Guideline for CITS SIARD

Guideline for the E-ARK Content Information Type Specification for Relational Databases using SIARD

Date: 31.08.2021
Version: 1.0.0

Guideline for CITS SIARD

Guideline for the E-ARK Content Information Type Specification for Relational Databases using SIARD

The European Commission eArchiving procurement recognizes the E-ARK specifications as the eArchiving specifications which are funded under the eArchiving Common Services Platform Agreement No. LC-01905904-CNECT/LUX/2021/OP/0077.

This specification is published, supported, and developed by the Digital Information LifeCycle Interoperability Standards (DILCIS) Board under the auspices of the DLM Forum.

This specification was previously developed with the support of the European Union:

E-ARK Grant No: 620998 CIP-ICT-PSP.2013.2.5
E-ARK4ALL Agreement No. LC-00921441 CEF-TC-2018-15
E-ARK3 Agreement No. LC-01390244 CEF-TC-2019-3

1 Preface

1.1 Aim of the specification

This document is one of several related specifications which aim to provide a common set of usage descriptions of international standards for packaging digital information for archiving purposes. These specifications are based on common, international standards for transmitting, describing and preserving digital data. They also utilise the Reference Model for an Open Archival Information System (OAIS), which has Information Packages as its foundation. Familiarity with the core functional entities of OAIS is a prerequisite for understanding the specifications.

The specifications are designed to help data creators, software developers, and digital archives to tackle the challenge of short-, medium- and long-term data management and reuse in a sustainable, authentic, cost-efficient, manageable and interoperable way. A visualisation of the current specification network can be seen here:

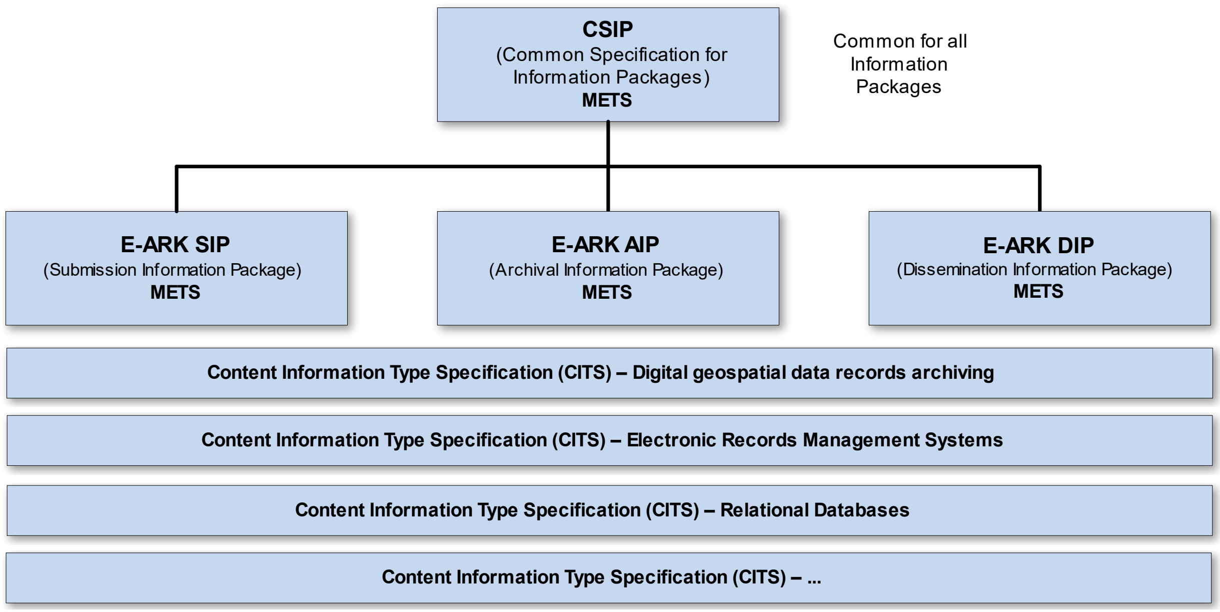

CSIP
(Common Specification for Information Packages)
METS
Common for all Information Packages
E-ARK SIP
(Submission Information Package)
METS
E-ARK AIP
(Archival Information Package)
METS
E-ARK DIP
(Dissemination Information Package)
METS
Content Information Type Specification (CITS) – Digital geospatial data records archiving
Content Information Type Specification (CITS) – Electronic Records Management Systems
Content Information Type Specification (CITS) – Relational Databases
Content Information Type Specification (CITS) – ...

Figure I: Diagram showing E-ARK specification dependency hierarchy. Note that the image only shows a selection of the published CITS and isn't an exhaustive list.

Specification	Aim and Goals
Common Specification for Information Packages	This document introduces the concept of a Common Specification for Information Packages (CSIP). Its three main purposes are to: • Establish a common understanding of the requirements, which need to be met in order to achieve interoperability of Information Packages. • Establish a common base for the development of more specific Information Package definitions and tools within the digital preservation community. • Propose the details of an XML-based implementation of the requirements using, to the largest possible extent, standards which are widely used in international digital preservation.

Specification	Aim and Goals
	Ultimately, the goal of the Common Specification is to reach a level of interoperability between all Information Packages so that tools implementing the Common Specification can be adopted by institutions without the need for further modifications or adaptations.
E-ARK SIP	The main aims of this specification are to: ● Define a general structure for a Submission Information Package format suitable for a wide variety of archival scenarios, e.g. document and image collections, databases or geographical data. ● Enhance interoperability between Producers and Archives. ● Recommend best practices regarding metadata, content and structure of Submission Information Packages.
E-ARK AIP	The main aims of this specification are to: ● Define a generic structure of the AIP format suitable for a wide variety of data types, such as document and image collections, archival records, databases or geographical data. ● Recommend a set of metadata related to the structural and the preservation aspects of the AIP as implemented by the eArchiving Reference Implementation (earkweb). ● Ensure the format is suitable to store large quantities of data.
E-ARK DIP	The main aims of this specification are to: ● Define a generic structure of the DIP format suitable for a wide variety of archival records, such as document and image collections, databases or geographical data. ● Recommend a set of metadata related to the structural and access aspects of the DIP.
Content Information Type Specifications	The main aim and goal of a Content Information Type Specification is to: ● Define, in technical terms, how data and metadata must be formatted and placed within a CSIP Information Package in order to achieve interoperability in exchanging specific Content Information. The number of possible Content Information Type Specifications is unlimited. For a list of existing Content Information Type Specifications see the DILCIS Board webpage (DILCIS Board, http://dilcis.eu/).

1.2 Organisational support

This specification is maintained by the Digital Information LifeCycle Interoperability Standards Board (DILCIS Board, http://dilcis.eu/). The role of the DILCIS Board is to enhance and maintain the draft specifications developed in the European Archival Records and Knowledge Preservation Project (E-ARK project, http://eark-project.com/), which concluded in January 2017. The Board consists of eight members, but no restriction is placed on the number of participants taking part in the work. All Board documents and specifications are stored in GitHub (https://github.com/DILCISBoard/), while published versions are made available on the Board webpage. The DILCIS Board have been responsible for providing the core specifications to the Connecting Europe Facility eArchiving Building Block https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eArchiving/.

1.3 Authors & Revision History

A full list of contributors to this specification, as well as the revision history, can be found in the Postface material..

TABLE OF CONTENT

Table of Figures

1 Context

1.1 Purpose

The purpose of this guideline is to further explain and describe the “Content Information Type Specification for Relational Databases using SIARD” (also called CITS SIARD in short). The goal is that as many people as possible will be able to understand the specification and therefore also to preserve relational databases. The guideline is an evolving document, and more concepts and standards will be explained following the needs of the users of the specification.

1.2 Scope

This guideline will provide further information and insights as to how to preserve relational databases using SIARD and using the specification landscape of CSIP, SIP, AIP, DIP, and SIARD.

This guideline is a guideline for CITS SIARD version 1.0.

1.3 Structure of the document

Section 2 contains an introductory section describing the concept of relational databases in general and their digital preservation. It also includes a recommended reading list for further interest in the topic. This section is meant for colleagues who are new to the field.

Section 3 contains an introductory section describing the SIARD specification and its history. It is important to understand the differences between SIARD specification and the CITS SIARD specification, which is why this is also elaborated in this section.

