identifier>info:ofi/tsp:http:openurl-by-ref</registry-identifier>
      <name>By-Reference OpenURL over HTTP</name>
    </transport>
    <transport>
      <registry-identifier>info:ofi/tsp:openurl-http:inline</registry-identifier>
      <name>Inline OpenURL over HTTP</name>
    </transport>
  </transports>
</profile>
```

Appendix B

Appendix B: Matrix defining the KEV Format for obtain interfaces

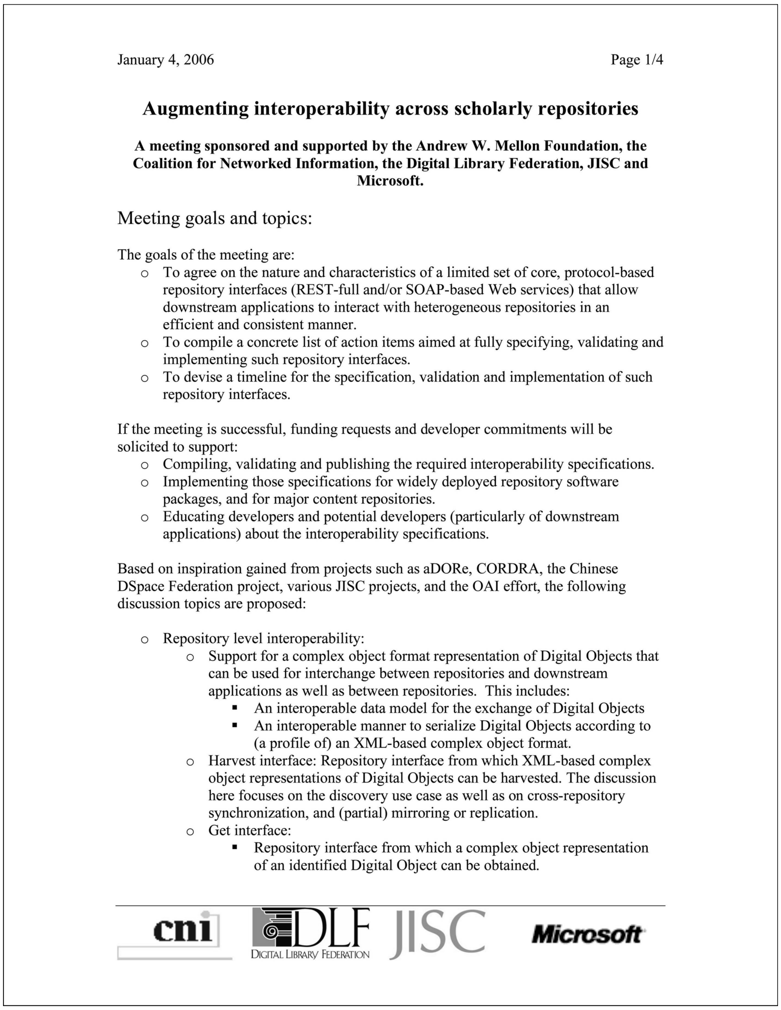
`dc:title`	KEV Metadata Format: pathways
`dc:creator`	NISO Committee AX, OpenURL Standard Committee
`dc:description`	This Matrix represents pathways related information as a string of ampersand-delimited Key/Encoded Value pairs
`dc:identifier`	http://www.openurl.info/registry/docs/info:ofi/fmt:kev:mtx:pathways
`dc:identifier`	info:ofi/fmt:kev:mtx:pathways
`dcterms:created`	2006-02-01

`<data>`	Character string
`<id>`	Character string for an Identifier, of the form: [ori:([a-zA-Z0-9]:)+:[^:]+ | uri:.+ | xri:.+]
`<fmt-id>`	Character string for a Format Identifier, of the form: [ori:fmt:[a-z]:[a-z]:[^:]+ | uri:.+ | xri:.+]
`<m-key>`	Character string for a Metadata Key, of the form: [a-zA-Z0-9]+
`<url>`	Character string for a URL
`<date>`	Character string representing a date to the complete date level of the W3CDTF profile of ISO 8601, of the form: [YYYY-MM-DD | YYYY-MM | YYYY]
`<time>`	Character string representing a date and time to the seconds level of the W3CDTF profile of ISO 8601, of the form: [YYYY-MM-DDThh:mm:ssTZD | YYYY-MM-DD]

Delim	Key	Value	Min	Max	Description
&	`version`	`<data>`	0	1	conveys a repository-specific value to distinguish between different versions of content
&	`arg`	`<data>`	0	1	conveys a Fragment Identifier pointing to a datastream constituting the identified version

Appendix C

Appendix C: Augmenting interoperability across scholarly repositories



January 4, 2006 Page 1/4

Augmenting interoperability across scholarly repositories

A meeting sponsored and supported by the Andrew W. Mellon Foundation, the Coalition for Networked Information, the Digital Library Federation, JISC and Microsoft.

Meeting goals and topics:

The goals of the meeting are:

- To agree on the nature and characteristics of a limited set of core, protocol-based repository interfaces (REST-full and/or SOAP-based Web services) that allow downstream applications to interact with heterogeneous repositories in an efficient and consistent manner.
- To compile a concrete list of action items aimed at fully specifying, validating and implementing such repository interfaces.
- To devise a timeline for the specification, validation and implementation of such repository interfaces.

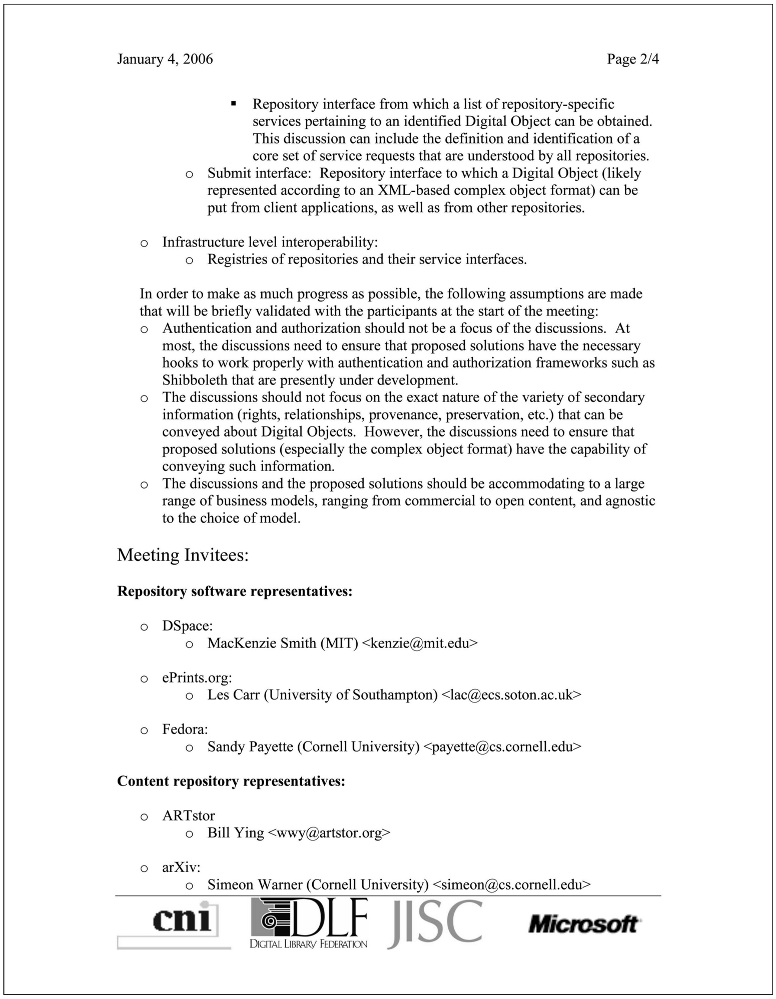
If the meeting is successful, funding requests and developer commitments will be solicited to support:

- Compiling, validating and publishing the required interoperability specifications.
- Implementing those specifications for widely deployed repository software packages, and for major content repositories.
- Educating developers and potential developers (particularly of downstream applications) about the interoperability specifications.

Based on inspiration gained from projects such as aDORe, CORDRA, the Chinese DSpace Federation project, various JISC projects, and the OAI effort, the following discussion topics are proposed:

- Repository level interoperability:
  - Support for a complex object format representation of Digital Objects that can be used for interchange between repositories and downstream applications as well as between repositories. This includes:
    - An interoperable data model for the exchange of Digital Objects
    - An interoperable manner to serialize Digital Objects according to (a profile of) an XML-based complex object format.
  - Harvest interface: Repository interface from which XML-based complex object representations of Digital Objects can be harvested. The discussion here focuses on the discovery use case as well as on cross-repository synchronization, and (partial) mirroring or replication.
  - Get interface:
    - Repository interface from which a complex object representation of an identified Digital Object can be obtained.

