Received July 2009
Revised October 2009
Accepted October 2009

THEME ARTICLE

Using XFDU for CASPAR information packaging

Matthew Dunckley
Science & Technology Facilities Council, Oxford, UK
Shahar Ronen, Ealan A. Henis, Simona Rabinovici-Cohen and Petra Reshef
IBM Research Laboratory, University of Haifa, Haifa, Israel, and
Esther Conway and David Giaretta
Science & Technology Facilities Council, Oxford, UK

Abstract

Purpose – The purpose of this paper is to describe the use of XML formatted data unit (XFDU) technology to implement OAIS-based information packaging in the CASPAR project.

Design/methodology/approach – The paper outlines new tools and techniques in working with XFDU for the purpose of preserving complex digital information.

Findings – The preservation of digital assets was facilitated by using the features of XFDU in the CASPAR project.

Practical implications – The paper is of interest to those responsible for the archival or long term preservation of digital assets.

Originality/value – The paper demonstrates new tools and techniques, which together provide an integrated system suitable for solving complex issues of preserving digital assets using information packaging.

Keywords Digital storage, Extensible markup language, Open systems

Paper type Technical paper

1. Introduction

Long-term digital preservation is a set of processes, strategies and tools for storing and accessing digital data for long periods of time, during which technologies, formats, hardware, software and technical communities are very likely to change. To allow the interpretation of the preserved data by future generations, the data should be packaged along with metadata describing them. The Open Archival Information System (OAIS) Reference Model (Consultative Committee for Space Data Systems (CCSDS), 2002), which laid the general framework for digital preservation, defined a model for such information packaging.

The work is partially supported by European Community under the Information Society Technologies (IST) program of the 6th FP for RTD – project CASPAR contract IST-033572. The authors are solely responsible for the content of this paper. It does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of data appearing therein.

OCLC Systems & Services:
International digital library
perspectives
Vol. 26 No. 2, 2010
pp. 80-93
© Emerald Group Publishing Limited
1065-075X
DOI 10.1108/10650751011048452

OAIS requires that the preserved content data object (CDO) and its metadata are stored in an archival information package (AIP). The metadata associated with the CDO is its representation information (RepInfo) and preservation descriptive information (PDI). The RepInfo is a set of objects used to interpret the CDO, such as a description of the object's structure, or a glossary of terms used in it. Each RepInfo may be described recursively by RepInfo of its own, creating a RepInfo network (RIN). The recursion ends when the RIN converges to the knowledge base of the designated community, which is assumed common knowledge, needing no further explanation (Patel and Ball, 2008). The PDI includes several types of metadata: provenance guarantees the documentation of the life cycle of the AIP, context documents its relation to its environment, fixity guarantees its integrity, and reference keeps a set of identifiers for the AIP (Consultative Committee for Space Data Systems (CCSDS), 2002). The AIP structure is depicted in Figure 1.

OAIS is a high-level reference model and does not provide implementation guidance. The European Union CASPAR project (www.casparpreserves.eu/) aims to validate OAIS by researching, implementing, and disseminating innovative OAIS-based solutions for the preservation of digital cultural, artistic, and scientific data. One of the challenges CASPAR deals with is that of information packaging, namely the organization of the metadata, their co-location with the CDO within an AIP, and the implementation of a self-describing package that is also easy to use. The authors used the OAIS-compatible XML formatted data unit (XFDU) (Consultative Committee for Space Data Systems (CCSDS), 2008) to address the information packaging challenge.

This paper describes the CASPAR packaging subsystem and its use of the XFDU packaging format. The authors demonstrate how XFDU can be adapted easily to support complex preservation concepts, such as RINs and AIP transformations, and present some of the tools they have created along the way. The paper is organized as follows. First the system's architecture is outlined, approaches to RepInfo packaging are described, and XFDU is introduced. Following this the various components of the packaging subsystem are described, including explanation of the application of XFDU. The paper concludes with a summary of the work.

2. Overview of the CASPAR packaging and storage solution

This paper focuses on three subcomponents of the CASPAR system: Packaging (by STFC, Science and Technology Facilities Council, UK), RepInfo Registry (by STFC and the Digital Curation Centre, UK), and Preservation DataStores (PDS, by IBM), as depicted in Figure 2. Together, these components provide an integrated preservation environment suitable for ingestion, storage and retrieval of information packages.

The packaging subsystem components have been applied to data used by various CASPAR testbed partners. Among these data are atmospheric data archived and

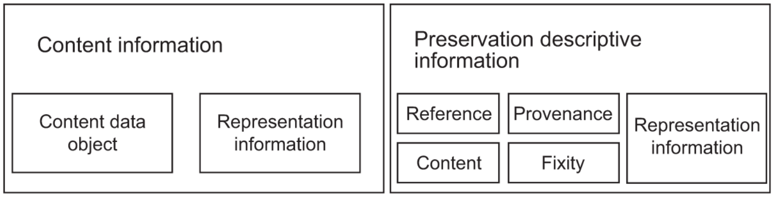

Content information
Content data object
Representation information
Preservation descriptive information
Reference
Provenance
Content
Fixity
Representation information