Section 4 provides a rationale for each of the requirements found in the CITS SIARD specification. This is meant to provide a better basis for understanding the reasons behind the requirements. This section is primarily meant for technicians and developers of the specification, and it is a prerequisite that the reader has knowledge about the SIARD specification and the Common Specification for Information Package and the SIP, AIP and DIP specifications.

Section 5 contains a description of the not cumbersome task of segmenting information packages in order to obtain scalability in the specifications and being able to easier to preserve large databases.

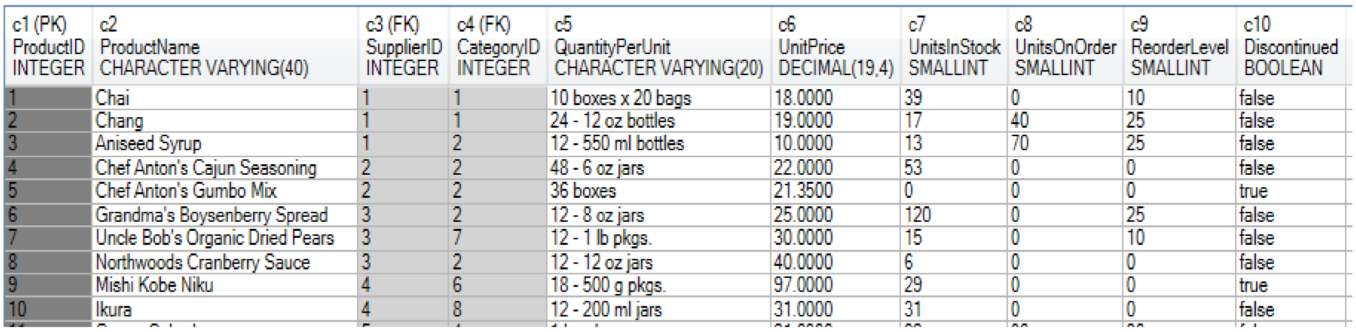
Section 6 contains an overview of available example packages, tools, and interest groups related to the CITS SIARD and SIARD specifications as a means to get your hands dirty and take action in developing the field of database preservation.

2 Relational databases - a short introduction

This section contains an introductory section describing the concept of relational databases in general and their digital preservation. It also includes a recommended reading list for further interest in the topic. This section is meant for colleagues who are new to the field.

2.1 A way to organise data

A database is in short an organised collection of data that is stored and accessed digitally.
There are different database models and also many different database management systems. One of the most widespread database models is the relational database model, which typically sorganises data into one or more tables of vertical columns (or attributes) and horizontal rows (or records or tuples), with a unique key to identify each row in the table. To illustrate this see the visualisation in Figure 1 of a table called “Products”. The table is taken from the example database “Northwind” and visualised via “ADA” which is an ingest validation application for the SIARDDK format. Throughout this guideline the example database “Northwind” is used in order to illustrate the handling of relational databases.[1]



c1 (PK)	c2	c3 (FK)	c4 (FK)	c5	c6	c7	c8	c9	c10
ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	UnitsInStock	UnitsOnOrder	ReorderLevel	Discontinued
INTEGER	CHARACTER VARYING(40)	INTEGER	INTEGER	CHARACTER VARYING(20)	DECIMAL(19,4)	SMALLINT	SMALLINT	SMALLINT	BOOLEAN
1	Chai	1	1	10 boxes x 20 bags	18.0000	39	0	10	false
2	Chang	1	1	24 - 12 oz bottles	19.0000	17	40	25	false
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10.0000	13	70	25	false
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22.0000	53	0	0	false
5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.3500	0	0	0	true
6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25.0000	120	0	25	false
7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	30.0000	15	0	10	false
8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	40.0000	6	0	0	false
9	Mishi Kobe Niku	4	6	18 - 500 g pkgs.	97.0000	29	0	0	true
10	Ikura	4	8	12 - 200 ml jars	31.0000	31	0	0	false

Figure 1: Visualisation of a database table

In Figure 1, the table “Products” consists of ten columns (from c1 to c10) and 77 rows, where only ten rows are visible.

Primary keys
“c1, ProductID” is a column with a primary key, which in this svisualisation is marked with a dark grey colour and the text: “(PK)” after c1. Primary keys should be unique and can therefore be used for identification of data. The row with the primary key which has the value 1 holds information about:

- product name (c2), Chai
- which supplier who supplies this product (c3), supplierID=”1”
- which category the product lies within (c4), categoryID=”1”
- how many quantities per unit there is (c4), QuantityPerUnit=”10 boxes x 20 bags”
- what the sales price is (c5), UnitPrice= ”18.0000”
- and so on.

[1] Northwind is an example database from Microsoft which is available at: https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/sql/linq/downloading-sample-databases.

Foreign keys

The columns c3 and c4 are marked in Figure 1 with a light grey colour which illustrates that these columns have been marked as having foreign keys. A foreign key is a key that points to another column. c3 points to the table "Suppliers" and the c4 points to the table "Categories" even though this cannot be seen in Figure 1. In Figure 2, all relations to and from the table Products can be seen. The table names are at the top of the grey entities, and the column names are listed under the table names, and therefore each grey entity depicts a table. Columns with keys have been marked yellow, and there are arrows between the tables showing in what direction the foreign keys go.

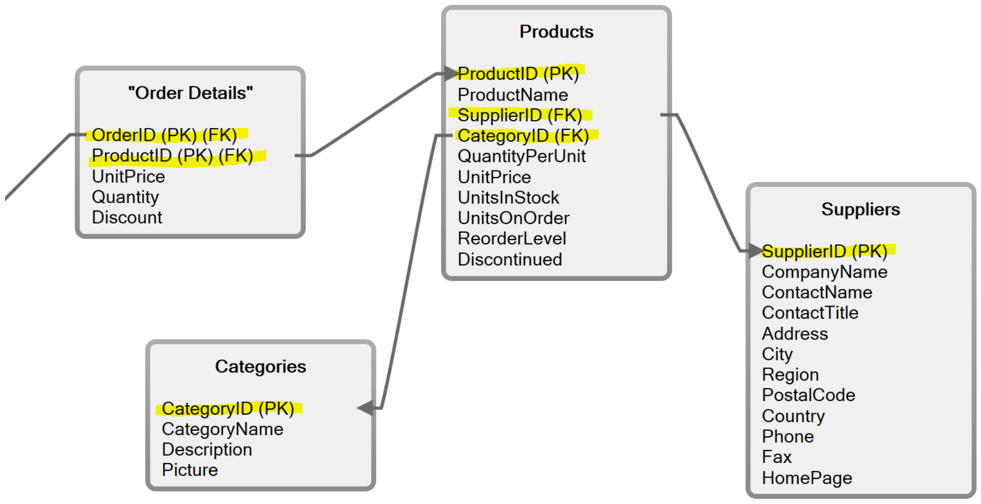
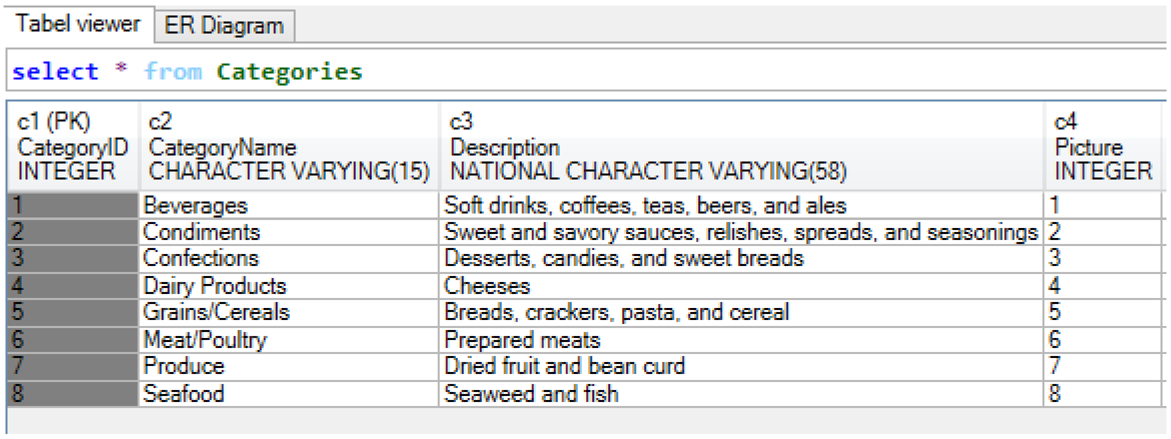


