DELIVERABLE

Project Acronym: **E-ARK**

Grant Agreement Number: **620998**

Project Title: **European Archival Records and Knowledge Preservation**

DELIVERABLE DETAILS

DELIVERABLE REFERENCE NO.	**MS10 - Version 3 of D5.3**
DELIVERABLE TITLE	**E-ARK Final DIP Specification**
REVISION	**3.0**

AUTHOR(S)	
Name(s)	Organisation(s)
Alex Thirifays	Danish National Archives
Zoltán Lux	National Archives of Hungary
Jože Škofljanec	National Archives of Slovenia
Gregor Završnik	National Archives of Slovenia
Anja Paulič	National Archives of Slovenia
Anders Bo Nielsen	Danish National Archives
Phillip Tømmerholt	Danish National Archives

Richard Healey	University of Portsmouth
Kuldar Aas	National Archives of Estonia

REVIEWER(S) Name(s)	Organisation(s)
Andrew Wilson	University of Brighton
Kuldar Aas	National Archives of Estonia
Jan Aspenfjall	Swedish National Archives
Anders Bo Nielsen	Danish National Archives

Project co-funded by the European Commission within the ICT Policy Support Programme Dissemination Level		
P	**Public**	X
C	**Confidential, only for members of the Consortium and the Commission Services**	

REVISION HISTORY AND STATEMENT OF ORIGINALITY

Submitted Revisions History

Revision No.	Date	Authors(s)	Organisation	Description
2.0	1/5/2016	Alex Thirifays	DNA	1st submitted version
3.0	13/3/2017	Alex Thrifays	DNA	Final published version (MS10)

Statement of originality:

This deliverable contains original unpublished work except where clearly indicated otherwise. Acknowledgement of previously published material and of the work of others has been made through appropriate citation, quotation or both.

Table of Contents

1 Executive Summary

The primary aim of this document is to present the *final* version of the E-ARK Dissemination Information Package (DIP) formats. The secondary aim is to describe the access scenarios in which these DIP formats will be rendered for use. This document is a revision of the deliverable D5.3 E-ARK DIP Pilot Specification, which was published in May 2016.

The above mentioned pilot version of the DIP formats has been used in the E-ARK pilots that are specific to Access. Feedback from these pilots has been taken into account to amend the specification of the formats where necessary. The results are the present deliverable.

In order to make this specification as operable as possible, it provides the concrete DIP specifications and access scenario descriptions that are associated with the content information types that the E-ARK project has dealt with: Databases, Data warehouses, Electronic Records Managements Systems, Simple File-System Based Records, and Geodata.

It also provides example metadata files for each content information type by linking to these files on GitHub.

The ultimate aim is to provide fully functional DIPs for each content information type, but this is outside the scope of this milestone deliverable.

This document is structured according to the specifications and access scenarios of the five aforementioned content information types. This structure has the advantage that each content information type section can stand alone. The drawback is that there are repetitions in each section.

2 Introduction

The purpose of this deliverable is to enable the dissemination of E-ARK content information types. This is done by specifying the E-ARK Dissemination Information Package (DIP) formats[1],[2],[3], and by describing the access scenarios[4] in which they are utilized[5].

The DIP reference format[6] represents the recommended practice for interoperable DIPs, and can be applied across different Access Software[7] and access systems. As such this format can in the future be supported as the default output format for preservation systems.

2.1 Methodology

The current document is part of an official milestone (MS10[8]), and has been created by the partners of the E-ARK project. It is mainly based upon two deliverables (D5.2[9] and D5.3[10]), but also on other existing work as well as on a series of requirements that have been identified employing both a bottom-up and a top-down approach.

The bottom-up approach identified relevant requirements by investigating the Common Specification[11]; analysing best practices[12] and user needs[13]; examining the E-ARK Submission Information Package (SIP)[14]

---

[1] Technical terms from OAIS, PREMIS, E-ARK, etc., can be found in the Glossary and will be available in the E-ARK Knowledge Center: http://kc.dlmforum.eu./home. The first time a term from the glossary is encountered in this deliverable a definition of it will be provided in a footnote.

[2] The Dissemination Information Package is an Information Package, derived from one or more AIPs, and sent by Archives to the Consumer in response to a request to the OAIS. Source OAIS. http://public.ccsds.org/publications/archive/650x0m2.pdf

[3] All OAIS terms are capitalised.

[4] Access scenario is used as a term to describe the environment (DIP and the Access Software) which is used to render archival records and their metadata.

[5] D5.4 Search, Access and Display Interfaces describes the tools that cater for the different access scenarios. http://www.eark-project.com/resources/project-deliverables/92-d54.

[6] The E-ARK DIP reference format refers to the E-ARK container format which is conceived to store the content information and its associated metadata. It is to a large extent built on the E-ARK Common Specification and on the E-ARK AIP Format and always holds content (i.e. one or more E-ARK DIP formats (e.g. SIARD or SMURF)).

[7] A type of software that presents part of or all of the information content of an Information Object in forms understandable to humans or systems. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf

[8] This specification is a part of E-ARK milestone "MS10 Final release of E-ARK formats and tools".

[9] D5.2 E-ARK DIP Draft Specification http://www.eark-project.com/resources/project-deliverables/31-d52

[10] D5.3 E-ARK Pilot DIP Specification http://www.eark-project.com/resources/project-deliverables/61-d53-pilot-dip-specification

[11] Common Specification, draft http://www.eark-project.com/resources/specificationdocs/67-e-ark-draft-common-specification-ver-017

[12] D3.1 E-ARK Report on Available Best Practices, http://www.eark-project.com/resources/project-deliverables/6-d31-e-ark-report-on-available-best-practices

[13] D5.1 E-ARK GAP report between requirements for access and current access solutions http://www.eark-project.com/resources/project-deliverables/3-d51-e-ark-gap-report.

[14] D3.4 Records export, transfer and ingest recommendations and SIP Creation Tools http://www.eark-project.com/resources/project-deliverables/93-d34-1
D3.3 E-ARK SMURF, http://www.eark-project.com/resources/project-deliverables/52-d33smurf

and the E-ARK Archival Information Package[15] (AIP)[16]; querying the pilot sites[17]; as well as scrutinising metadata elements from various metadata standards[18].

The top-down approach consisted in creating high-level workflows and descriptions of the generic steps in the whole Access[19] process, which is detailed elsewhere[20]. The main steps are showed here:

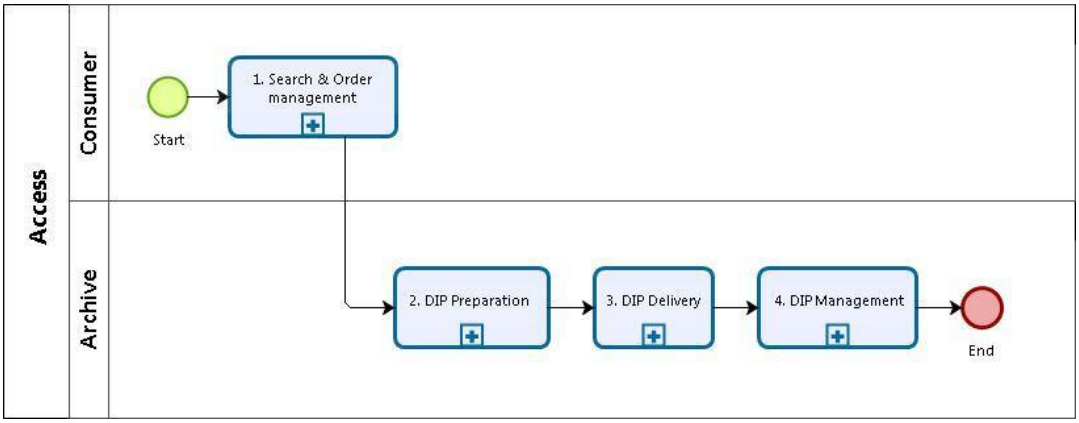

Figure 1 - Overview of the E-ARK Access process

If the reader is interested in the use cases and the DIP format requirements that underlie the format specifications, they are invited to consult respectively the aforementioned DIP draft and pilot specifications (D5.2 and D5.3).

2.2 Content information types, specifications, and tools

The E-ARK project has created a Common Specification (first row in Table 1 below), which constitutes the basis for the IP reference formats (second row). In addition the E-ARK project has created specifications for different content information types (bottom row), and created specific DIP representation formats for them (row above bottom row).

Generic	**Common specification** →	Common specification

[15] An Archival Information Package, consisting of the content information and the associated Preservation Description Information (PDI), which is preserved within an OAIS. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf

[16] D4.3 E-ARK AIP Specification, http://www.eark-project.com/resources/project-deliverables/53-d43earkaipspec-1

[17] D2.3 Detailed Pilots Specification, http://www.eark-project.com/resources/project-deliverables/60-23pilotsspec

[18] Internal E-ARK deliverable: E-ARK DIP Format Requirements - not published, but available on request.

[19] The OAIS Access functional entity contains the services and functions which make the archival information holdings and related services visible to Consumers. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf

[20] For the detailed BPMN models of the Access flow, see D2.1 General Pilot Model and Use Case Definition http://www.eark-project.com/resources/project-deliverables/5-d21-e-ark-general-pilot-model-and-use-case-definition. For both detailed illustrations and descriptions of the Access flow, see D5.2 E-ARK DIP Draft Specification http://www.eark-project.com/resources/project-deliverables/31-d52

	IP reference formats →	SIP, AIP, DIP				
Content information types and their specifications	**DIP representation formats →**	SMURF[21] ERMS[22]	SMURF SFSB[23]	SIARD[24]	SIARD; OLAP[25]	GML[26]; GeoTIFF[27]
	Content information types →	ERMS & case files	SFSB[28]	Database[29]	Data warehouse[30]	Geodata[31] (Vector[32], Raster[33])

Table 1 - Overview of the relation between content information types and the different specifications of the E-ARK project

In order to understand the content information types and their relationships to formats and tools of the E-ARK project, a table illustrating the information flow is helpful: Each column shows the transformations that each content information type undergoes on its way from the AIP format to the end-user consultation, as well as the tools that are used to perform these transformations.

[21] Semantically Marked Up Records Formats. SMURF is an IP format for ERMS systems and SFSB (simple file-system based records) conceived by the E-ARK project.

[22] Electronic Records Management System is a type of content management system and refers to the combined technologies of document management and records management systems as an integrated system.

[23] Simple File-System Based Records. Simple file-system based records: records that contain simple file-system based folders or files, including those originating from content and data management systems, such as SharePoint, that are not based on true file systems. They address the submission of computer files or folders from the file Producers rather than from an ERMS. They require manual enrichment with additional descriptive metadata.

[24] Software Independent Archiving of Relational Databases. IP format for databases. Currently there exist three versions: SIARD1.0, SIARDDK and SIARD2.0.

[25] In computing, online analytical processing, or OLAP, is an approach to answering multi-dimensional analytical (MDA) queries swiftly. OLAP is part of the broader category of business intelligence, which also encompasses relational database, report writing and data mining - https://en.wikipedia.org/wiki/Online_analytical_processing

[26] The Geography Mark-up Language: the XML grammar defined by the Open Geospatial Consortium (OGC) to express geographical features. GML serves as a modelling language for geographic systems as well as an open interchange format for geographic transactions on the Internet - https://en.wikipedia.org/wiki/Geography_Markup_Language

[27] GeoTIFF is a public domain metadata standard which allows georeferencing information to be embedded within a TIFF file. The potential additional information includes map projection, coordinate systems, ellipsoids, datums, and everything else necessary to establish the exact spatial reference for the file - https://en.wikipedia.org/wiki/GeoTIFF

[28] Simple File-System Based Records. Simple file-system based records: records that contain simple file-system based folders or files, including those originating from content and data management systems, such as SharePoint, that are not based on true file systems. They address the submission of computer files or folders from the file Producers rather than from an ERMS. They require manual enrichment with additional descriptive metadata.

[29] A database is an organised collection of data. It is the collection of schemas, tables, queries, reports, views and other objects - https://en.wikipedia.org/wiki/Database

[30] In computing, a data warehouse (DW or DWH), also known as an enterprise data warehouse (EDW), is a system used for reporting and data analysis, and is considered as a core component of a Business Intelligence environment - https://en.wikipedia.org/wiki/Data_warehouse

[31] Geodata is information about geographic locations that is stored in a format that can be used with a geographic information system (GIS). Geodata can be stored in a database, geodatabase, shapefile, coverage, raster image, or even a dbf table or Microsoft Excel spreadsheet. Throughout this document we use the abbreviation “geodata” for geographical data.

[32] Vector https://en.wikipedia.org/wiki/GIS_file_formats#Vector

[33] Raster https://en.wikipedia.org/wiki/GIS_file_formats#Raster

Content information types	AIP representation formats	DIP creation tools[34]	DIP representation formats	Access Software
Databases	SIARDDK[35]	DBPTK[36]; RDBMS[37]	SIARDDK	RDBMS; DBVTK[38]
	SIARD1.0	DBPTK; RDBMS	SIARD1.0	RDBMS
	SIARD2.0	DBPTK; RDBMS	SIARD2.0	RDBMS; DBVTK
		DBPTK; MDDBMS[39]	OLAP Cube	MDDBMS
Data warehouse	SIARD 2.0	DBPTK; MDDBMS	OLAP Cube	MDDBMS
ERMS	SMURF ERMS	Generic AIP2DIP converter	SMURF ERMS	ERMS-Viewer
SFSB	SMURF SFSB	Generic AIP2DIP converter	SMURF SFSB	SFSB-Viewer
Geodata	GML (Vector)	Generic AIP2DIP converter	GML	QGIS / GeoServer
	GeoTIFF (Raster)	Generic AIP2DIP converter	GeoTIFF / GML Frame	QGIS / GeoServer

Table 2 - Transformations that each content information type undergoes in the Access process

The ***content information types*** refer to the data types for which the E-ARK project is developing tools and formats. The content information types are ERMS systems, Simple File-System Based Records, databases, data warehouses, and geodata. Databases may or may not contain binary files (e.g. a pdf); the ERMS systems are records management systems that always contain binary files and that may or may not be MoReq compliant; geodata are found in the Vector file format or Raster graphics; data warehouse data are accessed via an OLAP strategy; and lastly, the Simple File-System Based Records originate, for example, from a writer's hard drive or a politician's inbox, and can be anything form a Word file to an EML file.

The ***AIP representation formats*** refer to different technical representations of the intellectual entity[40] which is preserved in an archive. Each of these intellectual entities is contained in at least one

[34] When 'Generic AIP2DIP converter' is indicated in this column it means that a local preservation system is employed to select and extract a specific AIP representation format from the AIP and then creates the corresponding DIP. In E-ARK, this can be the Repository of Authentic Digital Objects (RODA - http://www.roda-community.org/), the ESSArch Preservation Platform (EPP - http://epp.essarch.org/) or the E-ARK Web: https://earkdev.ait.ac.at:8443/cas/login?service (access can be granted upon request).

[35] DK is an abbreviation for "Denmark" – and this SIARD version a Danish variant of the original Swiss *SIARD1.0* format.

[36] The Database Preservation Tool Kit (DBPTK - http://keeps.github.io/db-preservation-toolkit/) is a piece of software which, from an Access perspective, enables the loading of a SIARD file into an RDBMS or into a Solr instance, making the database available in The Database Visualization Toolkit (DBVTK - http://visualization.database-preservation.com/) - a NO SQL solution for rendering databases stored in SIARD files.

[37] A relational database management system (RDBMS) is a computer software application that interacts with the user, other applications, and the database itself to capture and analyse data. A general-purpose RDBMS is designed to allow the definition, creation, querying, update, and administration of databases.

[38] The Database Visualization Toolkit - DBVTK - http://visualization.database-preservation.com/

[39] A MultiDimensional DBMS is a particular kind of *RDBMS* that is specifically geared towards OLAP (in fact MDDBMS is often used co-terminously with OLAP).

[40] A set of content that is considered a single intellectual unit for purposes of management and description: for example, a particular book, map, photograph, or database. An Intellectual Entity can include other Intellectual

representation folder of an IP. A simple example is an AIP holding one intellectual entity (e.g. a letter from Thomas Jefferson to Alexander von Humboldt) in two different representations: one representation is for example the originally scanned .jpg file and the other representation could be for example an indexable .txt file.

The ***DIP creation tools*** correspond to the tools that are needed to create the DIP from the AIP.

The ***DIP representation formats*** designate the formats that have been prepared for Access and that are ready to be rendered by Access Software.

The ***Access Software*** lists the tools that render the different DIP representation formats to the Consumer.

Each DIP representation format has its own physical folder structure as well as its own instances of metadata files (i.e. METS, PREMIS, and EAD). They all have the same foundation, which is described in the E-ARK Common Specification[41], and most of them resemble the AIP from which they were generated.

Entities; for example, a Web site can include a Web page; a Web page can include an image. An Intellectual Entity may have one or more digital representations. Source PREMIS, Intellectual entity http://www.digitizationguidelines.gov/term.php?term=intellectualentity

[41] E-ARK Common specification, draft http://www.eark-project.com/resources/specificationdocs/67-e-ark-draft-common-specification-ver-017

3 The DIP(s)

The DIP format is the last in sequence of the three IP formats defined in the OAIS reference model. The two others, the SIP and the AIP format, have - in their E-ARK context - already been defined, as noted above. In order to ensure consistency and compatibility between each other, all three formats build upon the same foundation - the E-ARK Common Specification.

The definition of an E-ARK DIP is that it corresponds to an IP which is ready to be processed by its designated Access Software; if it is not suited for processing and rendering by its designated Access Software, it is not (yet) a DIP.

This is a very generic, but handy, definition. To be more specific, the E-ARK DIP is:

1. an IP which is sent (or is ready to be sent) to the user in an Access environment;
2. supported by E-ARK tools, i.e. can be rendered by bespoke Access Software.

The DIP is to a very large extent similar to the AIP from which it is created. However, there are also significant differences.

First of all, the DIP looks like the AIP: It replicates the structure of the AIP from which it is derived. It also inherits all the metadata as well as the intellectual entities of the AIP, regardless of any format migrations that may have occurred during the AIP-DIP conversion process.

Secondly, the DIP is different from the AIP: The DIP allows for example for the inclusion of new DIP representation formats, which are more user-friendly than the AIP formats that are intended for long-term preservation purposes. It also allows for the updating of the metadata as well as for the addition of new metadata elements. Representation Information, which is required for rendering and understanding the intellectual content, is also added, and as a direct consequence, there may be a need for new folders and files, for example within the 'Documentation' folder.

3.1 Physical folder and metadata specifications

3.1.1 DIP Data Model and Physical Folder Structure

The physical structure of the E-ARK DIP must comply with the principles outlined in the E-ARK common specification. The basic E-ARK information package structure as presented in the common specification is seen in Figure 2 below. It is encapsulated in a ZIP or a TAR file, and the top-most folder carries the unique name of the DIP. The DIP consists of a "METS.xml" file that reflects the structure of the DIP and provides an inventory of its content (Packaging Information[42]). Any other metadata files are put in the "metadata/" folder (e.g. PREMIS, EAD). The 'Representations' folder contains any number of representations of the content. Any number of necessary folders and files may be placed inside these folders, and optionally a "schemas/" folder and a "documentation/" folder may be introduced.

[42] The information that is used to bind and identify the components of an Information Package. For example, it may be the ISO 9660 volume and directory information used on a CD-ROM to provide the content of several files containing content information and Preservation Description Information. OAIS, Space data and information transfer systems http://public.ccsds.org/publications/archive/650x0m2.pdf.

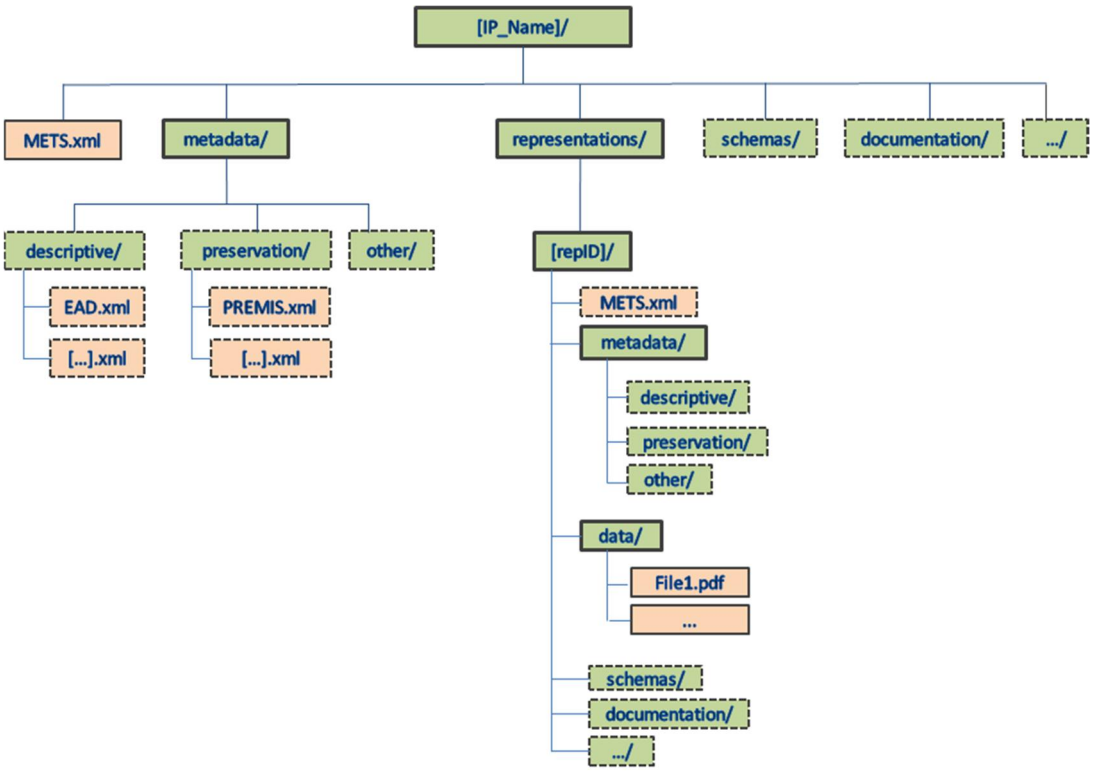

Figure 2 - E-ARK IP structure requirements

The diagram above represents the (D)IP structure. The DIP folder structure can change slightly depending on its content information type. The sections below that deal with the DIP format specifications will reflect these changes.

3.1.2 Metadata in the DIP

The DIP metadata is based upon the existing common, SIP and AIP specifications. The metadata descriptions provided in this document cover the three core metadata categories: structural[43] (METS[44]); preservation[45] (PREMIS[46]); and descriptive[47] (EAD[48])).

[43] Structural metadata describes the physical and/or logical structure of digital resources. The standard that E-ARK recommends for structural metadata is METS.

[44] Metadata Encoding and Transmission Standard. The METS schema is a standard for encoding descriptive, administrative, and structural metadata regarding objects within a digital library, expressed using the XML schema language of the World Wide Web Consortium. The standard is maintained in the Network Development and MARC Standards Office of the Library of Congress, and is being developed as an initiative of the Digital Library Federation - http://www.loc.gov/standards/mets/

[45] Preservation metadata is an essential component of most digital preservation strategies. Preservation metadata is information that supports and documents the digital preservation process - https://en.wikipedia.org/wiki/Preservation_metadata
The standard that E-ARK recommends for preservation metadata is PREMIS.

[46] The Preservation Metadata: Implementation Strategies. The PREMIS Data Dictionary for Preservation Metadata is the international standard for metadata to support the preservation of Digital Objects and ensure their long-term usability - http://www.loc.gov/standards/premis/

[47] Also named Descriptive Information in OAIS: The set of information, consisting primarily of Package Descriptions, which is provided to Data Management to support the finding, ordering, and retrieving of OAIS information holdings by Consumers - http://public.ccsds.org/publications/archive/650x0m2.pdf.
The standard that E-ARK recommends for descriptive metadata is EAD.

The DIP is based on the AIP, and structural and preservation metadata are thus always - slightly modified - present in the DIP. The METS file is in the root folder, and its schema file (mets.xsd) is in the /schemas folder. The greater part of the Access Software assumes the existence of an EAD and a PREMIS file in the root /metadata/descriptive folder and in the root /metadata/preservation/ folder respectively. Consequently it is also assumed that pertaining schema files (ead3.xsd; premis.xsd) are present in the /schemas folder.

3.1.2.1 METS

METS (Metadata Encoding and Transmission Standard) is a standard for encoding descriptive, administrative, and structural metadata expressed using the XML Schema Language.

The XML Schema for METS for an E-ARK DIP is the same XML schema as for an E-ARK AIP (i.e. the same mets.xsd file[49]).

The XML instance for METS for an E-ARK DIP is based on the same instance as an E-ARK AIP (i.e. different mets.xml files).

The differences between a METS instance for an E-ARK DIP vs an E-ARK AIP are small. Actually, most of the metadata differences between an AIP and a DIP are in the PREMIS and the EAD metadata.

The DIP specification is limited to include one and only one representation from an AIP (for which many may exist). The chosen representation is itself an E-ARK IP and therefore follows the same structure. This is reflected in the IP being migrated from an AIP to a DIP. Below is a broad overview of the METS file.

Elements				Values	Comments
mets					
	metsHdr				
		agent			software or archivist creating the DIP
	dmdSec				
		mdRef		*EAD*	information about the EAD file
	amdSec				
		mdRef		*PREMIS*	information about the PREMIS file
	fileSec				
		fileGrp		*Common Specification root*	
			fileGrp	*metadata*	
			fileGrp	*representations*	normally only one repr. in the DIP
			fileGrp	*schemas*	
			fileGrp	*documentation*	
	structMap				
		div		*metadata*	
		div		*representations*	mets pointer to mets file for the repr.
		div		*schemas*	
		div		*documentation*	

Table 3 - Broad overview of the METS file

In the following the major differences between an XML instance for METS for an E-ARK DIP vs an E-ARK AIP are listed.

Node level: mets

[48] Encoded Archival Description. A non-proprietary de facto standard for the encoding of Finding Aids for use in a networked (online) environment. EAD allows the standardization of collection information in Finding Aids within and across repositories. EAD3 About http://www.loc.gov/ead/eadabout.html.

[49] Over time the METS versions will change and therefore they may be a difference in version between the AIP and the DIP, but we assume that the AIP will be migrated and therefore that the difference will not occur or at least be small.