Figure 1. AIP logical structure according to OAIS

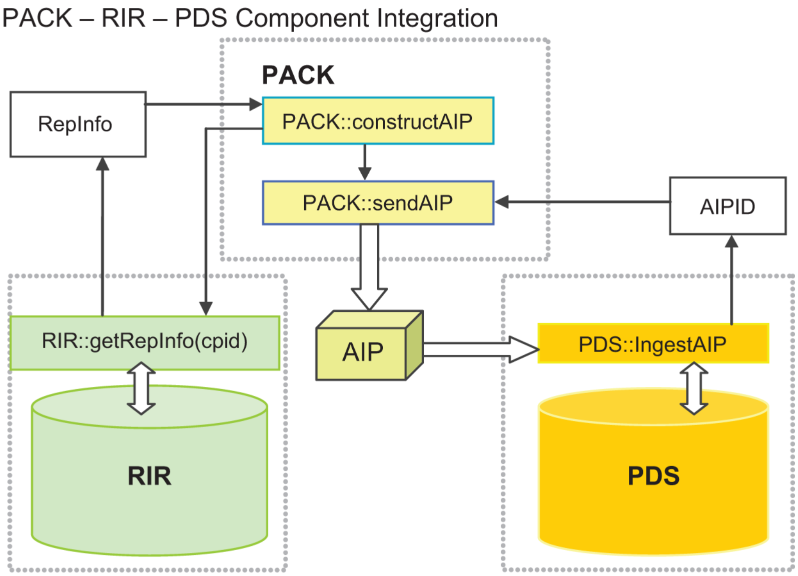


Figure 2.
The CASPAR packaging subsystem architecture

managed by STFC; earth-observation data captured by the European Space Agency's (ESA) global ozone monitoring experiment (GOME) satellite; and geographic information systems (GIS) data supplied by UNESCO, capturing information from world heritage sites. The importance and usage of these data types is discussed in (CASPAR Consortium, 2006).

To keep track of the different parts of an AIP and the relations among them, the authors use a "table of contents" called a manifest. The manifest is an XML document stored within the AIP, containing all the valuable information about the AIP contents. The manifest associates the CDO its RepInfo and PDI, allowing the expression of complex relationships between these information objects.

Given that the manifest is XML-based, it is self describing and platform independent, and can be moved, read and interpreted easily between heterogeneous data systems. Information can be added to the manifest to support archival curation services such as providing information to aid package discovery, digital rights management, format migration and transformation, data analysis, and data validation.

2.1 Approaches to RepInfo packaging

Accepting the idea that a CDO should be enhanced by an XML manifest and RepInfo to be properly preserved, it is necessary to decide how to package these three components together. The simplest solution is to keep all three components in the same container (e.g. a ZIP or TAR file). Keeping the RepInfo close to the CDO presents the lowest preservation risk, as it ensures that the RepInfo is maintained as long as the CDO is maintained, and that both can be retrieved. This method can become a storage hog when the same RepInfo is used by multiple AIPs, however, as the information must be duplicated and stored separately for each AIP. This solution also renders RepInfo updates quite inefficient, as it requires separate handling of each AIP that contains the updated RepInfo.

Repositories that extensively share RepInfo objects may benefit from using a *RepInfo Network (RIN)*. In this approach, the digital asset is separated from its RepInfo, which is held in a remote repository and referenced from the manifest. The remote repository is managed and maintained by the community or its proxy (e.g. a service provider). A single RepInfo object contained on the RIN may be referenced by multiple AIPs, reducing storage space and therefore the cost of preservation. Updating RepInfo also becomes easier: AIPs need not be updated separately, since an update of a RIN object is automatically reflected upon all AIPs that reference it. This ensures AIPs reflect the ever-changing knowledge of the designated community. While this solution introduces a possible loss of access to the RIN, this risk could be significantly reduced by taking proper precautions (e.g. the latest high-availability and data recovery methods).

In CASPAR the authors combined these two approaches, by storing specific (or less common) RepInfo with the data and general (or more common) RepInfo on the RIN. For example, the RepInfo describing the CDO's format might be kept on the RIN to allow AIPs containing CDOs of the same format to re-use it; for the same AIP, RepInfo describing terms used exclusively in this AIP's CDO would be kept with the data. Data producers and system administrators should determine which RepInfo should be stored with the data and which on the RIN, depending on the abundance of a specific RepInfo object in the system and the acceptable risks associated with the specific data it describes.

In addition, RepInfo sharing close to the data can be achieved using the PDS RepInfo Manager, which implements sharing at the storage level (see "RepInfo classification" in section 3.3.2). This approach attains all the benefits of a RIN while keeping the CDO and RepInfo in the same preservation storage.

2.2 XFDU packaging format

In searching for a solution to its Information Packaging requirements, the CASPAR Consortium study (CASPAR Consortium, 2007) identified three main contenders: the Metadata Encryption Transmission Standard (METS), MPEG-21, and the XML formatted data unit (XFDU), developed for the digital libraries, multimedia applications, and space domains, respectively. While the three were gaining popularity in their own domains, there was little take-up across domains. At the time of the survey, both METS (www.loc.gov/standards/mets/) and MPEG-21 (www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm) lacked built-in support for OAIS concepts; none of the CASPAR testbed-partners had actual experience with these standards and neither had standard and mature software tools available for implementation.