cni DLF DIGITAL LIBRARY FEDERATION JISC Microsoft



January 4, 2006 Page 2/4

        - Repository interface from which a list of repository-specific services pertaining to an identified Digital Object can be obtained. This discussion can include the definition and identification of a core set of service requests that are understood by all repositories.
    - Submit interface: Repository interface to which a Digital Object (likely represented according to an XML-based complex object format) can be put from client applications, as well as from other repositories.

- Infrastructure level interoperability:
    - Registries of repositories and their service interfaces.

In order to make as much progress as possible, the following assumptions are made that will be briefly validated with the participants at the start of the meeting:

- Authentication and authorization should not be a focus of the discussions. At most, the discussions need to ensure that proposed solutions have the necessary hooks to work properly with authentication and authorization frameworks such as Shibboleth that are presently under development.
- The discussions should not focus on the exact nature of the variety of secondary information (rights, relationships, provenance, preservation, etc.) that can be conveyed about Digital Objects. However, the discussions need to ensure that proposed solutions (especially the complex object format) have the capability of conveying such information.
- The discussions and the proposed solutions should be accommodating to a large range of business models, ranging from commercial to open content, and agnostic to the choice of model.

Meeting Invitees:

Repository software representatives:

- DSpace:
    - MacKenzie Smith (MIT) <kenzie@mit.edu>

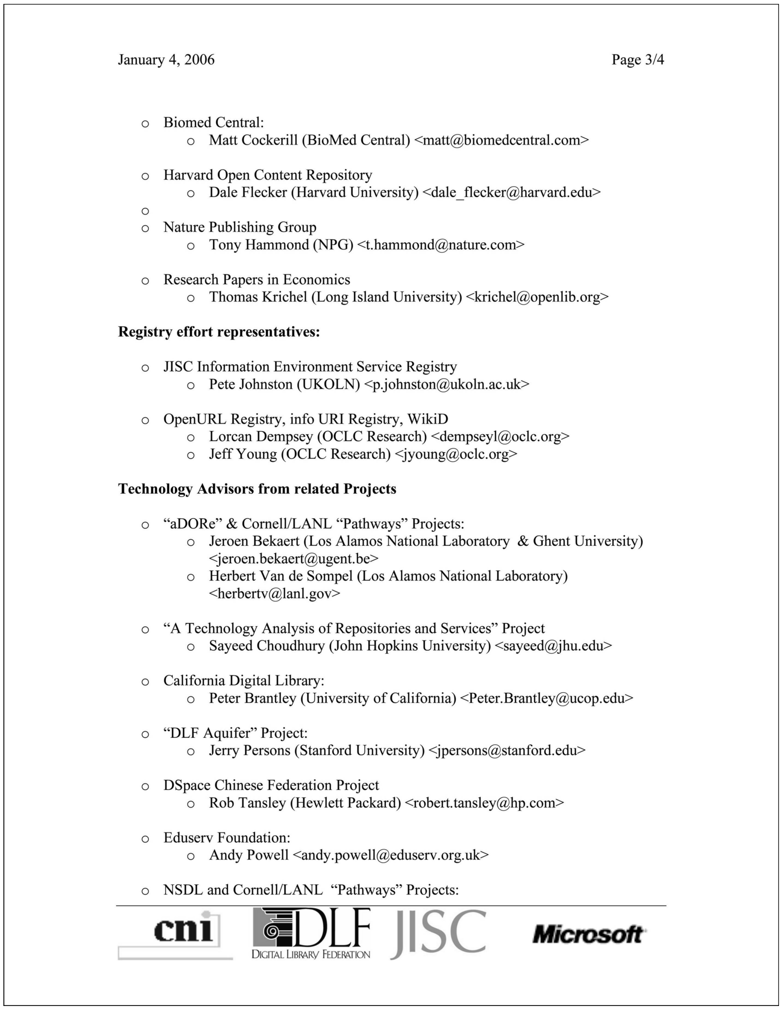
- ePrints.org:
    - Les Carr (University of Southampton) <lac@ecs.soton.ac.uk>

- Fedora:
    - Sandy Payette (Cornell University) <payette@cs.cornell.edu>

Content repository representatives:

- ARTstor
    - Bill Ying <wwy@artstor.org>

- arXiv:
    - Simeon Warner (Cornell University) <simeon@cs.cornell.edu>



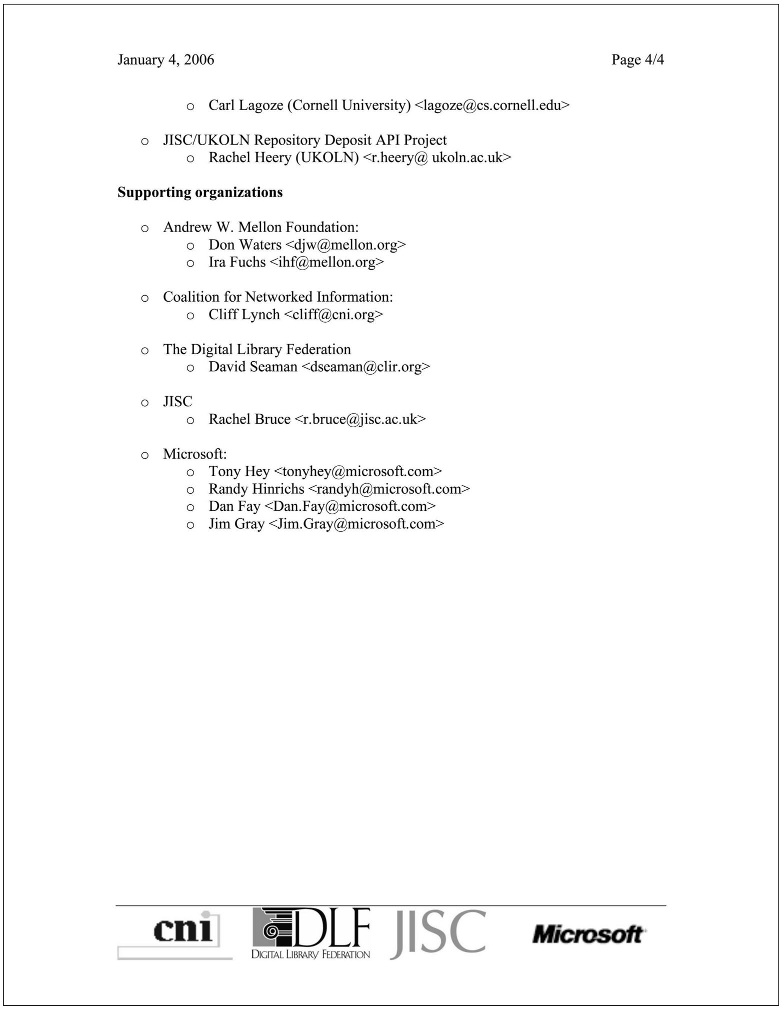
- Biomed Central:
    - Matt Cockerill (BioMed Central) <matt@biomedcentral.com>
- Harvard Open Content Repository
    - Dale Flecker (Harvard University) <dale_flecker@harvard.edu>
-
- Nature Publishing Group
    - Tony Hammond (NPG) <t.hammond@nature.com>
- Research Papers in Economics
    - Thomas Krichel (Long Island University) <krichel@openlib.org>

Registry effort representatives:

- JISC Information Environment Service Registry
    - Pete Johnston (UKOLN) <p.johnston@ukoln.ac.uk>
- OpenURL Registry, info URI Registry, WikiD
    - Lorcan Dempsey (OCLC Research) <dempseyl@oclc.org>
    - Jeff Young (OCLC Research) <jyoung@oclc.org>

Technology Advisors from related Projects

- “aDORe” & Cornell/LANL “Pathways” Projects:
    - Jeroen Bekaert (Los Alamos National Laboratory & Ghent University) <jeroen.bekaert@ugent.be>
    - Herbert Van de Sompel (Los Alamos National Laboratory) <herbertv@lanl.gov>
- “A Technology Analysis of Repositories and Services” Project
    - Sayeed Choudhury (John Hopkins University) <sayeed@jhu.edu>
- California Digital Library:
    - Peter Brantley (University of California) <Peter.Brantley@ucop.edu>
- “DLF Aquifer” Project:
    - Jerry Persons (Stanford University) <jpersons@stanford.edu>
- DSpace Chinese Federation Project
    - Rob Tansley (Hewlett Packard) <robert.tansley@hp.com>
- Eduserv Foundation:
    - Andy Powell <andy.powell@eduserv.org.uk>
- NSDL and Cornell/LANL “Pathways” Projects:



January 4, 2006 Page 4/4

- Carl Lagoze (Cornell University) <lagoze@cs.cornell.edu>

- JISC/UKOLN Repository Deposit API Project
  - Rachel Heery (UKOLN) <r.heery@ ukoln.ac.uk>

Supporting organizations

- Andrew W. Mellon Foundation:
  - Don Waters <djw@mellon.org>
  - Ira Fuchs <ihf@mellon.org>

- Coalition for Networked Information:
  - Cliff Lynch <cliff@cni.org>

- The Digital Library Federation
  - David Seaman <dseaman@clir.org>

- JISC
  - Rachel Bruce <r.bruce@jisc.ac.uk>

- Microsoft:
  - Tony Hey <tonyhey@microsoft.com>
  - Randy Hinrichs <randyh@microsoft.com>
  - Dan Fay <Dan.Fay@microsoft.com>
  - Jim Gray <Jim.Gray@microsoft.com>

Bibliography

Bibliography

<indecs> 2rdd Consortium. (2001, December). *<indecs>2rdd* (Input Document for the 58th MPEG Meeting, Pattaya, Thailand, No. ISO/IEC JTC1/SC29/ WG11/W7610). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Abrams, S. L., & Seaman, D. (2003, August). *Towards a global digital format registry*. Paper presented at World Library and Information Congress: 69th IFLA General Conference and Council, Berlin, Germany. Retrieved April 2, 2006, from http://www.ifla.org/IV/ifla69/papers/128e-Abrams_Seaman.pdf

Arms, W. Y. (1995, July). Key concepts in the architecture of the digital library. *D-Lib Magazine, 1*(7). Retrieved April 2, 2006, from http://hdl.handle.net/cnri.dlib/july95-arms

Arms, W. Y. (2000). *Digital libraries*. Cambridge, MA: MIT Press.