Figure 2: Visualisation of tables and their relations

In short, this means that if we find the ProductID with value "1" in the table "Products" (Figure 1), then we can find SupplierID= "1" and CategoryID="1" in the tables Suppliers and Categories depicted in Figure 2. If we, for example, go to the table "Categories", see Figure 3, then we can find the row where CategoryID="1" and in this row, we can find:

- CategoryName="Beverages"
- Description "Soft drinks, coffees, teas, beers, and ales".
- And a picture of the category, which in this case is an ID as a reference to a picture that lies outside the database.



```
select * from Categories
```

c1 (PK) CategoryID INTEGER	c2 CategoryName CHARACTER VARYING(15)	c3 Description NATIONAL CHARACTER VARYING(58)	c4 Picture INTEGER
1	Beverages	Soft drinks, coffees, teas, beers, and ales	1
2	Condiments	Sweet and savory sauces, relishes, spreads, and seasonings	2
3	Confections	Desserts, candies, and sweet breads	3
4	Dairy Products	Cheeses	4
5	Grains/Cereals	Breads, crackers, pasta, and cereal	5
6	Meat/Poultry	Prepared meats	6
7	Produce	Dried fruit and bean curd	7
8	Seafood	Seaweed and fish	8

Figure 3: Visualisation of tables and their relations

This way of organising data is a way of representing “real-world” entities, and since the logic is clear and builds on relational algebra, it might be one of the reasons why the relational database model and relational database management systems are so widespread and popular.

2.2 SQL Standardisation and proprietary software

There are many different relational database management systems, who have their own variants and flavours of, for example, data types and functionalities. Oracle, MS SQL Server, PostgreSQL, MySQL, MariaDB, DB2, Firebird, and SQLite, to name a few well-known. In many of the names, the abbreviation “SQL” appears. SQL is an abbreviation for Structured Query Language and is one of the first commercial languages created at IBM after learning about the relational model in the original papers from Edgar F. Codd in the 1970s[2]. Afterwards, standardisation work has taken place, and most relational database management systems are conformant in different degrees to the SQL standards. There are different versions of the SQL standard SQL-86, SQL-89, SQL-92, SQL:1999, SQL:2003, SQL:2006, SQL:2008, SQL:2011, SQL:2016.

2.3 Recommended reading list

This section provides a recommended reading list for those interested in the preservation of relational databases. It is a wish that more examples will follow. If you have a good example, please let us know via the GitHub portal https://github.com/DILCISBoard/CITS-SIARD.

The relational database model and SQL:

- W3Schools (2020). Introduction to SQL. https://www.w3schools.com/sql/sql_intro.asp
- *Codd, Edgar F. (1970). "A Relational Model of Data for Large Shared Data Banks". Communications of the ACM. 13 (6): 377–87. See e.g. https://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf*
- *Codd E. F. (1971). Further Normalization of the Relational Model. Courant Computer Science Symposium 6 in Data Base Systems edited by Rustin R*
- Chamberlin, D; Boyce, F (1974). "SEQUEL: A Structured English Query Language" (PDF). *Proceedings of the 1974 ACM SIGFIDET Workshop on Data Description, Access and Control*. Association for Computing Machinery: 249–64.

Preservation of relational databases:

- Jacinto et al. (2002). Bidirectional Conversion between XML Documents and Relational Databases. In INTERNATIONAL CONFERENCE ON CSCW IN DESIGN, 7, Rio de Janeiro.
- Ramalbo et al. (2007). Relational database preservation through XML modelling
- Freitas & Ramalbo (2010). Significant properties in relational databases International Journal of Engineering and Industrial Management 3
- Library of Congress (2015). SIARD (Software Independent Archiving of Relational Databases) Version 1.0. Retrieved the 4th of February 2021 at https://www.loc.gov/preservation/digital/formats/fdd/fdd000426.shtml

Miscellaneous:

- DILCIS. Guidelines around the specifications: https://dilcis.eu/guidelines

2

3 SIARD – a short introduction

This section contains an introductory text describing the SIARD specification and its history. It is important to understand the differences between SIARD specification and the CITS SIARD specification, which is why this is elaborated in this section.

A short historic overview of SIARD - the file format[3]

SIARD (Software Independent Archival of Relational Databases) is a normative description of an open file format for the long-term archiving of relational databases. SIARD is a nonproprietary, published open standard. The SIARD format is based on open standards, including the ISO standards Unicode, XML, and SQL, the URI Internet standard, and the industry-standard ZIP. The aim of employing internationally recognised standards is to ensure the long-term preservation of, and access to, the widely used relational database model, as well as the easy exchange of database content, independent of proprietary "dump" formats.

SIARD was developed as part of the Swiss Federal Archives (SFA) ARELDA project (ARchiving of ELectronic DAta) (2000-2004) and based on the archiving strategy of the ARELDA project of 11 April 2006. The SIARD 1.0 format was developed in 2008 by the Swiss Federal Archives, and in May 2008, SIARD 1.0 was accepted as the official format for archiving relational databases of the European Open PLANETS project in which the SFA participated.

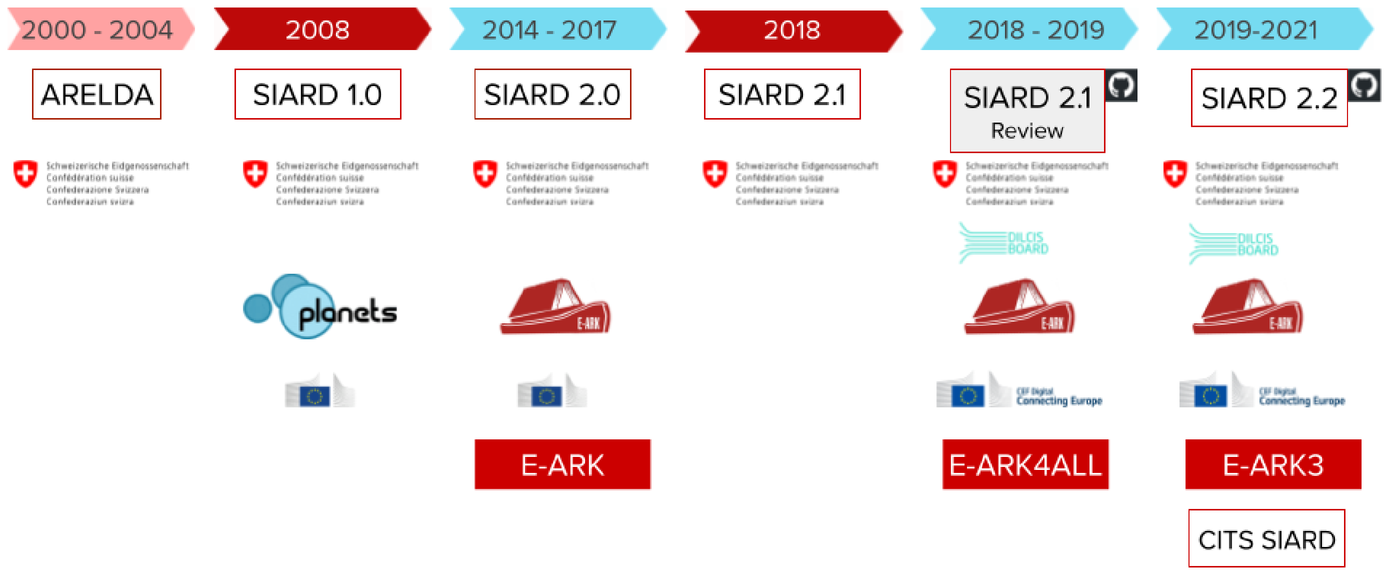


Figure 4: The history of SIARD

The SIARD 2.0 format was developed in 2015 by the Swiss Federal Archives (SFA) and the first E-ARK project (2014-2017).

The SIARD 2.1 format was developed in 2018 by the SFA after the end of the E-ARK project.

SIARD 1.0 and 2.0 are also official Swiss E-Government Standards and version 1.0 can be found here and version 2.0 here. (version 2.0 is currently not available at ech.ch).

SIARD 2.1 is not an official Swiss E-Government Standard but can be found here at the SFA website.

In the E-ARK4ALL project (2018–2019), a review was conducted for the SIARD 2.1 specification.

[3] This section is a copy of the text that can be found at https://github.com/DILCISBoard/SIARD/blob/master/README.md

The development and release of new versions will be coordinated in the DILCIS board (associated with the DLM Forum, created by the EC in 1994) following procedures proposed by the SFA.

