D-Lib Magazine
June 2005

Volume 11 Number 6

ISSN 1082-9873

A Standards-based Solution for the Accurate Transfer of Digital Assets

Jeroen Bekaert
Los Alamos National Laboratory, Research Library
Ghent University, Faculty of Engineering
<jbekaert@lanl.gov>

Herbert Van de Sompel
Los Alamos National Laboratory, Research Library
<herbertv@lanl.gov>

Abstract

This article describes results of a collaboration between the Research Library of the Los Alamos National Laboratory (LANL) and the American Physical Society (APS) aimed at designing and implementing a robust solution for the recurrent transfer of digital assets from the APS collection to LANL. In this solution, various recent standards are combined to obtain an asset transfer framework that should be attractive as a means to optimize content transfer in environments beyond the specific APS/LANL project. The proposed solution uses an XML-based complex object format (the MPEG-21 Digital Item Declaration Language) for the application-neutral representation of compound digital assets of all sorts. It uses a pull-oriented HTTP-based protocol (the Open Archives Initiative Protocol for Metadata Harvesting) that allows incrementally collecting new and updated assets, represented as XML documents, from a producing archive. It builds on an XML-specific technique (W3C XML Signatures) to provide guarantees regarding authenticity and accuracy of the transferred assets.

1. Introduction

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. Hence, it does not come as a surprise that, during the last few years, a growing interest in the standardization of content transfer frameworks can be observed. Several use cases motivate this need for standardization:

a. The transfer of content from information producers to parties that provide discovery-oriented services over the information collections. Examples include 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, etc.
b. The transfer of content from information producers to parties that provide digital preservation services. Examples include the transfer of collections to facilities that mirror the content to guarantee the existence of safety copies, and the transfer of content to services charged with content migration. [10]

c. The submission of content to a government agency for an official purpose such as the registration of copyright, as was pioneered in the CORDS effort [29].

Various projects have explored the possible standardization of the content transfer from publishers to libraries. The Networked European Deposit Library (NEDLIB) project (in 2000) [34] aimed at defining a workflow for ingesting, storing and accessing content in the context of deposit systems for electronic publications, while the BIBLINK project (in 1997) [31] focused on establishing authoritative rules for metadata transfer between publishers of electronic materials and national bibliographic agencies. Based on a thorough examination of the existing practices and enabling technologies, both projects concluded that it was unlikely that all content formats available on the market could be transferred through a single, standardized framework. As a result, the BIBLINK project recognized the existence of a heterogeneous environment by identifying a rather extensive list of formats and network protocols that should be supported by a national archive to facilitate metadata transfer 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 format required by the deposit system. Both projects felt that, under the existing circumstances, aiming for a single transfer framework was unrealistic. The NEDLIB report formulates this as follows:

The "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 content 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 [20] 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. Since the finalization of these projects, several new technologies have emerged that warrant revisiting the conclusions that were reached. As will be shown in this article, a framework built on the combination of such new technologies may bring us closer to having the ability to devise a standards-based content transfer framework.

Being a large-scale aggregator of published content, the Research Library of the Los Alamos National Laboratory (LANL) [note 1] has extensive experience with the significant overhead caused by the lack of standards in existing content transfer solutions. Therefore, it should come as no surprise that the Digital Library Research and Prototyping Team at LANL has 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) [note 2]. Under the terms of that agreement, LANL will mirror the complete 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. The project is partly funded by a grant from the Library of Congress's National Digital Information Infrastructure and Preservation Program (NDIIPP) [note 3]. 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. The core requirements for the transfer framework were formulated as follows:

Requirements regarding the content transfer mechanism:

- 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 independent from 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 independent of specific repository architectures.
- Reciprocity of the content transfer solution must exist: It must be possible for the producing archive to retrieve assets back from the consuming archive.

Requirement regarding accuracy and authenticity of the transfered content in light of digital preservation:

- The assets retrieved 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 Digital Item Declaration (MPEG-21 DID) [17,18], the MPEG-21 Digital Item Identification (MPEG-21 DII) [5,19], the W3C XML Signature Syntax and Processing standard [2], and the Open Archive Initiative Protocol for Metadata Harvesting (OAI-PMH) [23]. The core characteristics of the solution are:

To meet the requirements regarding the content transfer mechanism:

- Using the MPEG-21 DID Abstract Model that specifies a data model for the representation of digital assets as the "crosswalk data model" for the transfer of assets. 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 Digital Item Declaration Language (MPEG-21 DIDL). This XML document contains (pointers to) constituent datastreams of the asset, metadata pertaining to the asset, and the identifier of the 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.
- Including XML Signatures in the XML documents. These facilitate verification of authenticity and integrity of the constituent datastreams of the assets once they are collected by the consuming archive.