Bartel, M., Boyer, J., Fox, B., LaMacchia, B., & Simon, E. (2002, February 12). D. E. Eastlake, J. Reagle, & D. Solo (Eds.), *XML-Signature syntax and processing* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/xmldsig-core/

Beit-Arie, O., Blake, M., Caplan, P., Flecker, D., Ingoldsby, T., Lannom, L. W., et al. (2001, September). Linking to the appropriate copy: Report of a DOI-based prototype. *D-Lib Magazine, 7*(9). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/september2001-caplan

Bekaert, J. (on behalf of the Belgian National Standards Body) (2005, January). *BNB comments on ISO/IEC 21000-2 FCD 2nd edition* (Input Document for the 71th MPEG Meeting, Hong Kong, China, No. ISO/IEC JTC1/SC29/WG11/M11355). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Bekaert, J., Balakireva, L., Hochstenbach, P., & Van de Sompel, H. (2004, February). Using MPEG-21 DIP and NISO OpenURL for the dynamic dissemination of complex digital objects in the Los Alamos National Laboratory Digital Library. *D-Lib Magazine, 9*(11). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/february2004-bekaert

Bekaert, J., De Keukelaere, F., Van de Sompel, H., & Van de Walle, R. (2004, July). *Requirement for MPEG-21 DID* (Input Document for the 69th MPEG Meeting, Redmond, WA, No. ISO/IEC JTC1/SC29/WG11/M11081). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Bekaert, J., De Kooning, E., & Van de Sompel, H. (2006). Representing Digital Assets using MPEG-21 Digital Item Declaration. *International Journal on Digital Libraries, 6*(2), 159-173.

Bekaert, J., DeMartini, T., Rump, N., Barlas, C., Van de Sompel, H., & Van de Walle, R. (2005, April). *URI-reference of a DIDL document* (Input Document for the 72th MPEG Meeting, Busan, Korea, No. ISO/IEC JTC1/SC29/WG11/M11943). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Bekaert, J., DeMartini, T., Van de Walle, R., & Van de Sompel, H. (2004, July). *Identification of a DIDL document* (Input Document for the 69th MPEG Meeting, Redmond, WA, No. ISO/IEC JTC1/SC29/WG11/M10892). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Bekaert, J., Hochstenbach, P., De Neve, W., Van de Sompel, H., & Van de Walle, R. (2003, July). *Suggestions concerning MPEG-21 Digital Item Declaration* (Input Document for the 65th MPEG Meeting, Trondheim, Norway, No. ISO/IEC JTC1/SC29/WG11/ M9755). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Bekaert, J., Hochstenbach, P., & Van de Sompel, H. (2003, November). Using MPEG-21 DIDL to represent complex digital objects in the Los Alamos National Laboratory Digital Library. *D-Lib Magazine, 9*(11). Retrieved April 2, 2006, from http://dx.doi.org/ 10.1045/november2003-bekaert

Bekaert, J., Hochstenbach, P., Van de Sompel, H., & Van de Walle, R. (2003, December). *Suggestions concerning MPEG-21 Digital Item Declaration* (Input Document for the 67th MPEG Meeting, Hawaii, No. ISO/IEC JTC1/SC29/WG11/M10427). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Bekaert, J., Liu, X., & Van de Sompel, H. (2005). Representing Digital Objects for long-term preservation using MPEG-21 DID. In *Proceedings of PV-2005: Ensuring the Long-Term Preservation and Adding Value to the Scientific and Technical Data.*

Bekaert, J., Liu, X., Van de Sompel, H., Lagoze, C., Payette, S., & Warner, S. (in press). *Pathways Core. A Content Model for Cross-Repository Services.* Submitted to the Joint Conference on Digital Libraries '06.

Bekaert, J., & Rump, N. (Eds.). (2006, January). *ISO/IEC 21000-3 FPDAM/1. Digital Item Identifier relationship types* (Output Document of the 75th MPEG Meeting, Bangkok, Thailand, No. ISO/IEC JTC1/SC29/WG11/N7844). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Bekaert, J., & Van de Sompel, H. (2005a). Access Interfaces for Open Archival Information Systems based on the OAI-PMH and the OpenURL Framework for Context-Sensitive Services. In *Proceedings of PV-2005: Ensuring the Long-Term Preservation and Adding Value to the Scientific and Technical Data.*

Bekaert, J., & Van de Sompel, H. (2005b, June). A standards-based solution for the accurate transfer of digital assets. *D-Lib Magazine, 11*(6). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/june2005-bekaert

Berkman Center at Harvard Law (n.d.). *RSS 2.0 Specification.* Retrieved April 2, 2006, from http://blogs.law.harvard.edu/tech/rss

Berners-Lee, T., Fielding, R., & Masinter, L. (2005, January). *Uniform Resource Identifiers (URI): generic syntax* (IETF RFC 2986). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc3986.txt

Blinco, K., Mason, J., McLean, N., & Wilson, S. (2004, July). *Trends and Issues in E-learning Infrastructure Development* (A Paper prepared on behalf of the Australian Government Department of Education, Science and Training (DEST), and the JISC center for educational technology interoperability standards (CETIS)). Retrieved April 2, 2006, from http://www.jisc.ac.uk/uploaded_documents/Altilab04-infrastructureV2.pdf

Bollen, J., & Luce, R. (2002, June). Evaluation of digital library impact and user communities by analysis of usage patterns. *D-Lib Magazine, 8*(6). Retrieved April 2, 2006, from http:// dx.doi.org/10.1045/june2002-bollen

Boyer, J. (2001, March 15). *Canonical XML Version 1.0* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/xml-c14n

Boyer, J., Eastlake, D. E., & Reagle, J. (2002, July). *Exclusive XML Canonicalization* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/xml-exc-c14n/

Boyer, J., Hughes, M., & Reagle, J. (2002, November). *XML-Signature XPath Filter 2.0* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/ xmldsig-filter2/

Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., & Yergeau, F. (Eds.). (2004, February 4). *Extensible Markup Language (XML) 1.0 (Third Edition)* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/REC-xml/

Burner, M., & Kahle, B. (1996, September 15). *ARCfile format.* Retrieved April 2, 2006, from http://www.archive.org/web/researcher/ArcFileFormat.php

Burnett, I., Van de Walle, R., Hill, K., Bormans, J., & Pereira, F. (2003, October/December). MPEG-21: Goals and achievements. *IEEE MultiMedia, 10*(4), 60-70.