Element/ Attribute	Cardinality	Description and usage in Common specification	Change for DIP
mets	1..1	The root level element that is required in all METS documents	No change
@ID	0..1	Optional, no further requirements	No change
@OBJID	1..1	Mandatory in this specification. It is recommended that it be the same as the name or ID of the package (the name of the root folder). The OBJID must meet the Common Specification requirement of being unique at least across the repository	Change - the value of the attribute OBJID is changed to a new value reflecting the change in the IP from an AIP to a DIP.
@LABEL	0..1	Optional, if used should be filled with a human-readable description of the package	Change - Recommended to be ”METS file describing the DIP matching the OBJID”
@TYPE	1..1	Mandatory in this specification. The TYPE attribute must be used for identifying the type of the package (genre), for example ERMS, RDBMS, digitised construction plans. However, there is no fixed vocabulary and as such implementers are welcome to use values most suitable for their needs.	No change
@CONTENTTYPESPECIFICATION	0..1	An attribute added by this specification. It describes which content information type specification is used for the content. Values of the attribute are fixed in the following vocabulary: SMURFERMS * SMURFSFSB * SIARD1 * SIARD2 SIARDDK * GeoVectorGML * GeoRasterGeotiff NB The vocabulary is extensible as additional content information type specifications are developed.	Change to one of these: SMURFERMS SMURFSFSB SIARD1 SIARD2 SIARDDK OLAPCube GeoVectorGML GeoRasterGeotiff
@PROFILE	1..1	Mandatory in this specification. The PROFILE attribute has to have as its value the URL of the official Common Specification METS Profile.	No change

Table 4 - Differences between the AIP METS and the DIP METS on node level: mets

Node level: metsHdr

Element/ Attribute	Cardinality	Description and usage	Change for DIP
metsHdr	0..1	Element for describing the package itself	
@ID	0..1	Optional, no further requirements	No change
@ADMID	0..1	Optional, referring to the appropriate administrative metadata section, if used for metadata about the package as a whole.	No change
@CREATEDATE	1..1	Mandatory, the date of creation of the package	Change to creation date for DIP
@LASTMODDATE	0..n	Mandatory if relevant (in case the package has been modified)	Change to date of modification, if DIP has been created in several steps.
@RECORDSTATUS	0..1	Optional, no further requirements	No change
@PACKAGETYPE	1..1	An attribute added by the Common Specification for describing the type of the IP. The vocabulary to be used contains values:SIP, AIP, DIP, AIU, AIC The vocabulary is managed by the DAS Board and will be updated when required.	Change to DIP

agent	1..n	The *metsHdr* must include at least one agent describing the software which has been used to create the package (TYPE=”OTHER” ROLE=”CREATOR” OTHERTYPE=”SOFTWARE”). Description of all other agents is optional.	
agent/ @ID	0..1	An ID for the agent.	No change
agent/ @ROLE	1..1	The role of the agent. The Common Specification requires describing at least one agent with the *agent/@ROLE* value “CREATOR”. For other (optional) occurrences of *agent* this attribute shall use a value from the fixed list provided by METS.	No change
agent/ @OTHERROLE	0..1	A textual description of the role of the agent in case the value of *agent/@ROLE* is “OTHER”.	No change
agent/ @TYPE	0..1	The Common Specification requires that at least one instance of the *agent* element includes the *agent/@TYPE* attribute with the value “OTHER”. In other occurrences of the *agent* element the attribute is optional. If used, values defined in official METS documentation shall be followed (“individual”, “organisation”, “other").	No change
agent/ @OTHERTYPE	0..1	The Common Specification requires that at least one instance of the *agent* element includes the *agent/@OTHERTYPE* attribute with the value “SOFTWARE”. In other occurrences this attribute shall only be used in case the value of agent/@TYPE is “OTHER”.	No change
agent/ name	1..1	The name of the agent. In the Common Specification occurrence of the *agent* element this element must provide the name of the software tool which was used to create the IP.	Change to name of the agent that created the DIP from the AIP
agent/ note	0..1	Additional information about the agent. We recommend using this element to provide version information for the tool which was used to create the IP.	Change to version no. for the DIP creation software.
altRecordID	0..n	A container for an alternative ID for the package content.	No change
altRecordID/ @ID	0..1	An ID for the *altRecordID* element within the METS document.	No change
altRecordID/ @TYPE	0..1	Used to describe the type of ID assigned. It is recommended to use the Library of Congress vocabulary for this element when used.	No change
metsDocumentID	0..1	A unique identifier for the METS document itself.	No change
metsDocumentID/ @ID	0..1	The ID of the *metsDocumentID* element.	No change
metsDocumentID/ @TYPE	0..1	The type of the identifier assigned to the element.	No change

Table 5 - Differences between the AIP METS and the DIP METS on node level: metsHDR

Node level: dmdSec

Element/ Attribute	Cardinality	Description and usage	Change for DIP
dmdSec	0..n	Must be used if descriptive metadata about the package content is available.	No change. The DIP spec. assumes the

		NOTE: According to official METS documentation each metadata section must describe one and only one set of metadata. As such, if implementers want to include multiple occurrences of descriptive metadata into the package this must be done by repeating the whole *dmdSec* element for each individual metadata.	EAD is in the AIP. The EAD is needed to find the AIP with the desired content.
@ID	1..1	Mandatory, identifier must be unique within the package	No change
@GROUPID	0..1	Can be used to group together different metadata sections.	No change
@ADMID	0..1	In case administrative (provenance) metadata is available and described within METS about changes to the descriptive metadata, this element must reference the appropriate ID of the administrative metadata section.	No change
@CREATED	1..1	Required by the Common Specification. Creation date of the metadata in this section, needed to track changes to metadata files.	No change
@STATUS	0..1	Status of the metadata. Recommended for use to indicate currency of package. If used it is recommended to use one of the two values “SUPERSEDED” or “CURRENT”.	No change
/mdRef	0..1	Reference to the descriptive metadata file stored in the “**metadata**” folder of the IP. In each occurrence of the *dmdSec* exactly one of the elements mdRef or mdWrap must be present. The Common Specification recommends the use of *mdRef* over *mdWrap*	
@ID	0..1	Unique ID for the *mdRef* section within the METS document.	No change
@MIMETYPE	0..1	The IANA media type for the external file.	No change
@LABEL	0..1	A name for the referenced file.	No change
@XPTR	0..1	Locates the point within a file to which the *mdRef* element refers, if applicable, using any valid XPointer scheme.	No change
@LOCTYPE	1..1	Specifies the locator type used in the xlink:href which points to the file.	No change
@OTHERLOCTYPE	0..1	Required when *mdRef/@LOCTYPE* =“OTHER”.	No change
@MDTYPE	1..1	Specifies the type of metadata in the linked file. Values should be taken from the METS list provided.	No change
@MDTYPEVERSION	0..1	The version of the metadata type described in MDTYPE	No change
@OTHERMDTYPE	0..1	The type of metadata when MDTYPE=”OTHER”	No change
@SIZE	0..1	Size of linked file in bytes	No change
@CREATED	0..1	Date the linked file was created	No change
@CHECKSUM	0..1	The checksum of the linked file	No change
@CHECKSUMTYPE	0..1	The type of checksum used for calculating the checksum of the linked file	No change
/mdWrap	0..1	Wrapper for descriptive metadata embedded into the METS document. In each occurrence of the *dmdSec* exactly one of the elements mdRef or mdWrap must be present. The Common Specification recommends the use of *mdRef* over *mdWrap*	No change
@ID	0..1	Unique ID for the mdWrap section within the METS document.	No change
@MIMETYPE	0..1	The IANA mime type for the wrapped metadata.	No change
@LABEL	0..1	A name for the associated metadata.	No change

@MDTYPE	1..1	Specifies the type of embedded metadata. Values should be taken from the METS list provided.	No change
@MDTYPEVERSION	0..1	The version of the metadata type described in MDTYPE	No change
@OTHERMDTYPE	0..1	The type of metadata when MDTYPE=”OTHER”	No change
@SIZE	0..1	Size of associated metadata in bytes	No change
@CREATED	0..1	Date the embedded metadata was created	No change
@CHECKSUM	0..1	The checksum of the wrapped content	No change
@CHECKSUMTYPE	0..1	The type of checksum used for calculating the checksum of the embedded metadata	No change
/binData	0..1	A wrapper element to contain Base64 encoded metadata	No change
/xmldata	0..1	A wrapper element to contain XML encoded metadata	No change

Table 6 - Differences between the AIP METS and the DIP METS on node level: dmdsec

Node level: amdSec

Element/ Attribute	Cardinality	Description and usage	Change for DIP
amdSec	0..n	In case administrative / preservation metadata is available, it must be described using the *amdSec* element.	Mandatory for DIP
@ID	0..1	Unique ID for the *amdSec* within the METS document	No change
digiprovMD	0..n	The Common Specification recommends the use of PREMIS metadata for recording information about preservation events. If used, PREMIS metadata must appear in a *digiprovMD* element, either embedded or linked. It is mandatory to include one *digiprovMD* element for each external file in the “**metadata/preservation**” folder, or for each embedded set of PREMIS metadata.	The DIP spec. assumes that a PREMIS file is updated or created during the AIP to DIP migration
techMD	0..n	The use of *techMD* is not recommended. Instead, detailed technical metadata should be included into or referenced from appropriate PREMIS files	No change
rightsMD	0..n	Optional. The Common Specification recommends including a simple rights statement which describes the overall access status of the package with the following values: - *Open, Closed, Partially closed, Not known*. However, the exact schema and element is up to individual implementations to decide	No change
sourceMD	0..n	Optional, no further requirements	No change

Table 7 - Differences between the AIP METS and the DIP METS on node level: amdsec

The following attributes are available for use with each of the four specific metadata areas listed above (*xxx* below stands for *amdSec/digiprovMD*, *amdSec/techMD*, *amdSec/rightsMD* and *amdSec/sourceMD*).

Element/ Attribute	Cardinality	Description and usage	Change for DIP
xxx/ @ID	1..1	Mandatory for each of the four elements *amdSec/digiprovMD*, *amdSec/techMD*, *amdSec/rightsMD* and *amdSec/sourceMD*. Identifier must be unique within the package	No change
xxx/ @GROUPID	0..1	Optional, no further requirements	No change
xxx/ @ADMID	0..1	In case administrative (provenance) metadata is available and	No change

		described within METS about changes to the metadata occurrence described here, this element must reference the appropriate ID of the administrative metadata section.	
xxx/ @CREATED	0..1	Optional, no further requirements	No change
xxx/ @STATUS	0..1	Recommended for describing currency of metadata. If used, must include one of the two values “superseded” or “current”	No change

Table 8 - Available attributes

Node level: filesec

Element/ Attribute	Cardinality	Description and usage	Change for DIP
fileSec	0..1	Recommended to include one *fileSec* element in each METS file	No change
@ID	0..1	Recommended. The identifier must be unique within the METS file.	No change
/fileGrp	1..n	This specification requires that one specific occurrence of the *fileGrp* element is included as described above. Implementers are welcome to define and add additional file groups necessary for internal purposes. The main *fileGrp* element includes additional nested *fileGrp* elements, one for each folder of the package (except metadata described in *amdSec* and *dmdSec*).	No change
/fileGrp/@ID	0..1	Recommended, identifier must be unique within the package	No change
/fileGrp/ @VERSDATE	0..1	Version date of the file grouping	No change
/fileGrp/ @ADMID	0..1	In case administrative metadata is available and described within METS about the file group, this element must reference the appropriate ID of the administrative metadata section.	No change
/fileGrp/ @USE	1..1	Recommended in Common Specification with one occurrence bearing the values “Common Specification root” (for the root *fileGrp* element and the names of appropriate folders for nested *fileGrp* occurrences.	No change
/fileGrp/file	1..n	The Common Specification requires that *fileGrp* must contain at least one *file* element either pointing to content files with *FLocat* or wrapping the content files using *FContent*	No change
/fileGrp/ file/@ID	1..1	Mandatory, must be unique across the package	Add/update
/fileGrp/ file/ @MIMETYPE	1..1	The IANA mime type for the wrapped or linked file. Required by the Common Specification.	Add/update (reflecting change of format)
/fileGrp/ file/ @SEQ	0..1	Used to describe the sequence of files listed within the *fileGrp* element	No change
/fileGrp/ file/ @SIZE	1..1	Size of the linked or embedded file in bytes. Required by the Common Specification	Add/update
/fileGrp/ file/ @CREATED	1..1	Date the embedded/linked file was created. Required by the Common Specification	Add/update
/fileGrp/ file/ @CHECKSUM	1..1	The checksum of the embedded/linked file. Required by the Common Specification	Add/update
/fileGrp/ file/ @CHECKSUMTYPE	1..1	The type of checksum used for the embedded/linked file. Required by the Common Specification	No change (unless type is actually changed)

/fileGrp/ file/ @OWNERID	0..1	Unique ID of the file assigned by its owner	No change
/fileGrp/ file/ @ADMID	0..1	In case administrative metadata is available and described within METS about the file, this element must reference the appropriate ID of the administrative metadata section.	No change (e.g. link to EAD)
/fileGrp/ file/ @DMDID	0..1	Value for the ID attribute of the *dmdSec* containing metadata describing the content files listed in the file element.	Add/Update (e.g. link to PREMIS)
/fileGrp/ file/ @GROUPID	0..1	Provides an ID for a *fileGrp* containing related files.	No change
/fileGrp/ file/ @USE	0..1	Statement about intended use of the files	No change
/fileGrp/ file/ FLocat		The location of each external file must be defined by the <FLocat> element using the same rules as for referencing metadata files. All references to files should be made using the XLink href attribute and the file protocol using the relative location of the file. Example: xlink:href="file:schemas/mets.xsd"	Add/Update
/fileGrp/ file/ FLocat/ @ID	0..1	An ID for the <FLocat> element	Add/Update
/fileGrp/ file/ FLocat/ @LOCTYPE	1..1	Mandatory locator pointing to the external file.	Add/Update
/fileGrp/ file/ FLocat/@OTHERLOCTYPE	0..1	Description of the type of locator used	No change
/fileGrp/ file/ FLocat/ @USE	0..1	Statement about intended use of the linked file	No change
/fileGrp/ file/ FContent	0..1	Used for identifying content files wrapped within the METS file. The content file must be either encoded in base64 and inside an <binData> wrapper, or encoded in XML and included within an <xmlData> wrapper.	No change - not recommended
/fileGrp/ file/ FContent/ @ID	0..1	An ID for the <FContent> element	No change - not recommended
/fileGrp/ file/ FContent/ @USE	0..1	Statement about intended use of the embedded file	No change - not recommended

Table 9 - Differences between the AIP METS and the DIP METS on node level: filesec

Node level: structmap

Element/ Attribute	Cardinality	Description and usage	Change for DIP
structMap	1..n	Each METS file needs to include exactly one *structMap* element used exactly as described in this table. Institutions can add their own additional custom structural maps as separate *structMap* sections.	No change
@ID	0..1	Optional, but if used must be unique within the package	No change
@TYPE	1..1	Mandatory in this specification. The value must be “physical”	No change
@LABEL	1..1	Mandatory in this specification. The value must be “Common Specification structural map”	No change

/div	0..n	Each folder (and sub-folder) within the package must be represented by an occurrence of the element. Please note that sub-folders must be represented as nested div elements. Example: <structMap TYPE="physical" LABEL="Common Specification structural map"> 	No change
/div/@ID	1..1	Mandatory, identifier must be unique within the package	No change
/div/@TYPE	0..1	No specific requirements	No change
/div/@LABEL	1..1	Mandatory, value must be the name of the folder (“**metadata**”, “**descriptive**”, “**schemas**”, “**representations**”, etc). The LABEL value of the first *div* element in the package is the ID of the package	No change
/div/@DMDID	0..1	ID attribute values identifying the *dmdSec*, elements in the METS document that contain or link to descriptive metadata pertaining to the structural division represented by the current *div* element	Add/Update
/div/@ADMID	0..1	No specific requirements	Add/Update
/div/@ORDER	0	Not used in the specific Common Specification *structMap* occurrence	No change
/div/ @ORDERLABEL	0	Not used in the specific Common Specification *structMap* occurrence	No change
/div/ @CONTENTIDS	0..1	IDs for the content in this division. No specific use requirements.	No change
/div/fptr	0..n	If the folder which is described by the *div* element includes computer files these must be referenced by using the *fptr* element. The only exception is the description of representations (see below for the use of *mptr*). The *fptr* child elements *par*, *seq* and *area* must not be used.	Add/Update
/div/fptr/@ID	0..1	No specific requirements	No change
/div/ fptr/ @FILEID	1..1	Mandatory, must be the ID used in the appropriate *file* or *mdRef* element	Add/Update
/div/ fptr/ @CONTENTIDS	0..1	IDs for the content referenced by this *fptr* element. No specific requirements	No change
/div/div/mptr	0..n	In the case of describing representations within the package (i.e. representations/representation1) the content of the representations must not be described. Instead the of the specific representation should include one and only one occurrence of the <mptr> element, pointing to the appropriate representation METS file. The references to representation METS files must be made using the XLink href attribute and the file protocol using the relative location of the file. Example: xlink:href="file:representation/ representation1/mets.xml" The XLink type attribute is used with the fixed value “simple”. Example: xlink:type="simple" The LOCTYPE attribute is used with the fixed value ”URL”	No change
/div/ div/ mptr/@ID	0..1	Unique ID for this element	No change
/div/ div/ mptr/	0..1	The locator type used in the xlink:href attribute	No change

@LOCTYPE			
/div/ div/ mptr/ @OTHERLOCTYPE	0..1	Locator type in xlink:href when LOCTYPE="OTHER"	No change
/div/ div/ mptr/ @CONTENTIDS	0..1	The content ID for the content represented by the *mptr* element.	No change

Table 10 - Differences between the AIP METS and the DIP METS on node level: structmap

3.1.2.2 PREMIS

PREMIS is a standard that mainly caters for long-term preservation and technical usability, which for example is used to facilitate a range of preservation strategies including migration and emulation.

From an Access perspective, PREMIS especially satisfies the requirements pertaining to the recording of Representation Information. It is practical to state in a formalised and consistent way how the Access Software should behave and where it should look when dealing with different pieces of information, such as which representation formats are included in the DIP. Therefore all E-ARK Access Software assumes the availability of PREMIS metadata according to the specification below.

3.1.2.2.1 Metadata regarding Representations and Access Software

In PREMIS, a representation is a "set of files, including structural metadata, needed for a complete and reasonable rendition of an Intellectual Entity."[50].

One of the core concepts in PREMIS is the above formulated definition of a representation, but it is also important to note that the E-ARK Common Specification Information Package structure also incorporates physical management of different representations. When implementing PREMIS in E-ARK packages one must therefore choose if there must exist PREMIS files at representation level or at root level only (see Common Specification) and one must also choose how fine-grained each description should be.

In PREMIS, a representation is indicated using the semantic unit "1.1 objectIdentifier". In E-ARK Access, as already mentioned, the DIP representation formats are SMURF ERMS, SMURF SFSB, SIARD1.0, SIARD2.0, SIARDDK, OLAP, GML, and GeoTIFF. It is important to emphasise that the E-ARK project has neither created specifications nor tools for specific file formats[51] (fine-grained descriptions), but only for the aforementioned DIP *representation formats*.

Hence, the Access Software developed by the E-ARK project does guarantee the rendering of the E-ARK representations, but not of specific file formats contained *inside* an E-ARK representation. As an example, the SMURF ERMS could contain several file formats unknown to the E-ARK IP Viewer[52] (note: even though this is unlikely, because archives generally make sure that the number of file formats that they preserve is limited and their use widespread. As such for example PDF or TIFF files would be rendered).

To enable rendition, three pieces of information are needed in PREMIS: One identifying the representation to be rendered; one identifying the software to enable this; and one establishing a relationship between the two.

[50] PREMIS Editorial Committee (2015). "PREMIS Data Dictionary for Preservation Metadata", p.8.

[51] For example PDF or JPG.

[52] The generic Access tool that allows for the rendering of the DIP reference format, ie. the folder structure, descriptive metadata,and the most common file formats, cf. D5.4 Search, Access and Display Interfaces. http://www.eark-project.com/resources/project-deliverables/92-d54

The descriptions below therefore show how to:

1. Describe which DIP representation format is contained in the DIP (description 1 below);
2. Describe which piece(s) of Access Software is/are needed to render a specific DIP representation format. Several pieces of software may indeed be needed (description 2 below);
3. Describe the relationship between the DIP representation format and its Access Software (description 3 below).

3.1.2.2.1.1 Description 1 - The recording of DIP representation formats

In order to describe the specific DIP representation format the semantic component "1.4 significantProperties" is used. An example is:

```
<object xsi:type="representation">
   <objectIdentifier>
       <objectIdentifierType>filepath</objectIdentifierType>
       <objectIdentifierValue>xlink:href="representations\AVID.SA.18006.rep0"</objectIdentifierValue>
   </objectIdentifier>
   <significantProperties>
       <significantPropertiesType>DIP representation format</significantPropertiesType>
       <significantPropertiesValue>SIARD2</significantPropertiesValue>
   </significantProperties>
   <!-- PREMIS file continues but further elements left out in this example-->
```

Table 11 - Recording of DIP representation formats

Note that the object type is "representation" and that the objectIdentifierType value is "filepath", which according to the AIP specification is an IP scope value. The objectIdentifierValue is the filepath to the representation folder or could be a filepath to a file.

3.1.2.2.1.2 Description 2 - The recording of Access Software

In PREMIS 3.0 a description of an environment has become an object itself, so that both non-environmental objects and environmental objects exist. Access Software is therefore an environmental object which per default is an intellectual entity. The semantic unit "1.9 environmentFunction" is conceived to describe the environment object(s) with different levels of granularity. It is suggested to use the vocabulary from Library of Congress[53]. The semantic unit "1.10 environmentDesignation" is used for information identifying the environment by using human-readable language which can be expected to be understood outside of a digital repository.

See the example which follows this vocabulary:

```
<object xsi:type="intellectualEntity">
     <objectIdentifier>
         <objectIdentifierType>local</objectIdentifierType>
         <objectIdentifierValue>DBVTK</objectIdentifierValue>
     </objectIdentifier>
     <environmentFunction>
         <environmentFunctionType>software</environmentFunctionType>
         <environmentFunctionLevel>1</environmentFunctionLevel>
     </environmentFunction>
```

[53] Library of Congress. environmentFunctionType. Retrieved the 18th of January 2017 at: Environment Function Type http://id.loc.gov/vocabulary/preservation/environmentFunctionType.html

```
     <environmentFunction>
         <environmentFunctionType>software application</environmentFunctionType>
         <environmentFunctionLevel>2</environmentFunctionLevel>
     </environmentFunction>
    <environmentDesignation>
       <environmentName>Database Visualization Toolkit</environmentName>
       <environmentVersion>2.4.1</environmentVersion>
       <environmentDesignationNote>Lightweight web viewer for relational databases, specially
if preserved in SIARD 2, that uses SOLR as a backend, and allows browsing, search, and export.
Documentation at github.com/eark-project/software/DBVTK</environmentDesignationNote>
       </environmentDesignation>
</object>
```

Table 12 - Recording of Access Software

3.1.2.2.1.3 Description 3 - The recording of the relation between the representations and the Access Software

In order to establish a connection between the DIP representation format to be rendered and the Access Software to render it, it is necessary to use the semantic unit "1.13 relationship". The relationship element can bind both non-environmental objects together with environmental objects and it can bind environmental objects together with other environmental objects. The following example shows how the DIP representation format from Table 11 can be related to the Access Software from Table 12:

```
<object xsi:type="representation">
   <objectIdentifier>
      <objectIdentifierType>filepath</objectIdentifierType>
      <objectIdentifierValue>xlink:href="representations\AVID.SA.18006.rep0"</objectIdentifierValue>
   </objectIdentifier>
   <significantProperties>
      <significantPropertiesType>DIP representation format</significantPropertiesType>
      <significantPropertiesValue>SIARD2</significantPropertiesValue>
   </significantProperties>
   <!-- The following is the relation between the software and the DIP representation -->
   <relationship>
       <relationshipType>dependency</relationshipType>
       <relationshipSubType>requires</relationshipSubType>
       <relatedObjectIdentifier>
           <relatedObjectIdentifierType>local</relatedObjectIdentifierType>
           <relatedObjectIdentifierValue>DBVTK</relatedObjectIdentifierValue>
       </relatedObjectIdentifier>
       <relatedEnvironmentPurpose>render</relatedEnvironmentPurpose>
   </relationship>
</object>
```

Table 13 - Recording of Access Software

As can be seen in Table 13 (above) the nature of the relationship, <relationshipType> is used (value, e.g. 'dependency'); intimately linked to this is also the indication of a <relationshipSubType>, e.g. 'requires'.

In order to identify the Access Software, which is used to render the representation, the <relatedObjectIdentifier> is employed; and the <relatedEnvironmentPurpose> gives us a hint about what the purpose is (here: to 'render').

Since it is not always possible to render the DIP representation formats with one piece of Access Software, it can be necessary to model software dependencies and sequences between several pieces of software in PREMIS.

3.1.2.3 EAD

Descriptive metadata are used to describe the intellectual contents of archival holdings, and they support finding and understanding individual information packages. The E-ARK project allows for the inclusion of any kind of descriptive metadata in the E-ARK IP[54]. These go into the 'descriptive/' folder as seen in the example below, Figure 3 (cf. EAD.xml).

