RECOMMENDATION

for storing large objects outside the SIARD file

Project Acronym: **E-ARK**

Grant Agreement Number: **620998**

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

AUTHOR(S)	
Name(s)	**Organisation(s)**
Anders Bo Nielsen	Danish National Archives (DNA)
Kathrine Hougaard Edsen Johansen	Danish National Archives (DNA)
Phillip Tømmerholt	Danish National Archives (DNA)

REVIEWER(S)	
Name(s)	**Organisation(s)**
Kuldar Aas	National Archives of Estonia (NAE)
Bruno Ferreira	KEEPS
Andreas Voss	Swiss Federal 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				
Rev. No.	**Date**	**Author(s)**	**Organisation(s)**	**Description**
0.1	15-05-2015	Anders Bo Nielsen	DNA	First draft
0.5	20-07-2015	Anders Bo Nielsen	DNA	Revision
0.8	15-09-2015	Anders Bo Nielsen	DNA	Revision
0.9	15-10-2015	Anders Bo Nielsen	DNA	Revision
0.10	13-11-2015	Phillip Tømmerholt	DNA	Change of foldername containing dot "." replaced with underscore "_".
0.11	30-11-2015	Anders Bo Nielsen	DNA	Example updated
0.12	08-12-2015	Phillip Tømmerholt	DNA	Example updated
0.13	09-12-2015	Phillip Tømmerholt	DNA	Example updated
0.14	19-01-2016	Phillip Tømmerholt	DNA	Revision
0.16	23-02-2016	Anders Bo Nielsen	DNA	Typos corrected in XML example
0.17	23-04-2016	Anders Bo Nielsen	DNA	Change URL syntax in value of file attribute. Add numbered sections and align styles
0.18	03-05-2016	Anders Bo Nielsen	DNA	Updating example
0.19	18-05-2016	Anders Bo Nielsen	DNA	Naming convention for splitting binary files
0.20	20-05-2016	Anders Bo Nielsen	DNA	Update introduction
0.21	17-06-2016	Anders Bo Nielsen	DNA	Correcting typos
0.22	20-08-2016	Anders Bo Nielsen	DNA	Changing format for headings
0.23	21-09-2016	Anders Bo Nielsen	DNA	Example updated
0.24	05-10-2016	Anders Bo Nielsen	DNA	Terminology change
0.25	17-11-2016	Anders Bo Nielsen	DNA	Example updated
0.26	28-11-2016	Anders Bo Nielsen Phillip Tømmerholt	DNA DNA	RFC 3968 added AnyURI def. added
0.27	12-09-2017	Anders Bo Nielsen	DNA	Typos corrected
0.28	30-08-2018	Anders Bo Nielsen	DNA	Typos corrected

TABLE OF CONTENTS

1. Indholdsfortegnelse

1 Introduction

This document is a recommendation for storing large objects outside the SIARD file, a feature introduced with the SIARD 2.0 Format Specification.

The SIARD 2.0 format has its own specification, which deliberately does not specify details like the exact folder structure for storing large objects outside the SIARD file. This is deliberately avoided in order to get a high degree of durability and longevity, which requires flexibility.

Instead, such details are left to recommendations like this one.

Large object (LOB) is the common description for large character content or large binary content (such as video, sound, images, word processing documents etc.).

These LOBs can be stored inside a relational database as CLOBs or BLOBs or outside (SQL/MED)..

This recommendation will be used by the E-ARK open source tool Database Preservation Toolkit (DBPTK) for the export of relational databases to the SIARD 2.0 format.

2 Methods for binary data handling in relational databases

Binary data in regard to relational databases is defined as data for which no simple datatype (such as integer or date) exists. In addition the size of binary data is also important due to the need for efficient handling in databases. Binary data is mostly referred to as a binary large object (BLOB). Similarly large amounts of character data are named CLOB, they pose a problem due to size more than lack of a proper data type. For the rest of this recommendations CLOBs will be treated as BLOBs, and both are generally named LOBs (large objects).

Databases and handling of binary data has always been a challenge, regardless of whether the handling was based on:

1. internal BLOBs - where BLOBs are contained in the records.
2. external direct references (path and filename) - where BLOBs are stored as files.
3. external indirect reference (file ID)- where BLOBs are stored as files.

3 Binary data handling in SQL Standards

The first method using internal BLOBs has been available for many versions of the SQL standard. It is supported by all current relational database management systems.