Chiariglione, L. (1998). Impact of MPEG standards on multimedia industry. *Proceedings of the IEEE, 86*(6), 1222-1227

The China Digital Museum Project. (2005, November, 11). Retrieved April 2, 2006, from http://wiki.dspace.org/ChinaDigitalMuseumProject 2005-11-07

Clark, J., & DeRose, S. (1999, November). *XML Path Language (XPath)* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/xpath

Consultative Committee for Space Data Systems Panel 2. (2004a, May). *Producer-Archive Interface Methodology* (CCSDS Blue Book 651.0-B-1). Retrieved April 2, 2006, from http://www.ccsds.org/CCSDS/documents/651x0b1.pdf

Consultative Committee for Space Data Systems Panel 2. (2004b, August). *XML Formatted Data Units (XFDU). Structure and construction rules. White Book Version 2.* Retrieved April 2, 2006, from http://www.ccsds.org/docu/dscgi/ds.py/Get/File-1913/ IPRWBv2a.2.pdf

ContentGuard, Inc. (n.d.). *XrML... eXtensible rights Markup Language*. Retrieved April 2, 2006, from http://www.xrml.org/

Cowin, J., & Tobin, R. (2004, February). *XML Information Set (XPath) (Second Edition)* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/xml-infoset/

Darlington, J. (2003, October). PRONOM: a practical online compendium of file formats. *RLG DigiNews, 7*(5). Retrieved April 2, 2006, from http://www.rlg.org/preserv/ diginews/diginews7-5.html#feature2

Daniel, R. (1997, June). *A Trivial Convention for using HTTP in URN Resolution* (IETF RFC 2169). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2169.txt

Daniel, R., & Mealling, M. (1997, June). *Resolution of Uniform Resource Identifiers using the Domain Name System* (IETF RFC 2168). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2168.txt

Dempsey, L., & Lavoie, B. (2005, May 17). *DLF Service Framework for Digital Libraries* (A progress report for the DLF Steering Committee). Retrieved April 2, 2006, from http://www.diglib.org/architectures/serviceframe/dlfserviceframe1.htm

DeRose, S., Daniel, R., Jr., Grosso, P., Maler, E., Marsh, J., & Walsh, N. (Eds.). (2002, August 16). *XML Pointer Language (XPointer)* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/xptr/

DiLauro, T., Patton, M., Reynolds, D., & Sayeed, G. (2005, December). The Archive Ingest and Handling Test. The Johns Hopkins University Report. *D-Lib Magazine, 11*(12). Retrieved April 2, 2006, from http://dx.doi.org/ doi:10.1045/december2005-choudhury

Dublin Core Metadata Initiative Usage Board (2004, April). *DCMI Type Vocabulary* (DCMI Recommendation). Retrieved April 2, 2006, from http://dublincore.org/documents/ dcmi-type-vocabulary/

Dushay, N. (2001, December). *Using structural metadata to localize experience of digital content*. Retrieved April 2, 2006, from the Computing Research Repository (CoRR).

Dushay, N. (2002). Localizing experience of digital content via structural metadata. In *Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries, JCDL '02, Portland, OR* (pp. 244-252). New York, NY: ACM Press.

EPrints Free Software. (2005). Retrieved April 2, 2006, from http:// www.eprints.org/

Erdos, M., & Cantor, S. (2002). *Shibboleth-Architecture*. Retrieved April 2, 2006, from http://shibboleth.internet2.edu/docs/draft-internet2-shibboleth-arch-latest.pdf

Fallside, D. C. (Ed.). (2002, May 2). *XML Schema Part 0: Primer* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/xmlschema-0/

Fedora Project (2005a, January 26). *Introduction to Fedora Object XML (FOXML). Fedora Release 2.0*. Retrieved April 2, 2006, from http://www.fedora.info/download/2.0/ userdocs/ digitalobjects/introFOXML.html

Fedora Project (2005b, January 26). *Overview: The Fedora Digital Object Model. Fedora Release 2.0*. Retrieved April 2, 2006, from http://www.fedora.info/download/2.0/ userdocs/digitalobjects/objectModel.html

Fedora Project (2005c, January 28). *Fedora Content Versioning. Fedora Release 2.0.* Retrieved April 2, 2006, from http://www.fedora.info/download/2.0/userdocs/server/features/versioning.html

Fielding, R., Gettus, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., & Berners-Lee, T. (1999, June). *Hypertext Transfer Protocol -- HTTP/1.1* (IETF RFC 2616). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2616.txt

Freed, N., & Borenstein, N. (1996, November). *Multipurpose Internet Mail Extensions (MIME). Part Two: Media Types* (IETF RFC 2046). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2046.txt

Fielding, R., Gettus, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., & Berners-Lee, T. (1999, June). *Hypertext Transfer Protocol -- HTTP/1.1* (IETF RFC 2616). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2616.txt

Goland, Y., Whitehead, E., Faizi, A., Carter, S., & Jensen, D. (1999, February). *HTTP Extensions for Distributed Authoring -- WEBDAV* (IETF RFC 2518). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2518.txt

Grosso, P. (Ed.). (2003, September 12). *Proposal for XML Fragment Identifier syntax 0.9* (W3C Note). Retrieved April 2, 2006, from http://www.w3.org/TR/xml-fragid/

Gudgin, M., Hadley, M., Mendelsohn, N., Moreau, J.-J., & Nielsen, H. F. (Eds.). (2003, June 24). *SOAP version 1.2 part 1: messaging framework* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/soap12/

Gutteridge, C. (2002). GNU EPrints 2 Overview. In *Proceedings of the 11th Panhellenic Academic Libraries Conference*. Larissa, Greece.

Hakala, J., & Walravens, H. (2001, October). *Using International Standard Book Numbers as Uniform Resource Names* (IETF RFC 3187). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc3187.txt

Heery, R., & Anderson, S. (2005, February). *Digital Repositories Review* (A review prepared as background information for participants in the Joint Information Systems Committee (JISC) Digital Repositories Programme call). Retrieved April 2, 2006, from http://www.jisc.ac.uk/index.cfm?name=programme_digital_repositories

Hickey, T., Toves, J., & Young, J. (2005). *xISBN*. Retrieved April 2, 2006, from http://www.oclc.org/ research/projects/xisbn/

Hunter, J. (1999, September). MPEG-7: Behind the scenes. *D-Lib Magazine, 5*(9). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/september99-hunter

Hunter, J. (2001). An overview of the MPEG-7 Description Definition Language (DDL). *IEEE Transactions on Circuits and Systems for Video Technology, 11*(6), 765-772.

IEEE Learning Technology Standards Committee's, Learning Object Meta-data Working Group. (2002). *1484.12.1-2002 IEEE Standard for Learning Object Metadata* (IEEE Tech Rep.).

IMS Global Learning Consortium. (2003a, January). *IMS Digital Repositories version 1.0 Final specification*. Retrieved April 2, 2006, from http://www.imsglobal.org/

IMS Global Learning Consortium. (2003b, June). *IMS content packaging XML binding specification version 1.1.3*. Retrieved April 2, 2006, from http://www.imsglobal.org/content/packaging/

The International DOI Foundation. (2005, December). *The Digital Object Identifier Systems*. Retrieved April 2, 2006, from http://www.doi.org/

International Federation of Library Associations Study Group on the Functional Requirements for Bibliographic Records. (1998). Functional Requirements for Bibliographic Records. *UBCIM Publications-New Series, 19*. Retrieved April 2, 2006, from http://www.ifla.org/VII/s13/frbr/frbr.pdf

International Organization for Standardization. (2003a). *ISO 14721:2003. Space data and information transfer systems -- Open archival information system -- Reference model* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2003c). *ISO/IEC 21000-2:2003. Information technology -- Multimedia framework (MPEG-21) -- Part 2: Digital Item Declaration* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2003d). *ISO/IEC 21000-3:2003: Information technology -- Multimedia framework (MPEG-21) -- Part 3: Digital Item Identification* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2004a). *ISO 8601:2000: Data elements and interchange formats -- Information interchange -- Representation of dates and times* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2004b). *ISO/IEC TR 21000-1:2001: Information technology -- Multimedia framework (MPEG-21) -- Part 1: Vision, technologies and strategy* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2004c). *ISO/IEC 21000-5:2004: Information technology -- Multimedia framework (MPEG-21) -- Part 6: Rights Expression Language* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2004d). *ISO/IEC 21000-6:2004: Information technology -- Multimedia framework (MPEG-21) -- Part 6: Rights Data Dictionary* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2004e). *ISO/IEC 21000-7:2004: Information technology -- Multimedia framework (MPEG-21) -- Part 7: Digital Item Adaptation* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2005a). *ISO/IEC 21000-2:2005. Information technology -- Multimedia framework (MPEG-21) -- Part 2: Digital Item Declaration* (2nd ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2005b). *ISO/IEC 21000-9:2005. Information technology -- Multimedia framework (MPEG-21) -- Part 9: File Format* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization. (2006). *ISO/IEC 21000-10:2006. Information technology -- Multimedia framework (MPEG-21) -- Part 10: Digital Item Processing* (1st ed.) Geneva, Switzerland: Author.