XFDU was determined to be the most suitable for preservation of the three and the easiest to use. Standardized and well documented by Consultative Committee for Space Data Systems (CCSDS) (2008), XFDU had been designed from the start with support for OAIS terminology. It had already been used operationally by *T*he European Space Agency (ESA) (http://earth.esa.int/SAFE/index.html) a CASPAR partner that had developed the Standard Archive Format for Europe (SAFE), a packaging format fully-compatible with XFDU (ESA, 2006). Selecting XFDU also allowed CASPAR to leverage existing open-source Java toolkits and APIs – there were two toolkits available at the time, one by ESA (www.gael.fr/xfdu/site/index.html) and another by NASA (http://sindbad.gsfc.nasa.gov/xfdu/index.html) – for the construction, editing and analysis of XFDU-based information packages (Figure 3).

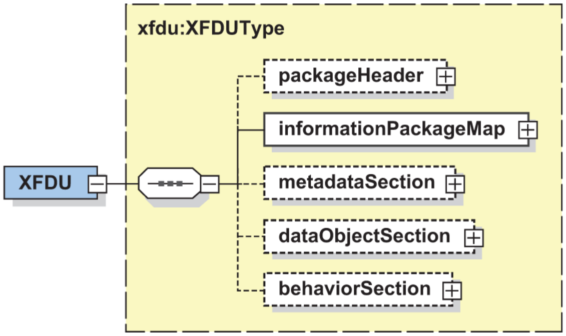


Figure 3.
Top-level manifest structure

XFDU uses an XML schema to describe a manifest file, which is separated into five sections (the behaviourSection will not be discussed in this paper). The packageHeader documents information about the package itself, its versioning, its position in a sequence or volume, and more.

The dataObjectSection and metadataSection are used to connect the CDO, or its RepInfo or PDI, respectively. Both data objects and metadata objects are either connected by reference, or encoded within the manifest itself. Each object is assigned an XML identifier, which is used to link objects between the two sections. Objects in both sections can be given built-in classifications or associated with user-defined classification schemes.

The contentUnit records information about content units, which are used to associate data in the dataObjectSection with metadata in the metadataSection. The association is done via XML identifiers, and maps to the OAIS concept of Content Information Object: a collection of a digital object and its RepInfo. Figure 4 shows the internal and external relationships that can be expressed using the Content Unit section of the manifest.

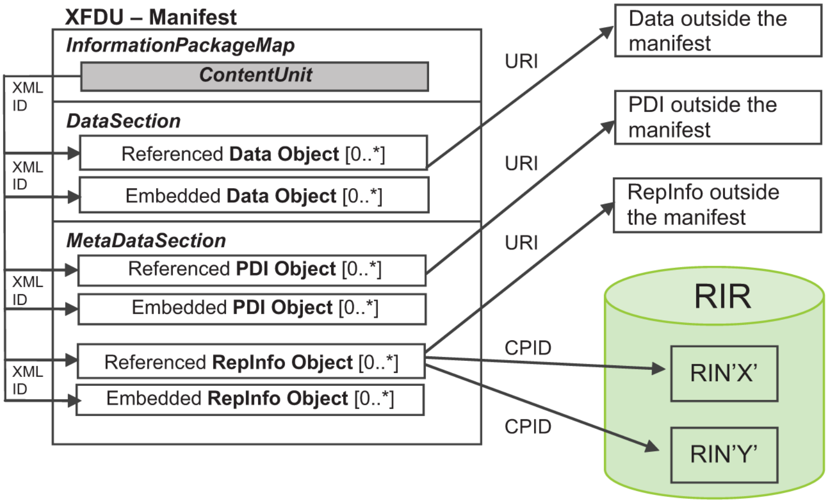

XFDU – Manifest
InformationPackageMap
ContentUnit
XML ID
DataSection
Referenced Data Object [0..*]
XML ID
Embedded Data Object [0..*]
MetaDataSection
XML ID
Referenced PDI Object [0..*]
Embedded PDI Object [0..*]
XML ID
Referenced RepInfo Object [0..*]
Embedded RepInfo Object [0..*]
URI
URI
URI
CPID
CPID
Data outside the manifest
PDI outside the manifest
RepInfo outside the manifest
RIR
RIN'X'
RIN'Y'

Figure 4.
Manifest associations

XFDU's standard schema syntax keeps all AIPs consistent while granting a flexible and adaptable implementation. ESA (2006) has demonstrated this by extending the XFDU schema into SAFE, which includes spacecraft mission-specific information embedded in the XFDU manifest. Such extension of the schema allows the inclusion of additional information while maintaining the standardization and consistency that are two of the main advantages of using XFDU for preservation.

3 The use of XFDU in the packaging subsystem

3.1 The CASPAR RepInfo registry repository

Previous preservation projects have identified the need to use registries to store networks of RepInfo. The main motivation for this is information re-use and sharing. CASPAR has also utilized a RepInfo registry (RIR) (Giaretta, 2005) to store the networks defined by its testbed data sets along with their classifications.

The CASPAR RIR provides an information model based on OAIS, extending it to provide sub-classifications of the main types of RepInfo defined by OAIS. The suite of registry tools includes a Java API and an intuitive GUI tool allowing the input, management and searching of RepInfo within the registry. A web-based thin client is provided allowing the searching and visualization of the RepInfo Networks (RIN).