The SFA is represented in the DILCIS board (as well as in the DLM Forum) together with other national archives using SIARD.

How to package SIARD: CITS SIARD

It is easy to mix up the SIARD and CITS SIARD specifications. It is vital to understand that SIARD is an independent format for archiving relational databases and hence has its own specification (find the SIARD specifications here: https://github.com/DILCISBoard/SIARD).

The SIARD specification deliberately states that packaging of the SIARD-file among other aspects is outside the scope of the SIARD specification:

> *"It should be noted that the SIARD format is only the long-term storage format for a specific type of digital documents (relational databases) and is therefore designed entirely independently of package structures such as the SIP (Submission Information Package), AIP (Archival Information Package) and DIP (Dissemination Information Package) in the OAIS model.*
> *It is assumed that a database in SIARD format is archived as part of such an information package together with other documents (externalised large object files, translation maps for external file names, database documentation, business documents relevant to the understanding of the database, etc.)."*

On the same page in the SIARD specification, this is svisualised by a diagram that is copied in Figure 5:

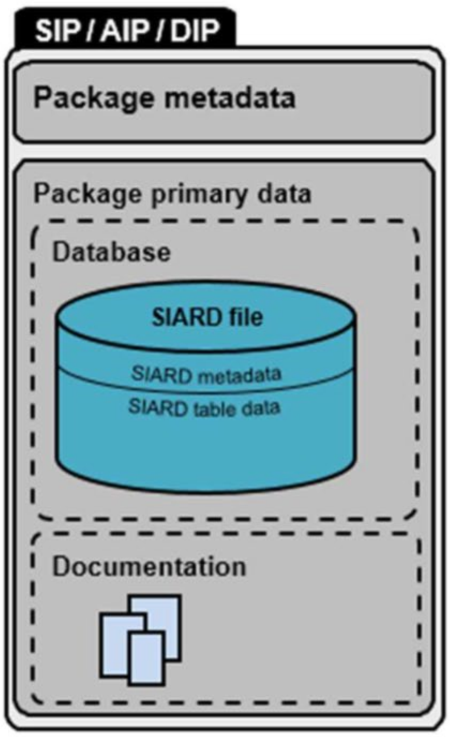


Figure 5: A copy of Diagram 1 in the SIARD specification

As seen from the quote above and Figure 5 a SIARD file can be packaged into many different kinds of packages, which is up to the different archives to decide. However, in the eArchiving Building Block and E-ARK projects there has been done a lot of work at creating Common Specifications for Information Packages which sets an international standard for interoperability of information packages. The CITS SIARD specification is a specification that describes how to package SIARD in a package that is compliant with the requirements stated in the Common Specification for Information Packages (can be found at https://earkcsip.dilcis.eu/) together with the other specifications under CSIP such as the SIP, DIP and AIP specifications.

In the CITS SIARD specification there is the following quote:

> *As in all classification issues, it is important to have collectively exhaustive and mutually exclusive categories, and even though the SIARD specification deliberately states that package structures are not part of the specification, then there are circumstances and scenarios where it is not clear whether an issue falls under the scope of a specification like this one or under the scope of the SIARD specification itself.*

SIARD2.2

The SIARD 2.2 specification has its focus on supporting files outside the database according to part 9 of SQL:2008 (ISO/IEC 9075-9:2008 – SQL/MED) and well as scalability supporting large databases and large objects stored outside the SIARD file.

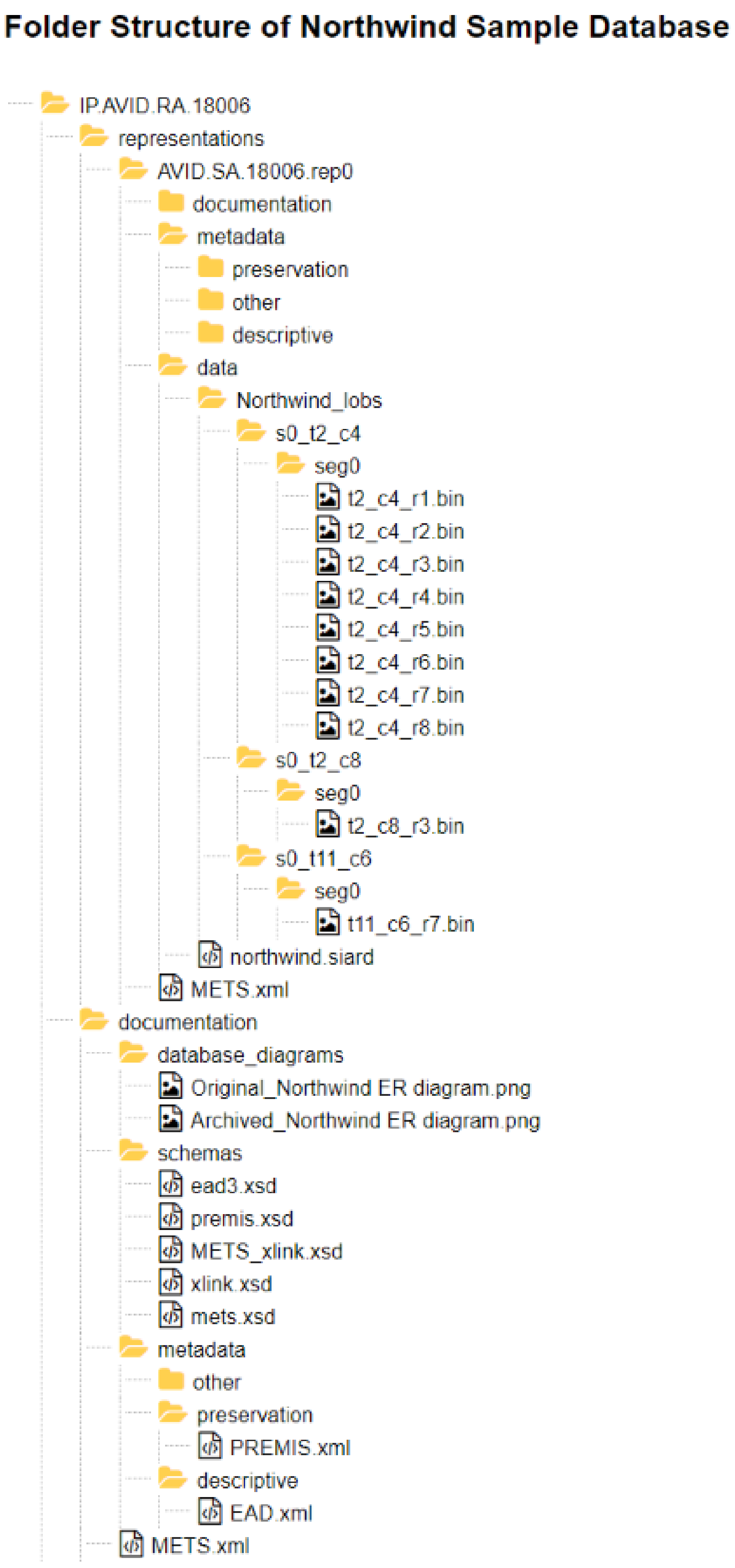
This scalability support is primarily proposed in order to ease, improve and thereby increase the use of SIARD itself.

Secondarily, it is also proposed to ease the archiving of a database in SIARD format as part of an information package.

A large database with GB size tables and TB of large objects can be a challenge for the creation, validation (and general handling) of a database in the SIARD file format. Therefore, segmentation (and brute binary division as a last resort) is proposed as an option for handling scalability issues with large databases, even though not all may need them for handling databases of that size.

Likewise, the creation and later validation and ingest of a SIP at the size of TBs packaging a large database with GB size tables and TB of large objects in SIARD format can be a challenge for some archival information systems.

An example of a complete IP package for CITS SIARD is shown below. Remember this diagram will help the reader understand some of the constructs described in this document.