International Organization for Standardization, Joint Technical Committee 1, Sub-Committee 34 (2004). *ISO/IEC FDIS 19757-3. Document Schema Definition Languages (DSDL) -- Part 3: Rule-based validation -- Schematron*. Retrieved April 2, 2006, from http://www.schematron.com/iso/Schematron.pdf

Jacobs, o., & Walsh, N. (Eds.). (2004, December 15). *Architecture of the World Wide Web, Volume One*. (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/webarch

Jerez, H. N., Liu, X., Hochstenbach, P., & Van de Sompel, H. (2004). The multi-faceted use of the OAI-PMH in the LANL repository. In *Proceedings of the 4th ACM/IEEE Joint Conference on Digital Libraries, JCDL '04, Tuscon, AZ* (pp. 11-20). New York, NY: ACM Press.

JISC e-Learning Programme - Technical Framework and Tools Strand. (2005, January 25). Retrieved April 2, 2006, http://www.jisc.ac.uk/index.cfm?name=elearning_ framework

JISC Information Environment. (2003, March3). Retrieved April 2, 2006, from http://www.jisc.ac.uk/ie/

JISC ITT. Linking UK Repositories Scoping Study. (2005, September 9). Retrieved April 2, 2006, from http://www.jisc.ac.uk/index.cfm?name=funding_repositoryservices

Josefsson, S. (Ed.). (2003, July). *The Base16, Base32, and Base 64 data encodings* (IETF RFC 3548) Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc3548.txt

Kahn, R., & Wilensky, R. (1995, May 13). *A framework for distributed digital object services*. Retrieved April 2, 2006, from http://hdl.handle.net/cnri.dlib/tn95-01

Koenen, R. (2001, December). *From MPEG-1 to MPEG-21: Creating an interoperable multimedia framework* (Output document of the 58th MPEG Meeting, Pattaya, Thailand, No ISO/IEC JTC1/SC29/WG11/N4518). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Lagoze, C. (1996, July/August). The Warwick framework: A container architecture for diverse sets of metadata. *D-Lib Magazine, 2*(7/8). Retrieved April 2, 2006, from http://hdl.handle.net/cnri.dlib/july96-lagoze

Lagoze, C., Lynch, C., & Daniel, R., Jr. (1996, June). *The Warwick Framework: A container architecture for aggregating sets of metadata* (Cornell University Tech. Rep. No. TR96-1593).

Lagoze, C., Payette, S., Shin, E., & Wilper, C. (in press). An architecture for complex objects and their relationships. *International Journal on Digital Libraries*.

Lagoze, C., Van de Sompel, H., Nelson, M. L., & Warner, S. (Eds.). (2002a, June 21). *Implementation guidelines for the Open Archives Initiative protocol for metadata harvesting (version 2.0): Specification and XML Schema for the OAI identifier format*. Retrieved April 2, 2006, from http://www.openarchives.org/OAI/2.0/guidelines-oai-identifier.htm

Lagoze, C., Van de Sompel, H., Nelson, M. L., & Warner, S. (Eds.). (2002b, June 14). *Implementation guidelines for the Open Archives Initiative protocol for metadata harvesting (version 2.0): XML Schema to hold provenance information in the 'about' part of a record*. Retrieved April 2, 2006, from http://www.openarchives.org/OAI/2.0/guidelines-provenance.htm

Lagoze, C., Van de Sompel, H., Nelson, M. L., & Warner, S. (2003a, September 26). *OAI-Rights White Paper*. Retrieved April 2, 2006, from http://www.openarchives.org/documents/OAIRightsWhitePaper.html

Lagoze, C., Van de Sompel, H., Nelson, M. L., & Warner, S. (Eds.). (2003b, February 21). *The Open Archives Initiative protocol for metadata harvesting* (2nd ed.). Retrieved April 2, 2006, from http://www.openarchives.org/OAI/2.0/openarchivesprotocol. htm

Leach, P., Mealling, M., & Salz, R. (2004, January). *A UUID URN Namespace* (3th ed.) (IETF Internet-Draft, expired on July 1, 2004).

The Libray of Congress. (2001). *Archival Information Package (AIP) design study* (LoC Tech. Rep. No. LC-DAVRS-07). Retrieved April 2, 2006, from http://www.loc.gov/rr/mopic/avprot/AIP-Study_v19.pdf

The Library of Congress: The Network Development and MARC Standards Office. (2005a, January). *Metadata Encoding and Transmission Standard (METS)*. Retrieved April 2, 2006, from http://www.loc.gov/Standards/mets/

The Library of Congress: The Network Development and MARC Standards Office. (2005b, April). *MARC 21 XML Schema (MARCXML)*. Retrieved April 2, 2006, from http://www.loc.gov/Standards/marcxml/

The Library of Congress: The Network Development and MARC Standards Office. (2005c, September). *Metadata Object Description Schema (MODS)*. Retrieved April 2, 2006, from http://www.loc.gov/Standards/mods/

The Library of Congress: The Network Development and MARC Standards Office. (2005d, November). *Search/Retrieve Web Service (SRW). Z39.50 International: Next Generation*. Retrieved April 2, 2006, from http://www.loc.gov/z3950/agency/zing/srw/

Liu, X., Balakireva, L., Hochstenbach, P., & Van de Sompel, H. (2005). File-Based Storage of Digital Objects and Constituent Datastreams: XMLtapes and Internet Archive ARC files. In A. Rauber, S. Christodoulakis & A. Min Tjoa (Vol. Eds.), *Lecture Notes in Computer Science: Vol. 3652 / 2005. Research and Advanced Technology for Digital Libraries: Proceedings of the 9th European Conference, ECDL '05, Vienna, Austria* (pp. 254-265). Heidelberg, Germany: Springer-Verlag.

Lynch, C. (1997, October). Identifiers and their role in networked information applications. *ARL newletter, 194*. Retrieved April 2, 2006, from http://www.arl.org/newsltr/194/identifier.html

Lynch, C. (2003, February). Institutional Repositories: Essential Infrastructure for Scholarship in the Digital Age. *ARL newsletter, 226*. Retrieved April 2, 2006, from http://www.arl.org/newsltr/226/ir.html

Marsh, J., & Orchard D. (Eds.). (2004, December 20). *XML Inclusion 1.0* (Xinclude) (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/xinclude/

McBride, B. (Series Ed.), Manola, F., & Miller, E. (Eds.). (2004, February 10). *RDF primer* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/RDF/

McLean, N. (2004). *The Ecology of repository services. A cosmic view* (A keynote presentation at Research and Advanced Technology for Digital Libraries: 8th European Conference, ECDL 2004, Bath, UK, September 12-17, 2004). Retrieved April 2, 2006, from http://www.ecdl2004.org/presentations/mclean/

McLean, N., & Lynch, C. (2004, May). *Interoperability between Library Information Services and Learning Environments. Bridging the Gaps* (A Joint White Paper on behalf of the IMS Global Learning Consortium and the Coalition for Networked Information). Retrieved April 2, 2006, from http://www.imsglobal.org/digitalrepositories/CNIandIMS_2004.pdf

Mills, D. (1996, October). *Simple Network Time Protocol (SNTP) version 4 for IPv4, IPv6 and OSI* (IETF RFC 2030). Retrieved April 2, 2006, from http://www.ietf.org/rfc/ rfc2030.txt

Moats, R. (1997, May). *URN Syntax* (IETF RFC 2141). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2141.txt

Mpeg21-DIDL distribution plug-in for DSpace Institutional Repositories. (2004, September). Retrieved April 2, 2006, from http://sourceforge.net/projects/didl-plug-in/

National Information Standards Organization. (2000, May). *ANSI/NISO Z39.84-2000: Syntax for the Digital Object Identifier*. Bethesda, MD: NISO Press.

National Information Standards Organization. (2001, September). *ANSI/NISO Z39.85-2001: The Dublin Core Metadata Element Set*. Bethesda, MD: NISO Press.

National Information Standards Organization. (2002, November). *ANSI/NISO Z39.50-2003: Information Retrieval (Z39.50): Application Service Definition and Protocol Specification*. Bethesda, MD: NISO Press.

National Information Standards Organization. (2005, April). *ANSI/NISO Z39.88-2004: The OpenURL Framework for Context-Sensitive Services*. Bethesda, MD: NISO Press.

Neslon, M. L. (2002). *Buckets: smart objects for digital libraries*. PhD Dissertation, Computer Science Department, Old Dominion University.

Nelson, M. L., Argue, B., Efron, M., Denn, S., & Pattuelli, M. (2001, December). *A survey of complex object technologies for digital libraries* (NASA Tech. Rep. NASA/TM-2001-

211426). Retrieved April 2, 2006, from http://techreports.larc.nasa.gov/ltrs/PDF/2001/tm/NASA-2001-tm211426.pdf

Nelson, M. L., & Maly, K. (2001a). Buckets: smart objects for digital libraries. *Communications of the ACM, 44*(5), 60-62.