2. An OAIS perspective on the content transfer solution

Each asset of the APS collection coincides with a 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. The Kahn/Wilensky framework [20] refers to such assets as Digital Objects. In the OAIS Reference Model [16], one may approximately equate an asset with the concept of a Content Information object. A Content Information object is a set of information that is the original target of preservation in an OAIS environment, and it is comprised of one or more Data Objects (or a constituent datastream) as well as secondary information related to the representation of these Data Objects. The identifier pertaining to a Content Information object is referred to as a Content Information Identifier. A Content Information object itself is encapsulated in a so-called Information Package, a container that holds and binds the various components making up the Content Information object. The OAIS Reference Model recognizes three subtypes of the Information Package: the Archival Information Package (OAIS AIP), the Submission Information Package (OAIS SIP), and the Dissemination Information Package (OAIS DIP). The definitions of these package types are based on the function of the archival process, which uses the package, and the translation from one package to another as it passes through the archival process. It is necessary to distinguish between an Information Package that is preserved by an OAIS and Information Packages that are submitted to, and disseminated from, an OAIS. This distinction is needed to reflect the reality that some submissions to a repository will have insufficient information to meet final requirements of that repository. In addition, different repositories may organize their content very differently, and hence, may warrant different environment-specific information to be contained within the archival packages.

In accordance with the OAIS Reference Model, the terms 'producing archive' and 'consuming archive' will be used throughout the article, 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. Indeed:

- As shown in Figure 1, the producing archive exposes OAIS DIPs through an OAI-PMH interface (2 in Figure 1). Those exposed OAIS DIPs result from – dynamically – mapping the assets in the producing archive (packaged in AIP_1 in Figure 1) 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 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 (2 in Figure 1) disseminated by the producing archive become OAIS SIPs to the consuming archive (3 in Figure 1). 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 1).
- The consuming archive itself may re-expose the retrieved assets using the same technique, thereby allowing the producing archive, as well as other archives to incrementally collect assets.

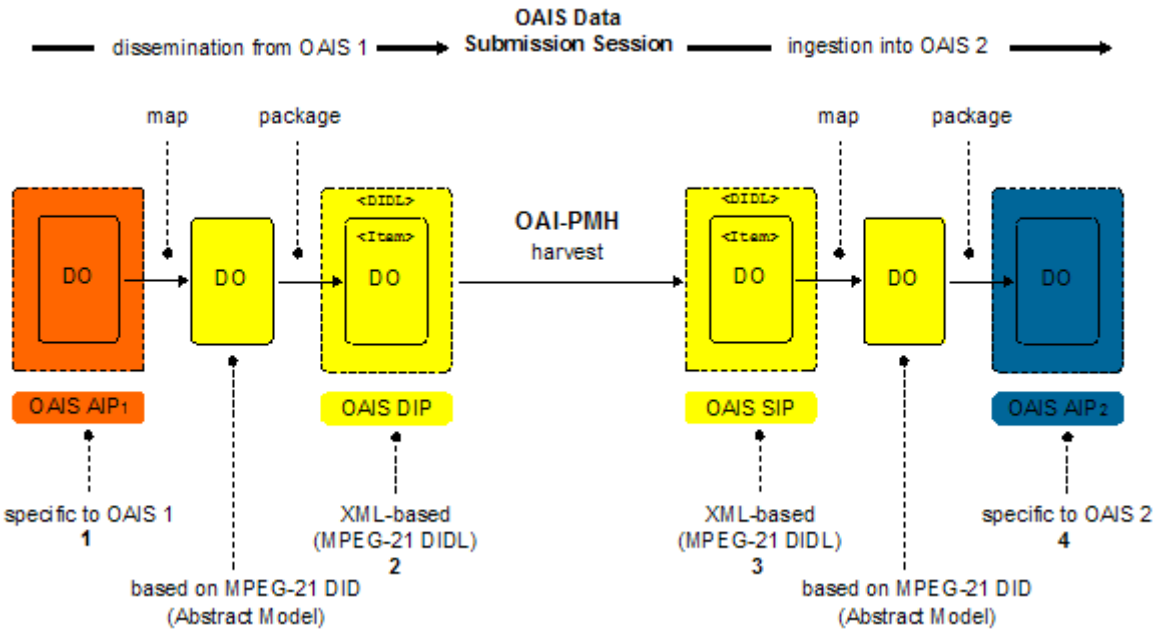

Figure 1: An OAIS perspective on content transfer between the producing archive and the consuming archive using the OAI-PMH.

3. Exposing OAIS DIPs from the producing archive

This Section describes the approach used to expose the 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 an asset of the producing archive as an application-neutral, XML-based OAIS DIP of that asset. The 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:

- W3C XML Signatures embedded in an OAIS DIP to allow verifying the integrity and authenticity of constituent datastreams of a represented asset.
- W3C 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.

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

An asset created by the APS typically coincides with an APS 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 Digital Object Identifier [27], which the OAIS Reference Model categorizes as a Content Information Identifier. In the remainder of this article, a sample APS asset will be used to illustrate several design choices made. The main characteristics of the sample asset are given in Table 1.

	Type	MIME	Content Information Identifier	Network location
APS Asset	publication	N/A	doi:10.1103/PhysRevB.69.174413	—
Constituent Datastream 1	metadata record	application/xml	—	http://oai.aps.org/filefetch?identifier=PhysRevB.69.174413&description=apsmeta
Constituent Datastream 2	fulltext file	application/pdf	—	http://oai.aps.org/filefetch?identifier=PhysRevB.69.174413&description=print

Table 1. The sample APS asset.