To create the recursive structure of the RIN (as explained in the introduction), the RepInfo objects are connected together using what has been termed as a RepInfo label; labels and RepInfo objects maintain a 1:*n* relationship. Non-digital RepInfo (e.g. books and other physical objects) can be recorded in the CASPAR RIR using identifiers that allow the user to track the RepInfo in the "real world", such as a book's ISBN number or an object's geographic location.

The CASPAR RIR is not only for storing RINs, but also allows them to be reused by multiple AIPs. In addition to sharing, the RIR allows discovery and creates the opportunity for a distributed community to maintain the RIN as the community itself evolves over time. These networks need to be curated as well, and may evolve over time, requiring a collaborative effort from key stakeholders.

3.1.1 Referencing a RepInfo network. A RIN referenced from an AIP becomes a logical part of it, even though it is physically separate from that AIP; it is therefore important to discuss how this was applied in CASPAR. RepInfo within the CASPAR RIR can be referenced in the XFDU manifest in either of two ways: by referencing the Curation Persistent Identifier (CPID) of a single RepInfo object directly, or by using a RepInfo Label to reference a set of RepInfo objects. Either way, the manifest reference provides an entry point into the RIN and its recursive structure.

CASPAR XFDU packages are connected to the RIN in the CASPAR RIR using the attributes of the XFDU metadataReference element, as demonstrated in the example below. Using OAIS terminology, the containing metadataObject is classified and categorized as data entity description (DED) RepInfo; the authors use the vocabularyName attribute to also identify the object as "SEMANTIC". The RepInfo object in the CASPAR RIR is referenced by a URI through the href attribute; the otherLocatorType attribute indicates that the URI is a CPID. The id attribute also contains the CPID.

```
metadataObject category = "REP" classification = "DED" ID = "REP_
DESCRIPTION01" >
< metadataReference
```

```
vocabularyName = "SEMANTIC" otherLocatorType = "CPID" locatorType = "OTHER"
href = "http://registry.dcc.ac.uk/omar/registry/http?interface = QueryManager&amp;
method = getRepositoryItem&amp;param-id = urn:uuid:40e0c3de-a405-4759-b116-
eda15d77df59"
textInfo = "Semantic Information about MST version 3 NetCDF data files "
ID = "cpid-40e0c3de-a405-4759-b116-eda15d77df59"/>
 < /metadataObject
```

Given the data to preserve and a CPID, the CASPAR packaging component can pull extra information from the RIR upon constructing a package, such as textual descriptions of the RepInfo, which can be inserted into the XFDU manifest. This method provides an entry point into the RIN, a first level dependency. Using the CASPAR packaging subsystem it is possible to download all further necessary RepInfo in the network to be added into an AIP.

Using the packaging and registry APIs for this purpose, the Packaging Visualization Tool provides the visual inspection and construction of RIR connected XFDU AIPs. Having been developed over the packaging API, the tool is flexible enough to allow alternative packaging formats to be used. For example, a METS toolkit could be used in place of the XFDU toolkit allowing the visual construction and visualization of METS based AIPs. Figure 5 shows an example of using the tool to construct an MST package, where the AIP's first level RepInfo dependencies are embedded within the package itself with subsequent levels stored in the RIR. The green square icon represents the data object, the triangles represent RepInfo embedded directly within the AIP, and the circles represent RepInfo stored within the CASPAR RIR.

3.2 The packaging component

The CASPAR *packaging* software component is a Java API closely based around OAIS concepts. It exposes operations that provide for the general management of AIPs as identified by the CASPAR Consortium (2006). The packaging components' main responsibilities are:

1. construction – providing operations to build aips conforming to OAIS standards;
2. unpackaging – providing access to the internal information objects or resolvable references to information objects if they are external to the package;
3. validation – providing operations to validate the contents and structure of an AIP;
4. transmission – providing operations to send an AIP to a location for storage; and
5. storage – provides operations to store packages by calling Preservation DataStores.

Since XFDU was chosen as the default AIP format, CASPAR utilized the NASA XFDU Java-based toolkit (http://sindbad.gsfc.nasa.gov/xfdu/index.html) to provide construction, unpackaging and validation of AIPs. Local or remote storage of AIPs is performed using the IBM PDS Demo Web Client (http://digitalpreserve.svn.