Nelson, M. L., & Maly, K. (2001b, February). Smart objects and open archives. *D-Lib Magazine, 7*(2). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/ february2001-nelson

Nelson, M. L., Van de Sompel, H., Liu, X., Harrison, T. L., & McFarland, N. (2005). mod_oai: An Apache Module for Metadata Harvesting. In A. Rauber, S. Christodoulakis & A. Min Tjoa (Vol. Eds.), *Lecture Notes in Computer Science: Vol. 3652 / 2005. Research and Advanced Technology for Digital Libraries: Proceedings of the 9th European Conference, ECDL '05, Vienna, Austria* (pp. 509-510). Heidelberg, Germany: Springer-Verlag.

Nierman, J. (1996, March). *Major Milestone: Copyright Office Receives First Digital Deposit.* Library of Congress Information Bulletin, March 4 1996. Retrieved April 2, 2006, from http://www.loc.gov/loc/lcib/9604/cords.html

Olivier, B., Roberts, T., & Blinco, T. (2005, July). *The e-Framework for Education and Research. An Overview* (A Paper prepared on behalf of the Australian Government Department of Education, Science and Training (DEST), and the Joint Information Systems Committee (JISC) center for educational technology interoperability standards (CETIS)). Retrieved April 2, 2006, from http://www.e-framework.org/

Online Computer Library Center. (2004). *ERRoL Resolver*. Retrieved April 2, 2006, from http://www.oclc.org/research/software/oai/errol.htm

Online Computer Library Center. (2005a). *OAICat*. Retrieved April 2, 2006, from http://www.oclc.org/research/software/oai/cat.htm

Online Computer Library Center. (2005b). *OAIHarvester2*. Retrieved April 2, 2006, from http://www.oclc.org/research/software/oai/harvester2.htm

Online Computer Library Center. (2005c). *OpenURL 1.0*. Retrieved April 2, 2006, from http://www.oclc.org/research/software/openurl/

The OCLC/RLG Working Group on Preservation Metadata. (2002, June). *Preservation metadata and the OAIS information model: A metadata framework to support the preservation of digital objects* (OCLC/RLG Tech. Rep.). Retrieved April 2, 2006, from http://www.oclc.org/research/projects/pmwg/pm_framework.pdf

The OWL Services Coalition (n.d.). *OWL-S 1.0 Release*. Retrieved April 2, 2006, from http://www.daml.org/services/owl-s/1.0/

Ozzie, J., Moromisato, G., & Suthar, P. (2006, January) *Simple Sharing Extensions for RSS and OPML*. Retrieved April 2, 2006, from http://msdn.microsoft.com/xml/rss/sse/

Paskin, N. (1999). Towards Unique Identifiers. *Proceedings of the IEEE, 87(7)*, 1208-1227

Paskin, N. (2003, January). On making and identifying a "copy". *D-Lib Magazine, 9*(1). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/january2003-paskin

Pathways. Lifecycles for Information Integration in Distributed Scholarly Communication. (n.d.). Retrieved April 2, 2006, from http://www.infosci.cornell.edu/pathways/

Payette, S., & Lagoze, C. (1998). Flexible and Extensible Digital Object and Repository Architecture (FEDORA). In C. Nikolaou & C. Stephanidis (Vol. Eds.), *Lecture Notes in Computer Science: Vol. 1513 / 1998. Research and Advanced Technology for Digital Libraries: Proceedings of the 2nd European Conference, ECDL'98, Heraklion, Crete, Greece (pp. 41-60). Heidelberg, Germany*: Springer-Verlag.

Pereira, F., & Ebrahimi, T. (Eds.). (2002). *The MPEG-4 Book.* Upper Saddle River, NY: Prentice Hall PTR.

Powell, A. (2005, November). *A 'service oriented' view of the JISC Information Environment.* Retrieved April 2, 2006, from http://www.ukoln.ac.uk/distributed-systems/jisc-ie/arch/soa/

Preservation Metadata Implementation Strategies Working Group (2005, May). *Data Dictionary for Preservation Metadata: Final Report of the PREMIS Working Group* (OCLC/RLG Tech. Rep.). Retrieved April 2, 2006, from http://www.oclc.org/research/projects/pmwg/

Rescorla, E. (2000, May). *HTTP over TLS.* (IETF RFC 2818). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2818.txt

Rosenthal, D., Robertson, T., Lipkis, T., Reich, T., & Morabito, S. (2005, November). Requirements for Digital Preservation Systems. A Bottom-Up Approach. *D-Lib Magazine, 11*(11). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/ november2005-rosenthal

Rust, G., & Bide, M. (2000, June). *The <indecs> metadata framework: Principles, model and data dictionary* (<indecs> Tech. Rep. No. WP1a-006-2.0). Retrieved April 2, 2006, from http://www.indecs.org/pdf/framework.pdf

Shirky, C. (2005, December). AIHT. Conceptual Issues from Practical Tests. *D-Lib Magazine, 11*(12). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/ december2005-shirky

Smith, M. (2004, July 19). *DSpace Versioning Feature Summary.* Retrieved April 2, 2006, from http://simile.mit.edu/dspace-mit-docs/versioning.pdf

Smith, M., Barton, M., Bass, M., Branschofsky, M., McClellan, G., Stuve, D., et al. (2003, January). DSpace: An open source dynamic digital repository. *D-Lib Magazine, 9*(1). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/ january2003-smith

Smith, M. K., Welty, C., & McGuinness, D. L. (Eds.). (2004, February 10). *OWL Web Ontology Language guide* (W3C Recommendation). Retrieved April 2, 2006, from http://www.w3.org/TR/owl-guide/

Staples, T., Wayland, R., & Payette, S. (2003, April). The Fedora project: An open-source digital object repository system. *D-Lib Magazine, 9*(4). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/april2003-staples

Stone, L., (2006, January). *A Lightweight Network Interface (for CWSpace)*. Retrieved April 2, 2006, from http://wiki.dspace.org/LightweightNetworkInterface

Sun, S., Lannom, L., & Boesch, B. (2003, November). *Handle System overview* (IETF RFC 3650). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc3650.txt

Sun, S., Reilly, S., Lannom, L., & Petrone, J. (2003, November). *Handle System Protocol (ver 2.1) Specification* (IETF RFC 3652). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc3652.txt

Sutton, C., & Clayphan, R. (1997, March). *BIBLINK - LB 4034 - D5.1 Transmission of Data*. Retrieved April 2, 2006, from http://hosted.ukoln.ac.uk/biblink/wp5/d5.1.rtf

Tansley R., Bass, M., & Smith, M. (2003). DSpace as an Open Archival Information System. Current Status and Future Directions. *Lecture Notes in Computer Science: Vol. 2769. Research and Advanced Technology for Digital Libraries: Proceedings of the 4th European Conference, ECDL '00, Lisbon, Portugal* (pp. 446-460). Heidelberg, Germany: Springer-Verlag.

Tansley, R., Smith, M., & Walker, J. H. (2005). The DSpace Open Source Digital Asset Management System: Challenges and Opportunities. *Lecture Notes in Computer Science: Vol. 2769. Research and Advanced Technology for Digital Libraries: Proceedings of the 9th European Conference, ECDL '05, Vienna, Austria* (pp. 242-253). Heidelberg, Germany: Springer-Verlag.

Tourte, G., & Powell, A. (2004, March). *Encoding full-text links in the eprint jump-off page*. Retrieved April 2, 2006, from http://www.rdn.ac.uk/projects/eprints-uk/docs/encoding-fulltext-links/

van der Werf-Davelaar, T. (1999, September). Long-term Preservation of Electronic Publications. *D-Lib Magazine, 5*(9). Retrieved December 10, 2005, http://dx.doi.org/10.1045/september99-vanderwerf

Van de Sompel, H. (1999, June). Repositioning libraries in the digital age. *Preservation & Access International Newsletter (6)*. Retrieved December 10, 2005, http://www.clir.org/pubs/pain/pain06.html#repositioning

Van de Sompel, H., Hochstenbach, P., & Beit-Arie, O. (2000, May). *OpenURL syntax description, version OpenURL/1.0f - 2000-05-16*. Retrieved April 2, 2006, from http://alcme.oclc.org/openurl/docs/pdf/openurl-01.pdf

Van de Sompel, H., & Beit-Arie, O. (2001a, March). Open linking in the scholarly information environment using the OpenURL framework. *D-Lib Magazine, 7*(3). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/march2001-vandesompel

Van de Sompel, H., & Beit-Arie, O. (2001b, July/August). Generalizing the OpenURL framework beyond references to scholarly works: The Bison-Futé model. *D-Lib Maga-*

zine, 7(7/8). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/july2001-vandesompel

Van de Sompel, H., Bekaert, J., Liu, X., Balakireva, L., & Schwander, T. (2005). aDORe. A Modular, Standards-based Digital Object Repository. *The Computer Journal, 48*(5), 514-535

Van de Sompel, H., Hammond, T., Neylon, E., & Weibel, S. (2005, August). *The "info" URI scheme for information assets with identifiers in public namespaces* (4th ed.). (IETF RFC-vandesompel-info-uri-04.txt) Retrieved April 2, 2006, from http://www.ietf.org/internet-drafts/ draft-vandesompel-info-uri-04.txt

Van de Sompel, H., Nelson, M. L., Lagoze, C., & Warner, S. (2004, December). Resource Harvesting within the OAI-PMH Framework. *D-Lib Magazine, 10*(12). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/december2004-vandesompel

Van de Sompel, H., Payette, S., Erickson, J., Lagoze, C., & Warner, S. (2004, September). Rethinking scholarly communication. Building the system that scholars deserve. *D-Lib Magazine, 10*(9). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/september 2004-vandesompel

Van de Sompel, H., Young, J. A., & Hickey, T. B. (2003, July/August). Using the OAI-PMH ... differently. *D-Lib Magazine, 9*(7/8). Retrieved April 2, 2006, from http://dx.doi.org/10.1045/july2003-young

Watkinson, J. (2004). *The Mpeg handbook* (2nd ed.). Oxford, UK: Focal Press.