In order to be able to use the OAI-PMH to transfer assets between the producing and the consuming archives, they must be represented as XML. The compound nature of the assets requires a representation by means of a complex object format. Various candidate formats exist, including MPEG-21 DIDL [17,18], IMS-CP [14], METS [25] and XFDU [11], and of those, MPEG-21 DIDL was selected. MPEG-21 DIDL is the XML-based instantiation of the data model (or Abstract Model) for assets, as defined by 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 aDORe repository environment [3,4,32], and from being actively involved in its ISO/MPEG standardization. Other, objective motivators, of importance to the described data transfer problem were:

- The existence of the MPEG-21 DID Abstract Model that underlies MPEG-21 DIDL. For the asset transfer problem, this Abstract Model provides valuable guidance regarding how to map an asset from the producing archive to an intermediate representation (called a Digital Item) compliant with the standardized MPEG-21 DID Abstract Model, and how to map from that intermediate representation to an asset in the consuming archive. Indeed, the MPEG-21 DID Abstract Model can function as the standard-based cross-walk between the data models for assets as used by various archives.
- The combination of MPEG-21 DID with the MPEG-21 Digital Item Identification (MPEG-21 DII) [19] results in an unambiguous way to map and convey Content Information Identifiers of assets and their constituent datastreams. In light of the data transfer problem, this is a core feature that is specified rather ambiguously in other complex object formats.
- MPEG-21 DID is an ISO standard. Using standards is deemed necessary in the context of the data

transfer, in which cross-repository and cross-community communication is required.

- The XML Schema that defines MPEG-21 DIDL is surprisingly elegant and simple, making the development of compliant tools achievable.

An asset represented according to the MPEG-21 DIDL XML syntax is packaged in a so-called DIDL document. 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. An example of a DIDL document resulting from the mapping and packaging of the sample APS asset is provided in Annex A. The core features of the mapping and packaging process are explained here and illustrated in Table 2. For simplicity, the explanation is given in terms of the XML elements of the package. Each such XML element corresponds to an Entity of the Abstract Model defined by MPEG-21 DID. The XML elements are shown in the `courier` font:

	MPEG-21 DIDL elements		Secondary info type	Value
1	`Item`			
2	`Descriptor/Statement`		`dii:identifier`	doi:10.1103/PhysRevB.69.174413
3	`Component`			
4	`Descriptor/Statement`		`dsig:Signature`	XML Signature pertaining to (6)
5	`Descriptor/Statement`		`dcterms:created`	2004-05-18T15:43:33Z
6	`Resource`	`@mimeType`		application/xml; charset=UTF-8
		`@ref`		http://oai.aps.org/filefetch?identifier=PhysRevB.69.174413&description=apsmeta
7	`Component`			
8	`Descriptor/Statement`		`dsig:Signature`	XML Signature pertaining to (10)
9	`Descriptor/Statement`		`dcterms:created`	2004-05-18T15:43:33Z
10	`Resource`	`@mimeType`		application/pdf
		`@ref`		http://oai.aps.org/filefetch?identifier=PhysRevB.69.174413&description=print

Table 2. An MPEG-21 DIDL perspective of the sample APS asset.

- In accordance with the MPEG-21 DID Abstract Model, an asset from the producing archive is mapped to a top-level DIDL `Item` element. Constituents of the asset are provided in child elements of this top-level `Item`. Hence, the sample APS asset is represented as an `Item`. (See also Row 1 of Table 2.)
- In DIDL, secondary information pertaining to an entity of the MPEG-21 DID Abstract Model are conveyed using `Descriptor/Statement` constructs attached to the entity. Hence, such a `Descriptor/Statement` construct is attached to the aforementioned top-level `Item` to convey the Content Information Identifier of the APS asset. This `Descriptor/Statement` construct is compliant with the MPEG-21 DII specification. The Content Information Identifier of the sample APS publication is 'doi:10.1103/PhysRevB.69.1744133'. (See also row 2 of Table 2.) If required, other `Descriptor/Statement` constructs can be attached to the `Item`.
- In accordance with the MPEG-21 DID Abstract Model, the structure of the 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 a 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 asset itself has a Content Information Identifier, and constituents don't. Hence, as can be seen in the example of Annex A (see also rows 3 and 7 of Table 2), all constituents of the asset are provided as `Component/Resource` constructs that are direct children of the top-level `Item`.

- In accordance with the MPEG-21 DID Abstract Model, a constituent datastream of an asset is always conveyed in a `Component/Resource` construct. Two approaches exist:

    - By-Reference: The network location of the constituent datastream of the asset is provided as the value of the `ref` attribute of the `Resource` element.
    - By-Value: The constituent datastream is provided by base64 encoding the binary data, wrapping the result in the `Resource` element and adding an `encoding` attribute with a value set to 'base64'.

  In both approaches, the `mimeType` attribute of the `Component` element specifies the MIME media type and subtype of the constituent datastream. Combining the `mimeType` attribute and the `contentEncoding` 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 project, because of performance reasons, several datastreams are being compressed using the 'GNU Zip Compression' algorithm before transfer both By-Value and By-Reference, or providing different network-locations of the same datastream. Furthermore, multiple Resource elements provided in the same Component are by definition considered to be bit-equivalent. This allows, for example, providing a datastream both By-Value and By-Reference, or providing different network-locations of the same datastream.

  As can be seen from Annex A, the constituent datastreams of the sample APS asset are provided By-Reference and have a MIME type 'application/xml' and 'application/pdf' respectively. (See also rows 6 and 10 of Table 2.) The network location of the metadata record is 'http://oai.aps.org/filefetch?identifier=PhysRevB.69.174413&description=apsmeta'. The PDF file can be found at 'http://oai.aps.org/filefetch?identifier=PhysRevB.69.174413&description=print'.