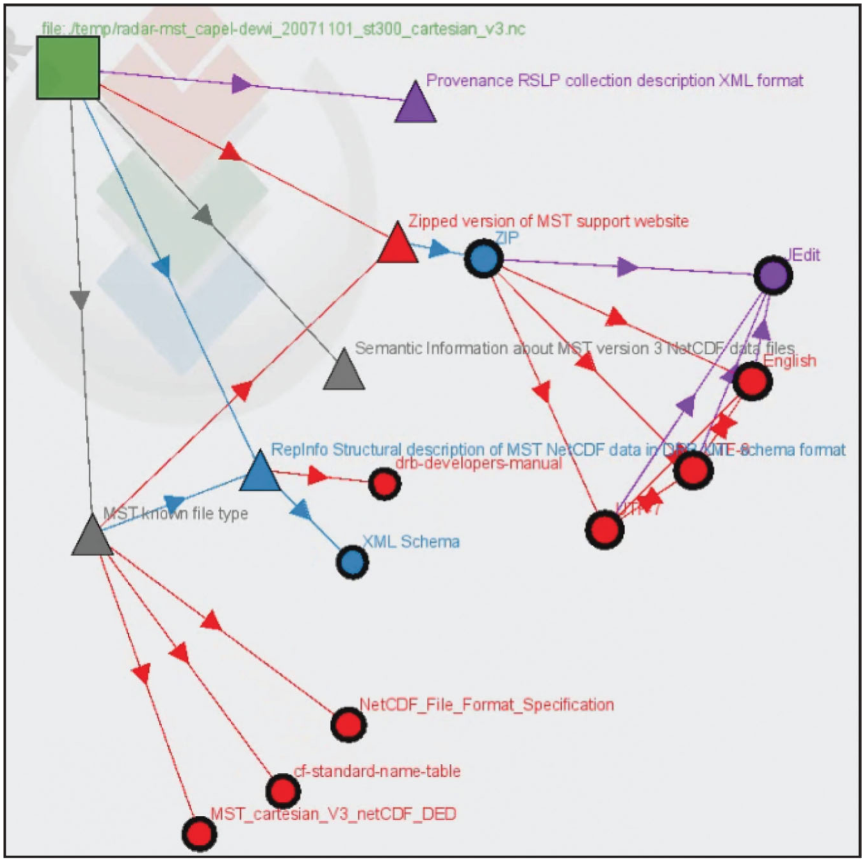
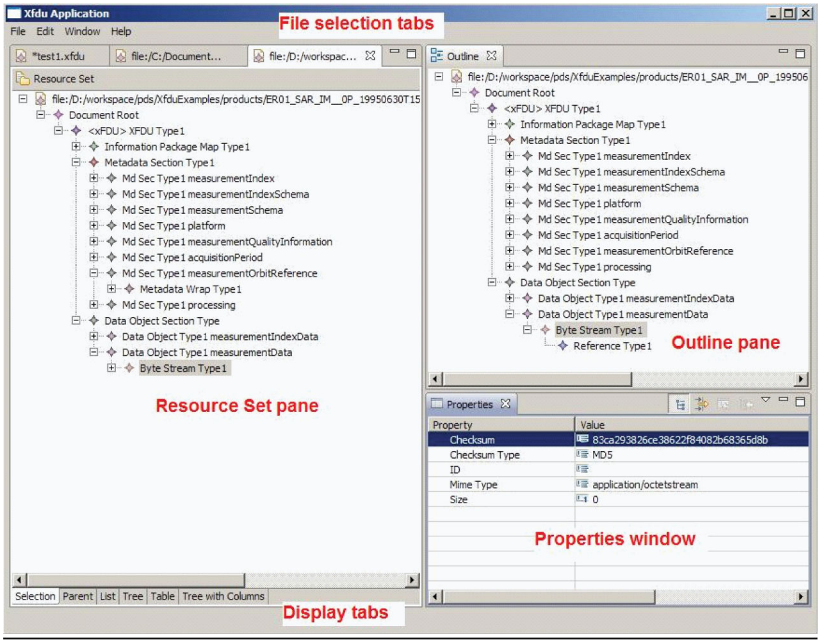

file: /temp/radar-mst_capel-dewi_20071101_st300_cartesian_v3.nc
Provenance RSLP collection description XML format
Zipped version of MST support website
ZIP
JEdit
Semantic Information about MST version 3 NetCDF data files
English
RepInfo Structural description of MST NetCDF data in DRB XML Schema format
UTF-8
drb-developers-manual
UTF-7
MST known file type
XML Schema
NetCDF_File_Format_Specification
cf-standard-name-table
MST_cartesian_V3_netCDF_DED

Figure 5.
A screenshot of the packaging visualization tool

sourceforge.net/viewvc/digitalpreserve/software/java/implementation/datastore/client/). Other clients may also be implemented for this purpose.

3.2.1 XFDU manifest editor. Packaging an AIP requires tremendous care, as errors made in the present are difficult to detect and correct in the distant future. XFDU manifests, which are extremely detailed and rely heavily on identifiers, are quite prone to errors. This is where the XFDU manifest editor (XME) yields an enormous benefit. Developed by the PDS team at IBM, XME – formerly known as XFDU AIP Generator (CASPAR Consortium, 2008) – is an easy-to-use graphical tool for viewing, creating and editing XFDU manifest files. Most graphical XML editors find errors only after they have been made; XME reduces error generation by limiting the user to specific valid entry options. For example, XME will decline non-numeral values entered for the *size* attribute, used to record the content data object's size in bytes; or, upon editing the metadataObject attribute classification, will present a drop-down menu listing only the possible values to select from. By removing irrelevant options, XME reduces the potential for confusion and facilitates the creation of correct XFDU manifests, thus significantly reducing errors (Figure 6).



```
file:/D:/workspace/pds/XfduExamples/products/ER01_SAR_IM__0P_19950630T15
  Document Root
    <xFDU> XFDU Type1
      Information Package Map Type1
      Metadata Section Type1
        Md Sec Type1 measurementIndex
        Md Sec Type1 measurementIndexSchema
        Md Sec Type1 measurementSchema
        Md Sec Type1 platform
        Md Sec Type1 measurementQualityInformation
        Md Sec Type1 acquisitionPeriod
        Md Sec Type1 measurementOrbitReference
          Metadata Wrap Type1
        Md Sec Type1 processing
      Data Object Section Type
        Data Object Type1 measurementIndexData
        Data Object Type1 measurementData
          Byte Stream Type1
```

```
file:/D:/workspace/pds/XfduExamples/products/ER01_SAR_IM__0P_199506
  Document Root
    <xFDU> XFDU Type1
      Information Package Map Type1
      Metadata Section Type1
        Md Sec Type1 measurementIndex
        Md Sec Type1 measurementIndexSchema
        Md Sec Type1 measurementSchema
        Md Sec Type1 platform
        Md Sec Type1 measurementQualityInformation
        Md Sec Type1 acquisitionPeriod
        Md Sec Type1 measurementOrbitReference
        Md Sec Type1 processing
      Data Object Section Type
        Data Object Type1 measurementIndexData
        Data Object Type1 measurementData
          Byte Stream Type1
            Reference Type1
```