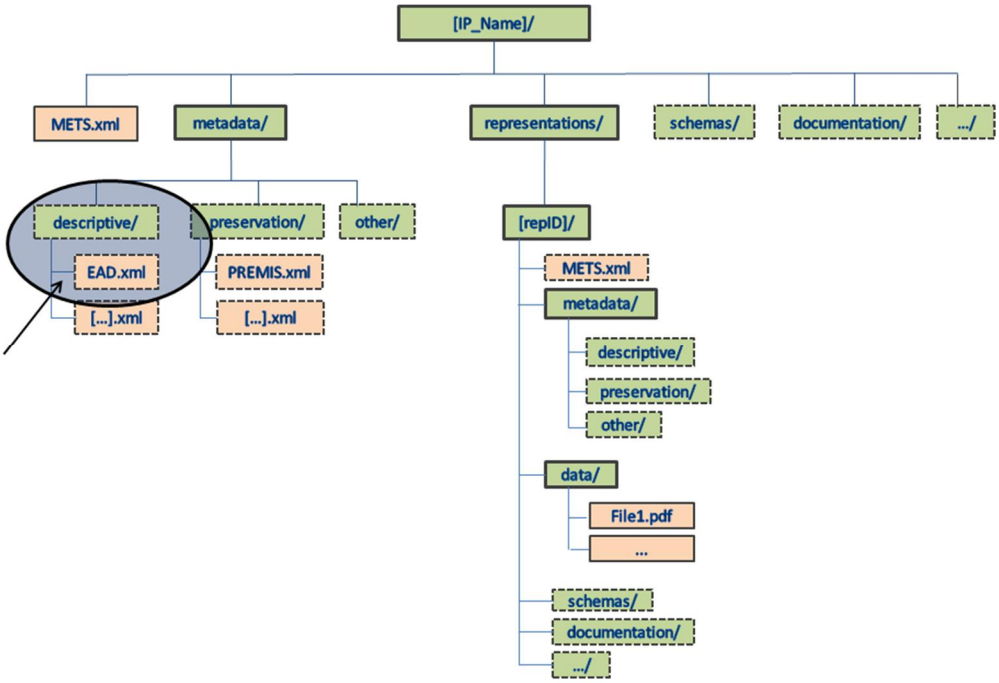

Figure 3 - E-ARK IP descriptive metadata

The METS descriptive metadata element <dmdSec> references descriptive metadata as seen in Figure 4 below and as such descriptive metadata are not to be included into the METS file.

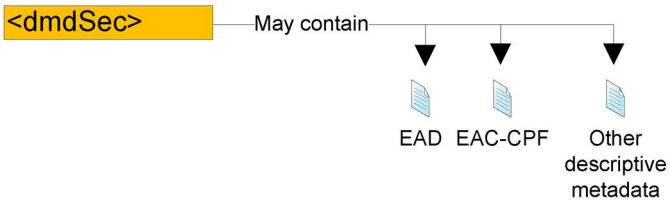

Figure 4 - METS descriptive metadata

The EAD file has three main inputs (Figure 5 below):

- Archival descriptions. Contains main archival descriptions (including metadata about aggregations and classification).

[54] For E-ARK pilots, most of the Access Software used the EAD3 standard. They can however be tweaked to use other descriptive metadata.

- Content. Contains links to computer files and folders as <dao> elements and @base attributes respectively.
- Additional metadata. Specific information that does not fit into the EAD3 standard elements can be saved as <odd> elements or localtype attributes in EAD3 elements (see localtype example below in section 'Search').

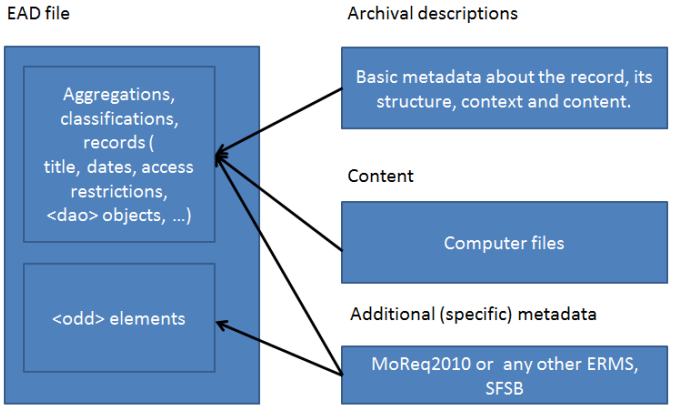

Figure 5 - Inputs to the EAD file

3.1.2.3.1 Tools

The tools that the E-ARK project is building will by default only be able to cope with EAD3, because EAD3 is the descriptive metadata standard that E-ARK has chosen. This does not mean that the tools cannot be configured so they can also cope with other descriptive metadata standards. For example, in E-ARK web there is a generic task for descriptive metadata. It would be easy to adapt this task to Dublin Core, for example, if required. The only thing that is required to do so is a convention which defines how to determine the metadata that belong to a file item in the context of a metadata format.

The tools in E-ARK that use EAD are:

- ERMS Export Module (EEM)
- EAD Editor
- RODA
- ESS tools
- E-ARK Web including Apache Hadoop, Lily and Solr.
- Access Software
    - Search GUI
    - Order Management Tool
    - DIP Viewer

3.1.2.3.2 Search

To support the development of a search interface, it is required to make certain metadata elements available in the Solr index[55], either by using the Lily indexer[56] or via the AIP indexing task, which adds content to the Solr index by creating one document[57] per contained file in the IP. These Solr documents

[55] Apache Solr Reference Guide Solr Indexing
https://cwiki.apache.org/confluence/display/solr/Introduction+to+Solr+Indexing
[56] The dm-file-ingest component: https://github.com/eark-project/dm-file-ingest
[57] A document in Solr terminology is Solr's basic unit of information, which is a set of data that describes something. A document about a person, for example, might contain the person's name, biography, favorite color, and shoe size. A document about a book could contain the title, author, year of publication, number of pages, and so on. Cf. Apache

include basic properties such as "path", "package", "contentType", "size", and can be further enriched by running a subsequent job which parses EAD metadata files to search for <dao> tags. If such a tag is found, the metadata fields of the first c-level tag in the ancestry path of the "data" element (e.g. the "title" field) are added to the Solr document. If no <dao> element is found, the entire EAD/c (component) is indexed and associated with all files in a given IP. This aims to support scenarios where no <dao> elements are provided as part of the description.

The following fields have been mapped to Solr and created in Lily_Solr and are thus indexed and searchable from the E-ARK Search GUI:

EAD element	Value access path	SolR field
ead:unitid	.	eadid_s (String)
ead:unittitle	.	eadtitle_s (String)
ead:unitdate	.	eaddate_s (String)
ead:unitdatestructured	ead:datesingle	eaddatestructuredfrom_dt (Date)
ead:unitdatestructured,	ead:datesingle	eaddatestructuredto_dt (Date)
ead:unitdatestructured	ead:daterange/ead:fromdate	eaddatestructuredfrom_dt (Date)
ead:unitdatestructured	ead:daterange/ead:todate	eaddatestructuredto_dt (Date)
ead:origination	ead:*/ead:part	eadorigination_s (String)
ead:abstract	.	eadabstract_t (Text)
ead:accessrestrict	ead:head	eadaccessrestrict_s (String),
ead:([Cc][0,1][0-9]|C)	@level (attribute)	eadclevel_s (String)

Table 14 - EAD to Solr mapping

Note that the element selector can be a regular expression: ([Cc][0,1][0-9]|C) matches either C01, C02, …, C20 or the C-element without numeric suffix. The value access path "@level (attribute)" means that the value of the attribute "level" of the selected C- or C{n}element is accessed. The expression ead:*/ead:part allows accessing the element's value with any EAD child element of ead:origination which has the child element ead:part.

Elements which are not part of the original EAD3 can be represented by <odd> elements, cf. the SMURF profile[58].

For example a new element for keywords:

```
<c level="item">
<did>
…
```

Solr Reference Guide Overview of Documents, Fields, and Schema Design
https://cwiki.apache.org/confluence/display/solr/Overview+of+Documents,+Fields,+and+Schema+Design

[58] D3.3 E-Ark SMURF http://www.eark-project.com/resources/project-deliverables/52-d33smurf

```
<!--System set date and time when the entity was created. “Created” is user-defined value.-->
<unitdate datechar="created">2004-05-25T00:00:00</unitdate>
…
<!--Other Descriptive Data. “Keyword” attribute is user-defined.-->
<odd localtype="Keyword">
   <list>
      <item>keyword A</item>
      <item>keyword B</item>
      <item>keyword C</item>
   </list>
</odd>
```

Table 15 - EAD example of new element for keywords

3.1.2.3.3 Hierarchical archival descriptions

The component tag (<c>) is “An element that designates a subordinate part of the materials being described.”[59]. The hierarchical archival descriptions (units of description) will be addressed using the unnumbered component tag (<c>).

The Access Software will be developed so that it can address all hierarchical levels in a description of the Archive's collections, thus providing user friendly information for all the levels of intellectual content, which have effectively been described in the descriptive metadata file.

The example below shows how the descriptions of the hierarchical levels can be displayed in the IP Viewer:

Figure 6 - Illustration of the E-ARK use of the EAD component tag <c>

The E-ARK project has chosen the unnumbered component tag (<c>) as opposed to the numbered one (<cNN>) because it is more generic, less resource demanding to implement, and because it provides more flexibility and more interoperability: First of all there is no upper limit to the number of levels when choosing <c> (whereas there are 12 in <cNN>). Also, a hierarchy is implicit from the archive's vocabulary (series, sub-series, etc.) and reflects each local archive’s way of describing descriptive units; the <c> component is defined as an argument using the level (@level) attribute[60], e.g. <c level="file">. With <cNN> we would need to define the meaning for each number and it would probably be too naive to expect that a

[59] EAD3 <c> http://www.loc.gov/ead/EAD3taglib/index.html#elem-c

[60] EAD3 @level https://www.loc.gov/ead/EAD3taglib/index.html#attr-level

prescriptive E-ARK solution would be adopted by all. Furthermore, it is also complicated to introduce new levels between the existing ones when using the <cNN> tag. Lastly, the Archives Portal Europe (APE) also uses the unnumbered tag in apeEAD and it is assumed that any compliance with APE is welcomed by the archival community.

EAD uses aggregation values as the "level" attribute on the elements <archdesc> and <c> to specify the aggregation level at which description belongs.

```
<archdesc level="fonds">
       …
   <dsc>
        <c level="series">
              …
          <c level="file">
          Records and computer files
          </c>
        </c>
   </dsc>
</archdesc>
```

Table 16 - EAD example of the use of the "level" attribute to specify the aggregation levels

The exact names of aggregation levels depend on the agreements between data producers and archives. EAD3 has defined a set of values (class, collection, file, fonds, item, otherlevel, recordgrp, series, subfonds, subgrp, subseries) for that purpose, but it allows using other values as well if they are defined with the @otherlevel attribute.

```
<archdesc level="collection">
       <did>
              <abstract>...</abstract>
       </did>
       <dsc>
              <c level="series">
                     <did>
                            <abstract></abstract>
                     </did>
                     <c otherlevel="case"> <!--A new aggregation level-->
                            <did>
                                   <abstract>Records and computer files from ....</abstract>
                            </did>
                     </c>
              </c>
       </dsc>
</archdesc>
```

Table 17 - EAD example of the use of <otherlevel>

3.1.2.3.4 Referencing files and folders from EAD

In order to be able to reference physical intellectual entities from within an EAD file, two mechanisms are being used: The digital archival object <dao> tag and the @base attribute.

@base is used to reference folders; <dao> is used to reference files.

The objectives of these mechanisms are to:

- Connect findings in the Finding Aid to specific descriptive units in the IPs (be it files (<dao>) or folders (@base));
- Allow for an appropriate visualisation of both data and EAD metadata inside the E-ARK Access Software GUIs.

The digital archival object <dao> tag is "A child element of <did> used for linking to born digital records or a digital representation of the materials being described."[61]. The <did> element is a wrapper element that encloses information essential for identifying the record:

3.1.2.3.4.1 The <did> tag

The <did> tag is "a wrapper element that encloses information essential for identifying the material being described".[62]. It binds together the elements that describe archival materials.

```
<did>
  <!--Unique identifier for a record that is generated automatically by the system.-->
  <unitid localtype="current">ERA.14-4-3-1-1</unitid>
  <!--The identifying name or title of the record.-->
  <unittitle>Final report</unittitle>
  <!--System set date and time when the entity was created. Can be represented as unitdatestructured
or unitdate.-->
  <unitdate datechar="created">2004-05-25T00:00:00</unitdate>
  <!--The unit used to describe the extent of the record (e.g MB, pages, num of files/components)-->
  <physdescstructured coverage="whole" physdescstructuredtype="spaceoccupied">
      <quantity>0.4453</quantity>
      <unittype>MB</unittype>
  </physdescstructured>
  <!--<daoset> allows grouping multiple links to born digital records or digital representations of
the materials being     described. If only one link is present then the <daoset> element can be
replaced by <dao>.-->
  <daoset label="Digital Objects">
      <dao … />
      <dao … />
  </daoset>
</did>
```

Table 18 - EAD example of the <did> element

3.1.2.3.4.2 The <dao> tag and the @base attribute

Both mechanisms are used to link to physical entities within an IP. The dao element links to files while the base attribute links to folders. It allows the Access Software to display metadata pertaining to specific files or folders so that the Consumer can easily understand what the selected element is about.

The dao element

The <dao> element is a linking element that uses href to connect to born digital records or digital representations of the record.

```
<dao id="dad603af-037b-44d5-8993-5754a42b3962" daotype="borndigital" linktitle="Report"
href=file:../../representations/rep1/data/Example1.docx" />
```

Table 19 - EAD example of the <dao> element and the href link

[61] EAD3 <dao> http://www.loc.gov/ead/EAD3taglib/index.html#elem-dao

[62] EAD3 <did> https://www.loc.gov/ead/EAD3taglib/index.html#elem-did

The ID attribute represents a machine-processable unique identifier for the file. The daotype specifies if a file is born digital (borndigital), digitized from physical holdings (derived) or other (otherdaotype). For example, for scanned files we can point out that the files are not “original”.

```
<dao id="baa703af-037b-44d5-8993-5754a42b3962" daotype="derived" linktitle="Page10"
href="file:../../representations/rep1/data/Page10.tif" />
```

Table 20 - EAD example of the <daotype> attribute

In more complex cases (e.g. files migration) it is recommended to use both attributes (daotype, otherdaotype) as using only one attribute may be not enough for describing the file origin. For example, if a file is a result of a file format conversion from DOCX to PDF/A then we recommend using daotype="borndigital" otherdaotype="migrated" because the PDF/A is a born-digital and a migrated file in the same time.

```
<dao id="dad603af-037b-44d5-8993-5754a42b3962" daotype="borndigital" linktitle="Report"
href="file:../../representations/rep1/data/Report.doc" />

<dao id="aad248bf-037b-34d5-7993-5754b42b3971" daotype="borndigital" otherdaotype="migrated"
linktitle="Report" href="file:../../representations/rep2/data/Report.pdf" />
```

Table 21 - EAD examples of the <daotype> and <otherdaotype> attributes

The base attribute

The @base attribute is “Used to specify a base URI that is different than the base URI of the EAD instance. This allows any relative URIs provided on attributes of a specific element or its descendants to be resolved using the URI provided in that element’s base”[63].

```
<c level="subseries" base="representations/[repID]/data/[seriesfolder]/[subseriesfolder]">
```

Table 22 - EAD examples of the base attribute

For reasons that have already been explained above, the <dao> element is not mandatory and as such an IP can just be described as a whole - ignoring all hierarchical levels -, for example using the <abstract> tag. The @base is not mandatory either.

The path used in @base is always absolute for reasons of efficiency when matching paths to folders. According to the EAD3 tag library doc, the data type of the base attribute is anyURI. The definition of anyURI is: "A Uniform Resource Identifier. This may be a Uniform Resource Locator (URL) or a Uniform Resource Name (URN). Both relative and absolute URIs are allowed.”[64]

3.1.2.3.4.3 Access restrictions

The <accessrestrict> tag is “An element for information about conditions that affect the availability of the materials being described.”[65]**.** The Access Rights Information that concerns the *end-user* has to be available in EAD - not in PREMIS - and <accessrestrict> is used for this purpose. The reasons being:

[63] EAD3 @base https://www.loc.gov/ead/EAD3taglib/index.html#attr-base

[64] Encoded Archival Description Tag Library, page 6, http://www2.archivists.org/sites/all/files/TagLibrary-VersionEAD3.pdf

[65] EAD3 <accessrestrict> http://www.loc.gov/ead/EAD3taglib/index.html#elem-accessrestrict

- It should be possible to find the Access Rights Information in one place and one place only, namely in the descriptive metadata, which, per default, are the metadata displayed in the Access Software (Finding Aids and different viewers).
- EAD supports the description of potentially very complex hierarchical levels of an IP and can therefore if necessary differentiate access restrictions all the way down to the individual file level.
- Descriptive metadata are very often added upon Ingest and Finding Aids can thus immediately be populated with this kind of information.

The tag in <accessrestrict> is repeatable and can be used in the following way:

```
<accessrestrict>
       <p>Restricted</p>
       <p>75</p>
       <p>...</p>
</accessrestrict>
```

Table 23 - EAD example of <accessrestrict>

If the value of the first is “Restricted” or ”” (empty - which also means that it is restricted) the tool will look for a second which specifies the restriction period. “Unrestricted” means that the IP is immediately accessible. The second can contain any text, for example This IP is available 20 years from November 14 2002 .

Note that the EAD3 schema validates even without the <head> tag inside <accessrestrict>.

For more complex scenarios, it is possible to use <chronlist> as follows:

```
<accessrestrict>
    <chronlist>
        <chronitem>
              <daterange>
                   <fromdate>01.01.2016</fromdate>
                   <todate>01.01.2041</todate>
              </daterange>
              <event>
                  <list>
                            <item>type of the restriction (e.g. personal data)</item>
                            <item>duration of the restriction in years (e.g. 25 years)</item>
                            <item>source of the restriction (e.g. Public access law AvTS §
                            7)</item>
                            <item>additional description of the access restriction (e.g. The
                           content can be made public if personal data is removed from the
                           DIP)</item>
                  </list>
              </event>
         </chronitem>
    </chronlist>
</accessrestrict>
```

Table 24 - EAD example of <chronlist>

3.1.2.3.5 Representations

Whenever new representations are created, a new EAD file needs to be created to include the new file items of the new representation. It replaces the old one for which a backup file with the date suffix is created at

each edit. However, the IP will still only include one valid EAD file (in the 'metadata/descriptive/' folder), which references all the representations of an IP.

3.1.2.4 Representation Information

Representation Information is particularly important to Access since it is the information which is required to understand and render both data and metadata, or, as it is stated in the OAIS, p. 1-13[66]: "Representation Information: The information that maps a Data Object into more meaningful concepts. An example is the ASCII definition that describes how a sequence of bits (i.e., a Data Object) is mapped into a symbol."

Representation Information can be subdivided into three classes:

- **Structural Information**: describes the format and data structure concepts to be applied to the bitstream, which result in more meaningful values like characters or number of pixels.
- **Semantic Information**: this is needed on top of the structural information. If the digital object is interpreted by the structural information as a sequence of text characters, the semantic information should include details of which language is being expressed.
- **Other Representation Information**: includes information about relevant software, hardware and storage media, encryption or compression algorithms, and printed documentation.

The Representation Information can be executable software, if that's helpful. Rather than having further Representation Information which defines what a PDF is, it's more helpful to simply use a PDF viewer, instead[67]. The OAIS needs to track this carefully though, because, one day, PDF viewers may cease to exist or work properly, and the original Representation Information would then need to be updated, together with other preservation strategies like file format migration or emulation, which will bring new requirements to the necessary representation information.

In the E-ARK projekt, we cover representation in different ways. For example by the use of the Documentation folder which is presented as a container where one can place all the information that one has collected during the pre-ingest phase which somehow helps the archivist and the Consumer to better understand the data stored in the IP. This folder fulfils well the requirements of the transfer of archival material from the producer to the archive. One does not want to over complicate the SIP creation step that is already a huge burden for most producers. Having a container that allows producers to include whatever extra documentation or information they have that may help to understand the data is a very welcome feature. Other examples of Representation Information in the E-ARK IP is the use of PREMIS to indicate rendering software of the use of EAD to increase the understandability of the preserved archival material.

On a long term, the E-ARK project proposes the use of relationship model that would enable the creation of links between an AIP and one or more "Representation Information packages". These "Representation Information packages" could very easily follow the same structure of the current E-ARK AIP, meaning that they could be composed of a metadata folder, representation folders, and all the other components that currently make up the E-ARK AIP. This particular type of AIP should be branded differently so it doesn't get confused with the regular AIPs (e.g. via a different type/role on the METS that wraps the entire AIP).

By following this strategy one could build an entire Representation Information network that follows the current specification so we could actually make use of the existing preservation action tools being developed in the project to preserve the RI. Moreover, exchanging RI in this standard way would greatly

[66] OAIS, blue book, 2002 https://siarchives.si.edu/sites/default/files/pdfs/650x0b1.PDF
[67] OAIS, blue book, pp. 4-21/22.

increase collaboration between institutions, enterprises and preservation experts while at the same time reducing the risk of an external registry going out of business.

The current AIP specification already includes a specific relationship between AIPs, namely a succession relationship, where an AIP point to its parent AIP (or AIC). We suggest using the same mechanism to refer to RI AIPs. The implementation would entail adding a new struct-map where the relationships with RI AIPs could be defined, and possibly also other links to external information (e.g. to the PRONOM registry). Also, we suggest adding a new AIP type called "Representation Information" which would clearly identify the AIP as not being part of the content of the repository *per se*, but extra information needed for the preservation of said content.

Since it has not been a part of the E-ARK project to implement such a proposition, we leave it at the conceptual level for the community to pick up whenever it reaches the right maturity level to do so.

3.2 Specification for databases: SIARDDK, SIARD1, and SIARD2

The DIP is created from a representation within an AIP that contains a relational database in the SIARD format.

3.2.1 AIP with more than one representation

In the case where the AIP contains more than one representation only one representation at a time can be chosen as basis for the DIP[68].

A representation may contain a SIARD file along which is a modified[69] version of the database in the form of de-normalization and/or supplemental views in order to make it transparent.

The difference between the representations must be indicated by specific PREMIS metadata.

Different representations must be treated as different (representation) object entities belonging to the same intellectual (object) entity. The relationship between the representations is expressed by the PREMIS semantic unit "relationship". This semantic unit includes "relationshipType" and "relationshipSubType" as semantic components. In regard to the recommended use of PREMIS, both components' values should be taken from a controlled vocabulary.

3.2.2 More than one database in a SIARD archive file

In the unlikely, but possible case, that the SIARD archive file itself contains more than one database, only one database at a time can be used for a DIP.

3.2.3 De-normalization of the transactional RDBMS

De-normalization of the transactional (and because of this) normalized databases may add essential value to the IP by making it more transparent and discovery-ready. The normalized RDB may have a very complex structure where it is difficult for the user to understand the connections between tables, or find references. A de-normalized data structure contains many fewer data tables and therefore it can be understood and used over time much easier than a normalized model. The problem is that a de-normalization process can be very time consuming depending on what information is available for the archivist within or outside the

[68] In the unusual case where the end-user requests access to more than one representation of an AIP it will be delivered as separate DIPs.

[69] The assumption being that a database is always archived in accordance with its original data model.

RDB to be archived. R. G. Healey undertook research[70] on how the de-normalization of a RDB could be done and how the necessary information for de-normalization could be archived.

As Healey pointed out, "The main initial problem is that there is no universally agreed and implemented standard for how information is stored within relational database dictionaries, either in terms of the structure of tables/views that may be employed, or in the naming conventions used."[71] The author proposes to develop a new vendor-independent standard that relevant non-standard data with constraints can be channelled into a common framework. The proposal is that two additional tables should be used in order to document join conditions between tables, where these conditions were part of the original database design prior to archiving[72]. In the case of Oracle databases, and if the constraints are already created in the RDB, the tables can be filled out in an automated way by running SQL scripts. Based on those two new tables automated de-normalization can be undertaken on a live Oracle RDBMS by specific SQL scripts. Both R. G. Healey and the Hungarian Pilot team tested this procedure[73]. Healey in a very simply case, the Hungarian team in a much more complicated case, but neither database contained LOB data. When the RDB contains LOB data or if it is very large, the automated de-normalization may produce very large tables and may become impractical. So the de-normalization in many cases can be made only semi-automated.

Another problem is that there is no uniform method to gather information about applied constraints in RDBs from different vendors. To extend the specification of tables, the necessary fields in them, and other RDB objects which are needed to store information about constraints when looking at all significant RDBMS vendors, is far beyond the scope of E-ARK project.

Both R. G. Healey's research and the Hungarian Pilot show that having a de-normalized version of an originally transactional RDB is very useful. But there may be a great variety in when and how this de-normalized RDB should be created and what should be stored within the AIPs and DIPs. As we now see, in each case the following should be considered by the archivists:

- Is the archived RDB reusable without the stored information on constraints between tables?
- Is it enough only to store the information about constraints, or is it preferable that the de-normalization takes place after submission and the de-normalized RDB should also be stored as a different representation of the original RDB?
- Whether the de-normalization can be done in an automated or semi-automated way. Even if it can be done in an automated way, for many reasons it could be decided to use semi-automated denormalization, for example, because of the possibility of generating superfluous data, or when table(s) contain LOB data as well.
- When it is decided to store only information about constraints, then it must be carefully considered what information and in which form will be stored in the documentation folder. For example, ER diagrams, table definitions, SQL scripts, proposed (see the research by R. G. Healey, cf. Annex A) new tables with information (see the research by R. G. Healey, cf. Annex A), etc.

[70] Healey, R. G. "Proposed E-ARK standard for relational database metadata table structure", cf. Annex B.
[71] Healey, R. G. "Proposed E-ARK standard for relational database metadata table structure", cf. Annex B.
[72] For details see Healey, R. G. "Proposed E-ARK standard for relational database metadata table structure", cf. Annex B.
[73] For details see Healey, R. G. "Proposed E-ARK standard for relational database metadata table structure", cf. Annex B.

3.2.4 SIARD versions and SIARD format structure

The SIARD version used could be SIARD 1.0, SIARDDK or SIARD 2.0, the latter being recommended by E-ARK for use with relational databases.

The SIARD format consists of the SIARD archive file (named [database name].siard)[74] and possibly associated files outside the SIARD archive file (not applicable for SIARD 1.0), representing LOBs from the database. These associated files are referenced from the SIARD table files inside the SIARD archive file, and are listed in the IP's METS file (or several METS files in case of segmented IPs due to size of the LOBs). The SIARD archive file and the associated LOBs files outside the SIARD archive file are referred to as a SIARD database hereafter.

3.2.5 Physical folder structure

A conflict exists between the SIARD format specifications and the E-ARK Common Specification. Requirement 3.1 in the Common Specification states that an IP MUST be built in such a way that its data and metadata can be logically and physically separated from one another. The physical folder structure of the E-ARK Information Packages also enforces this. However, in the case of relational databases that are compliant to the three different SIARD Format specifications, some metadata have to be located within the SIARD file or SIARD folder. This means that metadata of the SIARD file or the SIARD folder will be located in the representation and data folder.