- As will be described in Section 3.2, 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. (See rows 4 and 8 of Table 2.)
- Other `Descriptor/Statement` constructs can be attached to each `Component/Resource` construct, as required. For example, each `Component` may have a `Descriptor/Statement` construct that conveys the creation datetime of the constituent datastream contained inside the `Component/Resource`. Rows 5 and 9 of Table 2 show the use of the dcterms:created element to convey the creation datetime of the metadata record and the PDF datastream of the sample APS asset. Information related to the digital preservation of an asset could eventually be conveyed by elements from the PREMIS effort [note 4]. Also, investigations are ongoing aimed at documenting the genre, i.e., the form of the intellectual content or the presentation style of a datastream. For instance, one PDF datastream may represent an abstract of an article, while another may be a bibliography. The rendering software may not care about the difference between both, but applications and end-users of the content do. These values could be expressed using controlled vocabularies, such as the DCMI Type Vocabulary [12] and PRISM [15].
- 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 asset.

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

When transferring assets packaged in a DIDL document 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 project. We next explore the technologies used to achieve these goals.

3.2.1. 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 [2,30]:

- ***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 DIDL XML documents are transferred in the proposed data transfer solution, usage of the XML Signature & Processing specification [2] has been explored and adopted. This 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 3.2.2 . 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.

3.2.2. 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 information 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 category of the OAIS 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:

- 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. 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 retrieved from the `URI` attribute, but over the result of applying the specified transforms to that retrieved datastream. As such, the order in which the transforms are applied is important. The input to the first transform is the data retrieved 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 project (see columns 3 to 6 of Table 3):

    - XPath Transform [8] (Column 3). The main purpose of this transform is to provide a technique for computing 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 differerent 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 [6,7] (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 project, a base64-encoding algorithm is used to embed a binary datastream (Rows 2 and 3 of Table 3) in a `Resource` element (see also Section 3.1). 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 uncompression, that is used as a modifier to the datastream that is being transferred (Rows 2 and 4 of Table 3). 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 and registered by the Internet Assigned Numbers Authority (IANA).

	1	2	3	4	5	6	7

	Datastream Provision	Datastream Media Type	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 3. Use of W3C XML Signatures to sign datastreams referenced or embedded in a DIDL document.

- 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 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 4 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 XML Signature syntax follows the scenario outlined in Row 5 of Table 3.

```
<didl:Component id="uuid-279ee2d4-965c-11d9-81d6-ab2d295f8855">
  <didl:Descriptor>
    <didl:Statement mimeType="application/xml; charset=UTF-8">
      <dsig:Signature xmlns:dsig="http://www.w3.org/2000/09/xmldsig#"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
          <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?
                identifier=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>
    </didl:Statement>
```

```
      </didl:Descriptor>
      <didl:Resource mimeType="application/pdf" ref="http://oai.aps.org/
        filefetch?identifier=PhysRevB.69.174413&description=print"/>
    </didl:Component>
```

Table 4. W3C XML Signature pertaining to the PDF datastream of the sample asset

3.3. Exposing OAIS DIPs via the OAI-PMH

3.3.1. Use of complex object formats with the OAI-PMH

The Open Archives Protocol for Metadata Harvesting (OAI-PMH) [23] has been widely adopted as an approach to facilitate discovery of distributed resources. The OAI-PMH achieves this by providing a simple, yet powerful, framework for metadata harvesting. Harvesters can incrementally gather metadata records contained in distributed OAI-PMH repositories and use them to create services covering the content of those repositories.

Due to its origin in the realm of resource discovery, the OAI-PMH mandates the support of the Dublin Core [28] metadata format, but strongly encourages supporting more expressive formats. In essence, any metadata format that is defined by means of an XML Schema [13] can be used to describe resources in the OAI-PMH Framework. In typical use cases, metadata exposed by OAI-PMH repositories is descriptive, and it is expressed by means of metadata formats of varying complexity, such as simple Dublin Core or MARCXML [24]. In the described content transfer solution, the OAI-PMH is used to harvest metadata that are highly expressive and accurate in their representation of assets. Such metadata formats are typically referred to as complex object formats.

Introducing complex object formats as metadata formats in the OAI-PMH framework yields a robust and general solution to the resource-harvesting problem [33]. Especially, unlike other approaches aimed at gathering resources (not just metadata) based on OAI-PMH harvesting, an approach based on complex object representations of assets guarantees that a change to any constituent of a resource will result in a change of the OAI-PMH datestamp of the complex object representation of the resource. As a result, the OAI-PMH datestamp becomes a fully reliable trigger to incrementally harvest updated and added resources when those resources are represented using a complex object 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-to-one correspondence between an asset in the producing archive and an OAI-PMH item; the asset is the OAI-PMH resource.
- Metadata in various formats can be disseminated from each OAI-PMH item. Of interest to the proposed solution is the dissemination based on the MPEG-21 DIDL complex object format. A DIDL document that packages an asset is provided as metadata in OAI-PMH responses.
- In order to remain tightly synchronized with the producing archive, the granularity of the OAI-PMH repository is seconds-level. If tight synchronization is not required, a day-level granularity may be sufficient.
- The OAI-PMH datestamp of the OAI-PMH record that contains the DIDL document as metadata is the datetime of creation of an asset, or the datetime of the most recent change of any constituent of the asset, including datastreams, secondary information, etc. 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 complex object format, is essential to allow the consuming archive to remain permanently synchronized with the producing archive.
- In the current APS/LANL implementation, the OAI-PMH identifier of the OAI-PMH item is derived from the Content Information Identifier of the asset. As a matter of fact, the OAI-PMH identifier uses the Content Information Identifier value of the asset as part of an identifier of the oai-identifier scheme [21]. Investigations are ongoing aimed at understanding whether using the Content Information Identifier directly as the OAI-PMH identifier would simplify the content

transfer framework, especially if multiple nodes in a content transfer network are considered.