The other method using external files has been available since SQL:2003 and as the Management of External Data (SQL/MED) extension to the SQL standard. It is still poorly supported by current relational database management systems, and perhaps due to lack of a detailed specification in the SQL standard those RDBMS that support it do so differently.

4 Binary data handling in the SIARD 2.0 format

The SIARD 2.0 format is based on the SQL:2008 standard and other standards such as XML.

4.1 Support for internal BLOBS (ISO/IEC 9075-2:2008 - BLOBS) in SIARD 2.0

The SIARD 2.0 format specification supports the SQL:2008 method for using internal BLOBS (ISO/IEC 9075-2:2008), as did SIARD 1.0 (SQL:1999).

The SIARD 2.0 format supports BLOBS stored as files inside the SIARD table folder structure and describes this in detail in the SIARD 2.0 format specification (similar to SIARD 1.0).

The SIARD 2.0 format supports BLOBS stored as files outside the SIARD table folder structure (a new feature in SIARD 2.0), but does not describe this in detail, it is left for recommendations like this one to describe their own implementations.

4.2 Support for external files (ISO/IEC 9075-9:2008 – SQL/MED) in SIARD 2.0

The SIARD 2.0 format does not support the SQL:2008 method for using external files (ISO/IEC 9075-9:2008 – SQL/MED) due to a lack of specification in the SQL standard, and especially because of lack of support in many current RDBMS. The SIARD 2.0 format does not forbid the use of methods using external files, but leaves it to recommendations like this one to add such support.
Note that this recommendation does not contain any special elements to facilitate support for SQL/MED.

(Note that no software supporting SIARD 2.0 currently supports SQL/MED.)

5 Recommendation for the folder structure for LOBs stored outside the SIARD file

This recommendation specifies the folder structure for files stored outside the SIARD folder structure, i.e. outside of the SIARD file.

Due to file systems limitations and media limitations it is currently necessary to limit:

- the amount of files in one folder (eg. 100,000 files)
- the size of files in one folder hierarchy (eg. 4 GB)

In order to achieve effective and easy file and folder handling (such as copying, packing, hashing, mounting) new folders are created when reaching a folder amount or folder size limitation, whatever comes first.

These folders are placed at the same top level to enable effective and easy file and folder handling.

5.1 "Generic" example

Below is a "generic" example for a single column *k* in a single table *j*, in which the file amount limit is reached before the file size limit.

[xxx] to be replaced including the [ ]. (very rough BNF)

```
[databaseName].siard <!- packaged as a ZIP file ->
      content/
      header/
            metadata.xml
            metadata.xsd

[databaseName]_lobseg_[h]/..
      content/schema[i]/table[j]/lob[k]/record[1].bin
      ..
      content/schema[i]/table[j]/lob[k]/record[FILEAMOUNTLIMIT].bin
      <!-- file amount limit per folder reached -->
[databaseName]_lobseg_[h+1]/
      content/schema[i]/table[j]/lob[k]/record[FILEAMOUNTLIMIT+1].bin
      ..
      content/schema[i]/table[j]/lob[k]/record[FILEAMOUNTLIMIT+q].bin
      <!-- total file size limit per folder reached -->
[databaseName]_lobseg_[h+2]/
      content/schema[i]/table[j]/lob[k]/record[FILEAMOUNTLIMIT+q+1].bin
```

Note that the base value for records and tables is 0, whereas it is 1 for columns and rows.

5.2 Example with the example database Northwind

Below is an example of the folder structure for the example database Northwind with external LOBs. Northwind includes the following tables, which in our example are ordered this way:

Orders	(table0)
Products	(table1)
Categories	(table2) - has LOBs which exceed 2000 bytes or characters
Shippers	(table3)
Employees	(table4) - has LOBs which exceed 2000 bytes or characters
Territories	(table5)
CustomerDemographics	(table6)
CustomerCostumerDemo	(table7)
Suppliers	(table8)
EmployeeTerritories	(table9)
Customers	(table10)
Sysdiagrams	(table11)
Region	(table12)

Only the table 'Categories' (table2) below is used in this example

CategoryID	CategoryName	Description	Picture
1	Beverages	Soft drinks, coffees, teas, beers, and ales	BLOB (Size: 10151)
2	Condiments	Sweet and savory sauces, relishes, spreads, and seasonings	BLOB (Size: 12107)
3	Confections	Desserts, candies, and sweet breads	BLOB (Size: 12007)
4	Dairy Products	Cheeses	BLOB (Size: 9756)
5	Grains/Cereals	Breads, crackers, pasta, and cereal	BLOB (Size: 12131)
6	Meat/Poultry	Prepared meats	BLOB (Size: 11280)
7	Produce	Dried fruit and bean curd	BLOB (Size: 12338)
8	Seafood	Seaweed and fish	BLOB (Size: 12069)

5.2.1 Limits for file amount and for total file size per folder

The file amount limit per folder is in this case set to 4 and the total file size limit per folder is set to 45,000 bytes (unrealistic but useful as an example).