The metadata required by SIARD is mainly a description of the original database management system, and descriptions of the specific tables, columns, relations, views, etc. that the database had. It is necessary that the SIARD file/folder is kept as is in order to comply with the SIARD specifications. For enhanced access opportunities, however, metadata about the database and for example relevant tables could and should be given in the metadata folder. For a suggestive example on how to describe a database and some of its most relevant tables in EAD, see Chapter 3.2.3.3 EAD.

3.2.6 Data

The primary data of the relational database (RDB) to be delivered to the user is stored in a SIARD database. The actual content of the SIARD database (db) within an E-ARK DIP may be

1. all or a subset of the SIARD db from the AIP;
2. all or a subset of the SIARD db from the AIP supplemented with certain views[75] for better understanding and use of the database;
3. a de-normalised version of the SIARD db from the AIP or from a subset of it (i.e. case 1 or 2).

3.2.7 Metadata specifications

3.2.7.1 METS

For a full metadata example on how METS.xml for databases could look like, go to https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SIARD/SIARD2.0

[74] However, SIARDDK is based on a fixed folder structure, where the first folder has to be named AVID.XX.XXXXX.1.

[75] In database theory, a view is the result set of a stored query on the data, which the database users can query just as they would in a persistent database collection object. This pre-established query command is kept in the database dictionary. Unlike ordinary base tables in a relational database, a view does not form part of the physical schema: as a result set, it is a virtual table computed or collated dynamically from data in the database when access to that view is requested. Changes applied to the data in a relevant underlying table are reflected in the data shown in subsequent invocations of the view. In some NoSQL databases, views are the only way to query data. Source Wikipedia SQL https://en.wikipedia.org/wiki/View_(SQL).

download the zipped file called AVID.SA.Northwind.DIP.zip, and locate the METS.xml in the root folder and in representations/AVID.SA.18006_rep0/METS.xml.

3.2.7.2 PREMIS

For a full metadata example on how PREMIS.xml for databases could look like, go to https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SIARD/SIARD2.0 download the zipped file called AVID.SA.Northwind.DIP.zip and locate the PREMIS.xml in metadata/preservation/PREMIS.xml and in representations/AVID.SA.18006_rep0/preservation/PREMIS.xml.

In addition to the standard E-ARK DIP and SIARD metadata inside the SIARD archive file, the following information should be stored using the PREMIS semantic units:

1. Metadata on whether the SIARD db in the DIP is generated from the original database in the AIP or a modified version. If it was generated from a modification, then there should also be metadata about the modification.
    a. If the SIARD package is generated from the original database, then this fact can be indicated by <creatingApplicationExtension> or <formatNote>. In this case this metadata may be optional.
    b. If it is generated from a modified (e.g. de-normalized) version of the original database, then this fact can be indicated by <creatingApplicationExtension> or by <formatNote>. In this case this metadata is obligatory and because this object is a derivation from the original one, the relation between the representations also has to be given as follows:
        - relationshipType: “derivation”
        - relationshipSubType: “de-normalization”

2. Metadata (Representation Information) about recommended rendering tools and available detailed documentation within the IP.
    a. Metadata about recommended E-ARK tools using the PREMIS semantic units of <environmentFunction> and <environmentDesignation> (see section 3.1.2.2.1 Metadata regarding Representations and Access Software above).
    b. For a longer description about recommended rendering tools and scenarios use <environmentDesignationNote> (see section 3.1.2.2.1 Metadata regarding Representations and Access Software above).
    c. Metadata which is detailed documentation about rendering tools, and recommended further preparation(s) of the primary data for more efficient rendering and analysis. This is mainly a textual description of the possible rendering scenarios and the name of the documents stored in the “Documentation” folder, in which further detailed information can also be given. The semantic unit <environmentDesignationNote> is used for this purpose.

3.2.7.3 EAD

In order to describe a database in EAD it is suggested that the value of the attribute level should be set to “otherlevel” and that the value in the following otherlevel attribute should be set to “database”, see the following example where a database description is at the highest hierarchical level:

```
<archdesc level="otherlevel" otherlevel="database" lang="eng">
```

```
 <did>
  <unittitle>Northwind database</unittitle>
  <dao daotype="borndigital" href="/representations/AVID.SA.18006_rep0/data/northwind.siard"></dao>
  <unitdate>2000-2008</unitdate>
  <abstract lang="eng">Northwind Traders Access database is a sample database from Microsoft Office
suite. The Northwind database contains the sales data for a fictitious company called Northwind
Traders, which imports and exports specialty foods from around the world. The database contains 13
tables and also includes pictures of foods and employees. </abstract>
<!-- EAD file continues but further elements left out in this example-->
```

Table 25 - EAD level attributes for databases

If there are relevant tables the value of the attribute level should be set to “otherlevel” and the value in the following otherlevel attribute should be set to “table”:

```
<c level="otherlevel" otherlevel="table">
  <did>
     <unittitle>Orders</unittitle>
     <abstract>Table0. This table contains information about the orders that Northwind Traders
had</abstract>
  </did>
</c>
<c level="otherlevel" otherlevel="table">
  <did>
      <unittitle>Products</unittitle>
      <abstract>Table1. This table contains information about the products that Northwind Traders
had</abstract>
  </did>
</c>
```

Table 26 - EAD level attributes for database tables

For a full metadata example on how an EAD.xml for databases could look like, go to https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SIARD/SIARD2.0 download the zipped file called AVID.SA.Northwind.DIP.zip, and locate the EAD.xml in metadata/descriptive/EAD.xml.

3.2.8 Access scenarios

The first step is the search of the IP(s) which contain the data that meet the end-user’s requirements. One may find one or more relevant AIP(s). Then the following questions have to be taken into consideration:

1. Are there any access restrictions regarding the occurrence of sensitive data, copyright or other legal regulations?
2. Which parts of the IP(s) are needed by the user? The archivist has to make a decision based on the specific request of the user as to whether the whole data content of the IP(s) will be delivered to the user, or if it is sufficient to deliver only a part of the IP(s).
3. How many DIPs have to be created to fulfil the user’s requirements?
4. How will the data content and associated metadata and necessary documentation of the DIP be rendered to the user? An appropriate tool has to be chosen to make the data usable by the user.

The archivist may have considerable flexibility when looking for an appropriate solution for these issues, but the solution may also be highly dependent on the following circumstances:

1. How is the RDB in the AIP(s) archived?

    a. Is it archived as a simple SIARD db generated from the production database with or without documentation?
    b. Or is it archived as a modified version of the original database for better understanding?
2. The software tools and IT infrastructure available to the archivist.
3. The technical skills (or qualifications) of the archivist.
4. The inner regulation or guidelines of the archive regarding how much effort may be spent on the task of generating a DIP.

The access scenarios may appear quite different when taking into account the above mentioned issues.

The simplest scenario is when the whole data content (SIARD db) of an AIP will be delivered to the user[76] and the database can be properly rendered by a user friendly tool. For that purpose the Database Visualization Toolkit (DBVTK)[77] might be sufficient. This scenario may represent the easiest situation for the archive, but not necessarily for the user. The AIP may contain several representations of the db, and the user may need access to more than just one.

In most cases the production of an appropriate DIP for the user is rather complicated.

1. After having found the relevant AIP(s) the archivist has to decide in terms of the relevant access restrictions and user requirements from which part of the AIP(s) the DIP(s) will be generated. This involves:
    a. Selecting the relevant database
    b. Selecting the relevant records of the table(s);
    c. Selecting the relevant columns of the table(s);
    d. Changing/removing sensitive data if needed;
    e. Choosing the appropriate tool to make these modifications. For some of these a tool like the E-ARK DB Viewer would be sufficient but in most cases the SIARD db, stored in the AIP, has to be extracted from the AIP and imported with the E-ARK DBPTK into an RDBMS. Here the modifications have to be made in the GUI or more likely by SQL scripts. After that a new SIARD db may be exported from the RDBMS with the DBPTK.
2. In some cases, or for certain purposes, modifications have to be made on the structure of the data model to facilitate understandability or to allow for sophisticated analysis. These modifications can be done by SQL commands, scripts or by specialised tools after importing the SIARD db into a RDBMS. The modified database may be rendered by:
    a. standard tool like Database Visualization Toolkit (DBVTK) after generating a new SIARD db;
    b. or by a special tool connected directly to the RDBMS (using standard applications like SQL Navigator from Quest Software or specially developed applications like Sofia from the Danish National Archives).
3. In some cases it may need a special newly developed rendering tool (e.g. an Oracle APEX application for particular reporting).

[76] In specific cases (but in practice very often) the DIP may be totally identical to the AIP.
A) Because the AIP, as it is, is completely appropriate to the users' needs. Any modification wouldn't facilitate the user's investigation or analysis. As well, there are no access restrictions on the data;
B) Because the DIP is intended to be imported into another digital archive.
In both cases it has to be decided what should be done officially with the specific metadata regarding whether the IP is S/A/DIP.

[77] Developed within the E-ARK project, Database Visualisation Toolkit https://github.com/keeps/db-visualization-toolkit

4. In more specific cases the RDB may be transformed into a data warehouse (DW), so OLAP cubes can be created based on it, and tools like Oracle Business Intelligence, for example, may provide substantial research potential for analyzing the data of the original RDB.

Specific Access scenarios:

1. **For the user using the Database Visualization Toolkit (DBVTK) without structural modifications**
    - **a.** The archivist takes the SIARD db from the AIP.
    - **b.** The archivist checks the access restrictions of the data stored in the AIP.
    - **c.** If needed, the archivist selects the data meeting the user's requirements and data accessibility rules, and copes with sensitive data. This will be performed using SQL commands after loading the SIARD db into a RDBMS.
    - **d.** The archivist exports the modified content with DBPTK into SIARD 2.0 format and packages it as a DIP using a DIP creator tool.
    - **e.** The archivist delivers the DIP to the end-user.
    - **f.** The end-user may use the IP Viewer to explore the whole DIP.
    - **g.** The end-user extracts the SIARD db from the DIP.
    - **h.** The end-user uses the DBVTK to browse, search and analyse the database.

    Steps f, g, and h can be performed both in the archive institution and in the end-user's own environment.

2. **For the user using the Database Visualization Toolkit (DBVTK) with structural modifications**
    - **a.** The archivist modifies the data model by adding views, and possibly de-normalises the database to make the database transparent to the end-user.
    - **b.** The archivist creates a new SIARD db and DIP using DBPTK. Specific metadata will be added to the new DIP package about the new presentation and modification of the database.
    - **c.** The archivist delivers the DIP to the end-user.
    - **d.** The end-user may use the IP Viewer to explore the whole DIP.
    - **e.** The end-user extracts the SIARD db from the DIP.
    - **f.** The end-user uses the DBVTK to browse, search and analyse the database.

    Steps d, e, and f can be performed both in the archive institution and in the end-user's own environment.

3. **For the experienced user using RDBMS and specific tool(s)**. **Analysing the DIP using RDBMS and specific tool(s)**
    - **a.** The archivist modifies the data model by adding views, possibly de-normalises the database to make the database easier to understand (transparent) to the end-user.
    - **b.** The archivist provides access to the end-user to connect to the RDBMS.
    - **c.** The archivist provides specific tool(s) to search and analyse data. There is large variety in how the archivist may help the user to analyze the requested data. It depends on the potential of the archives what software tools it can afford to provide to the user.

4. **For archivist with IT knowledge when the DIP is intended to be imported into another digital repository**
    - **a.** The archivist creates a DIP from the requested AIP for the user (archivist). In special cases the AIP, as it is, can be the DIP. It will depend on the regulation of the archives.
    - **b.** The archivist (in the host archives) has to be careful about making the necessary changes in the DIP to be able to import it into the host repository.

3.3 Specification for Data warehouses: The OLAP Cube

This section is about Data Warehouses and Online Analytical Processing (OLAP)[78] of data from relational databases. Therefore it will refer to the previous section about relational databases.

In addition to this specification, the E-ARK project has produced an article on a possible approach to vendor-independent archiving of data warehouses[79].

3.3.1 Introduction: Data warehouse

The typical way of creating and using OLAP objects requires a data warehouse, for which there are different kinds of data sources: relational databases (RDB), spreadsheets, statistical file formats (SPSS, SAS, etc.), data in XML formats, etc. All the data from these data sources will be extracted and imported into a staging area and from there the data will be transformed and loaded into a particular kind of third normal form RDB (relational database) and into a "dimensional model", known as a star schema[80]. This is what we call a data warehouse.

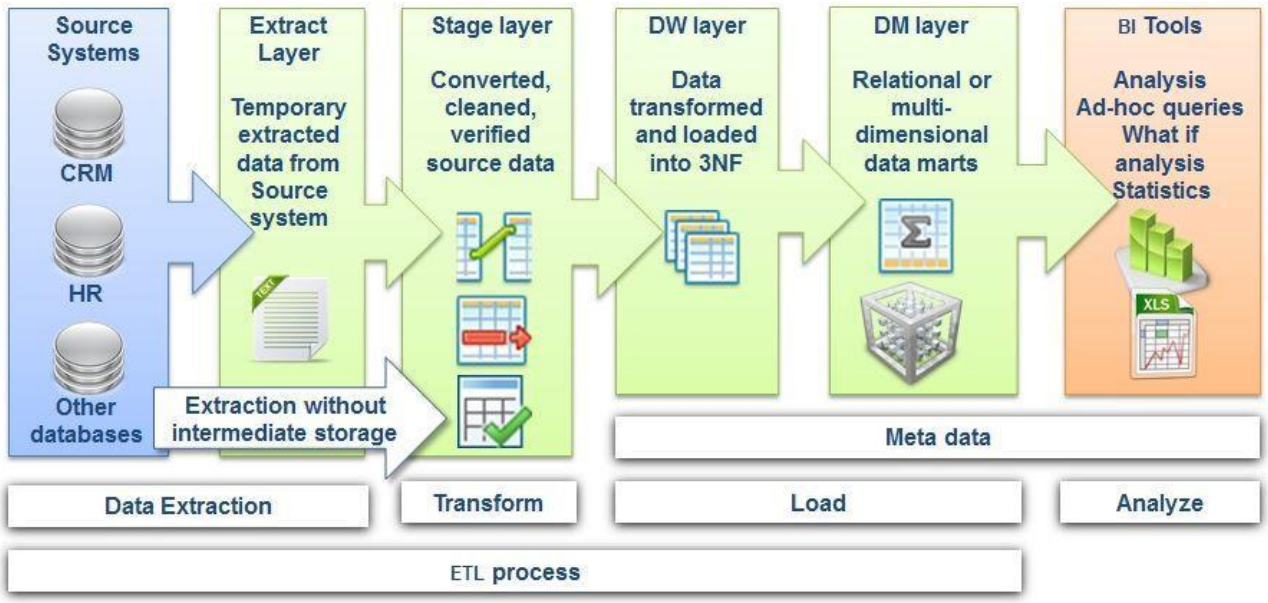

Figure 7 - Typical flow from source to BI

3.3.1.1 Data mart

For particular purposes, such as running specific analytical processes on the data, one or more data marts will be developed. These are usually based on the star schema and have a denormalised data model, but data marts do not necessarily contain all data that are stored in the data warehouse and the specific star schema model is highly dependent on the purpose of the required analysis[81].

---

[78] OLAP, Online analytical processing https://en.wikipedia.org/wiki/Online_analytical_processing provides definitions. See also D4.4 SIP-AIP conversion component (http://www.eark-project.com/resources/project-deliverables/89-d44) that describes OLAP and all its variants.

[79] Healey, R. G. A Proposed E-ARK Standard for Vendor-Independent Archiving of Data Warehouse, cf. Annex A.

[80] See deliverable D4.3 E-ARK AIP Specification available at http://www.eark-project.com/resources/project-deliverables/53-d43earkaipspec-1 for a full description of data mining, data warehousing, data marts, OLAP, star schemas, de-normalisation, MultiDimensional DBMSs (MDDBMSs) etc.

[81] It is possible to create a top-down data warehouse which then populates smaller data marts. This is the DW lifecycle set out by W.H. (Bill) Inmon. Conversely, Ralph Kimball's bottom-up approach advocates creating data marts first, then

3.3.1.2 OLAP Cube

An OLAP cube is a multi-dimensional dataset, typically from a data mart. The definition and creation of an OLAP Cube is extremely vendor specific and there is no unified exchange format for it. Therefore, the vendor specific export formats of OLAP Cube definitions or of OLAP Cubes (i.e. with data) are not interchangeable between the different OLAP applications.

3.3.1.3 Archiving Data Warehouses and reconstitution of such Warehouses for post-archival OLAP processing

There are significant differences in the ways different vendors have implemented their interpretations of data warehousing theory and how they handle the data warehouse specific objects of their systems. For example, Oracle has integrated an SQL level DW processing into its main kernel but it is not integrated into the MS SQL Server where it needs a separate OLAP engine. Another important point is that more and more data warehouse related design and implementation tasks are integrated into high-level end user data processing and analysis tools, like ORACLE ODI, OWB, ODI, or Oracle Visualization.

R. G. Healey also undertook research into how DW and related objects can be archived[82]. Taken into account this research the following possibilities may considered:

- Simply to archive DW structures in vendor-specific export formats, with very limited E-ARK specific metadata, but these formats will be subject to change over time (judging by past ORACLE experience). This will likely sharply reduce the long-term value of the archived data, and its restoration for inter-operability with more recent DW data could then give rise to major technical problems, which could defeat the original archiving purpose[83].
- Develop a low-level DW metadata table structure that could be used for automated generation of SQL level commands to (re-)build data warehouse dimensions, fact tables and the key linkages between them. These tables should also be treated as standard relational tables so standard E-ARK tools can be applied to archive them.
  "...the focus will be on preservation of the logical structure of the data warehouse, rather than any specialised physical storage structures … the DW was originally constructed." So the archiving of DW can bee seen as the capturing of the essential elements of the ROLAP structure."[84].)
- Healey also investigated, in the above mentioned paper, how a Big Data ecosystem could be applied to archiving DW data and related objects like facts and dimensions, and for providing an environment to analyse them. In spite of the many advantages of the Big Data approach, such as its ability to handle very large data sets, tables, rapidly improving tools, handling data in newer formats such as json, "BD approaches cannot replicate the capabilities of a properly configured SQL OLAP system sitting on top of a well-designed DW structure, without extensive additional programming to mimic the functionality that the latter systems already provide 'out-of-the-box'"[85]
- In specific cases the whole DW with its original software environment may also be archived. This can happen when very vendor specific DW objects and analysis tools are used. In this case the emulated environment should be put into a separate representation folder and very detailed

---

carefully joining them to form a DW. See Sansu George http://www.computerweekly.com/tip/Inmon-vs-Kimball-Which-approach-is-suitable-for-your-data-warehouse for a full discussion of such issues.

[82] Healey, R. G. A Proposed E-ARK Standard for Vendor-Independent Archiving of Data Warehouses, cf. Annex A.

[83] Healey, R. G. A Proposed E-ARK Standard for Vendor-Independent Archiving of Data Warehouses, cf. Annex A.

[84] Healey, R. G. A Proposed E-ARK Standard for Vendor-Independent Archiving of Data Warehouses, cf. Annex A.

[85] Healey, R. G. A Proposed E-ARK Standard for Vendor-Independent Archiving of Data Warehouses, cf. Annex A.

documentation about every software components should also be included in the "Documentation" folder of the IP. In certain cases the SIP (delivered to the archive) may have the same content.

3.3.1.4 Data warehouse stored as a RDB in SIARD format in the DIP

E-ARK has decided that everything that will be done with data after the data warehouse **stage** (as a simple RDBMS) will be treated as the rendering of the dataset/database, i.e. the de-normalised data will be exported into the SIARD format, and all specific information about that will be stored in the documentation folder in the DIP.

The following two subcases will be taken in account:

1. Specific data warehouses will be provided as a RDB in SIARD format in the DIPs (to allow for better understanding and easier use in a simple RDBMS)
2. Specific data warehouses will be provided as a RDB in SIARD format in the DIPs with sufficient information to generate and analyse OLAP Cubes.

3.3.2 Physical folder structure

The folder structure of a DIP containing relational databases in SIARD format is fully compliant with the folder structure of E-ARK Information Packages described in section 4.1 of the Introduction to the Common Specification *for Information Packages in* the E-ARK Project.

In the case where the IP contains more than one representation of the database, the difference(s) between the representations must be indicated using PREMIS metadata.

Different representations should be treated as different (representation) object entities belonging to the same intellectual (object) entity. The relationship between the representations can be expressed by the semantic unit (PREMIS) "relationship". This semantic unit includes "relationshipType" and "relationshipSubType" as semantic components. In regard to the recommendation to use PREMIS, both components' values should be taken from a controlled vocabulary.

The documentation folder should include all relevant OLAP specific data which may be:

1. A textual description (including diagrams) of the definition of the specific OLAP objects.
2. Vendor specific exports (files) of the OLAP cube.
3. Vendor specific documentation for particular software tools for creating and analysing OLAP data.
4. Software tools for creating and analysing OLAP data (always together with sufficient documentation).
5. SQL scripts or any other vendor specific scripts needed to create e.g. a star schema, or further OLAP objects.

The primary data of the RDB to be delivered to the user are stored in the form of a SIARD db. The actual content of a SIARD db within an E-ARK DIP will be a subset of the content of the SIARD db (from a certain representation) of the AIP.

3.3.3 Metadata specifications

3.3.3.1 METS

The agreement is that each author of the content information types provides the link to an example xml file.

3.3.3.1.1 METS files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/OLAP

3.3.3.2 PREMIS

In addition to the standard E-ARK DIP and SIARD 2.0 metadata, the following information should be stored:

1. Metadata on whether the actual representation is a SIARD package (with several types of documentation) or it is an emulation of the whole original DW within its original software environment. This information should be given by the PREMIS semantic unit <objectCharacteristics> or <format>.

   In the case of "emulation" all detailed information has to be added, as it is needed to be able to install and use the emulated environment using the semantic units of <creatingApplication>, <environmentDesignation> and its sub-components.

   If there is more than one representation stored within the DIP, the semantic unit <relationship> has to be used to specify the relationship between them.

2. Metadata on whether the actual SIARD db is generated from a modified schema or from the original database schema. If it was generated from a modified one, then there should also be metadata about the modification.
   a. If the SIARD package is generated from the original database, then this fact can be indicated by <creatingApplicationExtension> or by <formatNote>. In this case these metadata may be optional.
   b. If it is generated from a modified (e.g. a transformed version of the original one into another vendor's RDBMS) version of the original database, then this fact can be indicated by <creatingApplicationExtension> or by <formatNote>. In this case this metadata is obligatory and because the object is a derivation from the original one, the relation between the representations also has to be given as follows:

      relationshipType: "derivation"

      relationshipSubType: "other-RDBMS"

3. Metadata about the OLAP objects. Different vendors handle the OLAP specific objects very differently in their own systems in regard to how to create, handle and store them, and what kind of export functionalities they provide for them. At this stage of the research we can only say that all OLAP-specific information should be stored in the "Documentation" folder, but the fact that information is available about OLAP object(s) in the DIP, should be indicated by the PREMIS semantic unit <environmentDesignationNote >. This is also consistent with the point of view that everything that will be done with data after the data warehouse **stage** (as a simple RDBMS) will be treated as the rendering of the dataset/database. However, a user has to have all necessary information to reconstruct the particular OLAP objects based on the archived RDBMS, and this information may consist of documentations, vendor specific definition and export files, etc. Therefore all necessary information about the role of these files should also be given either within the semantic unit <environmentDesignationNote > or within a kind of "starting point document" which will be referred to within the <environmentDesignationNote > but placed in the

"Documentation" folder.

4. Metadata about recommended rendering tools.
    a. Metadata about recommended E-ARK tools using the semantic units of <environmentFunction> and <environmentDesignation> (see section 3.1.2.2.1 Metadata regarding Representations and Access Software above).
    b. A longer metadata description about recommended rendering tools, and scenarios can be put in the <environmentDesignationNote> semantic unit (see section 3.1.2.2.1 Metadata regarding Representations and Access Software above).
    c. Metadata with detailed documentation about rendering tools, and recommended further preparation(s) of the primary data for more efficient rendering and analysis. This is mainly a textual description of the possible rendering scenarios and the name of the documents should be stored in the "Documentation" folder, in which further detailed information can also be given. The semantic unit <environmentDesignationNote> is used for this purpose.

3.3.3.2.1 PREMIS files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/OLAP

3.3.3.3 EAD

3.3.3.3.1 EAD files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/OLAP

3.3.4 Access scenarios

3.3.4.1 Access scenarios with OLAP cubes

A data warehouse (without OLAP objects) can be rendered as a SIARD db by the E-ARK DBVTK. It can also be loaded into an RDBMS using the DBPTK and accessed by a vendor specific application (such as Oracle BI, Oracle BI Data Visualization) or by an in-house developed application.

Accessing and rendering OLAP objects is quite complicated. First, the SIARD db, which contains the data, has to be loaded into a RDBMS. After that the following scenarios may happen:

1. A particular vendor specific OLAP definition file is stored within the DIP (e.g. Oracle dimensional object definitions can be exported/saved either in an XML template or in an EIF format), and the required software infrastructure is also available, then:
    a. The definition file has to be imported into the particular RDBMS. In the case of Oracle this can be done:
        i. by running a PLSQL[86] package, and the definition file (name) will be a parameter of the package,
        ii. or by applying the Oracle Analytic Workspace Manager.
    b. A vendor specific tool has to be applied to perform the possible/required analytical processes on the OLAP cube. In case of Oracle, this specific tool is the Oracle BI.
2. The DIP contains only descriptions and documentation about how to generate (define) the needed OLAP objects (in the Documentation folder). These descriptions can be highly vendor specific, but they can be general, not requiring specific software.

[86] https://en.wikipedia.org/wiki/PL/SQL

a. An appropriate software infrastructure has to be chosen and installed and connected to the RDBMS, in which the OLAP objects can be created and the required analytical processes can be performed. This software infrastructure may differ from the one where the data were originally produced.
b. The OLAP objects have to be created. This step is very vendor specific. All vendors provide their own tool for this task. This step consists of two sub-steps:
    i. Creating the star schema;
    ii. Defining the OLAP cube(s).
c. After the definition of the OLAP objects, a specific tool is needed for the OLAP analysis. The previous two steps determine which analytical tools can connect to the database and interpret the particular OLAP definitions.