Property	Value
Checksum	83ca293826ce38622f84082b68365d8b
Checksum Type	MD5
ID	
Mime Type	application/octetstream
Size	0

Figure 6.
XFDU manifest editor screen capture

3.3 Preservation DataStores

Preservation DataStores (PDS) (Factor *et al.*, 2007a, b; Rabinovici-Cohen *et al.*, 2008) is the storage component for CASPAR. PDS is an OAIS-based, preservation-aware storage with the main goal of supporting logical preservation. Logical preservation is achieved by encapsulating the digital asset to be preserved with the metadata required to manage future access and use of the asset. These metadata allow PDS to perform preservation-related functions within the storage, such as fixity computations and AIP transformations, thus increasing robustness and improving overall performance. PDS was designed to provide maximum flexibility, and can interact with any packaging format and storage device. Support for these extensions can be easily added via publicly available interfaces (http://digitalpreserve.svn.sourceforge.net/viewvc/digitalpreserve/software/java/interfaces/datastore/)

In its CASPAR implementation, PDS works in concert with the packaging component, using XFDU as its default packaging format. PDS ingests packages constructed by the packaging component, and manages RepInfo and PDI for these packages. PDS also provides built-in support for preservation functions, such as fixity checks and transformations.

3.3.1 AIP roles. While all AIPs are built around a digital asset that needs to be preserved, some have additional functions in the preservation system, such as transformation modules, fixity modules, or even serving as RepInfo for another AIP. To handle these “special AIPs” properly, a preservation system needs to internally identify them as such. For this purpose, PDS supports various AIP roles, which are indicated upon ingest through the packageType attribute of the XFDU manifest’s contentUnit element. An AIP that also serves as another AIP's RepInfo should therefore be marked as follows:

```
contentUnit unitType = “ContentRepInfo” >
...
 < /contentUnit
```

Other roles include FixityModule for AIPs containing ingest modules for fixity calculation, CategoryRepInfo for grouping and classifying RepInfo. An AIP that is not “special” is indicated by packageType = “Standard”, or by default when the optional packageType attribute is omitted.

3.3.2 RepInfo classification. PDS implements a RepInfo sharing mechanism of its own, named RepInfo Manager (IBM, 2009). Under CASPAR, the RepInfo Manager is used to handle RepInfo objects that are not part of the RIR. The RepInfo Manager groups newly ingested RepInfo objects into categories, a concept resembling the RIR's RepInfo labels: RepInfos that describe the same type of objects (i.e. belong together in some sense) are listed in the same category (e.g. all JPEG viewers will be listed under the same category). Each category points to the RepInfo objects that belong to it. An object may, and usually will, belong to more than one category. The RepInfo Manager replaces the manifest reference to each RepInfo object with a corresponding reference to the category to which that RepInfo object belongs. This mechanism ensures that the most recent RepInfo (latest version of viewer software, a clearer specification, etc.) can be easily found without having to update each AIP individually.

Categories can be defined either by the system administrator or by the ingestor of the AIP. The ingestor assigns a RepInfo to a certain category via the XFDU manifest by using the otherClass attribute of the metadataObject element (classification must be set to “OTHER” for otherClass to take effect):

```
metadataObject category = “REP” classification = “OTHER”
otherClass = “Category_MSTDataFiles”
ID = “REP_DESCRIPTION_01” >
< metadataReference mimeType = “text/xml”
vocabularyName = “SEMANTIC”          otherLocatorType = “FILE”
locatorType = “OTHER”
href = “file:MST_cartesian_V3_netCDF_DED.doc”
textInfo = “Semantic Data Entity Description of MST data files”
ID = “MSTFileSemanticDescription” />
 < /metadataObject
```

In the example above, the Microsoft Word document containing the description of MST data files is assigned to the category of RepInfos that describe MST data files; the RepInfo manager will add this RepInfo object to the assigned category upon ingest to PDS.

PDS is also capable of automatically assigning RepInfos to categories, based on the format of the CDO. This is only applicable to RepInfos describing the syntax of the CDO, however, which are identified in the XFDU manifest by their classification as “SYNTAX”.

```
metadataObject    category = “REP”    classification = “SYNTAX”
ID = “REP_SYNTAX_01” >
...
 < /metadataObject
```

Upon ingest, PDS identifies the CDO format, then assigns the syntactic RepInfos referenced in the manifest to a category or categories. This is done according to the CDO's MIME type specified in the XFDU manifest. In the example below, syntactic RepInfos referenced in the manifest for the CDO will be added to the application/x-netcdf category.

```
dataObjectSection >
< dataObject ID = “dataObject1” >
< byteStream mimeType = “application/x-netcdf” >
...
 < /byteStream >
 < /dataObject >
 < /dataObjectSection
```

When possible, PDS uses the CDO's PRONOM unique ID (PUID) (Brown, 2006) for classifying its RepInfos. These identifiers are more accurate than the MIME type. PDS relies on the DROID tool (http://droid.sourceforge.net/) to indentify PUIDs.