Rows 1, 2, 3 and 4 will have their LOBs from column 4 ('Picture') (record0.bin, record1.bin, record2.bin, record3.bin) stored together in a folder named Northwind_lobseg_0.
Here the *file amount limit* of 4 is reached, meaning no more files can be stored in this folder, so a new folder is created named Northwind_lobseg_1.

Rows 5, 6, and 7 will have their LOBs from column 4 ('Picture') (record4.bin, record5.bin, record6.bin) stored together in a folder named Northwind_lobseg_1.

Row 8 will *not* have its LOB from column 4 ('Picture') (record7.bin) stored together with the ones from rows 5, 6 and 7 in the folder named Northwind_lobseg_1. Not because the *file amount limit* of 4 is reached, but because the *accumulated file size per folder limit* of 45,000 bytes is reached.

The LOBs from rows 5, 6 and 7 have respectively a size of 12,131, 11,280 and 12,338 bytes which accumulates to 35,749 bytes. Adding the LOB from row 8 with a size of 12,069 bytes to the 35,749 bytes of rows 5, 6 and 7 would accumulate to 47,818 bytes and exceed the *accumulated file size per folder limit* of 45,000 bytes. Therefore a new folder is created named Northwind_lobseg_2, and Row 8 will have its LOB from column 4 ('Picture') (record7.bin) stored in it.

5.2.2 Illustration of the folder structure for the example

```
Northwind.siard <!-- packaged as a ZIP file ->
        content/
        header/
                metadata.xml
                metadata.xsd
Northwind_lobseg_0/
        content/schema0/table2/lob4/record0.bin <!-- row 1, col 4 has record0.bin ->
        content/schema0/table2/lob4/record1.bin
        content/schema0/table2/lob4/record2.bin
        content/schema0/table2/lob4/record3.bin <!-- amount limit reached -->
Northwind_lobseg_1/
        content/schema0/table2/lob4/record4.bin
        content/schema0/table2/lob4/record5.bin
        content/schema0/table2/lob4/record6.bin <!-- size limit reached -->
Northwind_lobseg_2/
        content/schema0/table2/lob4/record7.bin
```

5.2.3 Extract of metadata.xml for the example

```xml
<dbname>Northwind</dbname>
<dataOwner>...</dataOwner>
<dataOriginTimespan>2015</dataOriginTimespan>
<lobFolder>file:///Archives/Northwind/</lobFolder>
```

5.2.4 Extract of table2.xml for the example

```xml
<row><c1>1</c1><c2>Beverages</c2><c3>Soft drinks, coffees, teas, beers, and ales</c3>
<c4 file="Northwind_lobseg_0/content/schema0/table2/lob4/record0.bin" length="10151"
messageDigest="md574f24080fc9d234d3ac221b8e743c763"/></row>

<row><c1>2</c1><c2>Condiments</c2><c3>Sweet and savory sauces, relishes, spreads, and
seasonings</c3>
<c4 file="Northwind_lobseg_0/content/schema0/table2/lob4/record1.bin" length="12107"
messageDigest="md522a0cbe8960b78ce48b07a285ce69e3c"/></row>

<row><c1>3</c1><c2>Confections</c2><c3>Desserts, candies, and sweet breads</c3>
<c4 file="Northwind_lobseg_0/content/schema0/table2/lob4/record2.bin" length="12007"
messageDigest="md53e2f2028a9147c29bdcd36ed4e5f25b3"/></row>

<row><c1>4</c1><c2>Dairy Products</c2><c3>Cheeses</c3>
<c4 file="Northwind_lobseg_0/content/schema0/table2/lob4/record3.bin" length="9756"
messageDigest="md512f588040e11cc2021ea37d46aa10c51"/></row>

<row><c1>5</c1><c2>Grains/Cereals</c2><c3>Breads, crackers, pasta, and cereal</c3>
<c4 file="Northwind_lobseg_1/content/schema0/table2/lob4/record4.bin" length="12131"
messageDigest="md5e2d8ef03e1b24edd946820dbbf44fdfd"/></row>

<row><c1>6</c1><c2>Meat/Poultry</c2><c3>Prepared meats</c3>
<c4 file="Northwind_lobseg_1/content/schema0/table2/lob4/record5.bin" length="11280"
messageDigest="md5814a3eb95253c08137f70bcfc279e00f"/></row>
```

```
<row><c1>7</c1><c2>Produce</c2><c3>Dried fruit and bean curd</c3>
<c4 file="Northwind_lobseg_1/content/schema0/table2/lob4/record6.bin" length="12338"
messageDigest="md5ee114cd7700f566b1f7c7e8e0f68ca0f"/></row>

<row><c1>8</c1><c2>Seafood</c2><c3>Seaweed and fish</c3>
<c4 file="Northwind_lobseg_2/content/schema0/table2/lob4/record7.bin" length="12069"
messageDigest="md52de1ac4c4e8ebb853e17db01af3fb7c3"/></row>
```