Folder Structure of Northwind Sample Database

```
IP.AVID.RA.18006
    representations
        AVID.SA.18006.rep0
            documentation
            metadata
                preservation
                other
                descriptive
            data
                Northwind_lobs
                    s0_t2_c4
                        seg0
                            t2_c4_r1.bin
                            t2_c4_r2.bin
                            t2_c4_r3.bin
                            t2_c4_r4.bin
                            t2_c4_r5.bin
                            t2_c4_r6.bin
                            t2_c4_r7.bin
                            t2_c4_r8.bin
                    s0_t2_c8
                        seg0
                            t2_c8_r3.bin
                    s0_t11_c6
                        seg0
                            t11_c6_r7.bin
                northwind.siard
            METS.xml
    documentation
        database_diagrams
            Original_Northwind ER diagram.png
            Archived_Northwind ER diagram.png
        schemas
            ead3.xsd
            premis.xsd
            METS_xlink.xsd
            xlink.xsd
            mets.xsd
        metadata
            other
            preservation
                PREMIS.xml
            descriptive
                EAD.xml
        METS.xml
```

Figure 6: The Northwind sample database

4 Rationale for requirements in CITS SIARD

This section is primarily meant for technicians and developers of the specification, and it is a prerequisite that the reader has knowledge about the SIARD specification and the Common Specification for Information Package and the SIP, AIP and DIP specifications.

In this section, all the requirements in the CITS SIARD are repeated, and a rationale and/or description is given for why the specific requirement stands. This is meant to provide a better basis for understanding the reasons behind the requirements and possibly help to validate any information package that strives to be CITS_SIARD compliant. The requirements are isolated in boxes like this:

Requirement:

SIARD_1	There **MUST** be a minimum of one representation and therefore a minimum of one Package METS.xml and a minimum of one Representation METS.xml in a CITS SIARD package.	1..1 MUST

After the box with the requirement, a text is provided which describes the rationale. The boxes with the requirements are located within the same numbering of the sections in the CITS SIARD specification. Also, text from the CITS SIARD specification is repeated here as a means to understand the requirements.

Rationales in 3.2 Package and Representation METS

CITS SIARD text:

"A CSIP can consist of zero to many representations, and this is an important feature that needs to be taken into consideration when packing SIARD files within CSIPs.

There can easily be different representations of the same database located within one CSIP. For example, one package could consist of:

- one representation where the native proprietary dump is located;
- one representation with SIARD-file that conforms only to an older version of the SIARD specification;
- one representation with the newest version of the SIARD specification;
- one representation where database normalisation and/or other dissemination tasks have taken place.

- There can be several DIP representations. There can also be other databases and, for example, geodata within the same package.

As for this specification, there always needs to be a minimum of one representation and, therefore, a minimum of two METS.xml. The Package METS.xml has to be a general METS.xml describing if the package itself is mainly a CITS_SIARD package, and then the single representations need to describe what specific SIARD versions they consist of."

SIARD_1 Rationale

Requirement:

SIARD_1	There **MUST** be a minimum of one representation and therefore exactly one Package METS.xml and a minimum of one Representation METS.xml in a CITS SIARD package.	1..n MUST

Rationale/description:
If there is not a database in a minimum of one data folder in a representation, then it does not make sense to call it a valid CITS SIARD package.

This first requirement is central for the CITS SIARD specification since it operates with two central terms: the Package METS.xml and the Representation METS.xml.

The "Package METS.xml" (there is only one) needs to be in the root of the package, and one "Representation METS.xml" need to exist in the root of each representation within the package. See the example with two representations (and therefore one "Package METS.xml" and two "Representation METS.xml"s) here https://citssiard.dilcis.eu/examples/IP_18007_SIARD2_2Rep_externallobs

In the CSIP, it is up to the user to define whether all files are described in the "Package METS.xml" or whether the user wishes to split it up and let "Representation METS.xml" describe the content within the representations. In the CITS SIARD specification, there need to be "Representation METS.xml".

Rationales in 3.3 Package METS requirements

SIARD_2 Rationale

Requirement:

SIARD_2 Ref CSIP2	Type mets/@TYPE	For information packages that primarily contain relational databases, the value in Package mets/@TYPE MUST be “Databases” as taken from the CSIP Vocabulary for Content Category.	1..1 MUST

Rationale/description:
This requirement is to make sure that the IP lives up to the requirement CSIP2 in CSIP which states that there MUST be a TYPE-attribute with a value taken from the provided vocabulary.

“Databases” is the most relevant value from the Content Category vocabulary found in CSIP.

SIARD_3 Rationale

Requirement:

SIARD_3 Ref CSIP4	Content Information Type Specification	For information packages that primarily contain relational databases, the value in Package mets/@csip:CONTENTINFORMATIONTYPE MUST be “CITS_SIARD” as taken from the CSIP Vocabulary for Content Information Type.	1..1 MUST

	mets/@csip:CONTENTINFORMATIONTYPE		

Rationale/description:
This requirement is to make sure that the IP lives up to the requirement CSIP4 in CSIP, which is a central way of handling which kind of content information type the package contains. In the case of multiple Content Information Types, then the value "MIXED" should be used.

When the "CITS_SIARD" value is used, this means that the package can be identified as stated to live up to this specification and therefore be validated accordingly.

SIARD_4 Rationale

Requirement:

SIARD_4 Ref CSIP5	Other Content Information Type Specification mets/@csip:OTHERCONTENTINFORMATIONTYPE	For information packages that primarily contain relational databases, the Package METS MUST NOT have a mets/@csip:OTHERCONTENTINFORMATIONTYPE	0..0 MUST NOT

Rationale/description:

The csip:OTHERCONTENTINFORMATIONTYPE-attribute is meant to specify which content information type is used if the csip:CONTENTINFORMATIONTYPE-attribute has the value "OTHER". It is not meant to exist if there are multiple Content Information Types. In the case of multiple Content Information Types then the value "MIXED" should be used. Therefore, for CITS_SIARD packages, there must not be a csip:OTHERCONTENTINFORMATIONTYPE-attribute in "Package METS.xml". Note that this is different from the "Representation METS.xml

SIARD_5 Rationale

Requirement:

SIARD_5 Ref CSIP6, SIP2	METS Profile mets/@PROFILE	For information packages that primarily contain relational databases the value in the @PROFILE MUST be "**https://citssiard.dilcis.eu/profile/E-ARK-SIARD-ROOT.xml**"	1..1 MUST

Rationale/description:
This requirement is to make sure that the IP lives up to the requirement CSIP6 in CSIP.

Since there are different requirements specific for the CITS SIARD, there have been a METS profile created for validation purposes.

SIARD_6 Rationale

Requirement:

SIARD_6 Ref CSIP62	fileSec Representation Content Information Type Specification mets/fileSec/fileGrp[@USE='Representations']/@csip:CONTENTINFORMATIONTYPE	There MUST be a minimum of one mets/fileSec/fileGrp[@USE='Representations']/@csip:CONTENTINFORMATIONTYPE with the value “citssiard_v1_0” as taken from the CSIP Vocabulary for Content Information Type that direct to the representation METS.xml in the representation containing a relational database.	1..n MUST

Rationale/description:
It is via the value "Representations" in the fileGroup USE-attribute on the filegroup element that one can mark up that within this filegroup will be a fileSec with a path to one or more METS-files in one or more representations. One METS-file per representation.

SIARD_7 Rationale

Requirement:

SIARD_7 Ref CSIP63	fileSec Other Content Information Type Specification mets/fileSec/fileGrp[@csip:CONTENTINFORMATIONTYPE=' citssiard_v1_0']/@csip:OTHERCONTENTINFORMATIONTYPE	For any mets/fileSec/fileGrp[@csip:CONTENTINFORMATIONTYPE that has the value “citssiard_v1_0”, there **MUST** be a @csip:OTHERCONTENTINFORMATIONTYPE attribute with a value taken from the vocabulary {SIARD_1.0; SIARD_2.0, SIARD_2.1, SIARD_2.2, Database_dump}.	1..1 MUST

Rationale/description:
This requirement is to make sure that the IP lives up to the requirement CSIP63 in CSIP.

SIARD_8 Rationale

Requirement:

SIARD_8 Ref CSIP105-CSIP112	StructMap METS pointer	For any fileGrp/@csip:CONTENTINFORMATIONTYPE with the value “citssiard_v1_0” there MUST be a corresponding @div-representation in the StructMap-element	1..1 MUST

Rationale/description:
This requirement is to make sure that the IP lives up to the requirement CSIP105 to CSIP112 in CSIP. They are all related to “how to create a StructMap-element”.

Rationales in 3.4 Representation METS requirements

Many of the requirements in this section are the same as in section 3.3 – however, it is important to notice the differences.