Watt, S., Huang, Z., Rodriguez, E., & Lauf, S. (Eds.). (2005, October). *ISO/IEC FDIS 21000-4 IPMP Components* (Output Document of the 74th MPEG Meeting, Nice, France, No. ISO/IEC JTC1/SC29/WG11/N7717). Retrieved April 2, 2006, from the NIST MPEG Document Register.

Williamson, S., Kosters, M., Blacka, D., Singh, J., & Zeilstra, K. (1997, June). *Referral Whois (RWhois) Protocol V1.5* (IETF RFC 2167). Retrieved April 2, 2006, from http://www.ietf.org/rfc/rfc2167.txt

Wolf, M., & Wicksteed, C. (1997, September 15). *Date and time formats* (W3C Note). Retrieved April 2, 2006, from http://www.w3.org/TR/NOTE-datetime

Publications

Publications

Journal articles listed in the ISI Journal Citation Index

Bekaert, J., Van De Ville, D., Rogge, B., De Kooning, E., & Van de Walle, R. (2002). Metadata-based access to architectural and historical archival collections: a review. *Aslib Proceedings: new information perspectives, 54*(6), 362-371.

Rogge, B., Bekaert, J., & Van de Walle, R. (2004). Timing issues in multimedia formats: review of the principles and comparison of existing formats. *IEEE Transactions on Multimedia, 6*(6), 910- 924.

Bekaert, J., De Kooning, E., & Van de Walle, R. (2005). Packaging models for the storage and distribution of complex digital objectsin archival information systems: a review of MPEG-21 DID principles. *Multimedia Systems, 10*(4), 286-301.

Van de Sompel, H., Bekaert, J., Liu, X., Balakireva, L., & Schwander, T. (2005). aDORe. A Modular, Standards-based Digital Object Repository. *The Computer Journal, 48*(5), 514-535.

Bekaert, J., De Kooning, E., & Van de Sompel, H. (2006). Representing Digital Assets using MPEG-21 Digital Item Declaration. *International Journal on Digital Libraries, 6*(2), 159-173.

Articles in international magazine

Bekaert, J., Hochstenbach, P., & Van de Sompel, H. (2003, November). Using MPEG-21 DIDL to represent complex digital objects in the Los Alamos National Laboratory Digital Library. *D-Lib Magazine, 9*(11). Available at http://dx.doi.org/10.1045/november2003-bekaert

Bekaert, J., Balakireva, L., Hochstenbach, P., & Van de Sompel, H. (2004, February). Using MPEG-21 and NISO OpenURL for the dynamic dissemination of complex digital objects in the Los Alamos National Laboratory Digital Library. *D-Lib Magazine, 9*(11). Available at http://dx.doi.org/10.1045/february2004-bekaert

Bekaert, J., & Van de Sompel, H. (2005, June). A standards-based solution for the accurate transfer of digital assets. *D-Lib Magazine, 11*(6). Available at http://dx.doi.org/10.1045/june2005-bekaert

Bekaert, J., Xiaoming, L., & Van de Sompel, H. (2005). aDORe, a modular and standards-based Digital Object repository at the Los Alamos National Laboratory. *IEEE Technical Committee on Digital Libraries 2*(1).

Articles in conference proceedings

Bekaert, J., De Kooning, E., & Van de Walle, R. (2001, December). *Data processing and research into architectural history*. In *Proceedings of the 2nd FTW PhD Symposium, Ghent University, Ghent, Belgium*.

Bekaert, J., De Kooning, E., & Van de Walle, R. (2002, December). *Metadata-based access to multimedia architectural and historical archive collections*. In *Proceedings of the 3th FTW PhD Symposium, Ghent University, Ghent, Belgium*.

Bekaert, J., Van de Ville, D., Rogge, B., Lerouge, S., De Sutter, R., De Kooning, E., & Van de Walle, R. (2002). Metadata-based access to multimedia architectural and historical archive collections. In J. R. Smith, S. Panchanathan, & T. Zhang (Vol. Eds.), In *Proceedings of SPIE: 4862: Internet Multimedia Management Systems III* (pp. 22-29). Bellingham, WA: SPIE Press.

De Sutter, R., Lerouge, S., Bekaert, J., Rogge, B., Van De Ville, D. & Van de Walle, R. (2002) Dynamic adaptation of multimedia data for mobile applications. In J. R. Smith, S. Panchanathan, & T. Zhang (Vol. Eds.), In *Proceedings of SPIE: 4862: Internet Multimedia Management Systems III* (pp. 240-248). Bellingham, WA: SPIE Press.

Lerouge, S., Rogge, B., De Sutter, R., Bekaert, J., Van De Ville, D. & Van de Walle, R. (2002) A generic mapping mechanism between content description metadata and user environments. In J. R. Smith, S. Panchanathan, & T. Zhang (Vol. Eds.), In *Proceedings of SPIE: 4862: Internet Multimedia Management Systems III* (pp. 12-21). Bellingham, WA: SPIE Press.

Rogge, B., De Sutter, R., Bekaert, J. and Van de Walle, R. (2002) An analysis of multimedia formats for content description. In J. R. Smith, S. Panchanathan, & T. Zhang (Vol. Eds.), In *Proceedings of SPIE: 4862: Internet Multimedia Management Systems III* (pp. 1-11). Bellingham, WA: SPIE Press.

De Sutter, R., Lerouge, S., Bekaert, J. & Van de Walle, R. (2003) Dynamic adaptation of streaming MPEG-4 video for mobile applications. In Furnell, S. & Dowland, P. (Vol. Ed.), In *Proceedings of EuroMedia 2003, Plymouth, England* (pp. 185-190). Ghent, Belgium: Eurosis.

Bekaert, J., De Sutter, R., De Kooning, E., & Van de Walle, R. (2003). Metadata-based access to complex digital objects in multimedia archival collections. In Furnell, S. & Dowland, P. (Vol. Ed.), In *Proceedings of EuroMedia 2003, Plymouth, England* (pp. 10-13). Ghent, Belgium: Eurosis.

Bekaert, J., Hochstenbach, P., De Kooning, E., & Van de Walle, R. (2003). An analysis of packaging formats for complex digtal objects: review of principles. In J. R. Smith, S. Panchanathan, & T. Zhang (Vol. Eds.), In *Proceedings of SPIE: 5242: Internet Multimedia Management Systems IV* (pp. 1-11). Bellingham, WA: SPIE Press.

De Keukelaere, F., DeMartini, T., Bekaert, J., & Van de Walle, R. (2005). Supporting rights checking in an MPEG-21 Digital Item Processing environment. In *Proceedings of the IEEE International Conference on Multimedia & Expo, ICME 2005*.

Bekaert, J., Xiaoming, L., & Van de Sompel, H. (2005). aDORe, a modular and standards-based Digital Object repository at the Los Alamos National Laboratory. In *Proceedings of the 5th ACM/IEEE Joint Conference on Digital libraries, JCDL '05, Denver, CO* (p. 367). New York, NY: ACM Press.

Bekaert, J., Xiaoming, L., & Van de Sompel, H. (2005). Representing Digital Objects for long-term preservation using MPEG-21 DID. In *Proceedings of PV-2005: Ensuring the Long-Term Preservation and Adding Value to the Scientific and Technical Data*.

Bekaert, J., Xiaoming, L., & Van de Sompel, H. (2005). Using standards in digital library design and development. In *Proceedings of the 5th ACM/IEEE Joint Conference on Digital libraries, JCDL '05, Denver, CO* (p. 423). New York, NY: ACM Press.