- As will be explained in Section 3.3.2, 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.

3.3.2. XML Signatures for DIDL documents

In addition to the XML Signatures provided at the level of each constituent datastream of an asset of the producing archive, an XML Signature is also created for the complete DIDL document that packages the 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 information (`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 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 information pertaining to the metadata provided in an OAI-PMH record. Table 5 summarizes the properties of this XML Signature. Annex B shows an example.

	1	2	3	4	5	6	7
	Provision	**Media Type**	**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 5. Use of W3C XML Signatures to sign DIDL documents.

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). As such, dependent on the request being issued, an OAI-PMH response may contain one or more OAI-PMH records. Two approaches are possible to identify the data that is being digested/signed (i.e., the DIDL document itself), 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 that is the OAI-PMH response. In addition, an XPath Filter transform 2.0 is provided (See column 3 of Table 5) 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 tools (as these tools are not aware of the semantics of the OAI-PMH model).

4. 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. As will be described, and as is illustrated in Figure 2, the process consists of:

Related to the transfer mechanism part of the solution:

- Harvesting, via OAI-PMH, DIDL documents that are XML-based packagings 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 dereferencing 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.

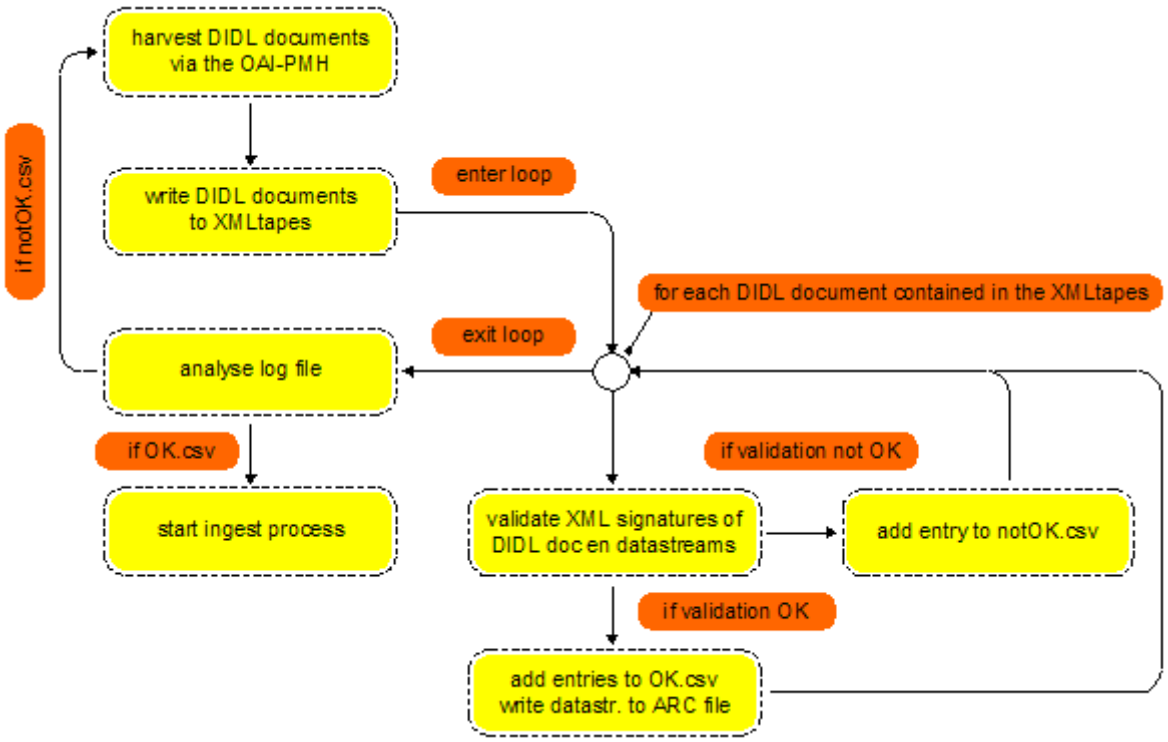

Figure 2: Gathering assets in the pre-ingest area of the LANL Repository.