3.3.3 Transformation AIPs. A key aspect of long-term digital preservation is the need to migrate data objects being preserved. This may be triggered by changes such as media decay, obsolescence of hardware or software, or a change in the copyright or external environment (e.g. organization). Unlike traditional systems, which perform these tasks at the application level, PDS performs transformations within the storage level, leveraging its awareness of the media characteristics and media health. Factor *et al.* (2007a, 2009) and Rabinovici-Cohen *et al.* (2008) further discussed PDS and storage-level transformations.

The PDS interface allows transformations to be uploaded dynamically as AIPs. Like “regular” AIPs, transformation AIPs contain a CDO (the transformation program or script) and the RepInfo and PDI describing it. RepInfo for a transformation should explain how to perform it, and preferably describe how the transformation works. In CASPAR, the authors use XFDU as the packaging format for transformation AIPs. The following example shows the RepInfo of a transformation that converts MST data in NetCDF format to PNG format. The CDO (not shown) is a BASH script.

```
contentUnit unitType = “Transformation” >
...
 < /contentUnit >
...
< metadataSection >
< metadataObject category = “REP” classification = “SYNTAX”
ID = “REP_SYNTAX_c3f254f7-385f-11dd-ba03-4f174c837f85” >
< metadataReference mimeType = “text/html”
vocabularyName = “SEMANTIC” locatorType = “URL”
href = “http://tiswww.case.edu/php/chet/bash/bashref.html”
textInfo = “This text is a brief description of the features
that are present in the Bash shell (version 3.2, 28 Sep 2006).”/>
 < /metadataObject >
...
 < /metadataSection
```

A transformation AIP should provide, in addition to RepInfo describing the transformation itself, post-transformation RepInfo to describe the output AIP created by executing the transformation. Such information is added as the RepInfo of the newly created AIP, and is necessary to increase its preservation chances in the future. Hence, in addition to RepInfo about BASH scripting, a NetCDF-to-PNG transformation AIP should also contain RepInfo about the PNG format, or the representation of MST data in PNG format. These will be added to the newly created AIP.

In PDS post-transformation RepInfo is added via the AIP's context. While XFDU was not designed to hold post-transformation RepInfo, it is nonetheless versatile enough to do so, as demonstrated in the (partial) manifest sample below:

```
metadataSection >
...
< metadataObject    category = "PDI"    classification = "CONTEXT"
ID = "PDI_DESCRIPTION21" >
< metadataReference locatorType = "URL"
textInfo = "PNG information. This record becomes the RepInfo of the transformed
AIP"
href = "http://www.w3.org/Graphics/PNG" />
 < /metadataObject >
...
 < /metadataSection
```

4. Conclusion

This paper demonstrates how the CASPAR EU project has utilized XML formatted data unit (XFDU) technology to implement an OAIS based information packaging environment. The core CASPAR packaging software subsystem was explained, describing how the main components – the packaging, PDS and the RIR components – integrated and interacted. Elaborating on new tools and techniques built around the features of XFDU, the paper showed how together they provide a generic packaging solution, and how XFDU could be expanded to support complex preservation concepts, such as transformations or RIN. The CASPAR implementation had yielded a flexible solution, which had facilitated preservation activities across the artistic, cultural and scientific domains.

References

Brown, A. (2006), *Digital Preservation Technical Paper No. 2: The PRONOM UID Scheme: A Scheme of Persistent Unique Identifiers for Representation Information*, The National Archives (UK), available at: www.nationalarchives.gov.uk/aboutapps/pronom/pdf/pronom_unique_identifier_scheme.pdf

CASPAR Consortium (2006), *D4101: User Requirements and Scenario Specifications (CASPAR-D4101-SCEN-0101-1_0)*, available at: www.casparpreserves.eu/Members/metaware/Deliverables/user-requirements-and-scenario-specifications/at_download/file

CASPAR Consortium (2007), *D1101: Review of State of the Art (CASPAR-D1101-TN-0101-1_0)*, available at: www.casparpreserves.eu/Members/cclrc/Deliverables/review-of-state-of-the-art-1/at_download/file

CASPAR Consortium (2008), *D2201: Preservation DataStore Interface (CASPAR-D2201-TN-0101-1_0)*, available at: www.casparpreserves.eu/Members/cclrc/Deliverables/preservation-datastore-interface/at_download/file

Consultative Committee for Space Data Systems (CCSDS) (2002), *Reference Model for an Open Archival Information System (OAIS)), Blue Book Issue 1*, available at: http://public.ccsds.org/publications/archive/650x0b1.pdf

Consultative Committee for Space Data Systems (CCSDS) (2008), *XML Formatted Data Unit (XFDU) Structure and Construction Rules, Blue Book Issue 1*, available at: http://public.ccsds.org/sites/cwe/rids/Lists/CCSDS%206610R1/Attachments/661x0r1.pdf