It is a different case when the original DW and analysis tools (the whole software infrastructure in emulated form) are part of the AIP. In this case data can delivered to the user in the original environment. In this case the DW can be provided to the user in its original environment.

- The emulated environment has to be restored onto an appropriate hardware and operating system.
- Access restrictions on the data stored in the DW must be checked. If needed, the archivist has to do some manipulation, selection, and deletion on the DW data. Deletion may also proceed when a user doesn't need the whole DW, only a well defined subset of it.
- A user may get access to the restored content and may use the original tools to analyse the DW.

It is an extremely rare and, alongside regular archiving practice, undesirable case, if only an emulated version of the original DW is stored within the AIP. In this case those parts of the DW to be delivered to the user, can be created within the original environment or outside of it. In the latter case the DW data has to be exported from the original DW, and the DW with necessary OLAP objects has to be (re)created within the new environment/system. This method usually needs significant effort.

3.3.4.2 Other access scenarios

1. **For the user using the DBVTK**
   In this case only DW or DM as RDBMS in SIARD (2.0, DK) format will be submitted to the user whether or not the AIP contains OLAP related information.
    a. The archivist takes the SIARD db from the AIP.
    b. The archivist checks the access restrictions of the data stored in the AIP.
    c. If needed, the archivist selects the data meeting the user requirements and data accessibility, and changes sensitive data. This will be performed by SQL commands after loading the SIARD db into a live RDBMS.
    d. The archivist exports the modified content with DBPTK into SIARD2 format and packages it into a DIP.
    e. The archivist delivers the DIP to the end-user.
    f. The end-user extracts the SIARD db from the DIP.
    g. The end-user uses the DBVTK to browse, search, export and analyse the database (data warehouse or data mart).

2. **For the experienced user using a special, vendor specific tool (without OLAP object)**
    a. The end-user or archivist loads the SIARD db into a live RDBMS.

b. The end-user uses the special, vendor specific tool (e.g. Oracle BI) to browse, search and analyse the database (data warehouse). The end-user can use different RDBMS and tools for data analysis than would have been available in the original database.
These steps can be performed both in the archives or in the end-user's own environment/equipment.

3. **For the experienced user using a special, vendor specific tool (with OLAP object)**
    - **a.** The archivist checks the description or definition of the OLAP object(s) stored in the AIP's documentation folder. If needed, the archivist modifies it.
    - **b.** The archivist packages the (modified) definition of the OLAP object(s) stored in the AIP's documentation folder into the DIP.
    - **c.** The archivist delivers the DIP to the end-user.
    - **d.** The end-user or archivist extracts the SIARD db and the documentation from the DIP
    - **e.** The end-user or archivist loads the SIARD db into a live RDBMS with the DPTK.
    - **f.** The end-user or archivist uses a special, vendor specific tool (e.g. Oracle BI, Oracle Analytic Workspace Manager) to create OLAP cubes based on the definition in the documentation folder. In the case of the Oracle RDBMS the definition can be stored as an XML file and the OLAP Cube can be created by importing it into a properly installed ORACLE environment.
    - **g.** The end-user uses the special, vendor specific tool (e.g. Oracle BI, Oracle Analytic Workspace Manager) to browse, search and analyse the database (data mart).
    Steps d-g can be performed both in the archives or in the end-user's own environment/equipment. When using the archive's infrastructure steps f-i may be skipped).

4. **For the experienced user using the original emulated environment for searching and analysing the data (with OLAP object)**
    - **a.** The archivist takes the emulated environment from the appropriate representation folder of the AIP.
    - **b.** The archivist installs the emulated environment onto the archive's infrastructure and selects the data regarding the user requirements and data accessibility, and changes sensitive data. This will be performed within the original environment.
    - **c.** The archivist provides the data to the user.
    - **d.** The archivist can give the user access to the data in the environment prepared for the end-user in the archives.
    - **e.** The archivist can create a DIP from the prepared emulated environment and submits it to the user, who may use it in their own infrastructure.
    - **f.** The archivist may generate a special export from the original environment and submit it to the user in the form of a SIARD package with or without special documentation regarding use.

3.4 Specification for ERMS Based Records: The SMURF ERMS

The SMURF (Semantically Marked Up Records Format) ERMS Content Information Type specification caters for the archiving of records which are either exported from an Electronic Records Management System (ERMS) or any other application or system with records management capabilities (e.g. capabilities of classifying and describing information which is stored in binary formats like .doc or .pdf). Note that the SMURF format in general can be used for archiving "loose" data organised as computer files and/or folders

on a hard drive. For this scenario please consult Chapter 3.5 Specification for Simple File-System Based Records: The SMURF SFSB.

3.4.1 Physical folder structure

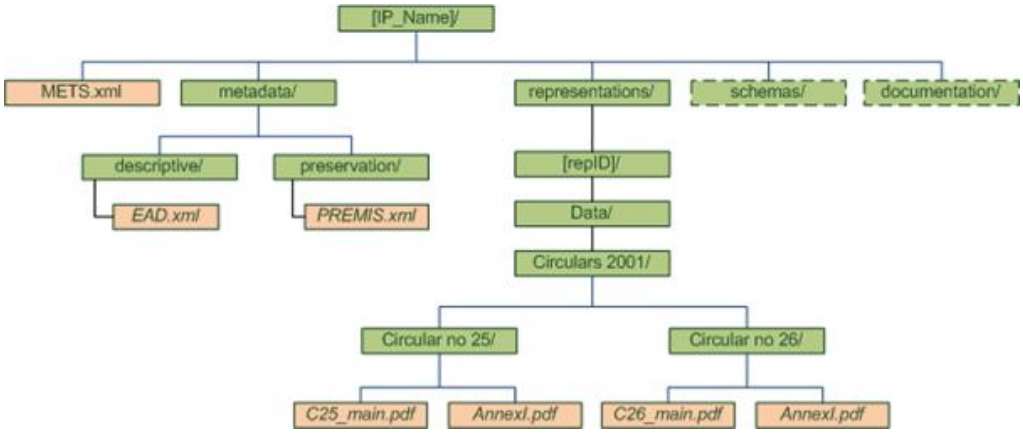

Figure 8 - Folder structure of the SMURF ERMS DIP

The physical folder structure of the SMURF ERMS is depicted in Figure 8 above. The mandatory (must) requirements for the folder structure (when compared to the core DIP specification) are as follows:

- The DIP **must** include only one representation (i.e. one folder with the id or name of the representation, within the "representations" folder);
- The DIP **must** include all metadata in the root "metadata" folder (i.e. no metadata is allowed within the "representations" folder);
- The DIP **must** include one *EAD.xml* file within the "descriptive" subfolder of the "metadata" folder;
- The DIP **must** include preservation metadata in PREMIS format and located within the "preservation" subfolder of the "metadata" folder;
- It **must** be possible to add relevant information about the original ERMS classification scheme, user rights etc. to the "documentation" folder of the DIP (in case the need for such additional information is part of the user's order). However, there are no further requirements on the naming, file format or structure of such documentation.

It is recommended that implementers follow these recommended (**should**) requirements to achieve improved usability and human-readability of the DIP packages:

- The internal structure of the "Data" folder **should** reflect the classification scheme (for example the hierarchy of series and case files delivered within the package);
- The names of these folders should follow the naming of the relevant classification units (for example "Circular no 25", see also Figure 8 above);

Note! The requirements above assume that one SMURF DIP is usually limited to not exceed GB range sizes. In case the user orders larger sets of data at once (i.e. exceeding the "normal" restrictions of DIP delivery media or channels) we recommend splitting the full order into reasonable chunks of classification units (as an example by series) and package these as individual DIPs. Solving this potential size issue with organisational measures rather than with a complex technical algorithm will hopefully keep the complexity and cost of access tools low.

3.4.2 Metadata specifications

The SMURF ERMS sub-specification requires the METS, PREMIS and EAD metadata files to be available. Individual implementers might also choose to add any additional metadata into the package.

For metadata in METS, EAD and PREMIS format the limitations expressed in the following chapters need to be taken into account.

3.4.2.1 METS

As mentioned in the "Physical folder structure" chapter above, the package **must** include exactly one METS.xml file in the root level of the DIP. This means that no further METS files are allowed in the representation folders.

When compared to the E-ARK Common Specification this means that:

- The <fileSec> element of the METS file must list all components of the representation folder (within appropriate <fileGrp> and <file> sub-elements);
- The <structMap> element of the METS file must list all components of the representation folder and not use the <mptr> sub-element;

The METS file must also indicate the use of the SMURF ERMS Content Information Type specification. This is done by setting the value of the <mets CONTENTTYPESPECIFICATION> attribute to "SMURFERMS".

```
<mets xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.loc.gov/METS/"
PROFILE="http://www.eark-project.com/METS/IP.xml" TYPE="ERMS"
CONTENTTYPESPECIFICATION="SMURFERMS" OBJID="5d378f86-28a1-41d8-a2b9-264b10fbd511" LABEL="METS
file describing a DIP with ERMS content." xsi:schemaLocation="http://www.loc.gov/METS/
schemas/IP.xsd http://www.w3.org/1999/xlink schemas/xlink.xsd">
```

Table 27 - METS content type specification attribute - ERMS

Further to the limitations above, implementers need to ensure that the requirements expressed within the E-ARK Common Specification and the core DIP format (see Chapter 3 The DIP) are followed.

3.4.2.1.1 METS files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SMURF/ERMS

3.4.2.2 PREMIS

The SMURF ERMS DIP **must** include a PREMIS metadata file which includes information about preservation events applicable to the records as well as all information pertaining to the rendering environment necessary for end users to view and use the records (i.e. the DIP viewer).

The recommendation (**should**) is to format all PREMIS metadata as a single metadata file which also embeds all technical metadata. Following this recommendation would increase the human-readability of the whole package.

There are no further requirements or restrictions to the use of PREMIS elements and attributes. As such, the requirements highlighted for the core DIP format specification (see Chapter 3 The DIP) **must** be followed.

3.4.2.2.1 PREMIS files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SMURF/ERMS

3.4.2.3 EAD

EAD metadata is the main tool for understanding and using the content of the SMURF ERMS DIP. In general EAD metadata must follow the rules set out in the core DIP specification (see Chapter 3 The DIP) and the

core SMURF specification (D3.4 E-ARK SIP Specification: http://www.eark-project.com/resources/project-deliverables/93-d34-1).

For the development of appropriate access tools (viewers) we expect that the following EAD elements are available and filled with appropriate information:

- The title for each descriptive unit within the package (element *did/unittitle*)
- The creation date of the record or descriptive unit (one of the elements *did/unitdate* or *did/unitdatestructured*)
- Each descriptive unit which has a physical counterpart as a folder or computer file(s) within the package's folder structure must use the element *did/dao* along with the *@href* attribute referring to the relative location of the folder or file. In case the reference is to a record which includes multiple computer files, the use of the element *did/daoset* is required
- Identification of aggregation level(s) and records (using the *@level* attribute on the *archdesc* or *c* element)
- archival history of the record (using the *custodhist* element)
- if applicable, EAD metadata **must** also include information about any access restrictions to the records (using the *accessrestrict* element)
- the original classification schema in full or in part (using the *fileplan* element);
- full ERMS originated metadata about the aggregation level(s) and records (embedded into EAD using the *odd* element at the appropriate c-level);
- relevant metadata about the actors and events as exported from the source ERMS and recorded in the extended EAD (sub-elements of the EAD element *odd* as described in D3.3 SMURF specification, for example information about signatures, opening and closing of the file, etc.)

Full example:

```
<?xml version="1.0" encoding="utf-8"?>
<ead xmlns="http://ead3.archivists.org/schema/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://ead3.archivists.org/schema/  ../../schemas/ead3.xsd">
<control>
    <recordid>f323992d-bfa0-4586-bc1b-090d06ec6a72</recordid>
       <filedesc>
           <titlestmt>
              <titleproper>University of Utopia</titleproper>
           </titlestmt>
       </filedesc>
       <maintenanceagency>
              <agencyname>University of Utopia</agencyname>
       </maintenanceagency>
       <maintenancehistory>
            <maintenanceevent>
                <eventtype value="created"></eventtype>
                <eventdatetime>2016-07-08T08:50:52.1200000</eventdatetime>
                <agenttype value="human">text</agenttype>
                <agent>Utopian Recordsmanager</agent>
            </maintenanceevent>
       </maintenancehistory>
</control>
  <archdesc level="fonds">
       <did>
```

```
            <unitid label="current">EAA.123-2</unitid>
            <abstract>Fonds of the University of Utopia</abstract>
            <origination label="Creator">
               <name>
                      <part>University of Utopia</part>
               </name>
            </origination>
         </did>
         <dsc>
            <c level="series">
               <did>
                  <unitid label="current">EAA.123-2-1</unitid>
                  <unitid label="original">1</unitid>
                  <unittitle>Statistical reports</unittitle>
                     <dao id="344ec022-eff4-872e-8771-72253ef5499ca" daotype="borndigital"
linktitle="Statistical reports" href="file:../../representations/rep1/data/Statistical reports" />
               </did>
                      <fileplan id="b6f03af0-c9f8-46f1-80bf-037f8d377ebf" localtype="original">
                             <head>Archival classification schema</head>
                             <fileplan>
                                    <head>Research</head>
                             </fileplan>
                      </fileplan>
            <c level="file">
               <did>
                      <unittitle>Statistical reports 2008</unittitle>
                      <unitid label="current">EAA.123-2-1-2</unitid>
                      <unitid label="original">1-2</unitid>
                      <dao id="e566c065-1e44-3444-3fea-55d6a736253a" daotype="borndigital"
linktitle="Statistical reports 2008" href="file:../../representations/rep1/data/Statistical
reports/Statisical reports 2008" />
               </did>
            <c level="item">
               <did>
                      <unitid localtype="current">EAA.123-2-1-2-1</unitid>
                      <unittitle>Report 01</unittitle>
                      <unitdate datechar="created">20.01.2008</unitdate>
                      <abstract>Report No. 3</abstract>
              <physdescstructured coverage="whole" physdescstructuredtype="spaceoccupied">
                      <quantity>0.0138</quantity>
                      <unittype>MB</unittype>
              </physdescstructured>
              <dao id="c90fc089-1986-4473-8a9f-55d6a73ee9dc" daotype="borndigital"
linktitle="repA136.doc" href="file:../../representations/rep1/data/Statistical reports/Statisical
reports 2008/repA136.doc" />
               </did>
               <scopecontent>
               <p>Report on grant execution in 2006 - 2007</p>
               </scopecontent>
            <accessrestrict>
                <chronlist>
                    <chronitem>
                        <daterange>
                                    <fromdate>2016</fromdate>
                                    <todate>2036</todate>
                          </daterange>
                          <event>
```

```
                                <list>
                                    <item>DEI</item>
                                    <item>20 years</item>
                                    <item>Source of the restriction.</item>
                                    <item>The report includes sensitive personal data</item>
                                </list>
                            </event>
                        </chronitem>
                    </chronlist>
                </accessrestrict>
            </c>
        </c>
       </c>
      </dsc>
     </archdesc>
</ead>
```

Table 28 - EAD example ERMS

3.4.2.3.1 EAD files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SMURF/ERMS

3.4.3 Access scenarios

The SMURF ERMS profile caters for three individual access scenarios.

1. **Access to single records.** In this scenario the records in question originate from an ERMS. The assumption for this scenario is that during ingest the content of the ERMS has been mapped to the hierarchical aggregations of the archival catalogue[87] and therefore the user is able to search for single records within the catalogue. As well, in addition to simple EAD metadata the record is assumed to include additional descriptive metadata originating from the ERMS. In most cases the record would include only a few computer files though in exceptional cases some records might even include tens of different computer files.
   In this scenario the user is again able to carry out either a full text search, a catalogue search or browse the archival hierarchy. If the user finds a record of interest through any of these means[88] he can order it as a DIP. The DIP includes the relevant EAD metadata, original records management metadata and the computer files. The user shall be able to view the DIP with an E-ARK viewer.

A potential tool developed to execute this scenario could implement the following requirements[89]:

- Ability to view the overall package metadata in a user friendly rendering
- Default presentation of the archival hierarchy as a navigable tree
- Alternate presentation of the DIP folder structure as a navigable tree
- Ability to view the metadata for each aggregation level in a user friendly way
- Clear highlighting of access and re-use restrictions (if relevant)
- Ability to view the names of computer files within the appropriate aggregation level

[87] Cf. Finding Aid in the Glossary.

[88] As an example, the user might find a computer file using full-text search. As the computer file is part of a record the user has the possibility of ordering the full record instead of accessing only the single file (which would lead back to scenario a).

[89] The IP Viewer does implement some of these requirements, cf. D5.4 Search, Access and Display Interfaces (http://www.eark-project.com/resources/project-deliverables/92-d54)

- Ability to view the technical metadata of the computer files
- Ability to extract the computer files for opening in external applications (for example Adobe Acrobat)
- Clear highlighting of relevant rendering information if the file in question is not expected to be in a "typical format" and the user is not expected to have relevant software available

2. **Access to a case file.** In this scenario the case files originate from an ERMS. The assumption for this scenario is that during ingest the content of the ERMS has been organised according to a case logic and the different case files have been mapped to the hierarchical aggregations of the archival catalogue. As well, both the records in the case file and the case file itself have specific metadata originating from the source ERMS.
   In this scenario the user is able to carry out either a full text search, a catalogue search or browse the archival hierarchy. If the user finds a case file of interest through any of these means he can order it as a DIP. The DIP includes EAD metadata, original records management metadata and the computer files. The user will be able to view the full DIP with an E-ARK viewer.

A potential tool developed to execute this scenario could implement the following requirements (in addition to the ones listed for the previous scenario)[90]:

- Ability to view the case file in its original context (ability to browse the original classification)
- Ability to view/browse the internal structure of the case file
- Ability to view the detailed case file metadata

3. **Access to an ERMS.** In this scenario the content originates from an ERMS. The assumption is that the ingest process included ERMS data (e.g. classification scheme, records, metadata) as a whole and the integrity of the whole transfer is also maintained within the archival data management and preservation layers. It is also worth noting that it is possible to treat subsequent ingests from the same ERMS either as additions to the same AIP or as a new AIP – the E-ARK Common Specification for Information Packages, the reference DIP specification and the current DIP representation format specification do not pose any restrictions as such, and institutions are able to follow local archival policies.
   In this scenario the user is able to carry out either a full text search, a catalogue search or browse the archival hierarchy. As well, there might be a dedicated search or browsing capability for "transfers". If the user finds a computer file, record or any aggregation which he wants to access he has the possibility to "order the whole transfer" and not only view the aggregation unit itself (which would fall under scenarios a) and b)). In response the archive prepares a large DIP from one or several AIPs which include the original classification scheme, records, metadata and additional elements defined in the SMURF profile. The user can access the DIP with a dedicated viewer which allows the user to browse and search using the original classification and view the records and computer files in a "close to original" environment. Ideally the full-ERMS DIP could be also exported and accessed in an end-user's own ERMS platform.

A potential tool developed to execute this scenario could implement the following requirements (in addition to the ones listed for the previous scenario)[91]:

[90] The IP Viewer does implement some of these requirements, cf. D5.4 Search, Access and Display Interfaces (ibid)

- Ability to view/browse the classification schema, preferably presented in hierarchal structure
- Ability to view the metadata of the whole classification schema
- Ability to view/browse the classes and their metadata
- Search over the full classification schema with the use of class metadata as filters
- Ability to view/browse the aggregations (e.g. case file) in original context (possibility to browse the original classification)
- Ability to view detailed aggregations metadata
- Search over classes with the use of aggregation metadata as filters
- Ability to view/browse the internal structure of the archived aggregation (e.g. case file) with the possibility to browse contained records
- Ability to view the detailed record metadata
- Search over aggregations with the use of records metadata as filters
- Ability to view/browse the archived record with the possibility to browse contained attachments (imbedded/related computer files)
- Ability to access content of contained attachments (embedded/related computer files)

3.5 Specification for Simple File-System Based Records: The SMURF SFSB

The SMURF SFSB profile caters for the archiving of "loose" computer files as available on hard drives, cloud storage providers etc. As such this profile is the most generic one out of all the DIP profiles and can be used to deliver virtually any content and/or metadata to the user.

Along with this, it is worth mentioning that the SMURF SFSB profile does also provide only very minor additions to the core DIP specification.

3.5.1 Physical folder structure

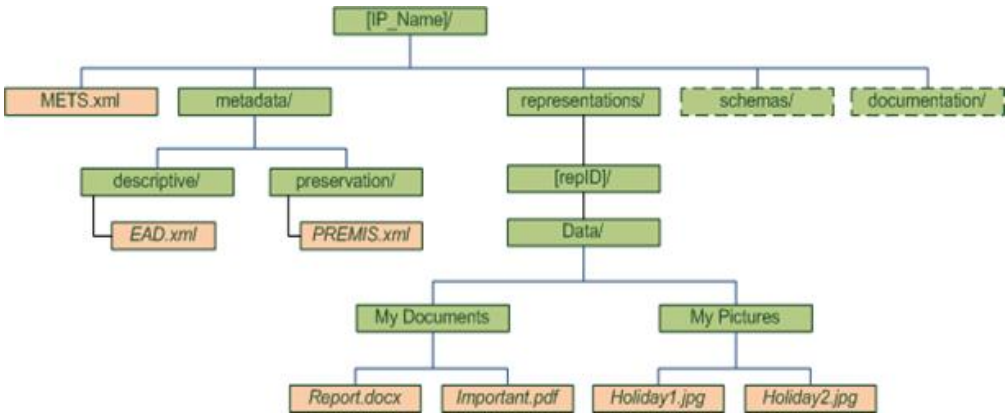

Figure 9 - Folder structure of the SMURF SFSB DIP

The physical folder structure of the SMURF SFSB is depicted on Figure 9 above. The mandatory (must) requirements to the folder structure are as follows:

- The DIP **must** include only one representation (i.e. one folder with the id or name of the representation, within the "representations" folder);
- The DIP **must** include all metadata in the root "metadata" folder (i.e. no metadata is allowed within the "representations" folder);

[91] The IP Viewer does implement some of these requirements, cf. D5.4 Search, Access and Display Interfaces (http://www.eark-project.com/resources/project-deliverables/92-d54)

- The DIP **must** include one *EAD.xml* file in the "descriptive" subfolder of the "metadata" folder;
- The DIP **must** include preservation metadata in PREMIS format and located within the "preservation" subfolder of the "metadata" folder;

It is recommended that implementers follow these recommended (**should**) requirements to achieve improved usability and human-readability of the DIP packages:

- The internal structure of the "Data" folder **should** replicate the initial ingest (SIP) or preservation (AIP) folder structure of the data;

3.5.2 Metadata specifications

The SMURF ERMS sub-specification requires the METS, PREMIS and EAD metadata files to be available. Individual implementers might also choose to add any additional metadata into the package.

For metadata in METS, EAD and PREMIS format the limitations expressed in the following chapters need to be taken into account.

3.5.2.1 METS

A SMURF SFSB package **must** include exactly one METS.xml file in the root level of the DIP. This means that it no further METS files are allowed within the representation folders.

When compared to the E-ARK Common Specification this means that:

- The <fileSec> element of the METS file must list all components of the representation folder (within appropriate <fileGrp> and <file> sub-elements);
- The <structMap> element of the METS file must list all components of the representation folder and not use the <mptr> sub-element;

The METS file must also indicate the use of the SMURF SFSB Content Information Type specification. This is done by setting the value of the <mets CONTENTTYPESPECIFICATION> attribute to "SMURFSFSB".

```
<mets xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.loc.gov/METS/"
PROFILE="http://www.eark-project.com/METS/IP.xml" TYPE="ERMS"
CONTENTTYPESPECIFICATION="SMURFSFSB" OBJID="5d378f86-28a1-41d8-a2b9-264b10fbd511"
LABEL="METS file describing a DIP with simple computer file content."
xsi:schemaLocation="http://www.loc.gov/METS/ schemas/IP.xsd http://www.w3.org/1999/xlink
schemas/xlink.xsd">
```

Table 29 - METS Content type specification attribute - SFSB

Further to the limitations above implementers need to ensure that the requirements expressed within the E-ARK Common Specification and the core DIP format (see Chapter 3 The DIP above) are followed.

3.5.2.1.1 METS files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SMURF/SFSB

3.5.2.2 PREMIS

The SMURF SFSB DIP **must** include a PREMIS metadata file which includes information about preservation events applicable to the records as well as all information pertaining to the rendering environment necessary for end users to view and use the records (i.e. the DIP viewer).

The recommendation (**should**) is to format all PREMIS metadata as a single metadata file which also embeds all technical metadata. Following this recommendation would increase the human-readability of the whole package.

There are no further requirements or restrictions to the use of PREMIS elements and attributes. As such, the requirements highlighted for the overall DIP format specification (**see above**) **must** be followed.

3.5.2.2.1 PREMIS files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SMURF/SFSB

3.5.2.3 EAD

The necessary prerequisite for the application of the SMURF SFSB profile is that a simple archival description has been created using EAD. In the most common scenario this EAD description would include a few aggregations (possibly following the original folder structure) and simple descriptions of the content itself. There are a few EAD elements which are deemed mandatory in order to allow for the development of common access tools:

- The title for each aggregation (element *did/unittitle*)
- The creation date of the record or aggregation (one of the elements *did/unitdate* or *did/unitdatestructured*)
- Each descriptive unit which has a physical counterpart as a folder or computer file(s) within the package's folder structure must use the element *did/dao* along with the *@href* attribute referring to the relative location of the folder or file. In case the reference is to a record which includes multiple computer files, the use of the element *did/daoset* is required

Besides these elements the rules for EAD as highlighted above for the core DIP and within the core SMURF SFSB specification need to be followed.