4.1 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 packagings of assets from the producing archive. The semantics of the OAI-PMH datestamp for the exposed DIDL documents ensures that all DIDL documents that are packagings of 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 as implemented by LANL are as follows:

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

  ```
  baseURL(producing archive)?
     verb=ListRecords
     & from=YYYY-MM-DDThh:mm:ssZ
     & metadataPrefix=DIDL
  ```

  Hereby, YYYY-MM-DDThh:mm:ssZ, is the seconds-granularity datetime of the previous harvest. 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 3- to 4-hour basis.
- The DIDL documents resulting from this harvest are streamed into XMLtapes [26]. An XMLtape is a general-purpose, file-based mechanism devised to store collections of XML documents of the same class by concatenating those XML documents into a well-formed XML file that validates against the XML Schema at http://purl.lanl.gov/STB-RL/schemas/2005-01/tape.xsd/schemas/2005-01/tape.xsd. XMLtapes are used as a persistent storage mechanism in LANL's aDORe repository [32]. But they are also used as a temporary storage medium for OAI-PMH harvesting processes, and are used as such in the described asset transfer process. The overall structure of an XMLtape can be derived from an introspection of its defining XML Schema. Here, the use of XMLtapes is explained for OAI-PMH harvesting, in general, and for the transfer process specifically; Annex C illustrates an XMLtape as used in the proposed content transfer solution. As can be seen, the XMLtape is organized as follows:
- A `tape-admin` header contains administrative information about the collection of XML documents stored in the XMLtape. The following information related to OAI-PMH harvesting is provided in this header:
    - The baseURL of the OAI-PMH repository from which the records were harvested. In the case of the proposed solution, this is the baseURL of the OAI-PMH repository operated by the producing archive – `baseURL(producing archive)`.
    - The XML Namespace URI of the metadata format of the `records` that have been harvested. In the case of the proposed solution, this is the XML Namespace URI of MPEG-21 DIDL - urn:mpeg:mpeg21:2002:02-DIDL-NS.
    - The response datetime of the first OAI-PMH `ListRecords` response of the harvesting process.
- Following the `tape-admin` header are multiple `tape-record` elements, each of which contains the following information related to the harvesting process:
    - The OAI-PMH identifier and the OAI-PMH datestamp of a harvested OAI-PMH record. In the proposed content transfer solution, they are the OAI-PMH identifier and the OAI-PMH datestamp of a harvested DIDL document.
    - Provenance information related to the harvested record, expressed using the OAI-PMH `provenance` element [22], and provided in the optional and repeatable `tape-record-admin` element of the XMLtape. This information includes the response datetime of the OAI-PMH response that resulted in the record being harvested, and hence, enables a repository to know when a DIDL document has been collected.
    - The complete harvested OAI-PMH record. In the proposed solution, this is the OAI-PMH record, which includes OAI-PMH header

information, the DIDL metadata, and the 'about' containers associated with that metadata, including the DIDL-level XML Signature.

4.2 Gathering constituent datastreams of assets

Once the harvesting of DIDL documents, as described in Section 4.1, 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 2):

- 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 4.3. 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:
    - 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 retrieved by dereferencing the URI provided in the `URI` attribute (i.e., 'http://oai.aps.org/filefetch?identifier=PhysRevB.69.174413&description=print') appended to the `Reference` element of the XML Signature. The PDF datastream is retrieved 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 ARC file [9], which in essence is a concatenation of bitstreams that are separated by a metadata section. Multiple ARC files 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) of *one* 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 ARC file. 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.

4.3 The verification of W3C XML Signatures

In the proposed data transfer approach, 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 packaging of an 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:

- 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 (e.g., 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 and when that stored digest is delivered as part of the XML Signature. 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 at the producing archive. Again, the consuming archive must try to re-collect the problematic datastream.

4.4. The pre-ingest area of the consuming archive

As a result of the processes described in Sections 4.1 to 4.3, the pre-ingest area of the consuming archive now contains a collection of harvested DIDL documents, datastreams associated with all assets of which those DIDL documents are XML-packagings, 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 the DIDL documents that resulted from OAI-PMH harvesting from the producing archive.
- Per XMLtape, one or more Internet Archive ARC files that contain datastreams that resulted from successfully processing complete DIDL documents. In such an ARC file, 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 that lists 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 6.
- A notOK.csv file that lists the DIDL documents for which processing was unsuccessful. Table 7 shows a sample entry from such a log.

Information elements in OK.csv entry	Sample value
OAI-PMH identifier of harvested DIDL document	oai:aps.org:PhysRevB.69.174413
XPath specifying the XML element from the harvested DIDL 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?identifier=PhysRevB.69.174413&description=apsmeta
Datetime of collecting the constituent datastream	2005-04-14T20:25:03Z
Name of the ARC file in which the constituent datastream is stored	aps_test_2005_d236cd58-88b2-49f3-9d84-c6450d3ebb7a
Identifier of the constituent datastream stored in the ARC file	info:lanl-repo/arc/881848de-9b6b-4401-924d-ad528877d34e
Digest over constituent datastream	urn:sha1:tKbU9kVeJsnvh2JfMXIbwTecO+Y=

Table 6. OK.csv: log information for all DIDL documents for which processing was successful.

Information elements in notOK.csv entry	Sample value
OAI-PMH identifier of harvested DIDL document	oai:aps.org:PhysRevB.69.198875
XPath specifying the XML element from the harvested DIDL 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?identifier=PhysRevB.69.198875&description=print
Datetime of collecting the constituent datastream	2005-04-15T15:10:31Z
Error status	Reference for URI ' has no XMLSignatureInput

Table 7. 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 that 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 4.3, 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 4.2.

As will be explained in Section 4.5, the OK.csv file is crucial for ingesting the obtained assets into the consuming archive.

4.5. 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 information 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 1).

In the LANL implementation, the OK.csv file (Table 6) unambiguously ties a DIDL document stored in an XMLtape to its constituent datastreams stored in one or more ARC files. As such, 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 ARC files stored in the pre-ingest area of the LANL aDORe repository.