Bekaert, J., & Van de Sompel, H. (2005). Access Interfaces for Open Archival Information Systems based on the OAI-PMH and the OpenURL Framework for Context-Sensitive Services. In *Proceedings of PV-2005: Ensuring the Long-Term Preservation and Adding Value to the Scientific and Technical Data*.

Bekaert, J., Liu, X., Van de Sompel, H., Lagoze, C., Payette, S., & Warner, S. (in press). *Pathways Core. A Content Model for Cross-Repository Services*. Submitted to the Joint Conference on Digital Libraries ’06.

Contributions to ISO/IEC JTC1 SC29/WG11

Bekaert, J., Hochstenbach, P., De Neve, W., Van de Sompel, H., & Van de Walle, R. (2003, July). *Suggestions concerning MPEG-21 Digital Item Declaration* (Input Document for the 65th MPEG Meeting, Trondheim, Norway, No. ISO/IEC JTC1/SC29/WG11/M9755). Available at the NIST MPEG Document Register.

Bekaert, J., Hochstenbach, P., De Keukelaere, F., Van de Sompel, H., & Van de Walle, R. (2003, December). *Suggestions concerning MPEG-21 Digital Item Declaration* (Input Document for the 67th MPEG Meeting, Hawaii, No. ISO/IEC JTC1/SC29/WG11/M10427). Available at the NIST MPEG Document Register.

De Keukelaere, F., Bekaert, J., Hochstenbach, P., De Sutter, R., Van de Sompel, H., & Van de Walle, R. (2003, December). *Issues related to the inclusion of DIP information in DIDs* (Input Document for the 67th MPEG Meeting, Hawaii, No. ISO/IEC JTC1/SC29/WG11/M10424). Available at the NIST MPEG Document Register.

Bekaert, J., De Keukelaere, F., De Sutter, R., & Van de Walle, R. (on behalf of the Belgian National Standards Body) (2004, March). *BNB comments on ISO/IEC 21000-10 CD* (Input Document for the 68th MPEG Meeting, Munich, Germany, No. ISO/IEC JTC1/SC29/WG11/M10797be). Available at the NIST MPEG Document Register

De Keukelaere, F., Bekaert, J., & Van de Walle, R. (2004, March). *Comments on ISO/IEC 21000-2 WD v2.0 - Digital Item Declaration 2nd edition* (Input Document for the 68th MPEG Meeting, Munich, Germany, No. ISO/IEC JTC1/SC29/WG11/M10611). Available at the NIST MPEG Document Register.

Bekaert, J., De Keukelaere, F., Van de Sompel, H., & Van de Walle, R. (2004, July). *Requirement for MPEG-21 DID* (Input Document for the 69th MPEG Meeting, Redmond, WA, No. ISO/IEC JTC1/SC29/WG11/M11081). Available at the NIST MPEG Document Register.

Bekaert, J., De Keukelaere, F., & Van de Walle, R. (on behalf of the Belgian National Standards Body) (2004, July). *BNB comments on ISO/IEC 21000-2 CD - Digital Item Declaration 2nd edition* (Input Document for the 69th MPEG Meeting, Redmond, WA, No. ISO/IEC JTC1/SC29/WG11/M10889be). Available at the NIST MPEG Document Register.

Bekaert, J., DeMartini, T., Van de Walle, R., & Van de Sompel, H. (2004, July). *Identification of a DIDL Document* (Input Document for the 69th MPEG Meeting, Redmond, WA, No. ISO/IEC JTC1/SC29/WG11/M10892). Available at the NIST MPEG Document Register.

Bekaert, J., De Keukelaere, F., De Sutter, R., & Van de Walle, R. (on behalf of the Belgian National Standards Body) (2004, July). *Preliminary BNB comments on ISO/IEC 21000-10 CD - Digital Item Processing* (Input Document for the 69th MPEG Meeting, Redmond, WA, No. ISO/IEC JTC1/SC29/WG11/M10621). Available at the NIST MPEG Document Register.

DeMartini, T., Van de Sompel, H., & Bekaert, J. (2004, July). *Status of mpegRA activity* (Input Document for the 69th MPEG Meeting, Redmond, WA, No. ISO/IEC JTC1/SC29/WG11/M11077). Available at the NIST MPEG Document Register.

Bekaert, J. (on behalf of the Belgian National Standards Body) (2004, October). *Preliminary BNB comments on ISO/IEC 21000-2 2nd edition FCD* (Input Document for the 70th MPEG Meeting, Palma de Mallorca, Spain, No. ISO/IEC JTC1/SC29/WG11/M11355). Available at the NIST MPEG Document Register.

De Keukelaere, F., DeMartini, T., Bekaert, J. & Van de Walle, R. (2004, October). *An object oriented approach for the DIBO APIs* (Input Document for the 70th MPEG Meeting, Palma De Mallorca, Spain, No. ISO/IEC JTC1/SC29/WG11/M11338). Available at the NIST MPEG Document Register.

Bekaert, J., De Keukelaere, F., De Zutter, S., & Van de Walle, R. (on behalf of the Belgian National Standards Body) (2005, January). *Preliminary BNB comments on ISO/IEC 21000-10 FCD* (Input Document for the 71st MPEG Meeting, Hong Kong, China, No. ISO/IEC JTC1/SC29/WG11/M11677). Available at the NIST MPEG Document Register.

Bekaert, J., Van de Sompel, H., Rump, N., Barlas, C., & Van de Walle, R. (2005, January). *Typing Identifiers* (Input Document for the 71st MPEG Meeting, Hong Kong, China, No. ISO/IEC JTC1/SC29/WG11/M11581). Available at the NIST MPEG Document Register.

Bekaert, J. (on behalf of the Belgian National Standards Body) (2005, January). *BNB comments on ISO/IEC 21000-2 FCD 2nd edition* (Input Document for the 71st MPEG Meeting, Hong Kong, China, No. ISO/IEC JTC1/SC29/WG11/M11355). Available at the NIST MPEG Document Register.

Bekaert, J., Van de Walle R., & Van de Sompel, H. (2005, January). *Using MPEG-21 technology in the Digital Archiving and Preservation domain* (Input Document for the 71st MPEG

Meeting, Hong Kong, China, No. ISO/IEC JTC1/SC29/WG11/M11515). Available at the NIST MPEG Document Register.

Bekaert, J. (on behalf of the Belgian National Standards Body) (2005, April). *BNB comments on ISO/IEC FCD 21000-10* (Input Document for the 71st MPEG Meeting, Hong Kong, China, No. ISO/IEC JTC1/SC29/WG11/M11355). Available at the NIST MPEG Document Register.

Bekaert, J., DeMartini, T., Rump, N., Barlas, C., Van de Sompel, H., & Van de Walle, R. (2005, April). *URI-reference of a DIDL document* (Input Document for the 72nd MPEG Meeting, Busan, Korea, No. ISO/IEC JTC1/SC29/WG11/M11943). Available at the NIST MPEG Document Register.

Bekaert, J., De Keukelaere, F., & Van de Walle, R. (2005, April). *Declaring DIMs and DIXOs using MPEG-21 DID* (Input Document for the 72nd MPEG Meeting, Busan, Korea, No. ISO/IEC JTC1/SC29/WG11/M11890). Available at the NIST MPEG Document Register.

Bekaert, J., & Rump, N. (2005, July). *Editors input towards DoC on PDAM1 of 21000-3.* (Input Document for the 73th MPEG Meeting, Poznan, poland, No. ISO/IEC JTC1/SC29/WG11/M12160). Available at the NIST MPEG Document Register.

Bekaert, J., & Rump, N. (2005, July). *Editors input towards FPDAM1 of 21000-3.* (Input Document for the 73th MPEG Meeting, Poznan, poland, No. ISO/IEC JTC1/SC29/WG11/M12161). Available at the NIST MPEG Document Register.

Bekaert, J., & Van de Sompel, H. (on behalf of NISO) (2005, October). *Liaison Statement from NISO. MPEG-21 DII mpegRA and NISO "info" URI* (Input Document for the 74th MPEG Meeting, Nice, France, No. ISO/IEC JTC1/SC29/WG11/M12536). Available at the NIST MPEG Document Register.

Bekaert, J., & Van de Sompel, H. (on behalf of NISO) (2006, January). *Liaison Statement from NISO. NISO 'info' URI published as RFC* (Input Document for the 75th MPEG Meeting, Bangkok, Thailand, No. ISO/IEC JTC1/SC29/WG11/M12940). Available at the NIST MPEG Document Register.

Bekaert, J., & Van de Sompel, H. (on behalf of NISO) (2006, January). *Liaison Statement from NISO. NISO request for patent information* (Input Document for the 75th MPEG Meeting, Bangkok, Thailand, No. ISO/IEC JTC1/SC29/WG11/M13020). Available at the NIST MPEG Document Register.