Full example:

```
<?xml version="1.0" encoding="utf-8"?>
<ead xmlns="http://ead3.archivists.org/schema/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://ead3.archivists.org/schema/  ../../schemas/ead3.xsd">
  <control>
       <recordid>f323992d-bfa0-4586-bc1b-090d06ec6a72</recordid>
       <filedesc>
              <titlestmt>
                     <titleproper>Personal letters of Mr Private Person</titleproper>
              </titlestmt>
       </filedesc>
       <maintenanceagency>
              <agencyname>National Archives of Utopia</agencyname>
       </maintenanceagency>
       <maintenancehistory>
              <maintenanceevent>
                     <eventtype value="created"></eventtype>
                     <eventdatetime>2016-07-08T08:50:52.1200000</eventdatetime>
                     <agenttype value="human">text</agenttype>
                     <agent>Utopian Recordsmanager</agent>
              </maintenanceevent>
       </maintenancehistory>
    </control>
```

```
    <archdesc level="series">
       <did>
              <unitid label="current">FNS.11-2</unitid>
              <abstract>Personal letters of Mr Private Person</abstract>
              <origination label="Creator">
                     <name>
                            <part>Mr Private Person</part>
                     </name>
              </origination>
       </did>
       <dsc>
              <c level="item">
                     <did>
                            <unitid localtype="current">FNS.112-2-1</unitid>
                            <unittitle>Report 01</unittitle>
                            <unitdate datechar="created">20.01.2008</unitdate>
                            <abstract>Report No. 3</abstract>
                            <physdescstructured coverage="whole"
                            physdescstructuredtype="spaceoccupied">
                                          <quantity>0.0138</quantity>
                                          <unittype>MB</unittype>
                            </physdescstructured>
                            <dao id="c90fc089-1986-4473-8a9f-55d6a73ee9dc"
daotype="borndigital"
                            linktitle="repA136.doc"
href="file:../../representations/rep1/data/My
                            Documents/important issue.doc" />
       </did>
       <scopecontent>
              <p>A document on an extremely important issue</p>
       </scopecontent>
         </c>
     </dsc>
  </archdesc>
</ead>
```

Table 30 - Full SMURF SFSB EAD example

3.5.2.3.1 EAD files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/SMURF/SFSB

3.5.3 Access scenarios

Access to folder structures

In this scenario the data are managed within an archive as a set of computer files and/or folder structures, for example as archived from a hard drive. In this scenario the data have very little descriptive metadata attached to them. However, we assume that there is a basic archival hierarchy available which in turn is described using the core elements of EAD.

In regard to the access use case, the user is able to carry out either a full text search or browse the basic hierarchy. Once the user finds the suitable computer file (by full text search) or aggregation (by hierarchical browsing or search within the catalogue) the user can order the DIP. The DIP includes the computer file(s) as well as the basic metadata which the user shall be able to view with an E-ARK viewer.

3.6 Specification for Geodata

This section is about Geospatial data (in short: geodata) and how they are represented in DIP format. Geodata is a combination of the graphical representation of objects in space and their descriptions or attributes. Additionally, geospatial formats include geospatially focused datasets or databases that contain primary information about a geographic location. Furthermore, ancillary and supplementary data – that can be either included or derived using spatial analysis – are considered necessary for rendering, interpretation and re-use of the data.

The basis of geodata is generally either vector or raster graphics data which can be stored as a set of files or as a database (if the database supports spatial data types). Vector data represents spatial objects as points, lines or polygons, or a complex combination of them. Descriptive or derived attributes can be stored along with the spatial data in tables (one row per spatial object) or as a separate table. Raster data represents spatial objects as cells in a grid. Those cells can contain different types of values from binary to decimal numbers. Raster data is often split into a number of images covering a larger area.

Some geodata is structured in a form that it is self-explanatory. In other cases, especially if geodata was a part of a greater system, we need additional information to properly render it in the way it was used in the original system.

For proper interpretation of geodata we also need some other elements that put the coordinates and attribute tables into context. The main additional elements are: Spatial referencing information (Coordinate system); geoprocessing information; and visualization.

3.6.1 Physical folder structure

The general folder structure of a DIP containing geodata is fully compliant with the folder structure of the E-ARK Information Packages described in section 4.1 of the Introduction to the Common Specification for Information Packages in the E-ARK Project.

Within the IP, geodata can be stored within one or more representations. The following schemas show some possible structures of packages, depending on the type of geodata (raster/vector) and different possible locations of the documentation.

3.6.1.1 DIP containing a representation with geodata and documentation

In this case, geodata is contained in one representation in a long-term preservation format for geodata – GML. Additional documentation for geodata is contained within the representations folder. This would be a common structure for the DIPu, which doesn't differ from the AIP.

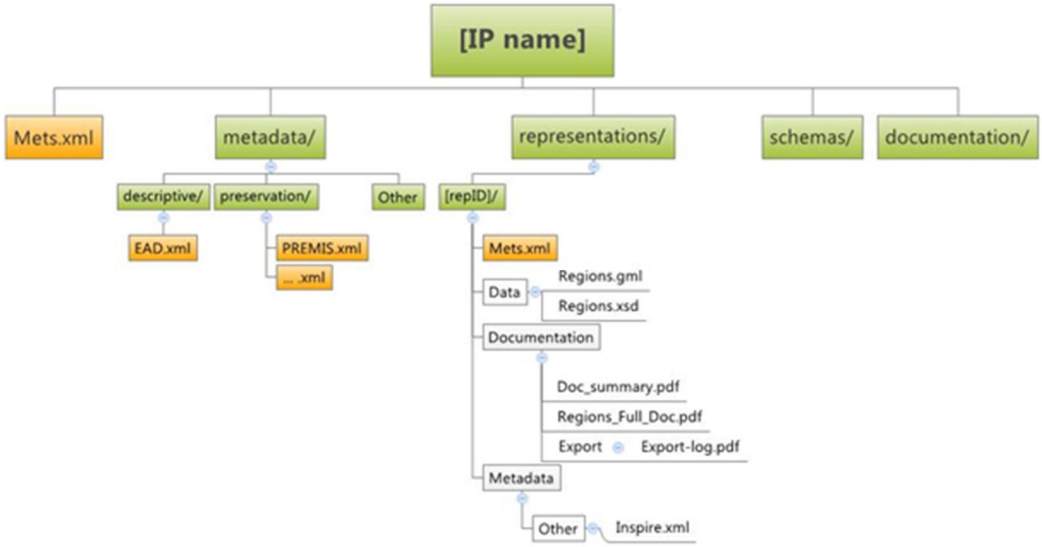

Figure 10 - Folder structure of the geodata DIP containing one vector representation and documentation within representation folder

3.6.1.2 DIP containing multiple vector representations and documentation on the top

In this case, the DIP package contains one representation in GML format and an original representation of the same data in ESRI Shapefile format. The representation with the GML file also contains the documentation specific to that representation – such as an export log from the shapefile to GML. All other documentation, required to properly interpret both representations, is put in the top-level documentation folder. This would be a common structure for the DIP which is created by combining 3 logically connected AIPs into one DIP.

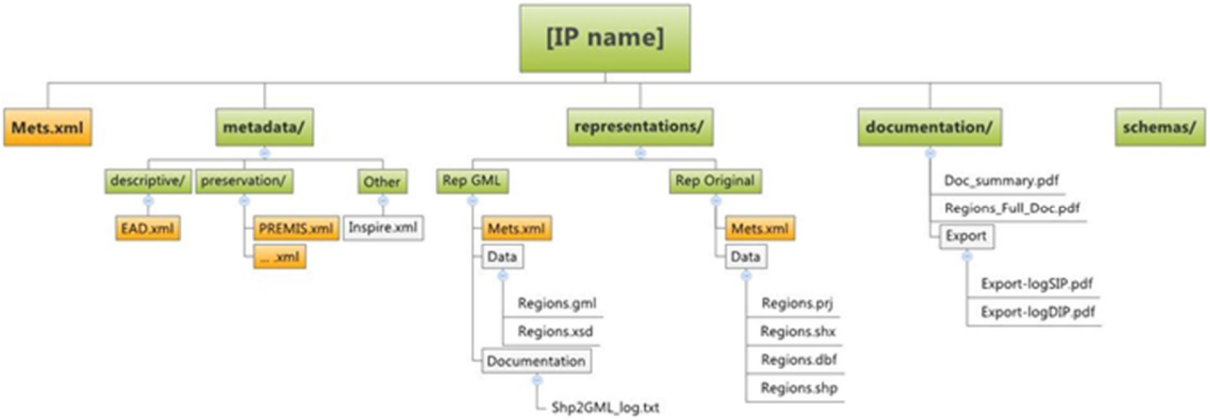

Figure 11 - Folder structure of the geodata DIP containing multiple vector representations and documentation in the top level

3.6.1.3 DIP containing one representation of multiple rasters and documentation on the top

In this case, the DIP package contains one representation consisting of multiple raster images covering an area with an accompanying vector file containing the positions of the raster images. Documentation for the raster datasets is located in the top-level Documentation folder. This is an example where a subset of a large set of geodata rasters is combined into one DIP along with the documentation from a separate AIP.

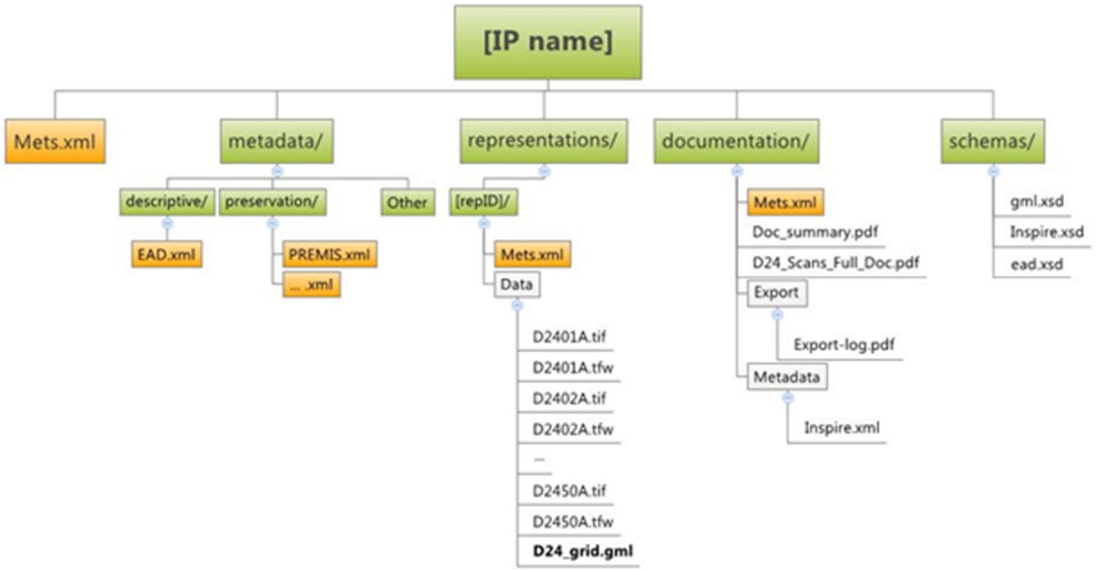

Figure 12 - Folder structure of the geodata DIP containing one representation of multiple rasters and separate documentation

3.6.1.4 Representations directory – the data folder

The representations directory contains at least one representation of geodata in a long term preservation format (GML for vector and GeoTIFF for Raster) and all additional information that is needed to properly render the information as described in the SIP SMURF[92] format for SFSB.

When creating a DIP containing extremely large datasets (more than 1 GB), we can experience difficulties in the validation process of such an XML file or with file manipulation. If the dataset was already split during the ingest (SIP) or data maintenance (AIP) process, we might need to combine it into one package, or use an appropriate strategy to present and deliver the data to the user. We can use the reverse of the process that was used during the Ingest process (should be described in PREMIS or in the Export section of the documentation). The same is required in the reverse process – when combining the data, the process needs to be described in the Export section of documentation and/or in PREMIS.

3.6.1.5 Documentation for geodata

Since geodata is rarely in a form that is self-explanatory enough to be used by itself, we need additional information in order to enable the user to properly understand, interpret and use geodata. Therefore a checklist of mandatory and optional elements of additional documentation is used in the ingest process. The required documentation for geodata can already exist within the AIP when extracted (Figure 10), or it can be stored within other AIPs connected with the geodata AIP on a logical level. In the first case, no extra steps are needed. However, in the second case, the archivist might need to combine the accompanying documentation with the geodata itself in order to produce the final DIP (Figure 11).

3.6.1.5.1 Documentation elements in a DIP

When combining a DIP from multiple AIPs, the following elements of documentation need to be taken into consideration:

Information	Description	Cardinality

[92] D3.4 E-ARK SIP Specification (http://www.eark-project.com/resources/project-deliverables/93-d34-1)

Documentation checklist	The documentation checklist is a document that needs to be present in the root of the documentation folder and which provides an index of locations of the required and optional information within the documentation folder. For instance, if we need to provide the *Feature catalogue, Logical Structure* and *Attribute* definitions, they can all exist within one document, or they can be in separate files. The documentation checklist serves as an index on geodata. If the geodata was manipulated in a DIP, the archivist should go through this checklist and see if any elements need to be updated.	1..1
Export Section	If archived geodata was not stored in the archival format, any transformation into another format needs to be documented. This part of the documentation should contain detailed information about the export and transformation process in any part of the archival package lifecycle. So, if during the DIP creation process geodata was transformed, the process needs to be described in order to ensure provenance and authenticity.	0..n
Feature Catalogue	The feature catalogue represents a logical structure of attributes. It provides a better understanding of the meaning, use and structure of the spatial data and provides a unified classification of spatial data in feature types (classes). Feature types are distinguished by their attributes (properties), by importance and by the relations between them. If during DIP creation some tables are merged or some attributes are simplified, the additional changes need to be described.	0..n
Visualization	Data visualization provides an illustration and representation of spatial data. The catalogue of cartographic symbols is a collection of agreed cartographic symbols, which are used during the process of visualizing spatial data sets to display objects in space. Cartographic symbols are shown in the legend, which explains their meaning.	0..n
Logical structure of layers	The logical structure of layers shows the organization of the data layers at the level of logical tables contained in the database or in a connection of unstructured objects and their attributes, organized in a GIS application.	0..n
Attribute definition	Contains descriptions of the attributes for each data layer and the code values for each attribute (if not described in the **feature catalogue**).	1..n
Table relations	In the case of a complex system of tables, the documentation directory should contain diagrams of the relationships between tables in a database or within a GIS project, in order to enable the reconstruction of queries and provide greater understanding of the usage of tables.	0..n
Metadata	Besides the standard archival metadata like METS, PREMIS and EAD, geodata is often accompanied by geodata specific metadata. We expect this data to be in a standardized structure based on standards such as EN ISO 19115, or the EC INSPIRE directive. This data could in the future be accessed by the GIS tools available at the time. Therefore, we expect an Inspire.xml file or a number of them named after their geodata counterparts. If metadata is available in any other self-explainable form – like a text file or a pdf document, it should be referenced under this section.	0..n

Common queries	A list and a description of the most common queries provide additional information about how the data set was used in the production environment from the end user's point of view. The main goal is to enable the re-creation of the original functionality of tools, which enables users to get information in the form of common queries and common reports, as they were present in the original production environment.	1..n
Other contextual documentation	This chapter combines all the documents (links) to the relevant documentation describing the lineage and provenance of the spatial data set. The list of documentation includes: user manuals, related practices in EU and worldwide, methodological rules, scientific articles, publications, etc.	0..n

Table 31 - Documentation elements in a geo DIP

The folder »Documentation« stores information which will not be already included in the »Metadata« or »Data« folders. It will help archivists and end-users to understand the data-set in a wider social context and will provide a better understanding of the meaning, use and structure of the spatial data. The goal is to provide the end-user with all the necessary information for a proper understanding and interpretation of the geodata data set and to ensure provenance and authenticity.

3.6.2 Metadata specifications

Since geodata IP's are in general a subtype of a SMURF SFSB, all the requirements for PREMIS metadata are in general equal to the specifications mentioned in the chapter 3.5.2.

3.6.2.1 3.6.2.1 METS

An IP package containing geodata **must** include exactly one METS.xml file in the root level of the DIP. This means that it no further METS files are allowed within the representation folders.

When compared to the E-ARK Common Specification this means that:

- The <fileSec> element of the METS file must list all components of the representation folder (within appropriate <fileGrp> and <file> sub-elements);
- The <structMap> element of the METS file must list all components of the representation folder and not use the <mptr> sub-element;

The METS file must also indicate the use of the GEODATA Content Information Type specification. This is done by setting the value of the <mets CONTENTTYPESPECIFICATION> attribute to "GEODATA".

```
<mets xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.loc.gov/METS/"
PROFILE="http://www.eark-project.com/METS/IP.xml" TYPE="ERMS"
CONTENTTYPESPECIFICATION="GEODATA" OBJID="5d378f86-28a1-41d8-a2b9-264b10fbd511" LABEL="METS
file describing a DIP with simple computer file content."
xsi:schemaLocation="http://www.loc.gov/METS/ schemas/IP.xsd http://www.w3.org/1999/xlink
schemas/xlink.xsd">
```

Table 32 - METS Content type specification attribute - GEODATA

Always when the content of a geodata IP is changed severely, as mentioned in the access scenarios – and result in geodata being transformed, clipped, generalised or recomputed, an external report **must** be written and referenced in the Administrative Metadata Section <amdSec>.

- <techMD> element is used for technical description about the creation, format and new use characteristics,

- <digiprovMD> element is used to describe digital provenance information, such as master/derivative file relationships, migration/transformation information, and other administrative decisions and actions affecting the geodata object.

3.6.2.1.1 METS files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/GEODATA

3.6.2.2 PREMIS

Since geodata IP's are in general a subtype of a SMURF SFSB, all the requirements for PREMIS metadata are in general equal to the specifications mentioned in the chapter 3.5.2.2.

Furthermore a geodata DIP **must** include a PREMIS metadata file which includes information about preservation events applicable to the records as well as all information pertaining to the rendering environment necessary for end users to view and use the geodata records (i.e. IP Viewer, QGIS, GeoServer).

The recommendation is to format all PREMIS metadata as a single metadata file which also embeds all technical metadata. Following this recommendation would increase the human-readability of the whole package.

According to different access scenarios, geodata is sometimes altered (transformed, clipped, generalised, computed…) in order to be make it usable for the user. Since this changes change the actual content of the IP, they are described in METS (see chapter 3.6.2.1).

3.6.2.2.1 PREMIS files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/GEODATA

3.6.2.3 EAD

Since geodata IP's are in general a subtype of a SMURF SFSB, all the requirements for EAD metadata are equal to the specifications mentioned in the chapter 3.5.2.3.

Always when the content of a geodata IP is changed in a way that affects positional accuracy (i.e. generalisation or transformation) the EAD metadata description must reflect the new accuracy values. The element "Lineage" must be updated and the element "Spatial resolution" must be updated to the new value.

In the case below an administrative unit's spatial dataset was generalised. The accuracy changes from 1:2880 to 1:50000.

```
<odd>
  <head>Lineage</head>
    <p>This dataset was created by manually digitizing the original cadastral maps in various
          scales (1:1000, 1:1440 and 1:2880). The whole dataset was created by merging
individual
       units, that were digitised by different authors, since different municipalities had
       different service providers. The base datasets were created in the period from 1984 to
       1994 and were subsequently added to create the final dataset. Boundaries were
       generalised for a faster display.</p>
</odd>
```

```
<odd>
 <head>SpatialResolution</head>
   <p>50000</p>
</odd>
```

Table 33 - EAD example of changing content of geodata IP (in this case elements Lineage and Spatial resolution)

3.6.2.3.1 EAD files

https://github.com/DLMArchivalStandardsBoard/E-ARK-DIP/tree/master/examples/GEODATA

3.6.3 Access scenarios

3.6.3.1 Geodata specific search

End-users can browse and search for geodata within archival Finding Aids. Beside standard E-ARK search tools (full-text search, catalogue search) geo-search is identified as the only geodata specific search available. Geo-search requires a separate tool with a spatial index that can be the base for executing geo-search functions. It allows the end-user to search for geodata contained in or intersecting a boundary or within a time window. Users can limit the area they are interested in or they can limit the search results using selected filters, respectively. The geo-web service allows end-users to set different filters to refine their search results, e. g. language, geodata formats (vector, raster, etc.).

All searches result in the end-user ordering one or more IP packages and then accessing the geodata DIP_u. Since geodata requires a special geodata viewer some Access specific circumstances emerge.

3.6.3.2 Access to geodata

Archives can offer two different forms of access to geodata: unstructured files; and geodata rendered via a web service. The manner in which geodata is accessed depends on the user's ability to work with raw geodata sets and their knowledge of geodata tools (QGIS, etc.).

3.6.3.2.1 Unstructured files

Advanced end-users such as geodata experts can work with raw data using QGIS. They can work with QGIS in the reading room using the archive's infrastructure or order a copy of a specific set of data to use within their own tools outside the reading room, according to the archive's policy.

3.6.3.2.2 Web service

Serving geodata through OGC web services[93], though technically more demanding, enables a broader access to geodata and enforces greater control over how data is accessed and manipulated; it can also prevent reuse. It brings geodata to those without specific knowledge of working with geodata. Access can be made possible via a Web or mobile application (login or some sort of authentication is needed) or externally (free web access for everyone), depending on the archival policy. The archive can also set up its own GeoServer, which is accessed only within the archival network.

A web service allows viewing layers of geodata - permanent DIPs that have no access restriction. Depending on archival policy, the DIP_u can be prepared for a certain end-user, based on the end-user's search results and order. Such access would require a user to log-in in order to see the order. The end-user can view his/her order from the reading room or from anywhere using the internet, again depending on the archival policy. It is also possible to recreate maps from multiple layers of geodata and add query access according to documentation.

[93] Open GIS Consortium standards for web services (WMS, WCS, WFS, WMTS…)
https://en.wikipedia.org/wiki/Open_Geospatial_Consortium

3.6.3.2.3 Edited and customized view in QGIS

The QGIS viewer enables end-user (archivist or the user in the reading room) manipulation of geodata. Access can again be full or restricted, which is negotiated in the process step of the initial order. The same rules (depending on archival legislation) as for any other data type apply also for geodata.

A. The user with full access (archivist – the keeper of the archived records and user in the reading room with allowance) can edit, manipulate and view geodata DIP_u in whatever way they see fit. Within QGIS they can perform several types of action, like simplification, selection of elements, transformation. With access to documentation, they can recreate the visualization and can recreate GIS projects.
B. If the geodata has to be anonymized, an archivist or employee with knowledge of geodata manipulation modifies the DIP_0. The DIP_0 is then transformed into a DIP_u and is made ready for the end-user for reading room use or reproduced for outside use.

The end-user with restricted access can work with a geodata DIP_u which has been modified by an archivist or employee with geodata knowledge. (S)he can manipulate data within QGIS in the same way as users with full access, only the data-sets are different.

3.6.3.2.4 Reproduction of geodata

The end-user can order a reproduction of a set of geodata. They can order an electronic copy or create maps in QGIS, which can be printed.

4 Annex A - A Proposed E-ARK Standard for Vendor-Independent Archiving of Data Warehouses

4.1 Introduction

Compared to the problem of archiving relational database tables, the problem of archiving data warehouse (DW) structures and data has received remarkably little in the way of research attention, and seems to have been largely left to the big system integrators, such as IBM and Informatica, to resolve using proprietary solutions sitting on top of final proprietary databases, such as DB2 or ORACLE. More recently, Big Data configurations based around the HADOOP open source ecosystem have become popular.

If research is limited, then successful standardisation efforts are even more difficult to identify and there is no equivalent of the SQL 2011 standard INFORMATION_SCHEMA tables, which is discussed in a separate paper. Equally lacking is a systematic overview of similarities and differences in the ways different vendors have implemented their interpretations of data warehousing theory, as originally propounded chiefly by Inmon and Kimball. These interpretations will doubtless be coloured by past design decisions relating to data dictionary structures, most of which predated the arrival of the INFORMATION_SCHEMA concept in the SQL standard.

The most significant standardisation initiative in this area was the Common Warehouse MetaModel (CWM) specification, introduced more than a decade ago and supported by a number of the then key commercial players in the field. While ambitious in the scale and scope of its conceptual coverage, this very breadth and generality (the specification document runs to more than 500 pages) may have been its undoing. By 2011, leading authorities in the DW field were declaring that as a standard it 'seems to be dead' and as a result, 'there appears to be no effective, active independent data warehouse metadata framework' (Mundy, Thornthwaite and Kimball 2011, p527).

Despite the perceived failure of a grand, over-arching scheme for DW metadata management, the growing need to archive DWs originating from different vendors, with multiple and sometimes incompatible versions of their own toolsets, means that a simpler, more pragmatic but operationally feasible approach must be sought. The purpose of this draft proposal is to initiate such a process.

4.2 The Range of Possible User Requirements for DW Archiving

A first step in clarifying and simplifying the archiving problem, is to assert that the focus will be on preservation of the logical structure of the data warehouse, rather than any specialised physical storage structures that may have been used for performance enhancement at the time the DW was originally constructed. Aside from the accepted division between logical and physical database design, which has a long pedigree in the literature, justification for this approach also derives from the continued pace of technological development on the hardware side – today's special purpose performance accelerator is matched by tomorrow's general purpose hardware/software configuration (with faster cpu/disk throughput and larger available memory).

A second, and possibly controversial (in some quarters) step, is to assert the greater generality of the Relational OLAP[94] approach to DW, as compared to the more specialised and less used MOLAP[95] or HOLAP[96]

[94] A Relational OLAP implementation means that all the dimensions and fact tables are represented as tables, while the metadata that specifies the linkages between facts and dimensions, and the hierarchical structure (if any) of the dimensions themselves, is all stored in data dictionary views. The SQL query processor then uses the SQL OLAP extensions, together with the metadata, to create virtual OLAP cubes or subsets of cubes at query

approaches. A particular reason for favouring the ROLAP approach in a DW archiving context is also that it will allow the use of tools already developed in E-ARK and other projects for archiving standard relational databases, thereby lessening the burden of required new development of concepts/tools specifically for DW archiving purposes. A possible corollary of going down the ROLAP only route is that older legacy MOLAP/HOLAP systems will need to provide the ability to move data into ROLAP-only structures. Further investigation will be needed to determine whether this is likely to prove a widespread impediment to effective archiving workflows in future, or whether its impact will be limited to isolated special cases where vendors have not already provided the requisite functionality.

It should be noted that adopting ROLAP to the exclusion of other approaches also has (potentially beneficial) implications for the problem of archiving OLAP cubes. This will be discussed in more detail in a later section.

The third basic tenet of the archiving approach to be adopted is that it will assume the use of the Kimball methodology[97] for DW design. This is based on the deployment of dimensional modelling techniques to identify fact/cube tables and their associated dimensions as the basis for subsequent OLAP querying. Compared to third normal form relational databases, such structures may demonstrate a high degree of de-normalisation. The alternative approach to DW design associated with the work of Inmon[98] is much more strongly based on standard normalised relational tables, which can be handled using existing relational database archiving methods and tools. The Kimball methodology will be adopted here as it differentiates the data warehouse approach more clearly from standard relational database approaches and it focuses on the use of dimensions which require additional metadata to be described properly for archival purposes.