As a result of the described data transfer process, and the consecutive ingestion process, 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 facilitates the archives to compare notes by exchanging digests.
- OAI-PMH Provenance information [20]: The inclusion of OAI-PMH Provenance information facilitates the automatic re-harvesting 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.

5. 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 project 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 will be 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 shows 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 seems 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 project. 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 project, 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, amongst 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 put on the wire. 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 retrieve those datastreams can be postponed to the dereferencing sub-process described in Section 4.2, 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 front-end to the producing archive. For completeness, it should be mentioned that the By-Reference approach also may introduce certain complexities because it requires the provision of a dereferencable URI per individual datastream of the producing archive. The URI should be independent of 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. URIs had to be created that combine the Content Information Identifier with metadata aimed at unambiguously typing individual datastreams of the identified asset. This requirement led to a significant reorganization of the APS archive to make it easier to map these URIs to specific files.

Another design choice that requires further attention is the use of the content-decoding transform indicating the use of compression as described in Secion 3.2.2. Such a transform specifies which content-decoding algorithms must be applied to identified data before a digest can be calculated. In the APS/LANL project, 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 make possible detecting 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 project, 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 specification, and hence requires additions to XML Signature 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 project.

LANL intends to publicly release several of the tools that were used in the context of this project. In the near future, a Perl package will be released on CPAN that facilitates the writing and reading of DIDL documents. This is the package that is being used by the APS to generate DIDL documents that are exposed through their OAI-PMH repository. This summer, LANL also plans to release a similar package written in Java. Around the same time, a bundle of packages will be released that, when combined, allow for the implementation of the OAI-PMH-based resource harvesting solution as described in Section 4. However, the resource harvesting capabilities provided by this bundle will not be limited to OAI-PMH repositories that support MPEG-21 DIDL and/or XML Signatures. Rather, harvesting from any kind of OAI-PMH repository will be possible, and a plug-in architecture will allow the use of code tailored at dereferencing datastreams. The code used will depend on the actual repository from which metadata is being harvested, the actual metadata format that is harvested, and the knowledge on how to interpret the harvested data in terms of locating datastreams. Harvested records will be written to XMLtapes, and collected datastreams to Internet ARC files; the resource harvesting process will be logged in control files.

6. Conclusion

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

- An XML-based complex object 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 collection of new and updated assets, represented as XML documents, from a producing archive,
- An XML-specific technique (XML Signatures) 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 assets from the producing archive without the need for human interaction.

It is hoped that the solution will attract the interest of content producers other than the APS, and content consumers other than LANL. Highly encouraging in this respect is the fact that the APS intends to start using the described mechanisms for the transfer of content with consuming archives other than LANL. 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 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 computing-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.

Annex A: APS DIDL document

Annex B: OAI-PMH GetRecord response containing an APS DIDL document

Annex C: XMLtape containing the harvested OAI-PMH records

Acknowledgments

The authors would like to thank their colleagues Lyudmila Balakireva, Mariella Di Giacomo, Xiaoming Liu, and Thorsten Schwander of the LANL Digital Library Research and Prototyping Team for their enormous contributions to the reported work. Many thanks also to Mark Doyle and Gerard Young from the American Physical Society for their input in the design of the mirroring process and for its concrete implementation at their end. Thanks also to Justin Littman from the Library of Congress for his efforts related to testing the OAI-PMH repository of the American Physical Society, and to Patrick Hochstenbach, at Ghent University, for his work on a previous version of the XMLtape.

Jeroen Bekaert wishes to thank the Fund for Scientific Research (Flanders, Belgium) for his Ph.D. scholarship.

The reported work is partially funded by a grant from the Library of Congress's National Digital Information Infrastructure Program.

References

1. *Apache XML Security for Java* (2005, March). Retrieved from <http://xml.apache.org/security/>.

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

3. 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). Retrieved from <doi:10.1045/february2004-bekaert>.

4. 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 from <doi:10.1045/november2003-bekaert>.

5. Bekaert, J., & Rump, N. (Eds.) (2005, January). *ISO/IEC 21000-3 PDAM1 Related Identifier Types* (Output Document of the 71th MPEG Meeting, Honk Kong, China, No. ISO/IEC JTC1/SC29/WG11/N6928). Retrieved from the NIST MPEG Document Register.

6. Boyer, J. (2001 March). *Canonical XML Version 1.0* (W3C Recommendation). Retrieved from <http://www.w3.org/TR/xml-c14n>.

7. Boyer, J., Eastlake, D. E., & Reagle, J. (2002, July). *Exclusive XML Canonicalization* (W3C Recommendation). Retrieved from <http://www.w3.org/TR/xml-exc-c14n/>.

8. Boyer, J., Hughes, M., & Reagle, J. (2002, November). *XML-Signature XPath Filter 2.0* (W3C Recommendation). Retrieved from <http://www.w3.org/TR/xmldsig-filter2/>.

9. Burner, M., & Kahle, B. (1996, September 15). *Arc File format*. Retrieved from <http://www.archive.org/web/researcher/ArcFileFormat.php>.

10. Consultative Committee for Space Data Systems (CCSDS) Panel 2. (2004, May). *Producer-Archive Interface Methodology* (CCSDS Blue Book 651.0-B-1). Retrieved from <http://www.ccsds.org/CCSDS/documents/651x0b1.pdf>.