European Space Agency (ESA) (2006), *Standard Archive Format for Europe (SAFE) Control Book, Vol. 1: Core Specifications*, available at: http://earth.esa.int/SAFE/download/Specifications/PGSI-GSEG-EOPG-FS-05-0001-20061120.pdf

Factor, M., Henis, E., Naor, D., Rabinovici-Cohen, S., Reshef, P., Ronen, S., Michetti, G. and Guercio, M. (2009), "Authenticity and provenance in long term digital preservation: modeling and implementation in preservation aware storage", paper presented at the USENIX Association First Workshop on the Theory and Practice of Provenance (TaPP), San Francisco, CA, February 23, available at: www.haifa.il.ibm.com/projects/storage/datastores/papers/Auth_Prov_CamReady_sent.pdf

Factor, M., Naor, D., Rabinovici-Cohen, S., Ramati, L., Reshef, P., Satran, J. and Giaretta, D. (2007a), "Preservation DataStores: architecture for preservation aware storage", *Proceedings of the IEEE Conference on Mass Storage Systems and Technologies (MSST), September 24-27, 2007, San Diego, CA, USA*, pp. 3-15, available at: http://doi.ieeecomputersociety.org/10.1109/MSST.2007.27

Factor, M., Naor, D., Rabinovici-Cohen, S., Ramati, L., Reshef, P. and Satran, J. (2007b), "The need for preservation aware storage: a position paper", *ACM SIGOPS Operating Systems Review, Special Issue on File and Storage Systems*, Vol. 41 No. 1, pp. 19-23.

Giaretta, D. (2005), *DCC Representation Information Registry*, Digital Curation Centre, available at: www.dcc.ac.uk/docs/RepresentationInformationRegistry.ppt

IBM (2009), *IPCOM000183331D: A System and Method for Managing Representation Information in a Preservation Data System*, Prior Art Database on IP.com, Portsmouth.

Patel, M. and Ball, A. (2008), "Challenges and issues relating to the user of representation information for the digital curation of crystallography and engineering data", *International Journal of Digital Curation*, Vol. 3 No. 1, pp. 76-88.

Rabinovici-Cohen, S., Factor, M.E., Naor, D., Ramati, L., Reshef, P., Ronen, S., Satran, J. and Giaretta, D.L. (2008), "Preservation DataStores: new storage paradigm for preservation environments", *IBM Journal of Research and Development*, Vol. 52 Nos 4/5, pp. 389-99.

Further reading

Abrams, S. (2006), "Global digital format registry, an interim status report", paper presented at the International Conference on the Preservation of Digital Objects (iPRES 2006), Ithaca, CA, October 8-10, available at: http://hdl.handle.net/1813/3689

Bekaert, J., Liu, X. and Van de Sompel, H. (2005), "Representing digital assets for long-term preservation using MPEG-21 DID", available at: http://arxiv.org/pdf/cs/0509084

Brown, A. (2008), *White Paper: Representation Information Registries*, IST-2006-033789, available at: www.planets-project.eu/docs/reports/Planets_PC3-D7_RepInformationRegistries.pdf

Conway, E. (2009), *SCRAP Case Study no. 2 – Curating Atmospheric Data for Long Term Use: Infrastructure and Preservation Issues for the Atmospheric Sciences Community*, Digital Curation Centre, available at: www.dcc.ac.uk/docs/publications/case-studies/SCARP_B4832_Atmospheric.pdf

Rödig, P., Borghoff, U.M., Scheffczyk, J. and Schmitz, L. (2003), "Preservation of digital publications: an OAIS extension and implementation", *Proceedings of the 2003 ACM Symposium on Document Engineering, Grenoble, France, November 20-22*, pp. 131-139.

Stanescu, A. (2004), "Assessing the durability of formats in a digital preservation environment", *D-Lib Magazine*, Vol. 10 No. 11.

About the authors

Matthew Dunckley is an Information Systems Developer, holding a BEng degree in Electronic Engineering and Computer Science from Aston University UK. He presently works on the CASPAR digital preservation project and for the Digital Curation Centre, UK. Matthew Dunckley is the corresponding author and can be contacted at: matt.dunckley@stfc.ac.uk

Shahar Ronen is a Research Staff Member in the Storage Systems and Performance Management group. He holds a BSc degree in computer science from the University of Haifa. Since joining IBM in 2006, he has worked on technology to support long-term digital preservation.

Ealan A. Henis holds a PhD degree in computer science from the Weizmann Institute of Science, Israel. Since 1997, he is a Research Staff Member at the IBM Haifa Laboratory, where he worked on a variety of computer science topics. He is currently focusing on long-term digital preservation.

Simona Rabinovici-Cohen holds BSc and MSc degrees in computer science, both from the Technion, Israel Institute of Technology. Since 1993, she has been a Research Staff Member at the Haifa Research Laboratory, working on various projects in the area of biomedical information integration, and on storage technology to support long-term digital preservation.

Petra Reshef holds a BSc degree in computer science from the University of Haifa. As a Research Staff Member in the Storage Systems and Performance Management group, she develops storage and network technologies, currently focusing on long-term digital preservation.

Esther Conway holds an MS degree in information systems and technology from City University London. She works as a systems analyst specializing in digital preservation projects.

David Giaretta is the director of the CASPAR digital preservation project and an associate director of the Digital Curation Centre (DCC).