Taken together, these three tenets mean that the DW archiving problem can be focused on capturing the essential elements of the ROLAP structure. These elements include :

- All Fact/Cube tables are represented as underlying relational tables
- The same applies to the basic data content of Dimension tables
- Linkages between Fact tables and Dimensions are implemented in the same way as primary/foreign key linkages between standard relational tables
- Dimensions may have additional internal structure in the form of hierarchies of attributes representing different levels of aggregation/dis-aggregation available to OLAP queries
- OLAP cubes are ‘virtual’ (see section below)

To provide the necessary metadata for these elements, the focus must be on the definition of dimensions, since the CREATE TABLE statement for the fact table is assumed to document which measures are foreign keys for its associated dimensions. This is achieved in the usual manner by declaring the appropriate referential integrity constraints.

---

time, but no OLAP cubes are actually stored, unless the performance enhancing facility of materialised views is deployed.

[95] Multidimensional online analytical processing. In a MOLAP implementation, specially optimised storage structures (which may have a hardware component), rather than standard relational tables, are used to hold the OLAP cube.

[96] A HOLAP or hybrid OLAP implementation uses a combination of relational OLAP and MOLAP technology, where what are essentially materialised views are off-loaded from the relational OLAP sub-system on to a MOLAP engine sub-system.

[97] Kimball, R. and Ross, M. (2013) The data Warehouse Toolkit: the Definitive Guide to Dimensional Modeling. 3rd Ed. Wiley: Indianapolis.

[98] Inmon, W. H. (2005) Building the Data Warehouse. 4th ed., Wiley: Indianapolis.

To characterise the essential structural elements of each Dimension, three metadata tables are required. The first, EARK_DIM, provides basic naming and level count information. The second table, EARK_DIM_HIERARCHY describes the hierarchical structure of the dimension, in terms of ordering of levels, if such a hierarchy is present (which may or may not be the case). The third and final table, EARK_DIM_LEVEL, lists the attributes associated with each level in each dimension.

4.3 Detailed metadata table definitions

Table **EARK_DIM**

Column Name	Data Type	Description
SCHEMA_NAME	VARCHAR2(128)	
DIMENSION_NAME	VARCHAR2(128)	Unique to SCHEMA
NO_OF_LEVELS	INTEGER	
SLOWLY_CHANGING_DIM_TYPE	INTEGER	Kimball and Ross (2013) classification of slowly changing dimension types
BASE_ROLE_PLAYING_DIM_NAME	VARCHAR2(128)	Dimension name for which this acts as role playing alias. The base dimension is expected to be in the same schema

Table 34 - EARK_DIM

Table **EARK_DIM_HIERARCHY**

Column Name	Data Type	Description
SCHEMA_NAME	VARCHAR2(128)	
DIMENSION_NAME	VARCHAR2(128)	
HIERARCHY_NAME	VARCHAR2(128)	
HIERARCHY_LEVEL_NAME	VARCHAR2(128)	
PARENT_LEVEL_NAME	VARCHAR2(128)	NULL at highest (most general) level of hierarchy

Table 35 - EARK_DIM_HIERARCHY

There is no row in this table for a given dimension if the number of levels = 1, i.e. there is no hierarchy present, as may be the case in factless fact tables.

There may be multiple overlapping hierarchies present in a single dimension.

Snowflake dimensions are deprecated by Kimball, as adding complexity without any gain in functionality, as linked sets of sub-tables can be reduced to a larger single denormalised dimension table. It is assumed that

this convention is to be followed when submitting organisations are advised of the requirements for archiving (advice best given at the outset of the DW project!).

Table **EARK_DIM_LEVEL**

Column Name	Data Type	Description
SCHEMA_NAME	VARCHAR2(128)	
DIMENSION_NAME	VARCHAR2(128)	
HIERARCHY_LEVEL_NAME	VARCHAR2(128)	
LEVEL_ATTRIBUTE_NAME	VARCHAR2(128)	
SCD_ATTRIBUTE	VARCHAR2(128)	Is attribute part of provision for a slowly changing dimension (YES?/NO)
RAGGED_ATTRIBUTE	VARCHAR2(128)	Is attribute part of ragged hierarchy (YES/NO)

Table 36 - EARK_DIM_LEVEL

The archival workflow associated with these tables is envisaged as follows:

- When preparing a DW for archiving, the above metadata tables will be populated, either manually or in (semi-)automated fashion from DW design documentation, the source system's data dictionary views, or equivalent metadata stores
- The completed tables will then be converted to SIARD 2.0 format for AIP storage
- The base (relational) data tables for the FACT and DIMENSION tables will also be converted into SIARD 2.0 format and stored in the same way in the AIP
- At the DIP stage, appropriate tools will translate the SIARD 2.0 format metadata and data tables (the latter with their accompanying referential integrity constraints) back into standard relational tables which can be stored in the target dissemination system
- The accessible contents of the metadata tables can then be read by software specific to the target dissemination system to generate either system specific SQL extension commands, such as the ORACLE CREATE DIMENSION command, or relevant programming utility calls (e.g. in SQL SERVER) to set up the DW structure for subsequent use by SQL OLAP commands in the target system. N.B. the base data tables will have to be re-established in the target system first, before the DW metadata can be super-imposed upon them.

4.4 OLAP Cube Considerations

Restricting the archiving process to ROLAP DW implementations has important implications for the question of OLAP cube storage.

In the ROLAP model, as noted above, by default the OLAP cube is *not* generated as a separate entity from the FACT and DIMENSION tables, i.e. it is a 'virtual' cube, the required elements of which are only generated at query time by the OLAP engine. If the SQL query containing the OLAP extensions is then stored as a view, this creates the appearance of a stored cube for the user, but the relevant OLAP cube cell values are still only computed at query time. For performance reasons, a further step, to create 'materialised views' may be undertaken. In this case, computed values are indeed stored separately from the original FACT and DIMENSION tables, but additional DW infrastructure is then required to ensure that

these materialised views remain current with the latest versions of data in the FACT table, if new data are being added to the latter over time.

In the archiving context, when the DW data will only be used on an occasional basis rather than for near real-time decision support, there does not appear to be any justification for storing materialised views. Conversely, the case for archiving stored OLAP queries as VIEWS seems to be very strong, as this helps retain aspects of the business logic associated with the main uses of the original DW system.

It should also be noted that the original CWM structure had no provision for 'the concept of pre-calculated aggregations', i.e. the storage of the results of OLAP queries, and it is proposed that the EARK project adopts the same stance.

Should there be a need to archive certain specific aggregated OLAP query results, e.g. census output tables, they can be stored as standard relational tables, subject to the usual SIARD 2.0 processing for archival processing, since the result of every OLAP SQL query, as with every SQL query, has the form of a table, albeit not necessarily a very convenient form for later post-DIP processing. A second approach, which is expected to gain further traction with governmental organisations in future, is to store such pre-calculated aggregations in the SDMX interchange format used by EUROSTAT and other European Census and Statistical Organisations. This route is therefore recommended for those organisations that require this type of functionality.

4.5 Appendix : Observations on the vendor-independent archiving of Data Warehouses in the context of developments in Big Data technology

A second possible route to DW archiving, which chimes with the E-ARK project interest in a data mining showcase, would be to make a 'Big Data' HADOOP ecosystem configuration the target for storing of DW data, both fact tables and dimensions. For use in archives, however, it may be wise to observe the following 'health warning', courtesy of one of the major vendors in the area of Big Data technology:

'*Getting started with the Apache Hadoop stack can be a challenge, whether you're a computer science student or a seasoned developer. There are many moving parts, and unless you get hands-on experience with each of those parts in a broader use-case context with sample data, the climb will be steep'*

(http://www.cloudera.com/developers/get-started-with-hadoop-tutorial.html viewed July 2016).

Since the lowest barrier to entry is set at the computer science student, this has unfortunate echoes of 1960s database technologies where high levels of expertise were required to make even basic use of the tools then available. The experience to date of the present writer and his team, which includes both database and parallel processing specialists, would seem to bear out these concerns.

That said, where groups have the requisite expertise, and this may include some larger archives but not smaller ones, open source Big Data (BD) approaches may have considerable mileage, especially as the tools improve and market leaders become more established. BD approaches excel for trawling through very large columnar datasets (provided the tables are not too wide…), multi-format data resources, and handling of data in newer formats such as json. They are also scalable on commodity hardware, provided equipment costs are not a constraint. It should, however, be noted that the big commercial vendors, such as IBM and ORACLE are not standing still in this area and are very rapidly reducing the gaps in their capabilities originally opened up by the advent of HADOOP-based BD approaches, so Archives which already have

established relationships with these vendors may wish to continue working with them, rather than moving to open source.

It must also be clearly stated, that at the present time, BD approaches cannot replicate the capabilities of a properly configured SQL OLAP system sitting on top of a well-designed DW structure, without extensive additional programming to mimic the functionality that the latter systems already provide 'out-of-the-box'. This point is conveniently under-emphasised in much of the technological marketing hype surrounding many of the new BD tools.

Possibly the most promising of the newer BD developments in the present context is the APACHE IMPALA project, supported by BD vendors such as CLOUDERA and MAPR. This provides an SQL interface to BD resources and was designed for user interactivity, unlike the original HADOOP, which was essentially conceived as a scalable batch processing engine. IMPALA is not yet very widely used in the marketplace, but is steadily maturing as a product. That said, some of the earlier expansive claims for how rapidly its functionality would expand, seem to have been scaled back recently. Importantly, at present it does not support SQL OLAP extensions such as the ROLLUP command, so cannot mimic DW operations and the present writer has not yet been able to find a recent product roadmap indicating when such support may be forthcoming.

It should also be emphasised that SQL OLAP extensions are not minor enhancements to functionality, because they presuppose the existence of metadata, which documents the hierarchical structure of DW dimensions and how they link to DW fact/cube tables. If IMPALA needs to provide such metadata, this will clearly move it away from the claimed BD benefit of working with minimally structured data resources, and it will effectively be re-inventing the wheel in comparison with the mature DW technology of the established vendors.

The only alternative to effective use of metadata is to 'denormalise' the DW structure of cube tables and dimensions, though for larger applications this would produce extremely wide and cumbersome tables with hundreds, if not thousands of columns (which require PARQUET format storage for efficient processing in IMPALA). This would simplify the query process, though it does not remove the need for the ROLLUP functionality entirely, provided the user could keep track of the meaning of all the columns in the monolithic structure. Since the PARQUET format is an efficient binary compression format (developed by TWITTER and CLOUDERA) it is interesting to ask whether the benefits of the compression are sufficient to offset the very substantial increase in the size of already large DW tables once they are denormalised. Also, IMPALA does not use indexes as commonly understood in RDBMS environments, and these RDBMS indexes also employ various forms of efficient compression in the trade-off between access speed and storage volumes. Published benchmark tests by CLOUDERA against an un-named RDBMS indicated that IMPALA was faster in the vast majority of cases examined (16 out of 19). However, every vendor will wish to present their own product in the best possible light, and without knowing details of the un-named RDBMS and whether it was optimally configured for the workload in question, it is difficult to assess the generality of these findings, though they are promising, at first sight. What was clear from other CLOUDERA tests, however, was that IMPALA was much faster, especially under multi-user load, than other candidate tools in the open source HADOOP ecosystem.

A fully automated general purpose approach to the extraction of DW metadata and the use of this metadata to (re-)construct operational warehouses does not seem feasible, without very large expenditure of time and resources, owing to the very different DW implementation strategies adopted by different vendors. That said, the concept of standardised tables of essential metadata, as developed in this paper,

based on Kimball's concept of dimensional hierarchies, to capture the essential aspects of each DW design, does seem to have significant

value for archiving ROLAP-based DWs, as well as long-term documentation and facilitation of data sharing/design convergence between different Archives and data generating organisations.

In the short- to medium-term, DWs certainly have value in their own right for holding tranches of data derived from transactional systems and denormalisation may be an appropriate strategy, depending on the DW design and user requirements. In the longer-term, these DWs will themselves need to be archived. Based on the above paper and this appendix, such archiving could be achieved in one of several ways. Firstly, using vendor-supplied export formats, though these may be subject to change, and backward compatibility may well become an issue over time. Secondly, the individual relational tables representing the fact/cube table(s) and the dimensions could be archived using the currently available tools, together with the standardised metadata tables proposed above (and in a separate paper), to preserve the details of the structure of dimensions and their key linkages to the fact/cube table(s). Thirdly, the DW structure could be 'flattened' into very wide tables and loaded into a BD system. For the latter, on say a three-year timescale, IMPALA currently looks like the best vehicle, provided ROLLUP type SQL extensions are implemented, although at present it could scarcely be regarded as a mature product, compared to the commercial DW systems. Of course, there would be nothing to prevent the standardised tables containing details of the original DW structure from also being loaded into the BD system for documentation purposes, as they may aid users in the formulation of sensible queries against the now relatively unstructured 'big table'. Indeed, this could prove to be a very useful thing to do over the longer term. The advantages of IMPALA for DW archiving are potential scalability (though this attribute is shared by the better commercial systems) and future-proofing, since there is no vendor-specific metadata that is retained. However, this future-proofing may not extend to changes in the structure of the data, such as the addition of new columns, e.g. when new attributes are added to a dataset, perhaps on a yearly basis, as a result of changing legislation and hence changing data collection requirements. Data re-organisation at the physical file level, as well as the logical level, will typically be required under these circumstances.

One final observation would be that from an Archives perspective, it is much more satisfactory in terms of data quality, to use BD solutions for archiving data coming from a DW, than it is to bypass the DW stage and only use BD technology. This is because the DW imposes stringent requirements on data quality and integrity, through its various in-built checking and structuring mechanisms. Such requirements are much harder to 'police' in the more flexible BD environment

5 Annex B - Proposed E-ARK standard for relational database metadata table structure

Earlier deliverables have reported the need to develop a project standard for relational database metadata pertaining to the use of constraints that can be used to document join conditions between tables, where these join conditions were part of the original database design prior to archiving. These constraints can subsequently be used for automated or semi-automated de-normalisation. In some cases the constraints may be explicit in the data dictionary of the database being archived, in which case it should be possible to 'harvest' them for insertion into the E-ARK metadata structure. In other cases, however, the constraints may be explicit (or even implicit) in various kinds of database documentation, such as E-R diagrams, whether in paper or digital form,. It may also be possible to infer the constraints from frequently used query macros stored along with the database to be archived (or held within the database as views).

The main initial problem is that there is no universally agreed and implemented standard for how information is stored within relational database dictionaries, either in terms of the structure of tables/views that may be employed, or in the naming conventions used. That said, since the introduction of SQL92 there has been a small and evolving section of the standard that introduced the concept of the 'INFORMATION_SCHEMA' whose job was to address aspects of this recognised deficiency. Unfortunately, this standard has not been implemented by all vendors, ORACLE being a key example, and even where vendors have provided implementations, these have begun to diverge to take account of system specific issues (e.g. MySQL and SQL Server databases).

Given this cacophony of non-standard voices, it is reasonable to propose a new vendor-independent standard, at least for the database archiving process, so relevant non-standard data on constraints can be channelled into a common framework. As a result, a series of vendor-specific lightweight conversion tools can be envisaged to support this process, in addition to a minimal data entry system that allows constraint data from non-data-dictionary sources to be added by archivists where it is lacking, thereby adding value to the archived database in question.

Triggers not addressed even though under certain circumstances they may contain information about the design and intended operation of database applications.

EARK_REFERENTIAL_CONSTRAINTS

Column Name	Data Type	Description
CONSTRAINT_SCHEMA	VARCHAR2(128	
CONSTRAINT_NAME	VARCHAR2(128)	
REF_UNIQUEKEY_CONSTR_SCHEMA	VARCHAR2(128)	
REF_UNIQUEKEY_ CONSTRAINT_NAME	VARCHAR2(128	
MATCH_OPTION	VARCHAR2(128)	DEFAULT=NONE
UPDATE_RULE	VARCHAR2(64)	

DELETE_RULE	VARCHAR2(64)	
TABLE_NAME	VARCHAR2(64)	
REFERENCED_TABLE_NAME	VARCHAR2(128)	

Table 37 - E-ARK Referential Constraints

CONSTRAINT_SCHEMA and CONSTRAINT_NAME identify the foreign key in TABLE_NAME. Constraint names are assumed to be unique to a schema.

REF_UNIQUEKEY_CONSTR_SCHEMA is short for REFERENCED UNIQUEKEY CONSTRAINT SCHEMA. Likewise for the associated name column

MATCH_OPTION is included for use in future versions of the SQL standard

Allowable UPDATE_RULE and DELETE_RULE options may vary between systems. The generally agreed possibilities are :

CASCADE, SET NULL, SET DEFAULT, RESTRICT, NO ACTION

In ORACLE, the only explicit UPDATE_RULE that can be set is NO ACTION, while the DELETE_RULE can be set to NO ACTION, CASCADE or SET NULL. If no rule is specified by the user, NO ACTION takes effect.

The difference between RESTRICT (not a valid ORACLE keyword) and NO ACTION relates to the timing of when the constraint is evaluated, e.g. with regard to transactions etc.

In ORACLE, SET DEFAULT requires the use of a database TRIGGER

While knowledge of UPDATE_RULE and DELETE_RULE options is not required for purposes of automated de-normalisation, these columns form part of the SQL 2011 INFORMATION_SCHEMA standard and they provide hints about how referential constraints between database tables were intended to operate. This information may be of value to archivists at a later stage.

EARK_KEY_COLUMN_USAGE

Column Name	Data Type	Description
CONSTRAINT_SCHEMA	VARCHAR2(128)	Name of Schema containing the constraint
CONSTRAINT_NAME	VARCHAR2(128)	Name of constraint
TABLE_SCHEMA	VARCHAR2(128)	Name of Schema containing the table
TABLE_NAME	VARCHAR2(128)	Name of table
COLUMN_NAME	VARCHAR2(128)	Name of column that is constrained
ORDINAL_POSITION	NUMBER	Ordinal position of column within constraint (not within table) since constraints may reference multiple columns
POSITION_IN_UNIQUE_CONSTRAINT	NUMBER	Value is NULL for UNIQUE and PRIMARY KEY constraints. For FOREIGN KEY constraints it is the ordinal position of the key column in a UNIQUE

		constraint in the table being referenced. FOREIGN KEYS are permitted to link to UNIQUE CONSTRAINTS in this way.
REFERENCED_TABLE_SCHEMA	VARCHAR2(128)	Name of schema containing table referenced in constraint
REFERENCED_TABLE_NAME	VARCHAR2(64)	Name of referenced table
REFERENCED_COLUMN_NAME	VARCHAR2(64)	Name of referenced column in referenced table

Table 38 - E-ARK Key Column Usage

This sub-table is required to handle cases where the constraints involve multiple columns.

There is seemingly some unnecessary redundancy between this table and EARK_REFERENTIAL_CONSTRAINTS, but this has not been removed in this first draft proposal, to maintain a higher level of conformity with the SQL standard, in terms of the columns which appear in the respective tables. To the present writer the ORACLE approach, in its data dictionary ALL_CONSTRAINTS (USER_CONSTRAINTS) and related tables appears more logical. This may explain why ORACLE have to date declined to implement the SQL standard in this area.

Automated denormalisation direct from ORACLE data dictionary views

This section is included to provide a comparison with the later implementation of automated denormalisation using the new E-ARK standard tables described above. It also shows just how different the ORACLE approach is from the SQL 'standard' INFORMATION SCHEMA.

The two ORACLE data dictionary views of interest here are firstly : USER_CONSTRAINTS (a system-wide view ALL_CONSTRAINTS can also be substituted, if required) :

Name	**Null**	**Type**
OWNER		VARCHAR2(120)
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)
CONSTRAINT_TYPE		VARCHAR2(1)
TABLE_NAME	NOT NULL	VARCHAR2(30)
SEARCH_CONDITION		LONG()
R_OWNER		VARCHAR2(120)
R_CONSTRAINT_NAME		VARCHAR2(30)
DELETE_RULE		VARCHAR2(9)
STATUS		VARCHAR2(8)
DEFERRABLE		VARCHAR2(14)
DEFERRED		VARCHAR2(9)
VALIDATED		VARCHAR2(13)

GENERATED	VARCHAR2(14)
BAD	VARCHAR2(3)
RELY	VARCHAR2(4)
LAST_CHANGE	DATE
INDEX_OWNER	VARCHAR2(30)
INDEX_NAME	VARCHAR2(30)
INVALID	VARCHAR2(7)
VIEW_RELATED	VARCHAR2(14)

The second view is USER_CONS_COLUMNS:

Name	**Null**	**Type**
OWNER	NOT NULL	VARCHAR2(30)
CONSTRAINT_NAME	NOT NULL	VARCHAR2(30)
TABLE_NAME	NOT NULL	VARCHAR2(30)
COLUMN_NAME		VARCHAR2(4000)
POSITION		NUMBER

For present purposes, a trivial example involving two tables MYTEST11 and MYTEST12, linked by a referential integrity constraint, will be used. Their CREATE TABLE commands are as follows;

```
create table mytest12
        (col1 number primary key,
        col2 number,
        col3 varchar2(20))

create table mytest11
        (col1 number,
        col2 number,
        col3 varchar2(20),
        constraint ref_check1 FOREIGN KEY (col1)
                REFERENCES mytest12(col1))
```

All the following queries have been tested in ORACLE, using the data in the data dictionary associated with these tables. Tests (not shown) have also been undertaken with other tables containing multi-column primary and foreign keys.

To introduce the use of the ORACLE data dictionary views, a series of illustrative queries will now be constructed.

The following query lists details of referential constraints for tables in the current ORACLE schema whose table names begin with the letters 'MY...'

```
select owner,constraint_name, constraint_type, table_name,
```

```
r_owner, r_constraint_name

from user_constraints

where constraint_type ='R'

and table_name like 'MY%'
```

The following query takes the above information from the USER_CONSTRAINTS and USER_CONS_COLUMNS data dictionary views on both the referential constraint (constraint_name) in the first table and the unique or primary key constraint it accesses (r_constraint_name) in the second table and uses them to find the primary key column name(s) in the second table.

```
select table_name,column_name

from user_cons_columns

where constraint_name in

(select r_constraint_name

from user_constraints

where constraint_type ='R'

and table_name like 'MY%')
```

The following query is a slightly modified version of the previous one and it returns the primary key column name (s) in the first table for the primary key that is linked to the foreign key in the second table.

```
select table_name,column_name

from user_cons_columns

where constraint_name in

(select constraint_name /* this is the changed column*/

from user_constraints

where constraint_type ='R'

and table_name like 'MY%')
```

The information derived from these two queries above is that required to reconstitute a join between the two tables for automated denormalisation purposes where the only information available about how joins should take place is that stored in the data dictionary when the tables were created with the relevant referential integrity constraints in place (or had had the constraints added subsequently using ALTER TABLE syntax)

If we then UNION the two queries, as follows, the building blocks for the join conditions between the two tables are then available to us :

```
select table_name,column_name

from user_cons_columns

where constraint_name in
```

```
(select constraint_name
from user_constraints
where constraint_type ='R'
and table_name = 'MYTEST11')
union
select table_name,column_name
from user_cons_columns
where constraint_name in
(select r_constraint_name
from user_constraints
where constraint_type ='R'
and table_name = 'MYTEST11')
```

Recasting the UNION query as a nested query with relational joins is one (slightly convoluted) way of allowing us to generate the SQL for the join condition for automated denormalisation. Further adding a prior query to generate the SELECT list, results in the following pair of SQL commands:

```
select tabnam from (select 'select * from '||ucc.table_name||',' tabnam, u.constraint_type typnam
from user_cons_columns ucc,user_constraints u
where ucc.constraint_name = u.r_constraint_name
and u.constraint_type ='R'
and u.table_name = 'MYTEST11'
union
(select ucc.table_name tabnam, u.constraint_type typnam
from user_cons_columns ucc,user_constraints u
where ucc.constraint_name = u.constraint_name
and u.constraint_type ='R'
and u.table_name = 'MYTEST11'))
order by typnam
/
select colnam from (select 'where '||ucc.table_name||'.'||ucc.column_name||'=' colnam, u.constraint_type typnam
from user_cons_columns ucc,user_constraints u
where ucc.constraint_name = u.r_constraint_name
and u.constraint_type ='R'
```

```
and u.table_name = 'MYTEST11'

union

(select ucc.table_name||'.'||ucc.column_name colnam, u.constraint_type typnam

from user_cons_columns ucc,user_constraints u

where ucc.constraint_name = u.constraint_name

and u.constraint_type ='R'

and u.table_name = 'MYTEST11'))

order by typnam

/
```

With minor Oracle embellishments, such as use of the SET HEADING OFF command, a SPOOL command to create an output SQL file and replacing the hard-coded table name 'MYTEST11' with an SQL*PLUS substitution variable, the result will be a functioning general purpose multi-query macro for automated generation of denormalisation macros for any pair of tables which are linked in a referential integrity constraint (stored in the data dictionary) by a single column.

Output from the above query for the simple case where the foreign key COL1 in MYTEST11 references the primary key COL1 in the linked table MYTEST12, produces the following output, which itself has the form of an SQL query:

```
select * from MYTEST12,

MYTEST11

where MYTEST12.COL1=

MYTEST11.COL1
```

Which can be run in ORACLE SQL*DEVELOPER to produce a simple denormalised data output that merges the contents of the two tables.

A further query on the USER_CONSTRAINTS view could also be added to the top of the multi-query macro above, to list all the names of tables for which referential integrity constraints were currently stored in the database. Any of these names could then be input via the substitution variable. The query would be of the following form :

```
select distinct table_name from user_constraints

where constraint_type='R'

order by table_name
```

The same general approach to querying of the data dictionary views applies for multi-column keys but generalising to this case without arbitrary limits on the number of allowed columns in the keys would best be handled by an ORACLE program written in JAVA or PL/SQL with embedded SQL. This is because each column in the key requires an additional row in the relevant data dictionary view and this situation is best handled via cursor processing.

A similar point about cursor processing applies in the case where a single large table has a number of coded columns and in the course of the denormalisation process we would wish to decode all of the coded columns. This will mean there are a number of referential integrity constraints ,all applying to the same table, and each of which will have a separate row (or rows) of data in the relevant data dictionary views.