5.2.5 Externally referenced files according to RFC 1738 required and RFC 3986 recommended

According to the SIARD 2.0 format specification (G_3.4-1) all externally referenced files are specified according to RFC 1738 “Uniform Resource Locators (URL)”, December 1994 - https://www.ietf.org/rfc/rfc1738.txt. According to this RFC a file url has three forward slashes "/" before the file path, namely two in the protocol name, i.e. "file://", and one separating the protocol name from the file path (the host element is optional).

```
5. BNF for specific URL schemes
; FILE
fileurl        = "file://" [ host | "localhost" ] "/" fpath
fpath          = fsegment *[ "/" fsegment ]
fsegment       = *[ uchar | "?" | ":" | "@" | "&" | "=" ]
uchar          = unreserved | escape
unreserved     = alpha | digit | safe | extra
alpha          = lowalpha | hialpha
digit          = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
safe           = "$" | "-" | "_" | "." | "+"
extra          = "!" | "*" | "'" | "(" | ")" | ","
```

RFC 1738 is sufficient for defining a base URI, a requirement for relative paths in URI, but the use of relative paths in URI is more explicitly specified in RFC 3986 “URI Generic Syntax”, January 2005 - https://tools.ietf.org/html/rfc3986#section-4.3 than in RFC 1738. It is therefore recommended to use RFC 3968 to better understand the use of relative paths in a URI[1].

The following excerpts are from RFC 3698:

```
4.2.  Relative Reference

A relative reference takes advantage of the hierarchical syntax (Section 1.2.3) to
express a URI reference relative to the name space of another hierarchical URI.

      relative-ref  = relative-part [ "?" query ] [ "#" fragment ]
      relative-part = "//" authority path-abempty
                    / path-absolute
                    / path-noscheme
                    / path-empty


A relative reference that begins with two slash characters is termed a network-path
reference; such references are rarely used.
A relative reference that begins with a single slash character is termed an
absolute-path reference.
A relative reference that does not begin with a slash character is termed a
relative-path reference.
...

5.1.  Establishing a Base URI
```

[1]see also the draft: https://tools.ietf.org/html/draft-ietf-appsawg-file-scheme-03#appendix-C.1.1

```
The term "relative" implies that a "base URI" exists against which the relative
reference is applied.  Aside from fragment-only references (Section 4.4), relative
references are only usable when a base URI is known.  A base URI must be established by
the parser prior to parsing URI references that might be relative.  A base URI must
conform to the <absolute-URI> syntax rule (Section 4.3).
...
```

5.4. Reference Resolution Examples

```
Within a representation with a well defined base URI of
      http://a/b/c/d;p?q
a relative reference is transformed to its target URI as follows.

5.4.1.  Normal Examples

      "g:h"           =  "g:h"
      "g"             =  "http://a/b/c/g"
      "./g"           =  "http://a/b/c/g"
      "g/"            =  "http://a/b/c/g/"
...
```

According to the above RFC the base URI and the relative-path reference in the example for row 4 will resolve to this:

Base URI:
`file:///Archives/Northwind/`

Relative-path reference for row 4: `Northwind_lobseg_0/content/schema0/table2/lob4/record0.bin`