SIARD_9 Rationale

Requirement:

SIARD_9 Ref CSIP2	Type mets/@TYPE	For representations that primarily contain relational databases, the value in Package mets/@TYPE MUST be “Databases” as taken from the CSIP Vocabulary for Content Category.	1..1 MUST

Rationale/description:
The same as SIARD_2. This requirement is to make sure that the IP lives up to the requirement CSIP2 in CSIP. “Databases” is the most relevant value from the Content Category vocabulary found in CSIP.

SIARD_10 Rationale

Requirement:

SIARD_10 Ref CSIP4	Content Information Type Specification mets/@csip:CONTENTINFORMATIONTYPE	For representations that primarily contain relational databases and that conforms to CITS SIARD, the value in Package mets/@csip:CONTENTINFORMATIONTYPE MUST be “citssiard_v1_0” as taken from the CSIP Vocabulary for Content Information Type.	1..1 MUST

Rationale/description:
The same as SIARD_3. This requirement is to make sure that the IP lives up to the requirement CSIP4 in CSIP, which is a central way of handling which kind of content information type the package contains; in this case, it is the representation.

When the “CITS_SIARD” value is used, this means that the representation can be identified as stated to live up to this specification, and therefore be validated.

SIARD_11 Rationale

Requirement:

SIARD_11 Ref CSIP5	Other Content Information Type Specification mets/@csip:OTHERCONTENTINFORMATIONTYPE	For representations where mets/@csip:CONTENTINFORMATIONTYPE has the value “citssiard_v1_0” then mets/@csip:OTHERCONTENTINFORMATIONTYPE MUST have a value taken from the vocabulary {SIARD_1.0; SIARD_2.0, SIARD_2.1, SIARD_2.2, Database_dump}	1..1 MUST

Rationale/description:

This requirement uses the CSIP5-requirement as a way for the “Representation METS.xml” to state which SIARD-version the representation contains. Note that it is different from the “Package METS.xml” where the csip:OTHERCONTENTINFORMATIONTYPE is not allowed. At the representation level, you MUST state the SIARD-version.

It is a wish that the vocabulary will exist as an html-file just as the vocabulary given in the Common Specification for Information Packages. This way, the vocabulary can be expanded as new versions will come.

For the database_dump value, see section 3.7.

SIARD_12 Rationale

Requirement:

SIARD_12 Ref CSIP6, SIP2	METS Profile mets/@PROFILE	For information packages that primarily contain relational databases the value in the @PROFILE MUST be “**https://citssiard.dilcis.eu/profile/E-ARK-SIARD-REPRESENTATION.xml**”	1..1 MUST

Rationale/description:
This requirement is to make sure that the IP lives up to the requirement CSIP6 in CSIP.

Since there are different requirements specific for the CITS SIARD, there also have been a METS profile created for validation purposes.

SIARD_13 Rationale

Requirement:

SIARD_13 Ref CSIP64-CSIP79	File Pointer fileSec/fileGrp/file@csip:OTHERCONTENTINFORMATIONTYPE	If the value in mets/@csip:OTHERCONTENTINFORMATIONTYPE is {SIARD_1.0, SIARD_2.0, SIARD_2.1, SIARD_2.2, Database_dump} then there MUST exist one and only one file in the fileGrp with @USE = “data” with an identical value in fileSec/fileGrp/file@csip:OTHERCONTENTINFORMATIONTYPE that is used to locate the relevant database file.	1..1 MUST

Rationale/description:
This requirement is to make sure that the “Representation METS.xml” points to the data file in the data folder in the representation. Since SIARD is a file format and proprietary database dumps in most cases are also single files, then this requirement expects to be one and only one file.

Rationales in 3.5 METS requirements between Package and Representation

SIARD_14 Rationale

Requirement:

SIARD_14	Type mets/@TYPE	If the value in representation mets/@csip:OTHERCONTENTINFORMATIONTYPE is {SIARD_1.0, SIARD_2.0, SIARD_2.1, SIARD_2.2, Database_dump} then the	1..1 MUST

	Package METS.xml fileGrp who refers to the Package METS.xml MUST have the same value.	

Rationale/description:
Since there is information about the SIARD version in the “Package METS.xml” fileGroup attribute csip:OTHERCONTENTINFORMATIONTYPE (see SIARD_7), and the same information is in the “Representation METS.xml”, this requirement makes sure that it MUST, in fact, be the same information.

We are aware of the redundancy and that this can have some disadvantages, but the reason why the same information needs to be at both package and representation is that if one “stands” at the “package level” it must be possible to see what representations with which SIARD-version are available, and if one stands at representation level the representation itself can state which SIARD version it contains.

Rationales in 3.6 {SIARD_1.0, SIARD_2.0, SIARD_2.1, SIARD_2.2} – requirements

SIARD_15 Rationale

Requirement:

SIARD_15	If the value in mets/@csip:OTHERCONTENTINFORMATIONTYPE is {SIARD_1.0, SIARD_2.0, SIARD_2.1, SIARD_2.2} then there MUST exist a file named [databaseName].siard in representations/[RepresentationName]/data	1..1 MUST

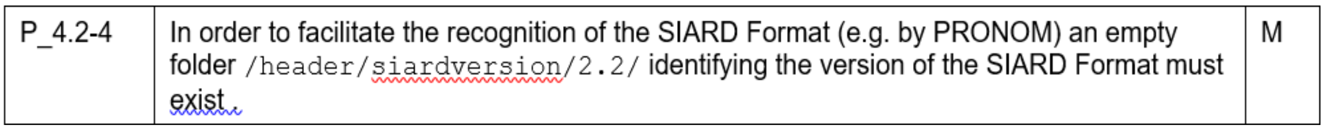
Rationale/description:
This requirement is made to make sure that if the “Representation METS.xml” states that it is a SIARD-representation then there MUST be a SIARD-file.

SIARD_16 Rationale

Requirement:

SIARD_16	The SIARD version of the SIARD-file MUST be the same as the version provided in mets/@csip:OTHERCONTENTINFORMATIONTYPE and fileSec/fileGrp/file@csip:OTHERCONTENTINFORMATIONTYPE	**1..1** MUST

Rationale/description:
This requirement is made to make sure that the version stated in the “Representation METS.xml” is in fact the same version. See also requirement P_4.2-4 in the SIARD specification which states that the SIARD version is stated in the header folder:



P_4.2-4	In order to facilitate the recognition of the SIARD Format (e.g. by PRONOM) an empty folder `/header/siardversion/2.2/` identifying the version of the SIARD Format must exist.	M

SIARD_17 Rationale

Requirement:

SIARD_17	The representations/[RepresentationName]/data/[databaseName].siard SHOULD be a valid SIARD file	**0..1** SHOULD

Rationale/description:
This is only a should requirement because invalid SIARD files might occur, and SIARD files can be split.

SIARD_18 Rationale

Requirement:

SIARD_18	There SHOULD be a minimum of one validation report in the documentation folder for the validation of the SIARD-file	1..n SHOULD

Rationale/description:
This should stand as proof that the SIARD-file in one point in time, by at least one validation tool was deemed valid. See for example https://kost-ceco.ch/cms/kost_val_de.html or https://database-preservation.com/ for SIARD-validators.

SIARD_19 Rationale

Requirement:

SIARD_19	The file name of the SIARD file representations/[RepresentationName]/data/[databaseName].siard MAY be the short database identifier of the database as specified in the <dbname> element of the metadata.xml file in the SIARD file, but it is not recommended.	0..n MAY

Rationale/description:
This is to make sure for the user of the specifications to avoid tight couplings.

SIARD_19a Rationale

Requirement:

SIARD_19a	From SIARD2.2 and onwards if a .siard file is larger than a desired or imposed implementation limit, then it may be physically split into file parts which are then placed in the same location that the .siard file would have been. Each file part must have the suffix _part[nnn] with nnn beginning with 001. In this case, SIARD_15, SIARD_16, SIARD_17, SIARD_18 and SIARD_19 refer to the complete SIARD file as if it was re-assembled from the constituent parts.	0..n MAY

Rationale/description:

It might be considered that the storage of extremely large SIARD files is either unwise, technically difficult or impossible due to operating system or file system limitations. In this case, the SIARD file may be split into parts. The size of each split is up to the discretion of the creator. It is, however conceived as a physical split, and no one part of the split SIARD file can be processed further on its own. To be processed, the parts would have to be re-joined in a fashion which reverses the process of the split.

A simple split command may be used. See for example https://man7.org/linux/man-pages/man1/split.1.html

Additionally, ZIP supports a method for splitting a zip file. See SIARD2.2, Appendix G for further information.