Example of inter-system transfer of data dictionary metadata to comply with the new E-ARK 'standard'– moving ORACLE data dictionary data into the new E-ARK metadata tables.

Again this is a somewhat convoluted process, but can be achieved using the same kind of approach as that indicated in the previous section. A combined INSERT…SELECT construct, allows the ORACLE data to be retrieved from multiple data dictionary views, then loaded into the appropriate columns in the E-ARK tables, as follows:

For the EARK_REFERENTIAL_CONSTRAINTS table:

```
insert into eark_referential_constraints
(CONSTRAINT_SCHEMA,CONSTRAINT_NAME, REF_UNIQUEKEY_CONSTR_SCHEMA,
REF_UNIQUEKEY_CONSTRAINT_NAME, DELETE_RULE, TABLE_NAME, REFERENCED_TABLE_NAME)
select user, u.constraint_name, u.r_owner, u.r_constraint_name, u.delete_rule,
u.table_name, uc.table_name
from user_constraints u, user_constraints uc
where uc.constraint_name = u.r_constraint_name
and u.constraint_type ='R'
```

N.B. the above query employs the ORACLE pseudo-column 'user', which returns the current database user, assuming this user owns the constraint in question. Otherwise, the column 'owner' from the USER_CONSTRAINTS view can be utilised instead.

For the EARK_KEY_COLUMN_USAGE table:

```
insert into eark_key_column_usage
(CONSTRAINT_SCHEMA,CONSTRAINT_NAME, TABLE_SCHEMA,TABLE_NAME,COLUMN_NAME,
ORDINAL_POSITION, POSITION_IN_UNIQUE_CONSTRAINT,
REFERENCED_TABLE_SCHEMA, REFERENCED_TABLE_NAME, REFERENCED_COLUMN_NAME)
select distinct u.owner, u.constraint_name, user, u.table_name,ucc.column_name,
ucc.position, ucr.position, u.r_owner,ucr.table_name,ucr.column_name
from user_constraints u, user_constraints uc, user_cons_columns ucc,
(select uccs.table_name,uccs.column_name, uccs.position,uccs.constraint_name from
user_cons_columns uccs, user_constraints u
where uccs.constraint_name = u.r_constraint_name) ucr
where uc.constraint_name = u.r_constraint_name
```

```
and u.constraint_name=ucc.constraint_name

and ucr.constraint_name=u.r_constraint_name

and u.constraint_type ='R'
```

N.B. For less regular SQL users, this query utilises both the SQL ability to join a table to itself and the use of a nested SELECT in place of a table name.

Automated denormalisation using the E-ARK 'standard' metadata tables

It is interesting to note that, in the case of data originating in ORACLE, the transfer process described in the previous section has undertaken the 'heavy lifting' of making the data dictionary information more accessible to the automated denormalisation process.

Thus the query required for automated generation of simple SQL for denormalisation purposes, is itself quite straightforward:

```
set heading off

select 'select * from '||table_name||','||referenced_table_name||

' where '||table_name||'.'||column_name||' = '||referenced_table_name||

'.'||referenced_column_name

from eark_key_column_usage

where table_name='MYTEST11'
```

N.B. the italicised *set heading off* **is an ORACLE directive to switch off column headings, it is not an SQL command**

The above query generates the following SQL:

```
select * from MYTEST11,MYTEST12 where MYTEST11.COL1 = MYTEST12.COL1
```

which can be run directly in ORACLE SQL*DEVELOPER to achieve the required denormalisation

As noted earlier, more complex queries embedded in programming logic will be required to process the general case of multi-column keys and multiple referential integrity constraints linking a main table to a number of smaller tables.

Adding referential integrity constraints to databases where this information is currently lacking

This information can be added to the database before it is subject to archival processing, in which case, SQL constructs such as the relevant version of the ALTER TABLE command can be used to add constraints into the data dictionary. These can then be transferred into the E-ARK standard metadata tables using the methods described in this paper. Alternatively, data pertaining to the constraints can be added directly into the E-ARK metadata tables, bypassing the data dictionary in the originating database. In the case of ORACLE users, this could be achieved via macros and substitution variables in SQL*PLUS, or via SQL*DEVELOPER or indeed SQL*FORMS and its successor technologies.

Once the relevant metadata are stored in the E-ARK tables, denormalisation can either be undertaken prior to conversion of the data tables into SIARD format or, if denormalisation is to take place at a later stage in

the AIP-SIP-DIP process, SIARD format versions of the E-ARK metadata tables can be created at the same time as the SIARD conversion of data tables, transferred into the archival system, and accessed when required for denormalisation purposes.

6 Glossary

Access	Access refers to the funtional entity from the OAIS reference model https://public.ccsds.org/pubs/650x0m2.pdf
Access Aid	A software program or document that allows Consumers to locate, analyse, order or retrieve information from an OAIS. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Access Functional Entity	The OAIS functional entity that contains the services and functions which make the archival information holdings and related services visible to Consumers. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Access Rights Information	The information that identifies the access restrictions pertaining to the content information, including the legal framework, licensing terms, and access control. It contains the access and distribution conditions stated within the Submission Agreement, related to both preservation (by the OAIS) and final usage (by the Consumer). It also includes the specifications for the application of rights enforcement measures. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Access scenarios	Access scenario is used as a term to describe the environment, the DIP and the Access Software which altogether are used to render content information and associated metadata.
Access Software	A type of software that presents part of or all of the information content of an Information Object in forms understandable to humans or systems. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Archival Information Package	An Archival Information Package, consisting of the content information and the associated Preservation Description Information (PDI), which is preserved within an OAIS. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Archival Catalogue	See Finding Aid.
Archival record	Materials created or received by a person, family, or organization, public or private, in the conduct of their affairs that are preserved because of the enduring value contained in the information they contain or as evidence of the functions and responsibilities of their creator. Source Society of American Archivists: http://www2.archivists.org/glossary/terms/a/archival-records#.VyB5VXqd9iN
Authenticity	The degree to which a person (or system) regards an object as what it is purported to be. Authenticity is judged on the basis of evidence. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Common Specification	The common IP specification for E-ARK IPs conceived to constitute a common basis for the E-ARK SIP, AIP and DIP Specifications.

Compound Object	A Digital Object composed of multiple Files: for example, a Web Page composed of text and image Files.
Consumer	The role played by those persons or client systems, which interact with OAIS services to find preserved information of interest and to access that information in detail. This can include other OAIS's, as well as internal OAIS persons or systems. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf In E-ARK "Consumer" is an umbrella term that designates all users of archival holdings, thus both internal users, cf. archivists, and external users, cf. end-user.
Content Data Object	The Data Object that together with associated Representation Information comprises the Content Information. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Content Information	A set of information that is the original target of preservation or that includes part or all of that information. It is an Information Object composed of its Content Data Object and its Representation Information. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Content Information Type	The data types for which format specifications have been created, cf. Electronic Management Systems (ERMS), Simple File-Based System Records (SFBS), databases, and geo-data.
Database	A database is an organised collection of data. It is the collection of schemas, tables, queries, reports, views and other objects. Source: Wikipedia: https://en.wikipedia.org/wiki/Database
Database Preservation ToolKit (DBPTK)	The Database Preservation Tool Kit is a piece of software which, from an Access perspective, enables the loading of a SIARD file into an RDBMS http://keeps.github.io/db-preservation-toolkit/. It is developed by KEEP SOLUTIONS which is a partner of the E-ARK project http://www.keep.pt/en
Database Visualization ToolKit (DBPTK)	GUI conceived by the E-ARK project to view and analyse databases. Based on No SQL technologies.
Data warehouse	In computing, a data warehouse (DW or DWH), also known as an enterprise data warehouse (EDW), is a system used for reporting and data analysis, and is considered as a core component of Business Intelligence [1] environment. DWs are central repositories of integrated data from one or more disparate sources. They store current and historical data and are used for creating analytical reports for knowledge workers throughout the enterprise. Examples of reports could range from annual and quarterly comparisons and trends to detailed daily sales analysis.
Descriptive metadata	Also named Descriptive Information in OAIS: The set of information, consisting primarily of Package Descriptions, which is provided to Data Management to support the finding, ordering, and retrieving of OAIS information holdings by Consumers. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf The standard that E-ARK recommends for descriptive metadata is EAD.
Digital Object	An object composed of a set of bit sequences. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf

Digital Provenance	Documentation of processes in a Digital Object's life cycle. Digital provenance typically describes Agents responsible for the custody and stewardship of Digital Objects, key Events that occur over the course of the Digital Object's life cycle, and other information associated with the Digital Object's creation, management, and preservation. Source PREMIS: http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf
DIP_0	A provisional Dissemination Information Package directly derived from one or more AIPs, which may or may not be ready for use, according to the user's order and access rights.
DIP_p	A permanent Dissemination Information Package, available to be accessed indefinitely by users due to frequent requests for the same data. The DIPP can be available on-line.
DIP_u	A Dissemination Information Package, ready to be accessed, and previously checked against user's order and access rights.
DIP reference format	Refers to the E-ARK container format which is conceived to store the content information and its associated metadata.
DIP Representation Formats	The DIP representation formats are content specific implementations of the DIP reference format and offer examples of content information type specific scenarios.
DIP Status	The E-ARK DIP can have three statuses: See DIP_0, DIP_u and DIP_p.
Dissemination Information Package (DIP)	Dissemination Information Package: an Information Package, derived from one or more AIPs, and sent by archives to the Consumer in response to a request to the OAIS. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
EAD	Encoded Archival Description. A non-proprietary de facto standard for the encoding of Finding Aids for use in a networked (online) environment. Finding Aids are inventories, indexes, or guides that are created by archival and manuscript repositories to provide information about specific collections. While the Finding Aids may vary somewhat in style, their common purpose is to provide detailed description of the content and intellectual organization of collections of archival materials. EAD allows the standardization of collection information in Finding Aids within and across repositories. http://www.loc.gov/ead/eadabout.html
Electronic Records Management System (ERMS)	Electronic Records Management System is a type of content management system and refers to the combined technologies of document management and records management systems as an integrated system.
End-User	The end-user designates an external user who seeks content information in archival holdings.
ERMS Viewer	GUI conceived by the E-ARK project to view ERMS systems.
Exchange	Refers to the DIP as an exchange format, and as such it is essential that it is possible to transfer DIPs, for example between a repository and various Access environments.

Finding Aid	A type of Access Aid that allows a user to search for and identify Information Packages of interest. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Geodata	Geodata is information about geographic locations that is stored in a format that can be used with a geographic information system (GIS). Geodata can be stored in a database, geodatabase, shapefile, coverage, raster image, or even a dbf table or Microsoft Excel spreadsheet.
GeoTIFF	GeoTIFF is a public domain metadata standard which allows georeferencing information to be embedded within a TIFF file. The potential additional information includes map projection, coordinate systems, ellipsoids, datums, and everything else necessary to establish the exact spatial reference for the file.
GML	The Geography Mark-up Language: the XML grammar defined by the Open Geospatial Consortium (OGC) to express geographical features. GML serves as a modelling language for geographic systems as well as an open interchange format for geographic transactions on the Internet.
Graphical user interface (GUI)	A Graphical user interface (GUI) is a graphical interface to a program on a computer. It takes advantage of the computer's graphics capabilities to make the program easier to use.
Information Object	A Data Object together with its Representation Information. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Information Package	A logical container composed of optional content information and optional associated Preservation Description Information. Associated with this Information Package is Packaging Information used to delimit and identify the content information and Package Description information used to facilitate searches for the content information. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Intellectual Entity	A set of content that is considered a single intellectual unit for purposes of management and description: for example, a particular book, map, photograph, or database. An Intellectual Entity can include other Intellectual Entities; for example, a Web site can include a Web page; a Web page can include an image. An Intellectual Entity may have one or more digital representations. Source PREMIS http://www.digitizationguidelines.gov/term.php?term=intellectualentity
METS	The METS schema is a standard for encoding descriptive, administrative, and structural metadata regarding objects within a digital library, expressed using the XML schema language of the World Wide Web Consortium. The standard is maintained in the Network Development and MARC Standards Office of the Library of Congress, and is being developed as an initiative of the Digital Library Federation. Source http://www.loc.gov/standards/mets/
MultiDimensional DBMS	A MultiDimensional DBMS is a particular kind of RDBMS that is specifically geared towards OLAP (in fact MDDBMS is often used co-terminously with OLAP).

Normalisation	The term is used in two meanings: Firstly, in the sense in which the digital preservation community is employing the word: On Ingest, Content Data Objects are transformed into long-term friendly formats. Secondly, in database normalisation where columns and tables are organised in order to reduce redundancy.
OAIS	The Open Archival Information System is an archive (and a standard: ISO 14721:2003), consisting of an organization of people and systems that has accepted the responsibility to preserve information and make it available for a Designated Community. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
OLAP	In computing, online analytical processing, or OLAP, is an approach to answering multi-dimensional analytical (MDA) queries swiftly. OLAP is part of the broader category of business intelligence, which also encompasses relational database, report writing and data mining. Typical applications of OLAP include business reporting for sales, marketing, management reporting, business process management (BPM), budgeting and forecasting, financial reporting and similar areas, with new applications coming up, such as agriculture. Source Wikipedia https://en.wikipedia.org/wiki/Online_analytical_processing
OLAP Cube	OLAP is an acronym for online analytical processing. An OLAP cube is an array of data understood in terms of its 0 or more dimensions. OLAP is a computer-based technique for analysing business data in the search for business intelligence.
Order Management Tool (OMT)	The E-ARK tool that manages orders created in the E-ARK Access system.
order.xml	The xml-file that specifies an order in the E-ARK Access system.
Packaging Information	The information that is used to bind and identify the components of an Information Package. For example, it may be the ISO 9660 volume and directory information used on a CD-ROM to provide the content of several files containing content information and Preservation Description Information. Source: OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
PREMIS	The PREMIS Data Dictionary for Preservation Metadata is the international standard for metadata to support the preservation of Digital Objects and ensure their long-term usability. Developed by an international team of experts, PREMIS is implemented in digital preservation projects around the world, and support for PREMIS is incorporated into a number of commercial and open-source digital preservation tools and systems. The PREMIS Editorial Committee coordinates revisions and implementation of the standard, which consists of the Data Dictionary, an XML schema, and supporting documentation. Source: http://www.loc.gov/standards/premis/
Preservation metadata	Preservation metadata is an essential component of most digital preservation strategies. As an increasing proportion of the world's information output shifts from analog to digital form, it is necessary to develop new strategies to preserve this information for the long-term. Preservation metadata is information that supports and documents the digital preservation process. Preservation metadata is sometimes considered a subset of technical or administrative metadata. Source https://en.wikipedia.org/wiki/Preservation_metadata

	The standard that E-ARK recommends for preservation metadata is PREMIS.
Producer	The role played by those persons or client systems that provide the information to be preserved. This can include other OAIS's or internal OAIS persons or systems. Source OAIS: http://public.ccsds.org/publications/archive/650x0m2.pdf
QGIS	A Free and Open Source Geographic Information System. http://www.qgis.org/en/site/
Record	Any 'information created, received and maintained as evidence and information by an organisation or person, in pursuance of legal obligations or in the transaction of business' (ISO 15489-1:2001, 3.15). In MoReq2010®, a record may be further characterised as follows. ● It has an extensible set of metadata that describe it. ● It has one or more components that represent its content. ● It is classified with a business classification. ● It has a disposal schedule that describes explicitly if, how and when it will be disposed of or destroyed. ● It belongs to an aggregation of records. ● Access to it is controlled and limited to authorised users. ● Its destruction may be prevented by a disposal hold. ● It may be exported to another MCRS while retaining all of the characteristics listed above. [MoReq 2010, v 1.1]
Relational Database Management System	A relational database management system (RDBMS) is a computer software application that interacts with the user, other applications, and the database itself to capture and analyse data. A general-purpose RDBMS is designed to allow the definition, creation, querying, update, and administration of databases.
Representation	The set of files, including structural metadata, needed for a complete and reasonable rendering of an Intellectual Entity. For example, a journal article may be complete in one PDF file; this single file constitutes the representation. Another journal article may consist of one SGML file and two image files; these three files constitute the representation. A third article may be represented by one TIFF image for each of 12 pages plus an XML file of structural metadata showing the order of the pages; these 13 files constitute the representation. Source PREMIS: http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf , p.8
Representation Information	Representation Information is metadata that that transforms a Digital Object into an Information Object and thereby making it understandable by a human being. It consists of Semantic and Structure Information. Source OAIS: http://public.ccsds.org/publications/archive/650x0m2.pdf
Semantically marked up records formats (SMURF)	The SMURF is an IP format for ERMS systems and SFSB (simple file-system based records) conceived by the E-ARK project.
SFSB Viewer	GUI conceived by the E-ARK project to view Single File-Based System Records.
Simple File-Based System Records (SFSB)	Simple file-system based records: records that contain simple file-system based folders or files, including those originating from content and data management systems, such as SharePoint, that are not based on true file systems. They address the submission of computer files or folders from the file Producers rather than from

	an ERMS. They require manual enrichment with additional descriptive metadata
SIARD	IP format for databases. Currently there exist three versions: SIARD1.0, SIARDDK and SIARD2.0.
SIARD 1.0	SIARD1.0 is the original SIARD format developed by SFA. Available at: http://www.ech.ch/vechweb/page?p=dossier&documentNumber=eCH-0165&documentVersion=1.0
SIARD 2.0	SIARD2.0 is developed in E-ARK in collaboration with the Swiss Federal Archives (SFA), and is based on the original SIARD format developed by SFA. Available at: http://www.eark-project.com/resources/specificationdocs/32-specification-for-siard-format-v20
SIARDDK	SIARDDK is a format used in Denmark since 2010 and is a slight deviation from SIARD1.0.
Simple File-System Based Records (SFSB)	Simple file-system based records (SFSB) are records that contain simple file-system based folders or files, including those originating from content and data management systems, such as SharePoint, that are not based on true file systems. They address the submission of computer files or folders from the file Producers rather than from an ERMS. They require manual enrichment with additional descriptive metadata.
Structural metadata	Structural metadata describes the physical and/or logical structure of digital resources; it expresses the intellectual boundaries of complex objects and can be used to describe relationships between an object's component parts. Structural metadata is commonly used to facilitate navigation and presentation of complex items by defining structural characteristics such as pagination and sequence. And, like METS, can be used to aggregate related metadata. Source http://www.library.illinois.edu/dcc/bestpractices/chapter_11_structuralmetadata.html The standard that E-ARK recommends for structural metadata is METS
Submission Information Package (SIP)	An Information Package that is delivered by the Producer to the OAIS for use in the construction or update of one or more AIPs and/or the associated Descriptive Information. Source OAIS http://public.ccsds.org/publications/archive/650x0m2.pdf
Views (SQL)	In database theory, a view is the result set of a stored query on the data, which the database users can query just as they would in a persistent database collection object. This pre-established query command is kept in the database dictionary. Unlike ordinary base tables in a relational database, a view does not form part of the physical schema: as a result set, it is a virtual table computed or collated dynamically from data in the database when access to that view is requested. Changes applied to the data in a relevant underlying table are reflected in the data shown in subsequent invocations of the view. In some NoSQL databases, views are the only way to query data. Source Wikipedia https://en.wikipedia.org/wiki/View_(SQL)

Table 39 - Glossary

7 References and associated links and documents

Apache Solr http://lucene.apache.org/solr/

Apache Solr Reference Guide Solr Indexing
https://cwiki.apache.org/confluence/display/solr/Introduction+to+Solr+Indexing

Apache Solr Reference Guide Overview of Documents, Fields, and Schema Design
https://cwiki.apache.org/confluence/display/solr/Overview+of+Documents,+Fields,+and+Schema+Design

Common Specification v0.17
http://www.eark-project.com/resources/specificationdocs/67-e-ark-draft-common-specification-ver-017

D2.1 General Pilot Model and Use Case Definition http://www.eark-project.com/resources/project-deliverables/5-d21-e-ark-general-pilot-model-and-use-case-definition

D2.2 Legal Issues Report: European Cultural Preservation in a Changing Legislative Landscape
http://www.eark-project.com/resources/project-deliverables/33-d22-legal-issues-report-european-cultural-preservation-in-a-changing-legislative-landscape

D2.3 Detailed Pilots Specification http://www.eark-project.com/resources/project-deliverables/60-23pilotsspec

D3.1 Report on available best practices http://www.eark-project.com/resources/project-deliverables/6-d31-e-ark-report-on-available-best-practices

D3.3 E-ARK SIP Pilot Specification http://www.eark-project.com/resources/project-deliverables/51-d33pilotspec

D3.3 E-ARK SMURF http://www.eark-project.com/resources/project-deliverables/52-d33smurf

D3.4 E-ARK Records export, transfer and ingest recommendations and SIP Creation Tools http://www.eark-project.com/resources/project-deliverables/93-d34-1

D4.3 E-ARK AIP Specification http://www.eark-project.com/resources/project-deliverables/53-d43earkaipspec-1

D4.4 SIP-AIP conversion component http://www.eark-project.com/resources/project-deliverables/89-d44

D5.1 GAP report between requirements for access and current access solutions http://www.eark-project.com/resources/project-deliverables/3-d51-e-ark-gap-report

D5.2 E-ARK DIP Draft Specification http://www.eark-project.com/resources/project-deliverables/31-d52

D5.3 E-ARK Pilot DIP Specification http://www.eark-project.com/resources/project-deliverables/61-d53-pilot-dip-specification

D5.4 Search, Access and Display Interfaces http://www.eark-project.com/resources/project-deliverables/92-d54

D6.1 Faceted Query Interface and API http://www.eark-project.com/resources/project-deliverables/34-d61-faceted-query-interface-and-api

D6.2 Integrated Platform Reference Implementation http://www.eark-project.com/resources/project-deliverables/54-d62intplatformref-1

D6.3 Data Mining Showcase
http://www.eark-project.com/resources/project-deliverables/90-d63

Database Preservation Toolkit http://www.database-preservation.com/

Database Visualisation Toolkit https://github.com/keeps/db-visualization-toolkit

Describing and Preserving Digital Object Environments, New Review of Information Networking, 2013 Dappert, A., Peyraud, S., Delve, J., Chou, C., ISSN 1361-4576, 106-173

DIP, ERMS and SFSB Viewer http://178.62.194.129/ipviewer/

Digital Content Creation
http://www.library.illinois.edu/dcc/bestpractices/chapter_11_structuralmetadata.html

dm-file-ingest https://github.com/eark-project/dm-file-ingest

E-ARK DIP Format Requirements, internal unpublished deliverable, available on request

E-ARK Knowledge Center http://kc.dlmforum.eu./home

E-ARK Web https://earkdev.ait.ac.at:8443/cas/login?service

EAD3 https://www.loc.gov/ead/

EAD3 About http://www.loc.gov/ead/eadabout.html

EAD3 <accessrestrict> http://www.loc.gov/ead/EAD3taglib/index.html#elem-accessrestrict

EAD3 <c> http://www.loc.gov/ead/EAD3taglib/#elem-c

EAD3 <dao> http://www.loc.gov/ead/EAD3taglib/index.html#elem-dao

EAD3 <did> https://www.loc.gov/ead/EAD3taglib/index.html#elem-did

EAD3 @base https://www.loc.gov/ead/EAD3taglib/index.html#attr-base

EAD3 @level https://www.loc.gov/ead/EAD3taglib/index.html#attr-level

Encoded Archival Description Tag Library http://www2.archivists.org/sites/all/files/TagLibrary-VersionEAD3.pdf

Environment Function Type http://id.loc.gov/vocabulary/preservation/environmentFunctionType.html

ESSArch Preservaion Platform (EPP) http://epp.esearch.org

GeoTIFF https://en.wikipedia.org/wiki/GeoTIFF

GIS https://en.wikipedia.org/wiki/Geographic_information_system

GML, Geography Markup Language https://en.wikipedia.org/wiki/Geography_Markup_Language

Hadoop https://hadoop.apache.org/

Healey, R. G.: A Proposed E-ARK Standard for Vendor-Independent Archiving of Data Warehouses. Annex A

Healey, R. G.: Proposed E-ARK Standard for Relational Database Metadata Table Structure. Annex B

Lily http://www.lilyproject.org

METS, Metadata Encoding and Transmission Standard http://www.loc.gov/standards/mets/ and http://www.loc.gov/standards/mets/mets-schemadocs.html

MDDBMS https://en.wikipedia.org/wiki/Online_analytical_processing#Multidimensional_databases

OAIS, blue book, 2002 https://siarchives.si.edu/sites/default/files/pdfs/650x0b1.PDF

OAIS, Space data and information transfer systems - Open archival information system (OAIS) - Reference model, ISO 14721:2012 http://public.ccsds.org/publications/archive/650x0m2.pdf

OLAP, Online analytical processing https://en.wikipedia.org/wiki/Online_analytical_processing

Open GIS Consortium standards for web services (WMS, WCS, WFS, WMTS…) https://en.wikipedia.org/wiki/Open_Geospatial_Consortium

PREMIS 3.0 https://www.loc.gov/standards/premis/v3/

PREMIS Editorial Committee, PREMIS Data Dictionary for Preservation Metadata, v.3.0, June-November 2015 http://www.loc.gov/standards/premis/v3/premis-3-0-final.pdf

PREMIS, Intellectual entity http://www.digitizationguidelines.gov/term.php?term=intellectualentity

Preservation metadata https://en.wikipedia.org/wiki/Preservation_metadata

QGIS http://www.qgis.org/en/site/

Raster https://en.wikipedia.org/wiki/GIS_file_formats#Raster

RDBMS https://en.wikipedia.org/wiki/Relational_database_management_system

Redmine https://e-ark-redmine.magenta-aps.dk/

Repository of Authentic Digital Objects (RODA) http://www.roda-community.org/

Sansu George: Inmon vs. Kimball: Which approach is suitable for your data warehouse? http://www.computerweekly.com/tip/Inmon-vs-Kimball-Which-approach-is-suitable-for-your-data-warehouse

SIARD2.0 http://www.eark-project.com/resources/specificationdocs/32-specification-for-siard-format-v20

SQL View https://en.wikipedia.org/wiki/View_(SQL)

Structural metadata
http://www.library.illinois.edu/dcc/bestpractices/chapter_11_structuralmetadata.html

Vector https://en.wikipedia.org/wiki/GIS_file_formats#Vector