Resolves into:
`file:///Archives/Northwind/Northwind_lobseg_0/content/schema0/table2/lob4/record0.bin`

5.4.2 Directions from the XML standard, escaping spaces

The SIARD 2.0 format uses the XML datatype xs:anyURI for representing URIs[2]. According to the W3C Recommendation[3] the datatype xs:anyURI can have values that can be absolute or relative, and may have optional fragment identifiers. The W3C recommendation refers to RFC 2396[4] and RFC 2732[5] for definitions. RFC 2396 and RFC 2732 are the standards that lie between RFC 1738 and the most current RFC 3698, all of which are already presented in this recommendation. The W3C recommendation also includes a note that the lexical space, i.e the set of valid literals for the datatype, in principle allows spaces, however, their use is highly discouraged unless they are encoded as %20. For spaces in URIs the SIARD 2.0 format follows the W3C recommendation, as can be seen on page 65 in the SIARD 2.0 format specification:

```
      <lobFolder>file:///D:/Projekte/SIARD/SIARD%20Suite/</lobFolder>
```

It is recommended not to have space characters in the content of the *lobFolder* elements, as shown in this example:

`file:///Archives/Northwind/Northwind_lobseg_0/content/schema0/table2/lob4/record0.bin`

[2] see *"Appendix D – XML Schema Definitions" in SIARD 2.0 Format Specification.*

[3] W3C recommendation. XML Schema Part 2: Datatypes Second Edition. 28 October 2004. Available at: http://www.w3.org/TR/xmlschema-2/

[4] Tim Berners-Lee, et. al. *RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax.*. 1998. Available at: http://www.ietf.org/rfc/rfc2396.txt

[5] R. Hinden. et al. *RFC 2732: Format for Literal IPv6 Addresses in URL's*. 1999. Available at: http://www.ietf.org/rfc/rfc2732.txt

5.2.5.1.1 Algorithm prefix to hash value required

Note that according to the SIARD 2.0 format specification the messageDigest value is prefixed with the algorithm followed by the hash value.

5.2.5.1.2 Lower case for messageDigest recommended

Note that the messageDigest indicates hexadecimal values and it is therefore not of strict importance whether they are set in upper or lower case. However, lower case is mostly used and enforced, see e.g. RFC 2831 https://www.ietf.org/rfc/rfc2831.txt

5.3 Splitting large files as binary chunks

It may happen that a file itself is larger than the limit size for a folder (for example, it could be a movie). In that case the file itself will have to be binary divided into smaller pieces using the naming convention with the filename and an incremental suffix ".[p]" ending with ".z".

In the rough example below record n+14 is larger than the limit size for a folder so is split into 5 smaller files, with segment 0 set to a size filling up to the total file size limit.

After the last segment z of the file is stored in folder [**databaseName**]**_lobseg_**[**h+4**]/ additional LOBs are stored in that folder.

(Smaller size optimization is also acceptable, such as storing the 5 segments alone in 5 lobseg_ folders, i.e. not filling up to the actual file size limit.)

```
[databaseName]_lobseg_[h]/
      content/schema[i]/table[j]/lob[k]/record[n].bin
      content/schema[i]/table[j]/lob[k]/record[n+1].bin
      ..
      content/schema[i]/table[j]/lob[k]/record[n+13].bin
      content/schema[i]/table[j]/lob[k]/record[n+14].bin.0
      <!-- large file splitted first seg -->
      <!-- total file size limit per folder reached -->
[databaseName]_lobseg_[h+1]/
      content/schema[i]/table[j]/lob[k]/record[n+14].bin.1
      <!-- large file splitted seg 1-->
      <!-- total file size limit per folder reached -->
[databaseName]_lobseg_[h+2]/
      content/schema[i]/table[j]/lob[k]/record[n+14].bin.2
      <!-- large file splitted seg 2-->
      <!-- total file size limit per folder reached -->
[databaseName]_lobseg_[h+3]/
      content/schema[i]/table[j]/lob[k]/record[n+14].bin.3
      <!-- large file splitted seg 3-->
      <!-- total file size limit per folder reached -->
[databaseName]_lobseg_[h+4]/
      content/schema[i]/table[j]/lob[k]/record[n+14].bin.z
      <!-- large file splitted last seg -->
      content/schema[i]/table[j]/lob[k]/record[n+15].bin
      content/schema[i]/table[j]/lob[k]/record[n+16].bin
```