11. Consultative Committee for Space Data Systems (CCSDS) Panel 2. (2003, August). *XML structure and construction rules* (CCSDS Tech. Rep. No. 727/0831XFDUv09). Retrieved from <http://www.ccsds.org/docu/dscgi/ds.py/Get/File-727/0831XFDUv09.pdf>.

12. DCMI Usage Board (2004, April). *DCMI Type Vocabulary* (DCMI Recommendation). Retrieved from <http://dublincore.org/documents/dcmi-type-vocabulary/>.

13. Fallside, D. C. (Ed.). (2002, May 2). *XML Schema Part 0: Primer* (W3C Recommendation). Retrieved from <http://www.w3.org/TR/xmlschema-0/>.

14. IMS Global Learning Consortium. (2003, June). *IMS content packaging XML binding specification version 1.1.3*. Retrieved from <http://www.imsglobal.org/content/packaging/>.

15. International Digital Enterprise Alliance, Inc. (2004, March). *PRISM: Publishing Requirements for Industry Standard Metadata. Version 1.2*. Retrieved from <http://www.prismstandard.org/specifications/>.

16. International Organization for Standardization. (2003). *ISO 14721:2003. Space data and information transfer systems -- Open archival information system -- Reference model* (1st ed.).

17. International Organization for Standardization. (2003). *ISO/IEC 21000-2:2003. Information technology -- Multimedia framework (MPEG-21) -- Part 2: Digital Item Declaration* (1st ed.).

18 . International Organization for Standardization. (2005). *ISO/IEC 21000-2:2005. Information technology -- Multimedia framework (MPEG-21) -- Part 2: Digital Item Declaration* (2nd ed.).

19. International Organization for Standardization. (2003). *ISO/IEC 21000-3:2003: Information technology -- Multimedia framework (MPEG-21) -- Part 3: Digital Item Identification* (1st ed.).

20. Kahn, R., & Wilensky, R. (1995, May 13). *A framework for distributed digital object services*.

Retrieved from <http://hdl.handle.net/cnri.dlib/tn95-01>.

21. Lagoze, C., Van de Sompel, H., Nelson, M. L., & Warner, S. (Eds.). (2002, 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 from <http://www.openarchives.org/OAI/2.0/guidelines-oai-identifier.htm>.

22. Lagoze, C., Van de Sompel, H., Nelson, M. L., & Warner, S. (Eds.). (2002, 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 from <http://www.openarchives.org/OAI/2.0/guidelines-provenance.htm>.

23. Lagoze, C., Van de Sompel, H., Nelson, M. L., & Warner, S. (Eds.). (2003, February 21). *The Open Archives Initiative protocol for metadata harvesting* (2nd ed.). Retrieved from <http://www.openarchives.org/OAI/2.0/openarchivesprotocol.htm>.

24. The Library of Congress: The Network Development and MARC Standards Office. (2004, June). MARC 21 XML Schema (MARCXML). Retrieved from <http://www.loc.gov/standards/marcxml/>.

25. The Library of Congress: The Network Development and MARC Standards Office. (2004, November). *Metadata Encoding and Transmission Standard (METS)*. Retrieved from <http://www.loc.gov/standards/mets/>.

26. Liu, X., Balakireva, L., & Van de Sompel, H. (accepted). Using XMLtapes and Internet Archive ARC files to store Digital Objects and constituent datastreams in aDORe. In *Proceedings of the 9th European Conference, ECDL '05, Vienna, Austria*. Heidelberg, Germany: Springer-Verlag. Retrieved from <http://arxiv.org/abs/cs.DL/0503016>.

27. National Information Standards Organization. (2000, May). *ANSI/NISO Z39.84-2000: Syntax for the Digital Object Identifier*. Bethesda, MD: NISO Press.

28. National Information Standards Organization. (2001, September). *ANSI/NISO Z39.85-2001: The Dublin Core Metadata Element Set*. Bethesda, MD: NISO Press.

29. Nierman, Judith (1996). *Major Milestone: Copyright Office Receives First Digital Deposit*. Library of Congress Information Bulletin, March 4 1996. Retrieved from <http://www.loc.gov/loc/lcib/9604/cords.html>.

30. Siddiqui, B. (2003, April). *Web Services Security Part 2*. Retrieved from <http://webservices.xml.com>.

31. Sutton, C., & Clayphan, R. (1997, March). *BIBLINK - LB 4034 - D5.1 Transmission of Data*. Retrieved from <http://hosted.ukoln.ac.uk/biblink/wp5/d5.1.rtf>.

32. Van de Sompel, H., Bekaert, J., Liu, X., Balakireva, & L., Schwander, T. (accepted for publication). aDORe. A modular standards-based Digital Object repository. In *The Computer Journal*. Oxford, UK: Oxford University Press. Retrieved from <http://arxiv.org/abs/cs.DL/0502028>.

33. 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 from <doi:10.1045/december2004-vandesompel>.

34. van der Werf-Davelaar, T. (1999, September). Long-term Preservation of Electronic Publications. *D-Lib Magazine, 5*(9). Retrieved from <doi:10.1045/september99-vanderwerf>.

Notes

1. LANL: <http://www.lanl.gov/>.

2. APS: <http://www.aps.org/>.

3. NDIIP: <http://www.digitalpreservation.gov/>.

4. PREMIS: <http://www.oclc.org/research/projects/pmwg/>.

doi:10.1045/june2005-bekaert