Also, see https://kb.winzip.com/help/winzip/help_splitdlg.htm and

https://www.addictivetips.com/windows-tips/single-file-multiple-zip-files/

SIARD_19b Rationale

Requirement:

SIARD_19b	From SIARD2.2 and onwards, if a mapping file is used to describe the physical location of segments, then a file mapping.txt **MUST** be provided at the same location as the .siard file. See SIARD2.2 section S_8.1.2.0 for further information regarding its format.	1..1 MUST

Rationale/description:

This rule is used to determine the placement of an optional mapping file.

In the case of scalability issues with many LOBS in one physical location, the segmentation of folders may not be sufficient as these LOBS will often be in the same column. A mapping file may be provided which advises the location of the actual segment. An example of the file is

Example

```
Northwind_lobs/s0_t2_c4/seg_0/ file://storagesrv1.sfa.ch/Home/stor/
Northwind_lobs/s0_t2_c4/seg_1/ file://storagesrv1.sfa.ch/Home/stor/
Northwind_lobs/s0_t2_c4/seg_2/ file://storagesrv2.sfa.ch/Home/stor/
```

The concept here is to give an indication of when LOBS in distinct columns are so large that they may need to be stored on physically separate servers or storage locations. This only has a usable value at the current archive location, and archives in transit or on reaching a new location may decide to update the file to reflect new storage parameters. This simply means that the mapping file is of zero use once the archive has left its source location and can be ignored by any receiving archive or when inspected during transport.

See SIARD2.2 section S_8.1.2-0 for further information.

Rationales in 3.7 {Database_dump} – requirements

CITS SIARD text:

> *"For authenticity and possible dissemination purposes, the OAIS might want to have a representation with a proprietary database dump from the original database management system."*

SIARD_20 Rationale

Requirement:

SIARD_20	If the value in mets/@csip:OTHERCONTENTINFORMATIONTYPE is "Database_dump" then there **MUST** exist a proprietary database dump in representations/[RepresentationName]/data	1..1 MUST

Rationale/description:
This requirement is made to make sure that if the "Representation METS.xml" states that it is a Database_dump-representation then there MUST be a database dump file.

SIARD_21 Rationale

Requirement:

SIARD_21	There **SHOULD** be preservation metadata describing the proprietary database dump	1..n SHOULD

Rationale/description:
At the representation level, there should be some kind of metadata.

Rationales in 3.8 {SIARD_lobs} – requirements

CITS SIARD text:

"A relational database can consist solely of table data, but it can as easily have large objects (LOBs). Large object (LOB) is the common description for large character content (CLOB) or large binary (BLOB) content – such as video, sound, images, word processing documents, etc.

These LOBs can be internal and stored inside a relational database as CLOBs or BLOBs within cells or be external and stored outside as external files – also called external LOBs (SQL/MED).

In the SIARD specification from SIARD 2.0 and onwards, the internal LOBs can be stored inline within cells or inside in the folder structure in the .siard-file or outside the .siard file. External LOBs can be stored outside the .siard file.

SIARD_22 Rationale

Requirement:

SIARD_22	If a database has LOBs outside the .siard-file then these **MUST** be stored in the same representation as the .siard-file in the directory "representations/[RepresentationName]/data"	1..n MUST

Rationale/description:

SIARD 2.2 and greater specifies that LOBS may be provided outside the SIARD file. In this case, the LOBS MUST be provided in the same representation as the SIARD file, which lies in the "data" area of the representation.

SIARD_22a Rationale

Requirement:

SIARD_22a	From SIARD2.2 and onwards, a manifest file **MAY** be used to increase interoperability to document LOBS stored outside the .siard file. In this case, then a file manifest.txt MAY be provided next to the .siard file. See SIARD2.2 section S_8.1.3-0 for further information	0..n MAY

Rationale/description:

To improve interoperability then a manifest file can be provided to document the LOBS outside the SIARD file. The LOBS will be provided either as a zip file (see SIARD22c) or as a folder structure outside of the SIARD file.

The location of the manifest.txt file should be outside the .siard file but directly next to it at the same level.

The LOBS files would also naturally receive documentation in the representation level of METS_xml. This should be done to be compliant with the requirements of the CSIP package.

If the LOBS files are individually provided, then the METS file will properly document them all. If a zip file is provided (see SIARD22c) then the METS file will only document the zip file and not the zipped files individually. Hence providing a manifest.txt only has an overall benefit if zipped LOB files are used.

SIARD_22b Rationale

Requirement:

SIARD22b	LOBS that are located as per SIARD22 **MUST** conform to a defined structure. • A main LOB folder named [databaseName]_lobs • A LOB folder for each column named after the schema no. i, table no. j, column no. k; i. e.: s[i]_t[j]_c[k] • A folder named seg_0 • A LOB file named after the table no. j, column no. k and row no. l of the LOB i.e. t[j]_c[k]_r[l] • A LOB file name suffix named bin (or a file extension associated with the MIME type of the lob file in case this is known *(see restrictions under SIARD22 section P_4.2-6).)* See SIARD2.2 section L_7.1.0 for further information.	1..1 MUST

Rationale/description:

The structure MUST have a predetermined format so that it can be verified and files properly locatable.

Segmentation can be performed to cater for folder limits or file size limits. In the latter case, a single LOB file can be split over more than one segment. See SIARD2.2 section 8.1.1

An example of a file structure from the above definition is :

```
Northwind_lobs/
     s0_t2_c4/
          seg_0/
               t2_c4_r1.bin
               t2_c4_r2.bin
               t2_c4_r3.bin
               t2_c4_r4.bin
          seg_1/              <!-- folder file number limit -->
               t2_c4_r5.bin
               t2_c4_r6.bin_part01     <!-- file size limit -->
          seg_2/
               t2_c4_r6.bin_part02
               t2_c4_r7.bin
               t2_c4_r8.bin
     s0_t2_c8/
          seg_0/
               t2_c8_r3.bin
     s0_t11_c6/
          seg_0/
                   t11_c6_r7.bin
```

```
The files themselves may be true binary files or tiff files, for example. They may use the
appropriate file extension such as .tiff, .jpeg and not just .bin
```

SIARD_22c Rationale

Requirement:

SIARD22c	The [databaseName]_lobs/seg_[]/folders **MAY** be packaged as ZIP files with the suffix .zip See SIARD2.2 section L_7.1-1 for further information	0..1 MAY

Rationale/description:

If there are hundreds of binary files, then it may be more pragmatic to provide them as a zip files. It may be necessary to use a 64 version of zip to overcome the 4GB limit of a 32 bit zip application

In this case, any manifest file should still document the individual LOBS and not the zip files. The METS file, by nature, will only document the zipped files.

The zip file should not be password encrypted.

Rationales in 4.1 Submission Agreement requirements

SIARD 23-28 Rationale

CITS SIARD text:

"There should be a submission agreement in the SIP representation that has been tailored to handle the preservation of relational databases. Since no standard for submission agreements for databases exist yet, the following requirements are not yet able to be automatically validated at this specification level. It is up to the business-specific specification layer or local implementation layer (see 1.2 Layered Data Model) to set up requirements that can be automatically validated. "

Rationale/description:

All the requirements are "should"-requirements, and they are not able to be automatically validated. Therefore a rationale/description is not provided at this point. However, the review answers show that there is a need to have a standard for Submission Agreements for relational databases. See also the *"CEF eArchiving Webinar #14 - Practical applications of digital archiving - submission agreements"* retrieved the 29-07-2021 at: https://www.youtube.com/watch?v=UlEdGoooZGw

SIARD_23	There SHOULD be a submission agreement in the SIP representation that has been tailored to handle the preservation of relational databases.	1..1 SHOULD
SIARD_24	The submission agreement SHOULD describe how many representations of the database that the Producer has to submit.	0..1 SHOULD
SIARD_25	The submission agreement SHOULD describe whether the submitted representations of a database is 1:1 with the running database (Full SIARD export) or if any alterations have been made (only a subset of tables).	0..1 SHOULD
SIARD_26	The submission agreement SHOULD list the tables that are required to be submitted to the archive and to be preserved.	0..1 SHOULD
SIARD_27	The submission agreement SHOULD list a set of SQL queries that are decided to be submitted to the archive and are to be preserved under the <views>-element in metadata.xml. The SQL queries SHOULD provide the most useful queries in the database for designated communities.	0..1 SHOULD
SIARD_28	The submission agreement SHOULD list the documentation that is decided to be submitted to the archive. See 7 Documentation requirements.	0..1 SHOULD

Rationales in 7 Documentation requirements

CITS SIARD text:

There should be documentation in the representations and/or in the information package. It is up to the business-specific specification layer or local implementation layer (see 1.2 Layered Data Model) to set up requirements that can be automatically validated for many of the requirements.

Rationale/description:

Many of the requirements are "should"-requirements, and they are not able to be automatically validated.

SIARD_29 Rationale

Requirement:

SIARD_29	.siard-file Documentation folder	Tables, columns/fields, keys, coded values **SHOULD** be explained, preferably in the metadata.xml and via code tables or the SIARD file or alternatively in the Documentation folder.	1..n SHOULD

Rationale/description:

There are many databases where coded values and relations between columns are handled in the application layer and even more databases where a description of the content of tables and column exist outside the RDBMS. When exporting the databases into a .siard-file, the meaning behind coded values and relations between columns and tables are often lost, and it is therefore important for understanding the structure of the database that this external information is preserved. The best way of preserving this is to describe the data content in the description-elements in metadata.xml, to mark up the foreign keys in metadata.xml, and to create code tables to the .siard.file.

The risk when only documenting these in a document in the documentation folder is that there is no validation. For example, if the descriptions do not fit the actual table content.

SIARD_29a Rationale

Requirement:

SIARD_29a	metadata.xml	Tables and columns that did not exist in the original database layer (such as code tables created for archiving purposes) **SHOULD** be named with the prefix "Arch_" in metadata.xml	1..n SHOULD

Rationale/description:

This requirement is a way to differentiate between the information from the originating system and information in the archived version. This is to ensure authenticity and provenance. The SIARD specification itself has a few takes on this since there are elements to describe original datatypes versus SQL datatypes, but there is no way to differentiate between tables/columns that existed in the original database versus if it has been created later. Examples, where this might fit, is when coded values in the database were only translated at the application layer.

SIARD_30 Rationale

Requirement:

SIARD_30	Documentation folder	There **SHOULD** be a system diagram in the Documentation folder. Preferably an Entity/Relationship Diagram.	1..n SHOULD

Rationale/description:
See for example https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model
Often a good diagram is the best way to understand the data and its relations.

SIARD_30a Rationale

Requirement:

SIARD_30a	Documentation \database_diagrams	The system diagrams SHOULD be located in a subfolder called database_diagrams	1..1 SHOULD

Rationale/description:

By having a folder and name convention, it is possible to support interoperability for showing and creating system diagrams. It is recommended to be able to differentiate between different documentation types, and therefore all system diagrams are placed in their own subfolder.

SIARD_30b Rationale

Requirement:

SIARD_30b	Documentation \database_diagrams	System diagrams **MAY** be archived as PNG-files.	0..n MAY

Rationale/description:
It is up to the different organisations and individuals to define their preferred preservation formats, but the authors behind the CITS SIARD specification recommends that system diagrams are archived as .PNG-files in order to facilitate interoperability.

SIARD_30c and SIARD_30d Rationale

Requirement:

SIARD_30c	Documentation \database _diagrams	System diagrams visualising the original database **SHOULD** be named with the prefix "Original_"	1..n SHOULD
SIARD_30d	Documentation \ database _diagrams	System diagrams visualising the archived database **SHOULD** be named with the prefix "Archived_"	1..n SHOULD

Rationale/description:
Sometimes the archived database and the original database is not identical. This could be if only a subset of tables has been chosen to be archived or if the archived database has been populated with extra tables such as code tables. It can therefore be good to preserve a diagram of the original diagram in order to ensure authenticity and provenance.

SIARD_31 Rationale

Requirement:

SIARD_31	.siard-file Documentation folder	The (main) system-user dialogues **SHOULD** be documented, down to the identification of the database columns/fields involved in the dialogues, documented as a combination of: · Screenshots, annotated with column/field descriptions, stored in the Documentation folder. · User documentation describing the system-user dialogue stored in the Documentation folder. · Views, if available, as part of the SIARD file. · If views are not present, additional descriptions of the system (application) logic, stored in the Documentation folder.	1..n SHOULD

Rationale/description:

SIARD_32 Rationale

Requirement:

SIARD_32	Documentation folder	Documentation of the legal context of the database and associated system **SHOULD** be provided in the Documentation folder.	1..n SHOULD

Rationale/description:

SIARD_33 Rationale

Requirement:

SIARD_33	Documentation folder	There **MAY** be videos or screen dumps from the system as seen from the user's point of view in the Documentation folder.	1..n MAY

Rationale/description:

5 Tools, examples, communities

This section contains an overview of available example packages, tools, and interest groups related to the CITS SIARD and SIARD specifications as a means to "get your hands dirty" and take action in developing the field of database preservation.

Examples

Examples are often the best teacher. In this section, we will guide the reader to examples of valid CITS SIARD packages. These can, in general, be found at the GitHub-site for this specification: https://citssiard.dilcis.eu/examples but in this section, we are providing a table

Currently, there are two examples:

Link	Name and Description	Number of representations	Proprietary DBMS	BLOBs
https://citssiard.dilcis.eu/examples/IP_18006_SIARD2_1Rep_externallobs	Northwind example database	1 (SIARD2.1)	SQL Server	Yes, outside the SIARD file
https://citssiard.dilcis.eu/examples/IP_18007_SIARD2_2Rep_externallobs	Northwind example database	2 (SIARD2.1) (database_dump)	SQL Server	Yes, outside the SIARD file

It is a plan that more examples will follow. If you have a good example, please let us know via the "Issues"-function in GitHub portal https://github.com/DILCISBoard/CITS-SIARD/issues.

Tools

There are freely available tools that can create SIARD-files and/or validate SIARD-files. These can be found in the following table:

Name	Description	Link
SIARD suite	From the official link: "SIARD Suite is a software developed by the Federal Archives to simplify archiving of relational databases. It complies with international standards and is used in over 50 countries around the globe. It is provided by the Federal Archives free of charge."	Official link: https://www.bar.admin.ch/bar/en/home/archiving/tools/siard-suite.html Copter: https://coptr.digipres.org/SIARD_Suite
KostVAL	From the GitHub page: The KOST-Val application is used tos validate TIFF, SIARD, PDF/A, JP2, JPEG-Files and Submission Information Packages (SIP).	Official link: https://kost-ceco.ch/cms/kost_val_de.html GitHub page: https://github.com/KOST-CECO/KOST-Val
DBPTK	Database Preservation Toolkit From the official link: "Desktop: Desktop application to store database to archival format, validate it and browse the content. Enterprise: DBPTK Enterprise deployment using docker" Developer: A command-line tool and development library for automation and systems integration."	Official link: https://database-preservation.com/

If you know of other good examples of freely available tools - please let us know via the "Issues"-function in GitHub portal https://github.com/DILCISBoard/CITS-SIARD/issues.

Communities

The Relational Database Archiving Interest Group[4]

The DILCIS Board and eArchiving Building Block maintain a “Relational Database Archiving Interest Group” which documents and shares best practices on database archiving, the application of the SIARD and SIARD CITS specifications and related tools. If you are interested in joining the interest group please register following the information at https://dilcis.eu/content-types/cs-siard

For now, the Interest Group has published two international case studies on the webpage:

- Case Study 1 - Preserving databases using SIARD: Experiences with workflows and documentation practices
- Case Study 2 - Preserving databases using SIARD: Experiences working with large databases and their preservation

If you have other case studies, please contact the Relational Database Archiving Interest Group through the groups email list..

[4] Text taken from https://dilcis.eu/content-types/cs-siard

2 Postface

AUTHOR(S)	
Name(s)	Organisation(s)
Phillip Aasvang Tømmerholt	The Danish National Archives
Anders Bo Nielsen	The Danish National Archives
Martin Dew-Hattens	The Danish National Archives

REVIEWER(S)	
Name(s)	Organisation(s)
Ann-Kristin Egeland	The Danish National Archives
Anonymous	Review round spring 2021
[Name]	[Affiliation]

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
Review version	1st of February 2021	Phillip Aasvang Tømmerholt Anders Bo Nielsen	The Danish National Archives	First draft
0.9	1st of August 2021	Phillip Aasvang Tømmerholt Anders Bo Nielsen Martin Dew-Hattens	The Danish National Archives	Update to CITS SIARD 1.0.0
1.0	31st of August 2021	Various	Various	Publication of version 1.0

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.